POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Understanding the Practice and Supporting the Creation of Design Systems

YASSINE LAMINE
Département de génie informatique et génie logiciel

Mémoire présenté en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
Génie informatique

Août 2020

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Ce mémoire intitulé :

Understanding the Practice and Supporting the Creation of Design Systems

présenté par **Yassine LAMINE**
en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
a été dûment accepté par le jury d'examen constitué de :

KHOMH FOUTSE, président
CHENG JINGHUI, membre et directeur de recherche
DOYON-POULIN PHILIPPE, membre

DEDICATION

I dedicate this work to my loving family, you're the reason I'm where I'm today.

ACKNOWLEDGMENTS

I would like to express my deepest appreciation to my advisor Prof.Jinghui Cheng of the Computer and Software Engineering department of Polytechnique Montreal. His door was always open to my questions whether they were about my research or other issues I encounter with my studies. I would also like to extend my very profound gratitude to the jury members who reviewed this thesis.

Many thanks for all the people who participated in this research. Thank you for your time and the knowledge you provided us.

Finally, I'm extremely grateful to Al Ghurair Foundation for education for their generous scholarship that lightened my financial burden which allowed me to focus more on learning and expanding my knowledge.

RÉSUMÉ

L'utilisabilité spécifie à quel point un système logiciel doit être facile, efficace, anti-erreur et agréable pour être utilisé par des utilisateurs humains. Au cours des deux à trois dernières années, de nombreuses entreprises et organisations qui développent des technologies de l'information et de la communication ont commencé à adopter des systèmes de conception qui aident à améliorer la convivialité de leurs produits. Ces entreprises et organisations ont généralement plus d'un produit sur différentes plates-formes, ce qui présente divers défis de conception tels que la cohérence, la maintenabilité, des coûts élevés et une mauvaise collaboration entre les équipes de conception. Les systèmes de conception aident à standardiser la conception de l'interaction utilisateur en créant des directives et des composants réutilisables pouvant être utilisés dans tous les produits et / ou versions de produits.

Pour comprendre la popularité croissante des systèmes de conception, nous avons mené une étude d'analyse de contenu des problèmes de système de conception open source de GitHub. Nous avons constaté que les problèmes du système de conception consistent principalement en des rapports de bogues ou des demandes de fonctionnalités et en améliorant le comportement de l'interface utilisateur ou la conception visuelle des composants du système de conception. En outre, nos résultats suggèrent que la communauté des systèmes de conception open source utilise le système de suivi des problèmes comme canal de communication pour discuter de divers sujets liés au système de conception. Nous avons également mené une étude exploratoire basée sur des entretiens avec des praticiens qui développent et maintiennent ces projets open source. Sur la base des résultats rapportés, nous avons identifié plusieurs avantages et défis concernant le développement et l'utilisation d'un système de conception pour la création de produits logiciels. De nombreux participants à notre étude d'entrevue ont mentionné que le principal avantage du système de conception est la cohérence qu'il offre dans la conception de l'interface utilisateur, cette cohérence améliore la communication sur la conception de l'interface utilisateur au sein de leur organisation, ainsi que l'efficacité et la vitesse de développement, ce qui réduit le coût du développement frontal. Cependant, ils ont rencontré des défis majeurs pour créer, faire évoluer et faire adopter le système de conception au sein de l'entreprise. Pour relever ces défis, nous avons conçu une application Web qui génère un système de conception complet et personnalisé en quelques étapes simples. Nous avons proposé une solution d'outils avec diverses fonctionnalités dont nous avons évalué sa convivialité et son efficacité par deux groupes de praticiens du logiciel: (1) des praticiens expérimentés qui ont travaillé sur des systèmes de conception et (2) des utilisateurs novices qui n'ont pas une expérience significative des systèmes de conception. Nous avons reçu des

commentaires positifs sur l'utilité de notre application pour nos participants, ainsi que des suggestions d'améliorations possibles ainsi que certains problèmes qui doivent être résolus dans une future version bêta de notre outils.

ABSTRACT

Usability specifies how easy, efficient, error-preventing, and pleasant a software system must be to be used by human users. In the recent two to three years, many companies and organizations that develop information and communication technologies have started to adopt design systems that help address the usability of their products. These companies and organizations usually have more than one product across different platforms, which introduces various design challenges such as consistency, maintainability, large costs, and poor collaboration between design teams. Design systems help standardize user interaction design by creating guidelines and reusable components that can be used across products and/or product versions.

To understand the growing popularity of design systems, we first conducted a content analysis study of open-source design system issues from GitHub. We found that design system issues consist mainly of bug reports, feature requests and improvement of the UI behavior or visual design of design system components. Furthermore, our findings suggest that the open-source design system community use the issue tracking systems as a communication channel for discussing various topics related to the design system. We then conducted an exploratory interview-based study with practitioners who develop and maintain these open source projects. Based on the reported results, we identified several benefits and challenges concerning the development and the use of a design system for crafting software products. Many participants in our interview study mentioned that the design system's key advantage is the consistency they offer in UI design; this consistency improves communication about the UI design inside their organization alongside efficiency and development speed which reduces the cost of front-end development. However, they experienced major challenges in creating, evolving, and getting adoption to the design system inside the company. To address these challenges, we designed a web-based application that generates a complete and custom design system in a few simple steps. We proposed a tool solution with various features that we evaluated its usability and effectiveness by two groups of software practitioners: (1) experienced practitioners that have worked on design systems and (2) novice users who do not have significant experience with design systems. We received positive feedback on the usefulness of our application from the participants along with suggestions for possible improvements and also certain issues that need to be addressed in a future version of our tool.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ACRONYMS

OSS	Open Source Software
UI	User Interface
UX	User Experience
ITS	Issue Tracking System
UCD	User-Centered Design
SUS	System Usability Scale
SMEQ	Subjective Mental Effort Question

LIST OF APPENDICES

CHAPTER 1 INTRODUCTION

User interface design is an essential aspect of software development that aims at optimizing the ways in which end-users interact with a computer system. It directly affects user experience and adoption, and thus pose significant financial impacts to software organizations. Over the past decade, front-end design tools and development technologies have experienced a rapid evolution.

The design system is one of such novel concepts that is being adopted in recent years by many companies and organizations who develop information and communication technologies. This novel approach is defined as "a collection of reusable components, guided by clear standards, that can be assembled together to build any number of applications" [1]. The prominent examples of design systems include Google's Material design[1] and Microsoft's fluent UI[2]. Motivations of adopting a design system is usually originated from the fact that the companies or organizations have more than one product across different platforms, which introduces various design challenges such as consistency, maintainability, large costs and poor collaboration among design and development teams [1]. Design systems help standardize user interaction design within an organization by creating guidelines and reusable components that can be adopted across multiple products and/or product versions. They ensure that the desired system characteristics concerning usability (such as efficiency, accessibility, and performance) are consistently met while helping designers and developers efficiently build cross-platform user interfaces.

Take Google's Material design as an example, it is a design system that is currently used across multiple Google products (e.g Gmail, Youtube, Google drive, etc). It includes a variety of UI components for displaying content, navigation, actions, inputs for entering information, and alerting users. It has a large open-source community and is used by developers and companies other than Google to build their products after customizing and adapting it for their needs.

Despite the popularity in the industry, there is little academic work focused on the creation and use of design systems. Particularly, little knowledge has established as to how organizations currently manage (i.e. create, evolve, maintain, and use) the design systems and the effects of this management. More interestingly, although usually considered as private property, many design systems have become open source projects that are accessible by other

[1] https://material.io/

[2] https://www.microsoft.com/design/fluent/

designers and the general public. Knowledge about the motivation and the benefits (to both the organization who own the design system, and external software practitioners that use it) of putting them as open-source projects are also not established. Besides, there are currently no tools to address the challenges practitioners experience during the development, maintenance, and use of design systems.

In this work, we address these issues and investigate the management, development, and customization of design systems through three studies: (1) a content analysis on the issues from the Issue Tracking Systems (ITSs) of open-source design systems hosted on GitHub [3] to understand the open-source communities' interests towards design systems, (2) an interview study with software practitioners who develop and maintain open-source design systems to explore their practices and challenges, and finally (3) the creation and evaluation of a prototype tool for efficiently customizing design systems to address some prominent challenges experienced by the practitioners.

1.1 Research Objectives

This master thesis aims at studying the design system development process, maintenance and customization. More specifically, we focus on the following research questions:

RQ1: What types of issues do the open source communities raise about design systems?
RQ2: How do practitioners who create design systems perceive and approach this new technique (e.g. their perceived benefits, effective practices, and major challenges)?
RQ3: What are the features practitioners value in a tool designed to address the challenges related to the creation of design systems?

To answer RQ1, we conducted a content analysis of the data collected from the Issue Tracking Systems (ITSs) of 41 open-source design systems hosted on GitHub. Through analysis of 4714 issues about these design systems, we find that open source design systems have the same characteristics of an open-source software project with active use of ITS for discussions related to evolving the design system and also to address community-related issues.

To answer RQ2, we first conducted in-depth interviews with practitioners who have contributed to open-source design systems. Based on a preliminary investigation, we find that open source design system projects usually have a small number of core developers who build most of the system and a large number of peripheral contributors who contribute to smaller, and more specific parts of the system. We recruited 9 core contributors that work on

[3] https://www.github.com

different design systems for our interview study. These participants mentioned multiple benefits of using design systems such as consistency and coherence of the design, efficiency and development speed, improvements in communication and collaboration, a more optimized onboarding process, and reduced development costs. They also talked about the challenges that they face while working with this design approach like the difficulty of evolving it and the amount of communication required for that, also getting adopted can prove challenging. Their solutions to overcome these challenges include building their design system on top of existing UI components, along with having a dedicated team that is responsible for it. Additionally, documentation and communication are also required. Finally, they described how open-sourcing their design system benefits them in terms of the hiring process, code quality, gaining publicity, and having an open-source community. In the meantime, open sourcing has drawbacks such as losing control over it and additional management work to address privacy and security concerns.

To answer RQ3, we conclude this work by addressing a prominent challenge reported by the participants in the interview study, namely the challenge of efficiently extracting common design elements and elevate them into design systems. We created a web-based prototype tool that supports the customization of design systems based on a design system template and existing web application products (from which the design styles are to be extracted). We conducted a user study to evaluate the usability and effectiveness of our tool by two groups of software practitioners: (1) experienced practitioners that have worked on design systems ($N = 6$) and (2) novice users who do not have experience with design systems ($N = 9$). The choice of these two groups of participants is to ensure that our application is usable by users with different backgrounds and experience in software development. Our prototype received positive feedback by the participants, they found our tool useful for prototyping which can come in handy for designers to test their designs, additionally, it can be used by novice users that aren't experienced by UI development technologies. Also, it is an efficient solution for companies that don't have enough resources to develop a design system from scratch. We also received suggestions for possible improvements that will ameliorate the overall user experience in our tool.

1.2 Thesis Plan

This thesis organized as follows. In chapter 2, we present the related literature review on design patterns and Frameworks, user interaction designers, and open source development. Chapter 3, reports the content analysis study of design system issues on GitHub. Chapter 4 presents the interview study with experienced design system contributors. Chapter 5 introduces the evaluation of our proposed tool to address the design system's creation. Finally, Chapter 6 concludes and discusses future work. Figure 1.1 provides an overview of the three studies we conducted during this research project.

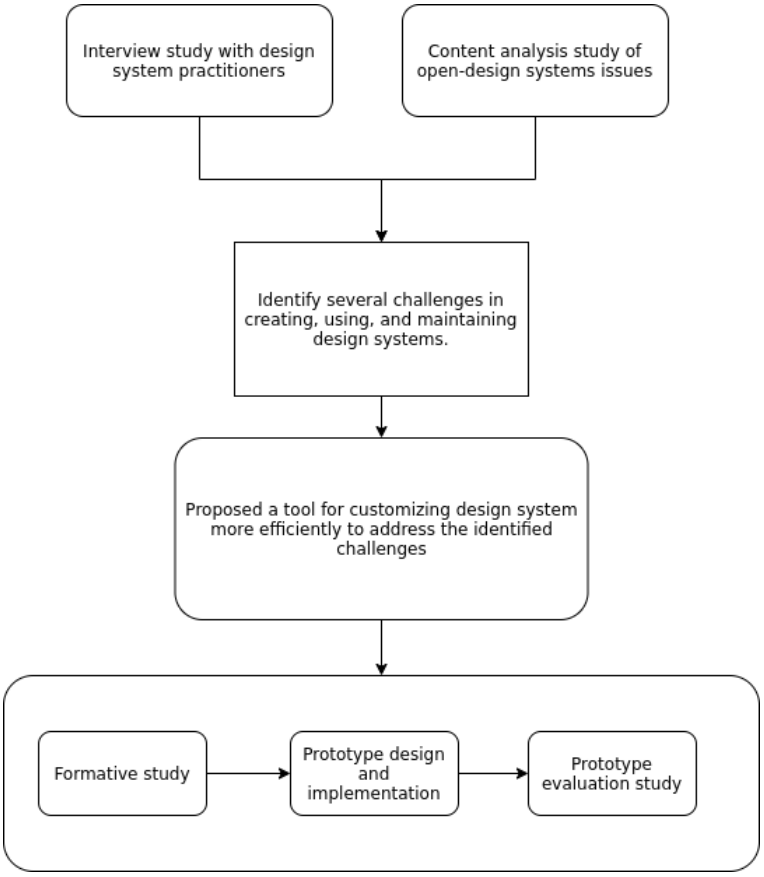


Figure 1.1 Thesis overview

CHAPTER 2 LITERATURE REVIEW

This work is the first academic research that focuses on various aspects of the design system approach, currently, there are no previous studies addressing this topic directly despite its increasing popularity in the industry. However, studies focused on (1) design patterns and frameworks,(2) understanding user experience designers, (3) collaboration between design and development teams, and (4) open-source development are closely related to design systems as open-source tools for user interface creation. In the following, we briefly review the literature in each category.

2.1 Design Patterns and Frameworks

The concept of design systems is related to design patterns and frameworks. Design patterns originated from Christopher Alexander's work in the field of architecture to describe a collection of common solutions that have solved recurring problems in corresponding design contexts [2]. It has been then adopted in many fields, including user interaction design. For example, Erickson conceptualized interaction design pattern language as a *lingua franca*, a common language support communication among various kinds of stakeholders (including users) in a design process [3]. Tidwell has summarized more than 100 interactive design patterns, put into 11 categories [4]; this pattern library has covered a wide range of aspects in user interaction design, such as content organization, navigation, form design, and data visualization. Many researchers have also focused on identifying design patterns in a specific domain, including web design [5, 6], mobile app design [7], information retrieval system design [8], ubiquitous computing [9], video games [10, 11], and, most recently, intelligent systems [12, 13]. In general, user interaction design patterns suggest high-level design solutions based on specific problems that the designers face. They are usually descriptive and include elements such as a name (for shortcut communication), a problem statement, a solution description, and several examples that realized the solution. Research has identified that design patterns are useful tools in participatory design and stakeholder communication [10]; however, communication breakdowns can still appear when design pattern is an unfamiliar concept [14].

Different from design patterns, design frameworks focused on not only user interaction design, but also on supporting the implementation of user interface elements [15]. Popular

design frameworks include Bootstrap[1], Foundation[2], Pure [3], Siimple [4], etc. They usually include reusable and customizable code to support construction of UI layout (e.g. many include a grid system for easy layout) and creation of UI components (e.g. buttons, lists, navigation bars, etc.). The development of these frameworks are often managed by an organization or company. However, they are aimed to serve the purpose of general UI design and implementation.

While design patterns and frameworks are related concepts to design systems, they focused on different aspects of design support. Different from patterns, which usually discuss highly generalized design solutions, design systems incorporate specific guidelines and components to create concrete support to both design and implementation. Different from frameworks, which often aimed to support general User interface (UI) design and implementation, design systems are more organization specific, incorporating branding-related elements and artifacts. Our study builds on top of the work on these related areas to explore design systems as a less represented but highly impactful topic.

2.2 Understanding User Interaction Designers

User interaction designers are a special group of practitioners who links the users needs and desires to the technical abilities provided in software systems. On one hand, they share many characteristics of UX (e.g user experience) and UI practitioners, considering, as Schön identified, the specific practical problem through "reflection-in-action" (i.e., constant assessing and adjusting actions in an unfolding situation) [16]. On the other hand, UI designers tackles the unique design task, which Cross argued to be "ill-defined" and requires solution-focused strategies [17]. In a seminal paper, Gould and Lewis outlined three principles that defined a "user-centered" approach: (a) early focus on the user, (b) empirical measurement, and (c) iterative design [18]. In a survey conducted in the 1980s with attendees of a HCI conference, they have found that only a small fraction of professional practitioners at that time considered these principles [18]. Since then, much work have been put to bridge the gap between HCI research and interaction design practice. For example, Goodman et al. proposed a shift in which "HCI researchers turn their attention to producing theories of interaction design practice that resonate with practitioners themselves" [19]. Gray has also found that design competence of newly graduated students evolve in real-world design practices and proposed pedagogical changes for interaction design practitioners [20].

[1]https://getbootstrap.com
[2]https://foundation.zurb.com
[3]https://purecss.io/
[4]https://www.siimple.xyz/

Many previous works have focused on understanding user interaction designers and their design practices that contains both practical and creative aspects. Stolterman et al. advocated the concept of *Designerly Tools* aimed at exploring "methods, tools, techniques, and approaches that support design activity in a way that is appreciated by practicing designers" [21]. They have found that designers considered physical or digital tools and conceptual tools (theories and approaches) in the same manner. Further, the designers used these tools to support two different types of activities: (1) to support design thinking and (2) to support creation of an artifact [21]. Leveraging these concepts, Gray has identified that interaction designers rely on a user-centered mindset to guide their adoption of design methods and tools [22]. Zhang and Wakkary have also pointed out that the personal experience of interaction designers can have a major impact to their practice [23]. More recently, Biskjaer et al. proposed a framework for analyzing the creativity methods in design. This framework supports the examination of a creativity method from three aspects: design space aspect, conceptual aspect, and concrete aspect [24].

The content analysis process and the design of the interview questions in our study were informed by the insights gained from the aforementioned studies. Particularly, we consider design systems a type of "designerly tools" to support both (1) ideating consistent and high-quality design through consideration of the design guidelines and (2) efficiently constructing mockups and working user interfaces with reusable components. We investigate how designers and the broader communities work with such tools.

2.3 Collaboration between UI/UX designers and developers

Collaboration between UI/UX designers and developers is an essential aspect in the development of user-facing software, in which both actors communicate and work with each other to produce a software product. Ferreira et al. investigated the integration of UI design into agile development. They reported on a case in which agile iterations facilitated usability testing, which allows developers to incorporate the results of these tests into subsequent iterations. Embracing the iterative development process had in fact led to improvement in developers-designers relationship [25]. Brown et al. have conducted a grounded theory field study with eight software teams and found that collaborative events between designers and developers happens frequently and often go beyond planning activities [26]. They also noticed that a few collaborations targeted what they called explicit alignment work (e.g prioritizing tasks) while the majority deals with implicit alignment. In other words, "designers and developers share an implicit understanding of how collaborative work should be carried out" and engage in collaborative activities that are not directly targeting work alignment (activities

such as assessing the project tensions, presenting and negotiating ideas, and exploring the feasibility of a design). They found that these types of implicit alignment activities happen in various interactions between designers and developers, including during scheduled meetings, impromptu events and work-related chats [26]. Both of the above mentioned studies show that designer-developer collaboration is positioned in the center of software production process.

However, the multi-disciplinary nature of the teamwork in software organization can be challenging. Chamberlain et al. described the existence of underlying communication issues between designers and developers. They observed project teams while working within UCD and agile development, from this observation they identified common themes (e.g user involvement, collaboration and culture, prototyping, and project life cycle). Based on theses themes they proposed a framework to alleviate the tension when integrating agile development and user-centred design (UCD) [27]. The authors pointed out that the issues involved in this integration process are the consequence of "a culture of defensiveness" between the two disciplines. In other words, both designers and developers defend their discipline in response to decisions made by the other party. Furthermore, Lundström et al. also investigated designers' perception of the developers' empathy towards designers and design work [28]. The authors identified that while developers were perceived as lacking understanding of design work, developers often considered user interaction design as an optional process and sometimes take their own design decisions with no justification or communication with designers, which results in mismatch in final product with the original design [28]. Leiva et al. also classified three types of breakdowns during the designer-developer collaboration while exploring the mismatches in the processes, tools, and considerations between designers and developers [29]. They found that breakdowns tended to occur when (1) a specific design detail is not communicated by designers, (2) a particular case is not covered in the design, or (3) the design failed to consider developers' technical constraints [29].

Researchers have also investigated several tools and tactics to mitigate these collaboration issues. For example, Wale-Kolade described five tactics needed by usability designers to integrate usability activities into agile development projects; these tactics included (1) engaging in persistent negotiation during the project, (2) seeking front-end developers' approval to increase influence, (3) following a placatory approach to gain a more complete and comprehensive overview form the users about usability of the system, (4) demonstrating the designers' competence and expressing their intentions about the project, and (5) establishing authority by diffusing designs [30]. On the developers side, getting involved in the design process in early phase mitigates design breakdowns although new breakdowns may still appear subsequently and require continuous monitoring [29].

Our work builds on these previous studies to explore the way design systems serve as a medium for facilitating the collaboration between UI/UX designers and developers. We also explore the challenges design system practitioners face while collaborating to create effective design systems.

2.4 Open Source Development

Open source is a software development model in which the source code of the software product is open for access under a certain license. This development model has gained popularity over the past decades and become a common practice in many software-intensive application domains and a mainstream focus for academic research [31,32]. Open source software projects are usually hosted on a public repository management platform (e.g. GitHub.com) and rely on various tools for tasks such as version control, project management, community engagement, and communication. Through these tools, geographically dispersed community members make diverse contributions to open source projects [33].

Many researchers and practitioners has pointed out that the development of open source software rely on a healthy community. A traditional view of a typical open source community resembles an "onion model" [34, 35]. This model suggests a hierarchical structure of responsibilities among community members that included a small number of core members and an increasingly larger number of various types of peripheral developers [34]. However, recent studies suggested that this structure is not stable and constantly evolving. While the size of the community increases, the boundaries of the hierarchical layers among the peripheral members tend to be blurred [36]. Additionally, several factors, such as the developer's motivation in participating in different projects and the social structure around the developer, can influence their evolution from a peripheral member to a core member [37].

Previous works have suggested that individuals were motivated to join an open source project because of both internal factors (such as socialization [38,39], learning opportunities [35], and self-perceived identity with respect to the project [40]) and external factors (such as human capital and monetary rewards [41]). Participating in open source projects, however, is not always easy. Newcomers in an open source community can experience major barriers when trying to make a contribution. For example, Lee et al. conducted a survey with "one-time" contributors of open source projects. They found that while this group of people usually focus on fixing bugs to "scratch their own back," many of them have deterred by barriers such as tedious contribution process and complex code base that prevented them from making further contribution [42]. Steinmacher et al. have also identified various social barriers such as communication breakdowns and reception issues that discouraged community newcomers

to make their first contribution [43]. Ways to help open source communities remove these barriers are then important for a sustainable project.

Our project builds on these previous studies and focus on open source design system projects to investigate core developers' motivations and practices, as well as the communities' concerns.

CHAPTER 3 ISSUE ANALYSIS OF DESIGN SYSTEM PROJECTS

Open-source design systems have become a trend among flagship IT companies, including Google with their material design [1], Microsoft with the Fluent design system [2], IBM with the Carbon design system [3], and many others. The open-source communities around design systems are also growing, serving as a key element in the success of these projects. However, design systems are a new type of artifact and their collaborative development within and beyond the companies and organizations is a new phenomenon. As a result, it is not clear what types of issues and tasks the open source communities around design systems are interested in and what challenges they face. To understand the main concerns of open-source communities around design systems, in this chapter, we focus on a content analysis study on the data collected from the Issue Tracking Systems (ITSs) of open-source design systems. The ITS is an important type of software engineering tool for open source development that has a forum-like functionality. It allows the entire community around an open-source project (both within and beyond the design and development team) to create, discuss (through comments), and manage the status of various *issues* that can include system and project-related tasks, enhancements, problems, and questions.

3.1 Methods

This study aims to identify the types of issues open source communities raise about design systems (i.e RQ1). We conducted a qualitative content analysis of 4714 issues collected from a wide range of open-source design system projects to identify the common themes manifested in these issues. In the following, we describe our methodology including the selection criteria of the open source software (OSS) design systems, the data collection process, and the content analysis approach. Figure 3.1 describes the different stages of our data collection process.

3.1.1 Design system project selection

To identify a wide range of open source design systems, we referred to the Adele repository[4], which is a curiated list of publicly available design systems, style guides and pattern libraries. We acknowledge that the definition of design system is not fully established. However, a

[1] material.io/design
[2] microsoft.com/design/fluent/
[3] carbondesignsystem.com/
[4] adele.uxpin.com

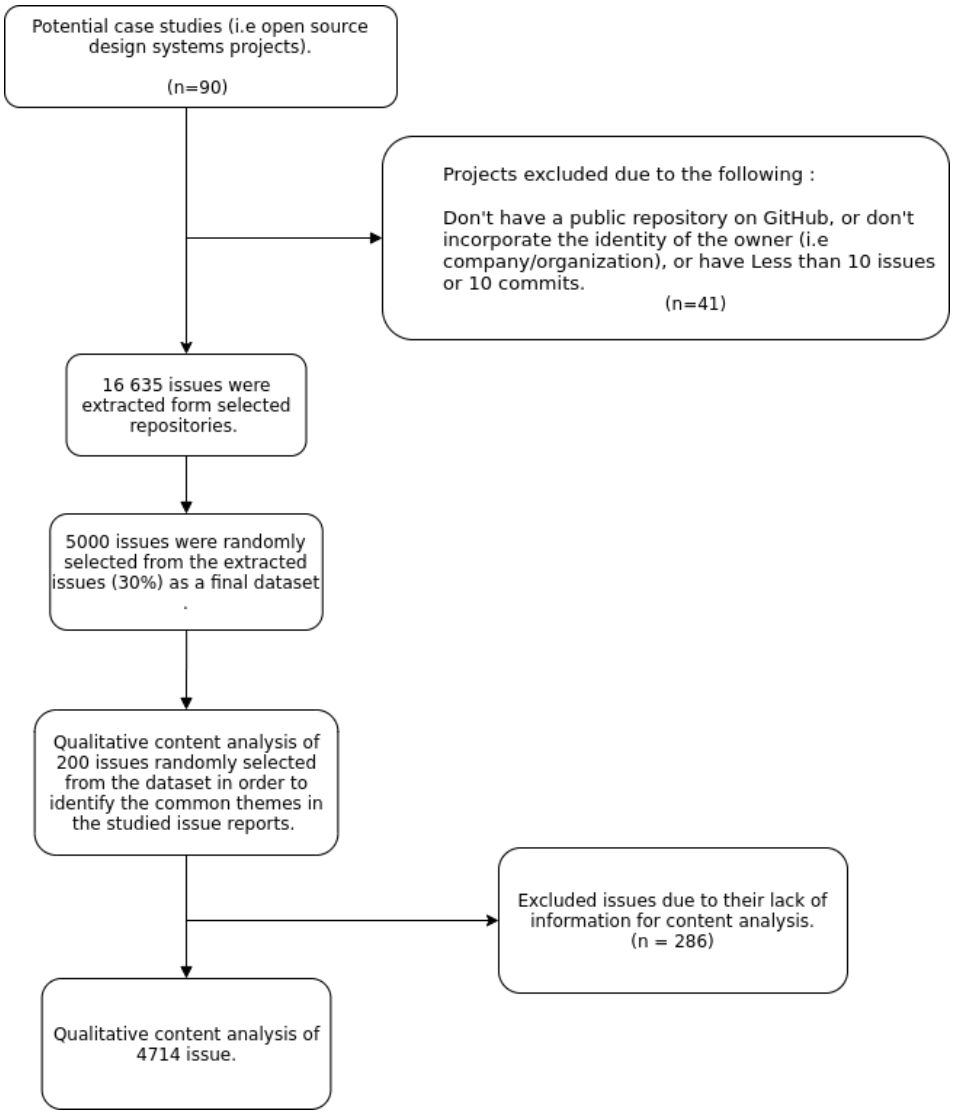


Figure 3.1 Content analysis study flow diagram

clear characteristic that differentiates design systems from style guides and pattern libraries is that design systems usually incorporate the identity of a company or an organization. We compared the design systems visual design with products of their owners using this criteria and selected 52 design systems among the 90 projects listed in the Adele repository. We then only included design system projects that have a public repository on GitHub; we focused on GitHub because of its increasing popularity over the past years among the open source communities as an integrated platform for open source project hosting and management. Finally, we filtered out projects that have less than 10 issues and 10 commits. This strategy allowed us to focus on a wide range of active design system projects and resulted in 41 projects in our dataset. Table 3.1 summarizes the characteristics of these design system projects.

Design system Repository	Open issues	Closed issues	Total issues	number of contributors	active contributors	stars	Forks
alfa-laboratory/arui-feather	12	41	53	64	5	411	83
alphagov/govuk-design-system	52	199	251	46	4	74	65
Altinn/DesignSystem	5	8	13	28	7	31	11
audi/audi-ui	15	3	18	5	0	155	25
auth0/cosmos	144	500	644	21	5	395	85
auth0/styleguide	12	46	58	20	4	143	51
brainly/style-guide	52	551	603	21	6	114	17
bring/hedwig	13	95	108	21	4	21	1
buzzfeed/solid	2	245	247	27	3	108	20
cfpb/capital-framework	64	255	319	17	6	50	35
Dropbox	14	14	28	12	1	794	46
Financial-Times/ft-origami	4	306	310	31	4	81	12
FirefoxUX/photon	58	217	275	31	3	186	49
fs-webdev/fs-styles	2	49	51	17	1	34	17
gctools-outilsgc/aurora-website	43	87	130	11	8	13	6
govau/design-system-components	43	307	350	20	4	584	59
instacart/Snacks	18	34	52	19	2	50	35
JetBrains/ring-ui	2	874	876	31	14	2216	107
liferay/lexicon-site	5	22	27	15	4	21	25
lonelyplanet/rizzo	6	55	61	57	4	728	86
mesosphere/cnvs	20	19	39	12	1	27	3
mineral-ui/mineral-ui	72	315	387	13	3	424	42
mozilla/protocol	70	160	230	11	4	50	26
OfficeDev/office-ui-fabric-react	404	3294	3698	388	25	4625	1016
pinterest/gestalt	22	75	97	44	9	3093	181
pivotal-cf/pivotal-ui	2	261	263	72	7	605	82
pluralsight/design-system	43	285	328	20	2	115	23
pricelinelabs/design-system	58	148	206	34	4	399	76
primer/css	75	195	270	67	4	8215	604
rei/rei-cedar	4	25	29	20	4	40	11
salesforce-ux/design-system	19	521	540	48	14	2530	560
seek-oss/seek-style-guide	2	33	35	42	5	256	37
Shopify/polaris-react	184	691	875	126	13	2433	385
SpareBank1/designsystem	55	189	244	47	7	71	40
sparkdesignsystem/spark-design-system	197	448	645	17	4	40	25
USAJOBS/design-system	34	119	153	8	1	27	15
uswds/uswds	33	1480	1513	98	2	5016	691
uswitch/ustyle	22	113	135	31	2	16	3
vmware/clarity	309	1820	2129	51	7	4743	458
vtex/styleguide	50	123	173	47	7	45	7
wework/plasma	36	23	59	23	4	17	4
Averages	55.53	347.43	402.97	42.268	5.3170	951.12	124.97

Table 3.1 List of the studied design systems; all counts were conducted on 41 design system, 2019

3.1.2 Data collection

We used the GitHub REST API[5] to collect the issue data from our list of design system repositories. Particularly, for each issue, we extracted the title, the description, the comments, the state (open or closed), and other identification information such as the creator and timestamp. Figure 3.2 provides an example of the issue data collected from this process. We then selected a random set of 4714 issues from 16 635 issues we extracted from the 41 design system repositories as our final dataset.

-Repository: alphagov/govuk-design-system
-Title: Cookie page is a bit confusing
-Issue number: #252
-State: closed
-Url: https://github.com/alphagov/govuk-design-system/issues/252
-Created by: joelanman
-Created at: 2018-04-25 11:43:20
————————————Description————————————
@amyhupe I don't think this section of the cookie page is right:
————————
![image](https://user-images.githubusercontent.com/1132904/39243631-06a1e190-4886-11e8-99d1-571705361f3b.png)
————————
Google Analytics doesn't set the Introductory message cookies. I suggest taking the Introductory message section out and putting it before the Google Analytics section. Happy to do this work as a PR if you're ok with that.
————————————Comments————————————
CommentId: 384297949
Created by: amyhupe
Created at: 2018-04-25 14:02:05
Hi Joe - if that's true, we should definitely change it. Happy for you to do that if you have the time.
————————————————————
CommentId: 396530278
Created by: 36degrees
Created at: 2018-06-12 09:40:59
Believe this was fixed in #255
————————————————————

Figure 3.2 Issue data example

[5]https://developer.github.com/v3/

3.1.3 Analysis Method

Content analysis is a qualitative analysis method that can be used following either an inductive or deductive approach depending on the nature of the study [44] [45]. While the deductive approach is beneficial if the overall goal was to test a preceding theory in a distinct scenario or to compare categories at different time periods, in case of lack of previous work, or when the knowledge about the studied phenomenon is fragmented, an inductive approach is recommended [44]. In our case, there is no previous work that focuses on exploring issues raised by open source communities about design systems. Thus, we adopted the inductive approach in this study.

We started our analysis by conducting an inductive qualitative coding on a random sample of 200 issues from the collected data in order to identify the common themes in the issue reports. We particularly focused on (1) the nature of the issues themselves, (2)the aspects of the design system projects that the issues address. This analysis was initially done independently by two researchers and was then followed by a thorough discussion and codes comparison to establish the themes. We then used these themes and coded 4514 issue reports after excluding 286 issues due to their lack of enough information to understand the problem they reported. The themes that emerged from this step were discussed among the two researchers and added to the coding schema.

3.2 Results

In this section, we describe the themes in each main category we identified during the content analysis process. For each category, we report its identified themes along with the number of their occurrences and the examples presented in the analyzed data. Results summary is included in Appendix A and B.

Issue nature:

In this category, we aimed to identify the root cause of why the issue is reported. We describe each theme and provide examples from the analyzed dataset.

Bugs (N=2808):

About half of the issues we analyzed are reported because of a bug; while the majority were related to **UI components**, the rest were associated with **documentation** and **peripheral code linked to software engineering aspects**. These issues usually described the steps or the process to reproduce the bug. The UI bugs can include a wide range that ad-

dresses two main UI issues. The first is **styling bugs**, including the layout of components, the overall layout of the page, and the color scheme issues. For example, Issue 4936 of the OfficeDev/office-ui-fabric-react project reported a color scheme issue that *"Close button for the Message Bar is not visible properly in the High Contrast Black mode."* And Issue 877 in the auth0/cosmos repository reported a layout issue that *"Contents from the docs sidebar are cut off screen."* The second type of UI components bugs is **behavioral bugs**, including issues about the navigation behavior, state change behavior, personalization, compatibility, and animation. For example, Issue 149 of the alphagov/govuk-design-system reported that *"Copy button does not give user feedback."* And Issues 380 of the vmware/clarity project discussed a bug in the progress bar animation. The **documentation bugs** mainly reported missing information in the documentation. For example, issue 4708 OfficeDev/office-ui-fabric-react was created to highlight lack of documentation for a certain component: *"TooltipHost directionalHint default value is not documented"*. Finally, **peripheral software engineering bugs** can be related to issue in testing or deployment code as well as third-party libraries. For instance, issue 5587 of the OfficeDev/office-ui-fabric-react repository reports a problem with the auto deployment code that has broken links to the other codes.

Requests for new elements (N=1370):

Interestingly, many design system issues are also requests for adding certain components. The frequently requested components included (1) **content containers** such as panel (e.g. Issue 55 in alphagov/govuk-design-system, "Add panel component"), card (e.g. Issue 281 in gctools-outilsgc/aurora-website, "Add profile card design to Components - Cards page as an example"), and dialogue; (2) **navigational components** such as sidebar, breadcrumb, and link (e.g. Issue 154 in auth0/cosmos, "[Link] Create Link component"); (3) **informational components** such as banner and progress indicator (e.g. Issue 827 in auth0/cosmos, "Add loading indicator to Table."); and (4) **page components** such as header, footer, 404 error content (e.g. Issue 47 in auth0/cosmos, "Create Page Header component"). These requests focused on both unique, complex components (e.g. dialogue, avatar system) and basic, universal ones (e.g. banner, header). This phenomenon indicated that the design system projects are not created as a static library, but are constantly evolving. It also shows that the components in the design systems are carefully selected, may only include the most relevant ones, and are enriched while the projects around it evolve.

Suggestions for improvements (N=378):

Some issues have also focused on requesting or suggesting improvements to an existing component or functionality of the design system. For example, in brainly/styleguide, issue 365, a contributor proposed to change the implementation of a certain component: *" I'd propose*

to use content-box as a mixin, could be used in places where we use it already, but would require us to create less generic components (i.e mint-ranking, mint-panel etc)).". We also encountered multiple issues that suggest improvements to increase community engagement and organization. For example, in issue 85 of seek design system, the issue reporter is suggesting adding a markdown file, a type of file for wiring documentation for GitHub repositories to assist contributors fo the design system: *"We should add a 'CONTRIBUTING.md' to assist those who are working on the style guide itself, not just consuming it ")*

Questions (N=158): Some of the issues posted in design system projects did not report problem of the system, but instead focused on asking questions and requesting information about design system components, functionalities, and the development process. For example, in Issue 154 of the cfpb/capital-framework repository, the contributor wondered whether they should stop using the auto-generated docs.

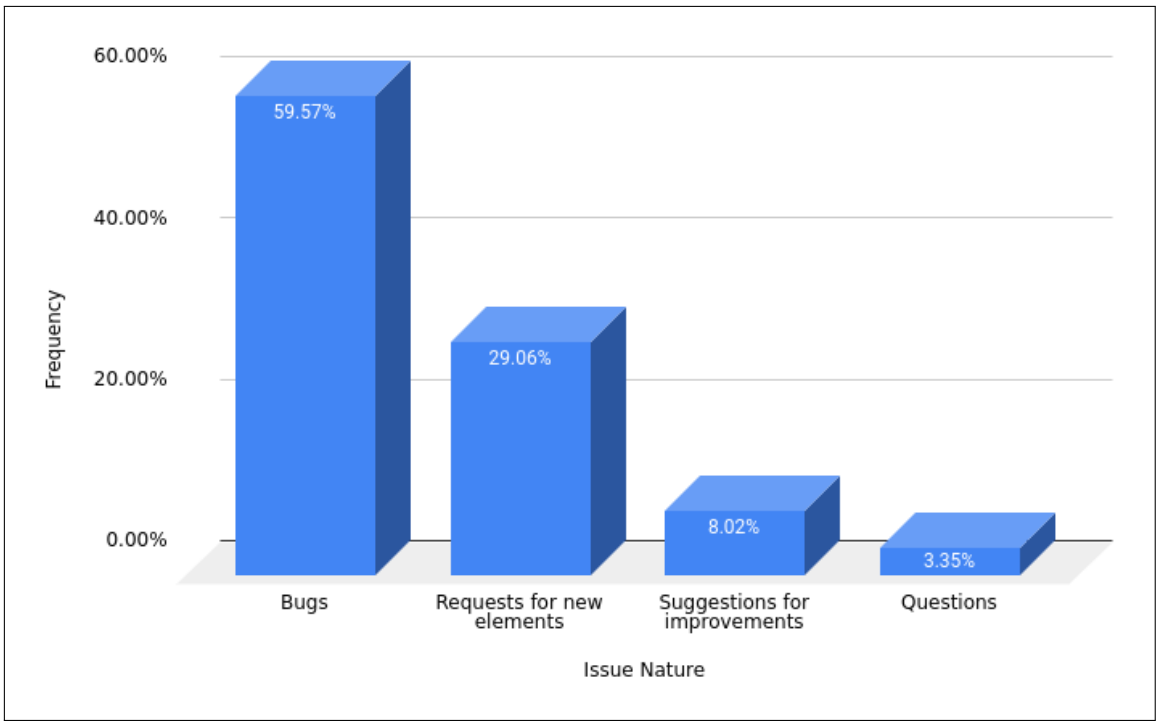


Figure 3.3 Issue nature themes frequency

Project aspect

Analyzing OSS design systems issues allowed us to categorize the specific core aspects of design system projects that are frequently discussed in the issue tracking system. In this section, we list all the aspects that we identified during our study.

The communities around open source design systems have raised issues for various aspects of the project; some issues have also dealt with multiple aspects. In this category, we aim to identify these main aspects. No surprisingly, most issues (*N=3261*) in design systems projects have addressed specific UI components. The common components discussed in these issues included content containers (e.g. tables, boxes, accordion), navigational components (e.g. links, menus, breadcrumb), form inputs, and informational components (e.g. progress bars, toolips, notifications). Besides these obvious UI component-related issues, we have found that the design system communities have focused on other project aspects. We categorised them into the following five groups. These additional aspects are our major focus of interest.

Behavior of UI components (N=2015): Many issues were focused on the behaviour and functionalities of the UI components. Some issues mentioned more than one behavior aspects. Among them, 245 issues discussed **state change behaviors** of UI components to provide appropriate user feedback, for example, when the component is hovered, focused, or disabled. E.g., the reporter of Issue 190 of brainly/style-guide requested to add a DISABLED state for all form inputs.There are 153 issues focused on **animations** in the UI. E.g., Issue 429 of the mineral-ui/mineral-ui repository reports the need to improve home page animations. **Navigational behaviors** such as pagination and scrolling are another frequent subject, discussed in 101 issues. E.g., in Issue 147 of mineral-ui/mineral-ui, a user mentioned: *"as a developer, I would like to implement pagination for tables in order to decrease the screen space and render/load times that my large data sets consume"*. **Input verification** is also a commonly discussed behavior, in 112 issues. E.g., the reporter of Issue 17 of alphagov/govuk-design-system repository asked for adding an email address verification pattern. Another 93 issues discussed **keyboard shortcuts** that could enable more efficient user interaction. Some of them are related to accessibility features that support screen readers; e.g. Issue 8424 of the OfficeDev/office-ui-fabric-react repository discussed a malfunction with the keyboard navigation feature. The **responsiveness** (i.e. allowing UI components and the page structure to adjust to the screen size) of the design system components is also a common topic, mentioned in 47 issues. E.g., the reporter of Issue 434 of the brainly/style-guide repository requested to make the popup menu responsive by default. Lastly, 17 design systems have considered **accessibility** as an important aspect that was addressed in 124 issues for our data-set. For example, the Microsoft design system, Office UI Fabric, includes a screen reader support

component, called Narrator that was frequently addressed in the issues; in Issue 1907, the contributor reported a bug: *"Visually impaired users, who rely on screen reader will not be aware of the suggestions as they type the search keyword because even though the suggestions might have changed there'd be no alert since the suggestions availability state has not changed. "* In fact, many issues addressing the accessibility aspects of the design systems are focused on supporting users with visual impairments, including having sufficient foreground-background contrast; this was mentioned in Financial-Times/ft-origami design system, *e.g. Issue 203 Some use cases in o-colors do not meet accessibility standards* , supporting keyboard shortcuts as shown in this example in offic-fabric, *e.g issue 1595, With JAWS on you are unable to navigate the nodes in the breadcrumb. Note that even without JAWS, one can't go to the overflow "..." node. Expected: Using arrow keys will navigate all the nodes.*

The visual design of UI components (N=480): In addition to addressing the UI behavior, many issues in design systems projects were addressing the visual design of UI components including **color scheme**, **Spacing**, and **typography**. For example, Issue 233 in wework/plasma repository, the issue reporter presented his fix to a coloring bug in the container background: *(e.g "fix removes the fill="#FFFFFF" line, enabling the container to determinate the background color..."*. Meanwhile, Issue 181 of vmware/clarity repository reported a bug in buttons spacing, in margins to be specific: *(i.e "Buttons should be assigned margins for specific context (for eg: in forms or modals footers). In general you should be able to use buttons inline without any margins").* Another example from shopify/polaris in Issue 2906 where the issue reporter updated the fonts used in the design system: *(e.g " Update polaris-tokens to use new font stack ".*

Documentation (N=321): The documentation is an aspect that is frequently discussed in the design system issues. In most issues about this aspect, contributors usually reported missing documentation, pointed out error or inaccuracy, and requested to improve documentation of a certain component. For example, in Issue 1647 of the OfficeDev/office-ui-fabric-react project, the issue creator reports an error in the design system version documentation: "*On the website's home page, we show the version of Fabric Core and Fabric React that the website documents. These are showing as a version range when we want to show the latest version only."*

Software engineering (N=310): Many issues were also focused on the aspects related to the software development process, tooling, infrastructure, programming, the use of frameworks, etc. The most frequently discussed topics in this category included (1) testing issues (e.g in issue 183, cfpb/capital-framework, *"The node-wcag tests seem to be failing for many (if not all) cf components. Here's the output:..."*), (2) deployment and release issues (e.g in issue

624, mineral-ui/mineral-ui , *"As a Mineral UI developer, I would like to improve the release script to make it easier to use, and to enable additional common tasks."*), (3) issues related to the design system's code (e.g in issue 4478, OfficeDev/office-ui-fabric-react, *"Component Classes should not define their methods with lambdas, as it prevents inheritance.."*), and (4) dependency related bugs (e.g issue 569, Shopify/polaris-react, *"Move 'app-bridge' to peer dependency..."*). Additionally, Because many design systems are built with third-party libraries and frameworks such as React and Bootstrap, these libraries and packages (usually managed by a package manager such as NPM (i.e a package manager for the JavaScript programming language)) were sometimes the focus of discussion in design systems issues. For example, a contributor of Issue 170 of the Altinn/DesignSystem repository requested to "*make it possible to run "npm start" from a chosen project.*" Some contributors also discussed the problems of relying on third-party libraries and proposed to reduce such reliance. For example, a contributor of Issue 53 of the fs-webdev/fs-styles repository reported that he prefers to drop the use of Bootstrap in the design system documentation: "*Some of the tag level styles are being inherited from bootstrap.css ... I'd love to see bootstrap dropped from the styleguide demo."*

Community (N=135): The issue tracking systems are also used as a communication tool for the design system teams to address community-level tasks and processes. Many issues aimed to start a discussion about attracting external **contributions** to the design system repository. For example in issue 725 in govuk-design-system repository, a user reported that *e.g "We want to enable more people to contribute, and to make it easier to make smaller contributions to the Design System"*. Some issues discussed tasks to help better **communicate** with the design system users. For example, the reporter of Issue 361 of the auth0/cosmos project suggested a *"need to publish a changelog to inform users of changes that we make to Cosmos, especially to the API. The changelog should be part of the documentation site."* Issue discussions also evolved around how to better **satisfy** the needs of the design system users. For example, the reporter of Issue 223 of govau/design-system-components indicated that *"the team needs a method for tracking what technology our users have access to and currently use."*. Some have also discussed process and practical issues to support the **growth** of the community. For example, the reporter of Issue 2301 of the uswds/uswds repository initiated the discussion about the importance to *"assure long-term growth with a small core team by adopting better open-source practices and growing our community of clients and contributors."*

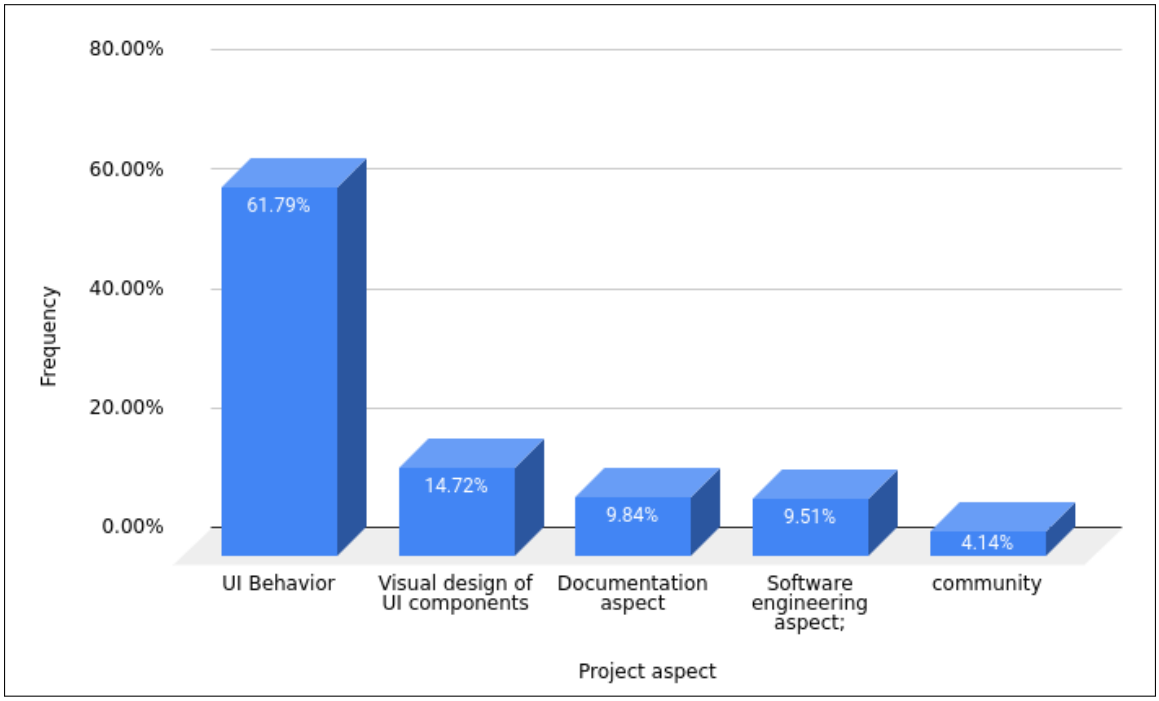


Figure 3.4 Project aspect theme frequency

3.3 Discussion

Our study of open source design system issues revealed that these projects share most characteristics with other open source projects. We noticed that the majority of issues (59.57%) are reporting bugs and requesting improvements and new features (29.06%), Figure 3.3 summarizes the frequency of all themes. Bissyandé et al. found similar results during their analysis of 20000 open source projects on Github, they reported that contributors are interested in both reporting bugs along with requesting new features in issues they report [46]. Furthermore, our findings revealed that design system community is using the issue tracking system not only for bug reports and feature requests, but also as a communication channel

to discuss various aspects related to the project. About 8.02 % of the issues we analyzed are reported to suggest improvements for a certain element or asking questions (3.35%) to clarify specific details about it. This finding echos with the results of Bertram et al.'s qualitative study of issue tracking systems [47]. They found that software engineering community use issue trackers as communication channels to discuss various aspect of the project in hand.

Additionally, we managed to categorize the most important aspects of design systems from the perspective of the open source communities. Visual design and behaviors of UI components is, no surprisingly, the center of design systems, comprising the majority of the issues analyzed (about 76.51%). Figure 3.4 represents the frequency of all identified themes. Because of the nature of design systems as both component libraries and UI design language and guidelines, we have speculated that the documentation issues would also be a frequent topic. To our surprise, however, they only comprised about 9.84% of the analyzed issues, a similar frequency as the issues associated with software engineering topics (9.51%). This phenomenon of less emphasized documentation issues indicated either (1) documentation aspects were discussed in other channels such as internal communications and were not an interest to the community outside of the organization that creates the design system, (2) the design system open source communities are still not mature enough to break through the common pitfall of open source development (i.e. focus on system-related aspects rather than user-related aspects [48]), (3) the documentations were usually well maintained by the team who manage the design systems and has little problems, or (4) users of OSS design systems are familiar with the use of UI templates and libraries and that the existing documentation is sufficient for their use-case. . Future studies need to be conducted to examine these hypotheses. What was also surprising in our findings is how community engagement is a topic that was also included in issue discussions (4.14% of the issues), which implies that design system owners are interested in getting their design systems adopted and surrounded by external developers. However, the direct motivations that the design system owners have in engaging with such activities are still not clear and will be a focus in the interview study presented in the next chapter.

CHAPTER 4 CONTRIBUTOR INTERVIEW STUDY

The issue analysis study allowed us to highlight the types of issues open source communities raise about design systems. However, this study does not explain the reasons behind those issues. In this chapter, we present an interview study with design system practitioners to further explore the common asked questions about design systems including its concept in general, the benefits, challenges, and best practices to create and evolve a design system.

4.1 Methods

To understand the value and practice of design systems contributors and answer RQ2, we conducted an interview study with nine practitioners from nine different companies.

4.1.1 Participants

We aimed to recruit practitioners who are highly experienced with design systems creation. In order to identify the qualified participants, we first collected the most active contributors in each of the design system projects we analyzed in chapter 3 based on their commit history. Among these contributors, 674 provided a public email address on their GitHub profile. We directly contacted them using these email addresses. In the recruitment email, we attached an informed consent form explaining the interview procedure and how the collected information will be used as well as the followed measures to protect the participants' personal information. Nine participated in our interview study. Most of our participants occupied high-level design-related positions in large companies (e.g Facebook, Shopify, Dropbox, etc) that own a design system that is used in one product or in multiple products across different plate-forms such as Shopify polaris [1] or single plate-form like origami design system [2] that is used for web development only. Table 4.1 summarizes the characteristics of our participants.

4.1.2 Interview procedure

The interviews were conducted and recorded using Zoom[3] and were semi-structural, supported by an interview guide. Each interview took about 40 minutes to complete. During the interview sessions, we focused on the participants' knowledge and experience in developing

[1] https://polaris.shopify.com/
[2] https://origami.ft.com/
[3] https://zoom.us/

Participant	Location	Experience with Design systems (Years)	Job Title
P1	UK	7	developer/business owner
P2	US	5	Design lead
P3	Russia	3	Senior product designer
P4	Spain	3	Product design manager
P5	UK	6	Head of engineering
P6	Canada	4	UX Development Manager
P7	Spain	2	Software engineer
P8	Spain	4	UX design lead
P9	Canada	4	Design system manager

Table 4.1 Interview participants information

or contributing to design systems. Particularly, we started by asking about their definition of a design system considering the fact that the concept is relatively new and unclear. We followed by questions about the benefits of design systems in order to explain their increasing popularity. We then asked the challenges the participants experienced during the development, maintenance, and usage of design systems; we also asked about the best practices our practitioners follow to mitigate the challenges they face. Finally, we concluded our interviews by asking different aspects of open-sourcing design systems including motivation, benefits, and drawbacks. The complete interview guide is included in Appendix C.

4.1.3 Analysis

To analyze our interview data, we fully transcribed the recordings and performed an inductive qualitative analysis using the following steps. First, we categorized the common topics our participants discussed in the following categories:

1. **Concepts** : We asked our participants about their definition of a design system. This theme groups all the definitions our participants provided.

2. **Benefits**: Our participants mentioned several benefits and advantages of design systems for building user interfaces for products inside their companies.

3. **Challenges**: This theme categorizes the challenges that come with design system development, maintenance, or usage.

4. **Key to success**: Our participants recommended numerous strategies and best practices they identified during their experience to mitigate the challenges they face. This theme groups these practices.

5. **Open sourcing**: All our participants have worked at-least on one open-source design system, they reported the benefits, drawbacks, and motivations behind open sourcing design systems according to their individual experience.

Once the inductive coding process is concluded, we created a codebook for describing the themes identified in the categories. Then another researcher (i.e. the blind coder) is involved to use the codebook to code the interview transcripts. We used Cohen's kappa [49] to evaluate the inter-rater reliability between the codebook creator and the blind coder for the codes generated. Among all codes included in the codebook, the average kappa is 0.88%, indicating almost perfect agreements between the coders about the identified themes.

4.2 Results

In this section, we report the results of our interview analysis following the themes (indicated in bold fonts below) we classified during the coding process.

1. **Concepts** :

   Four participants described the approach as a method for **scaling product design**. In their opinions, They provide all the necessary tools and resources for the design work. It can be described as an ecosystem where all the design development activities are carried out. As P2 described, it is an umbrella that covers all the required element to manage the design aspect of digital product development at scale: *"I tend to think about design systems as the term is an umbrella, term for resources, tools, groups of support that facilitate product development really. It's more about consistent and scalable digital product development than necessarily about design."*. P5 also thinks the same, for him, the design system includes every element that is related to the product design: *"a system that allows you to use design at scale and design components at scale. A design system includes more than just a simple component library or a color palette. It would include the brand guidelines. It also includes fully interactive and shareable components that you can reuse across applications and the web, and whatever*

other pieces of technology you choose to use there."

Meanwhile, three participants considered the design system as **a set of guidelines and rules** for creating UI components. In their minds, a design system is a highly efficient technique to standardize the product design development process. Multiple teams can all work together following specific patterns and rules; the design system promotes the common practices in designing and developing UI components that can be defined according to the needs of the company. These practices result in the consistency and coherence of UI components across all the products the company owns, which facilitates the integration of the brand and reinforcement of the identity in the products. For example, P4 talked about how a design system groups practices, fundamentals, and limitations while creating UI: *"... the design system that is another layer that is on top of the implementation, that could be any language and so on, it tells you like, "These are the best practices. These are the limitations that we want you to know. These are the rules. You should use it in this way, otherwise, this would create an extension point to your own case, please follow the fundamentals that we have." Everything is inside the same language. It looks consistent, and it doesn't create much trouble for our users."*

Another three participants provided a more simplified definition. They considered it as **a tool for crafting** digital products. Essentially, a design system provides the required components and artifacts for developers to build and implement design ideas in digital products. For them, its the most complete tool to perform their development tasks; it widely eases their work and increases their performance. For example, P8 highlighted this point and considered the design system as a toolkit that has all the necessary elements for the developer and designer to build their product: *"it's the system that includes everything that the designers and the developers need to craft their products "*. P7 also described the design systems as a set of components and libraries to built products: *"we describe our design system as a set of libraries. A set of components that help us build our product more easily and that help us not having to focus on design and just having to focus on structure. Is more like a helper tool for us."*

A tool for unifying design is another key concept our participants included in their definitions, mentioned by three participants. Apparently, design systems are great tools for communication between different parties that are involved in the creation of a digital product. Design systems act as a common language they all understand, as described by

P4: *"basically design system is a language, that allows your products to talk in the same way"*. To illustrate, having a design system means that developers work with the same UI components in all the company's products. This creates a global understanding of the UI design in the company. To put it another way, companies that own and maintain multiple products without using a design system develop UI components independently for each product and as a consequence, making it challenging to track and manage the design activity in the company. Additionally, design systems communicate the UI functionalities to users, using the same components across multiple products helps users to form mental models of the UI components use. This was highlighted by P1: *"I would say a design system is something that allows a consumer to get the information without having to understand the design. That's the basic principle of it. We want to show that all of these things are of the same type and the same kind and then once you've learned one of them, you understand how all of them work. For example, if we have buttons that are inconsistently colored then a user goes and uses a feature of one of our websites and presses the blue button, and then they start to have a mental model of what blue buttons do..."*.

Figure 4.1 summarizes the concept theme that was provided by our participants.

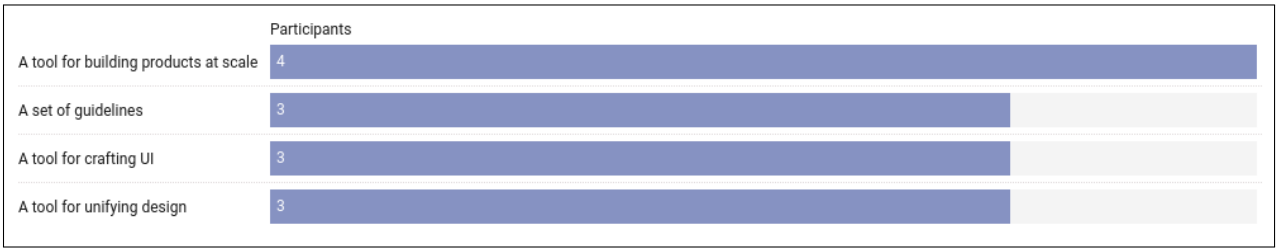


Figure 4.1 Design system definition concepts

2. **Benefits**:

   Our participants have discussed various benefits that design systems offer to their companies (e.g see Figure 4.2). First, five participants commented on how the use of a design system results in **efficiency and development speed**, when you have a consistent design and implementation of UI components, the process of creating new products and applications becomes easier and more efficient, both developers and designers have a clear understanding of the UI development process, and they both follow pre-defined patterns to accomplish their task which improves their overall productivity. Five participants discussed this aspect of design systems. For example, P2 mentioned how design systems allow faster development: *"but the main benefit is actually speeding of development. You can very quickly go from a nascent idea to a full-fledged prototype or product using a designed system in a way that would be much more difficult without one."*. P4 also highlighted this aspect stating how the centralized approach of design system reduces development effort and time: P4 *"there is a reduction in the effort since everything is centralized... There is a point of reference that everyone can follow. It reduces development time ..."*.

   Secondly, design systems help **reduce costs** of product development. This is a natural result of other benefits we identified. The consistency and re-usability lead to more efficiency in the design process. As a result, the company will not have to create UI for each product – the design system can be used across all products, which means the cost of UI development and design is reduced. For example, P3 mentioned that: *"It's a great economy when we develop, when we make design and when we develop a product so we don't have to reinvent the wheel every time."* P6 confirmed this by stating: *"Cost-cutting is huge. An example could be if you have 60 scrum teams who are building the same thing over and over again because they need a drop-down or whatever"*.

   Consistency or coherence is another main benefit of design systems according to three participants. According to them, using a design system ensures that all the components have a consistent design and implementation across multiple products. The UI design and prototyping follow a defined and clear pattern that is centered around the company's brand, which results in consistent products with similarly looking and behaving components. For example, P5 mentioned: *"I think the biggest benefits from a design system are reusability, and consistency setting. I think it's also very dependent on the company that you're working at and the benefits that those give. For us, the*

biggest benefits were that shared consistency and understanding of how to best build the components.".

Well documented design systems also **facilities and optimize the on-boarding process**. New hired developers or designers can easily understand how UI design activities are carried out inside the company. For example, P3 emphasized on this as an advantage of design systems: *"we have nice documentation so when the team changes and it happens every time, newcomers can easily join the work and understand how it works."* P8 also mentioned: *"You could say one of the main benefits is speed or money or onboarding for new designers or new developers, etc."*

Finally, design systems **improve communication and collaboration** inside IT organizations. They provide a common language that is spoken and understood by different entities involved in the project development, which results in better collaboration inside the organization. For example, P8 mentioned how design systems influence positively communication/collaboration between developers and designers; the designer can communicate their ideas more efficiently when discussing UI with developers because all the UI components are predefined and well recognized by both parties: *" If it's easier for a designer to communicate something to a developer, and that's just a pen and a paper saying, I need this button and this is a primary button and both are thinking the same thing, obviously, like, okay, I know you're referring to this UI kit and you are referring to this UI component, but we're talking about the same thing."*

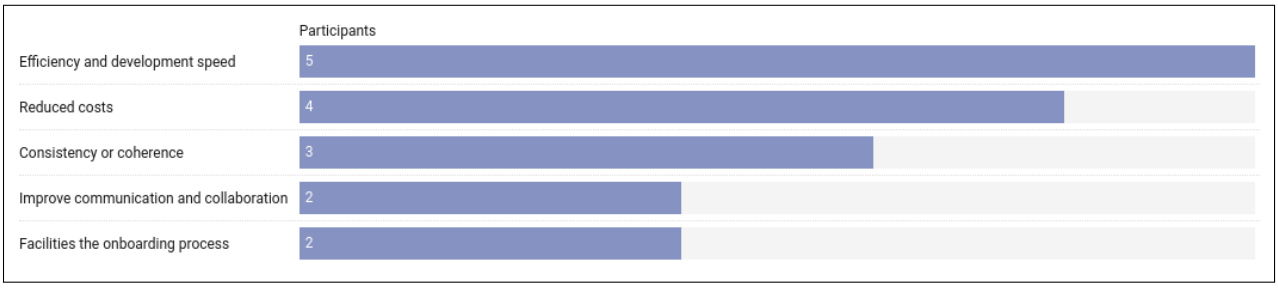


Figure 4.2 Design system benefits

3. **Challenges**:

   Despite the benefits, the design systems' centralized approach to create and maintain UI design in IT organizations poses some challenges according to our participants (e.g refer to Figure 4.3).

   Our participants had different experiences with design systems, but the thing that the majority (six participants) agreed about is **the difficulty of evolving the design system**. Maintaining consistency is a tricky process that proves more challenging than the creation of a design system. The fact that the design is centralized and that different teams are expected to use the design system results in serious conflicts about which components need to be added. Ideally, a dedicated team is responsible for making this kind of decision. This team must understand the needs of the users to make sure the design system is relevant, up to date, and getting adopted. P6 clarified (e.g *"developing it, it's not too complicated. It's getting adoption, maintaining it, making sure it's still relevant even after a few years, making sure it stays somewhat nimble if you can still update it easily and change management. How do you put it? Making sure it's still changeable, and it doesn't become crystallized..."*). P7 also expressed this challenge: *"the biggest issue that we have is having to decide what goes into design system and what is not worthy to have. But it's very subjective...*. The evolution of design systems demands the involvement of all the parties that use it and that the approach is well understood and accepted inside the company. If not, each team will just build what suits their need, as P2 states: *"That's an interesting balance for us to strike because sometimes those teams will come to us and they need a new component or patent that we don't see that need in the products that we're supposed to be serving. So, we have to have this debate about how realistic is it for us to work on that thing, or can we say, "No." and what will that mean in a year's time, if that team decides to build something themselves. That's always a little challenging."*.

   We mentioned in the benefits section how design systems can improve communication and collaboration when used in product development. However, this aspect is also considered a challenge during the development and maintenance of the design systems. **Achieving consistency requires quality communication and collaboration** between all the actors in order to fully implement the concepts and the philosophy of design systems. For example, developers and designers need to work hand in hand with a common goal and understanding of what they want to create, P3 mentioned the

fundamental role this plays in creating design systems: *"if you don't have this mindset at both designers and developers, you don't have a clear way to communicate. If the designer is sitting somewhere remotely and he's not in the team with his developer, he turns his design and then developer takes it and they don't talk before, there's a huge chance that you can't create a design system like this"*. Also, designers can introduce ambitious ideas without considering the technical aspect which results in conflict with developers, as mentioned by P9: *"Before we had a team, some designers are spending a lot of time designing components. That sounded really nice, that's very useful, but sometimes the scope was so big that no developer wanted to touch it. They say I had my day job to do, there's no reasonable way I can make all this happen!"* Overall, communication is obviously a must in the entire process that involves teams with objectives that meet at some level; this applied perfectly to design systems.

Getting the design system adopted and used by the development teams can be a challenge due to the fact that, it leads to a major change in the way people used to do the design work in the company, before its introduction as the main design tool, developers can create any UI component according to their needs but the focus on consistency with design system takes away that flexibility and freedom. It introduces various limitations on which component can be included and how it will look and behave. This requires increasing communication efforts to attain the fixed goals for following this approach, as described by P4: *"I used to get some complaints because people used to think that the design system was limiting them somehow, but it took an effort a huge or big effort on communication to tell people that, "Hey, you can do whatever you want, here is the design system." The design system offers you some rules, some help to build upon from that, nothing limits you."*. Overall, the human factor cannot be negligible when talking about the creation and use of design systems, as mentioned by P8: *"I think (the biggest challenge) is the human aspect of the design system. It's trying to convince someone that, "Hey, I told you this three times." If something goes in the design system, it's because it's reusable and it will be good for someone from other vertical and this really makes sense to be there. If not, it's just a test. All the other things that come with human, like, hey, you didn't attend the weekly and we spoke about that feature in that weekly and now three weeks, you're trying to complain because of this thing that we spoke back then, remember?"*.

4. **Key to success**: Our participants recommended several practices in order to facilitate and optimize the creation, maintenance, and use of design systems to overcome the

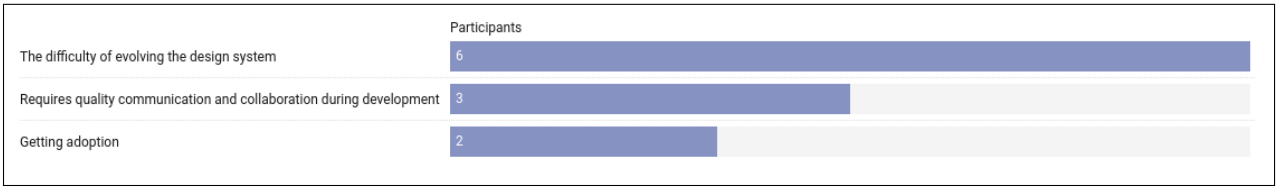


Figure 4.3 Design system challenges

challenges. Figure 4.4 summarizes their recommendations .

The best method to build a design system according to P4 is to start from the UI components that you already have and build up your design system following a **bottom-up process**: (e.g *"We made a base layer with that, and then people can build on top of that, as many things as they want."*). P5 agrees about this and he described how in his company, started by a simple style guide that evolved and expanded into multiple design systems (e.g *" I had built design systems before previously in an agency that I worked at. We had started creating shared styles for our Rails framework, which then expanded into a design system. I think I will go simpler than that and say we were building a very simple straightforward component library or a style guide to begin with. That started off six years ago to try and simplify a lot of the engineering processes that we had, and the disparate designs that we had across multiple of the team. We kicked off with our very simple straightforward style guide. This then expanded into a design system, and now has expanded into multiple other design systems."*).

Having a **dedicated team** that is responsible for developing and maintaining the design system is essential according to two participants. Having an owner of the design system will maintain the consistency of the design. The dedicated team can also guide the use of the design system by different teams in the company and advocate for the design system approach to better integrate it into the company's culture. P5 described this in his statement: *"I think that you have to have, I would say, ambassadors of a design system within a company to actually get that to be continuously used, and you have to set in place processes that allow teams that are not always working on the design system to feedback into the design system, and then also push their changes into that system because if you don't get that, then you still get divergence."*

Three participants emphasized the importance of good **communication** among all the parties that interact and involved in the design system during the development and maintenance. Essentially, it is a continuous process that requires feedback and discussion to justify the decision being made, as P2 describes: *"We have to communicate those changes and think about the implications and how we deprecate the old thing in favor of the new thing, how we continue to support the old thing for systems where that did make sense."* P9 shares the same idea by mentioning the importance of including the stakeholders in conversations about the design system: *"talking to people in your organization including the right stakeholders in every conversation and not just doing it because it's cool and other people do it."*

Finally, **documentation** was also mentioned by three participants as an essential piece for any design system project. They considered good documentation as a factor that increases development speed, resolves communication challenges, and facilitates the use of design systems. For example, P4 considered the documentation as a very important element and he gave the example of Microsoft's design system and how documentation helps better promote their design system: *"we don't have the same number of users material design has, but somehow we need to serve them the same way. we don't have the number of users the same as Mircosoft or Fiori or others. but if they didn't have the documentation they would have failed in the first year probably.so Documenting is really important,"*, P9 also discussed this aspect and mentioned that better documentation would improve developer experience: *"You have to spend so much time on documentation. If the documentation goes all the way to the developer experience, you can obviously be super helpful".*

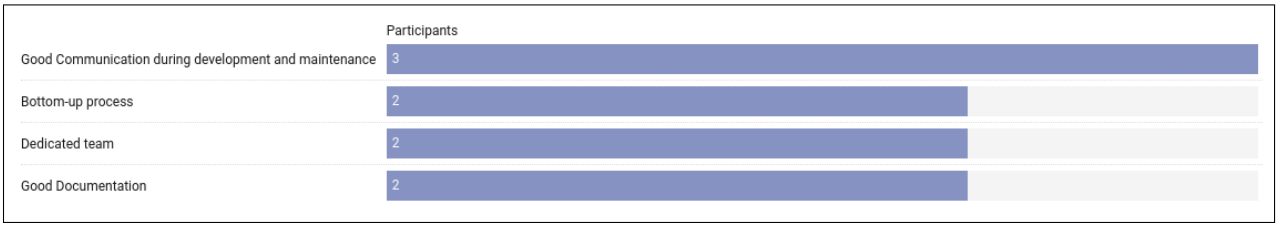


Figure 4.4 Design system's creation best practices

5. **Open-sourcing**

   Based on our preliminary study, open-sourcing design systems are common among IT companies. During our interviews, we focused on this to better understand the benefits and the challenges involved in this process (e.g see Figure 4.5 and Figure 4.6).

   Hiring is the main advantage of making a design system open source. Open source projects are a great tool for hiring as explained by six participants. For companies, active contributors to open-source design systems are great candidates since they are already familiar with the design system and also with the company's culture due to their interaction with the designers, the developers, and the project owners. For example, P7 mentioned that their company has actually hired people that contributed to their design system: *"We have hired people due to the contributions that they have made to the design system."*. For developers and designers, contributing to open source projects also allows them to showcase their work, thus gain more credibility while searching for new job opportunities, as P9 mentioned: *"the developers on the team and the designers on the team would be able to point to this open-source work and say, I did that, and that's an achievement. If they're looking for jobs in the future it gives them some credibility."*.

   Four participants talked about how open-sourcing helps improve **the quality of the design system code**. According to these participants, knowing that the code is public, developers pay more attention to details while coding and become more motivated to address software design and programming comprehension issues in the code. This tendency positively impacts the code quality and consequently the project quality in

general. For example, P5 mentioned: *"the reason that having it in the open can be useful is that it enforces a more correct way of doing things."* P9 also thinks the same way: *"There's the quality gate if you will. When you open-source work like this, it gives you very high motivation to make sure that the code you write is very high quality."*.

Additionally, according to four participants, open-sourcing a design system allows it to **gain publicity** and reach a larger audience. For example, P3 mentioned: *"The nice part that you get a lot of public relations so you can come to the conference and say that, "Hello, that was the motivation for us, because we developed it in our company and we used it a lot..."*".

Finally, having **an open-source community** contributing to the evolution of the design system in various aspects is a motivation mentioned by two participants. It is very common in the open-source world that the community can help test, detect, and fix bugs in the design system, as P3 experienced: *"We're shown that we have been here, we have very clean and easy-to-use components, ... Suddenly, people started to use it. Sometimes, yes, when they saw a bug. That was easy for them to contribute. That's a big advantage for people who use it, and for open-source, of course. Even developers of other packages that we use came to us and said, "Okay, so you have an old version," or, "You use it the wrong way." We're like, "Okay, thank you." "*)

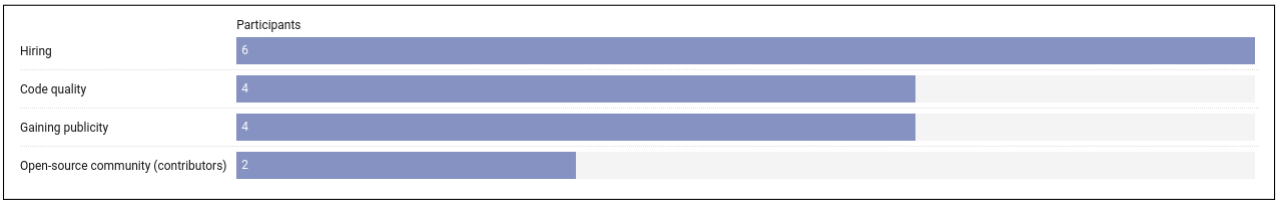


Figure 4.5 Design system open-sourcing benefits

On the other hand, our participants revealed three main drawbacks of open-source design systems, starting with **management**. Having a public repository necessitates management resources. And sometimes the developers can be overwhelmed with the extra work, especially when you have multiple repositories. This issue is highlighted by P1: *"That means that you immediately have a massive administration challenge because you have all these repositories and nobody knows what you have. Nobody's going to go and browse GitHub and find the one out of the 400 repositories that contain the*

component they need." Also, hosting the repositories on GitHub necessitates guidelines and a form of code of conduct for the community to follow when contributing or using the design system. For example, P6 mentioned: *"there's complete open development, where everything's open, you can fork it, you can reuse it, you can add features to it. If you do that, then it's a lot to take in because you have to have community guidelines, and code of conduct, you have to think about the contribution model, not only internally, but externally. There's just a lot of thoughts to put to it if you want to do it well. That's one of the disadvantages. It's not something that's, like, willy-nilly, just push a button on GitHub or GitLab, and make it public, because there are legal implications to it and all that. Choosing the correct license, etc. It's a lot to think about."*

Privacy and security concerns are another drawback according to two participants, due to the risk for sensitive information leakage. For example, P1 mentioned: *"There are security risks associated with revealing the nature of your internal architecture. Your design system will obviously mirror to some extent the nature of your organization. You might have a bunch of styles that have names that actually are products that you haven't released yet."* P7 also commented: *"We need to be careful with that and not to use the same system until the feature is let's say public because we can leak information. ".*). This drawback can be addressed by implementing measures to protect the company from security breaches. Like limiting the use of GitHub issue tracking system as also mentioned by P7: *(e.g "In the past, we were a team that used, for example, a lot of GitHub issues and GitHub projects to manage the project. Now that we have more responsibilities that include private things, we are not using it that much anymore. ").*

Finally, Open sourcing the design system means that anyone can use it and modify it as they like. Once it becomes open source, the company will be **losing control** over it, as mentioned by two participants. An interesting risk is that the concept of ownership can shift from the company to the open-source community, making the focus of the design system potentially drift, as P4 described: *"I think another one of open source specifically is when you open-source software, there is often an expectation that it then belongs to a community outside of your own.".*

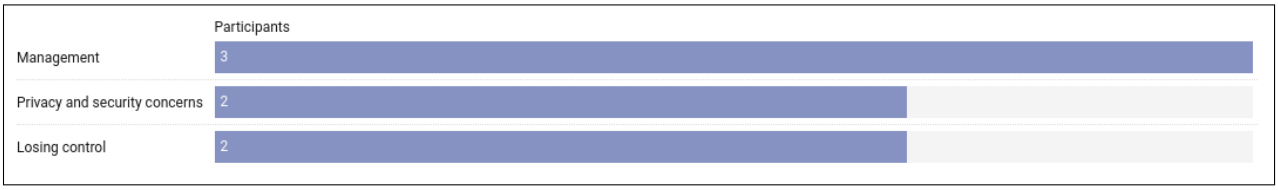


Figure 4.6 Design system open-sourcing challenges

4.3 Discussion

Our interview results helped us get more details and answer the questions we had from the issue analysis study. Considering the concept of the design system, our participants talked about four main factors, some see design systems as tools for building products at scale, while others see them as sets of guidelines and rules for creating UI, or tools for crafting. An interesting definition from our participants is that design systems are a common design language that unifies the UI design. All these definitions reflect an important element of design systems from the practitioners' point of view. Combining these elements, we can provide a living definition of design systems, as: **design systems are a set of tools, guidelines, and rules that are used for designing and creating digital products design at scale while maintaining consistency and coherence**.

Design systems' advantages over traditional methods for creating UI vary according to each participant's needs and his usage of this approach. However, we noticed that all the identified benefits are connected, as illustrated in Figure 4.7. Design system provides consistency and coherence in UI design which leads to improve communication and collaboration inside the company because everyone is working with the same approach and UI. These two advantages result in facilitating the on-boarding process, and all of these elements lead to more efficient development process, which finally reduces costs of developing UI and maintaining it.

On the other hand, three challenges were identified, first is the difficulty to evolve the design system due to the tension to maintain consistency and remain flexible; this challenge explains the huge number of requests of new elements in the issues we analyzed. The remaining challenges consist of communication challenges during the creation of the design system (also because of the aim to achieve consistency) and problems in getting it adopted inside the company. To overcome these challenges, good communication and documentation are

necessary. The latter is often finalized inside the organization; P2 for example mentioned how they had a wiki in his company where they put all the documentation for their projects and that he was the sole maintainer of the documentation website of their open-source design system which supports two of our hypothesis (2 and 3) for explaining the low frequency of discussions about documentation in the issues we analyzed in Chapter 3.

Finally, for the motivation why design system owner open-sourced these design solutions, our precipitants mentioned four advantages including code quality, hiring benefits, publicity, and community engagement. Morgan et al reported similar results from an exploratory study with IS/IT managers in 13 companies operating in the secondary software sector in Europe, the majority of their interviewees expressed how peer reviews and the quality of developers and testers base enhances the overall quality of the open source project [50]. Additionally, seeking better jobs and advancing professional career is one of the main motivations of open source contributors as reported by Lakhani and Wolf in their survey study of 684 software developers in 287 F/OSS projects [51], these software developers invest hours of their free time in contributing to the OSS project which makes them the ideal hiring candidate targets.

The benefits of OSS design systems explain the actual fierce competition among giant IT companies on promoting their design systems. However, open-sourcing can bring some challenges that necessitate allocating more resources in management and addressing privacy and security concerns. Although the design system is only UI components, the risk of exposing the internal architecture of other similar products is always present, open design system developers must be extra cautious not revealing sensitive information to the public.

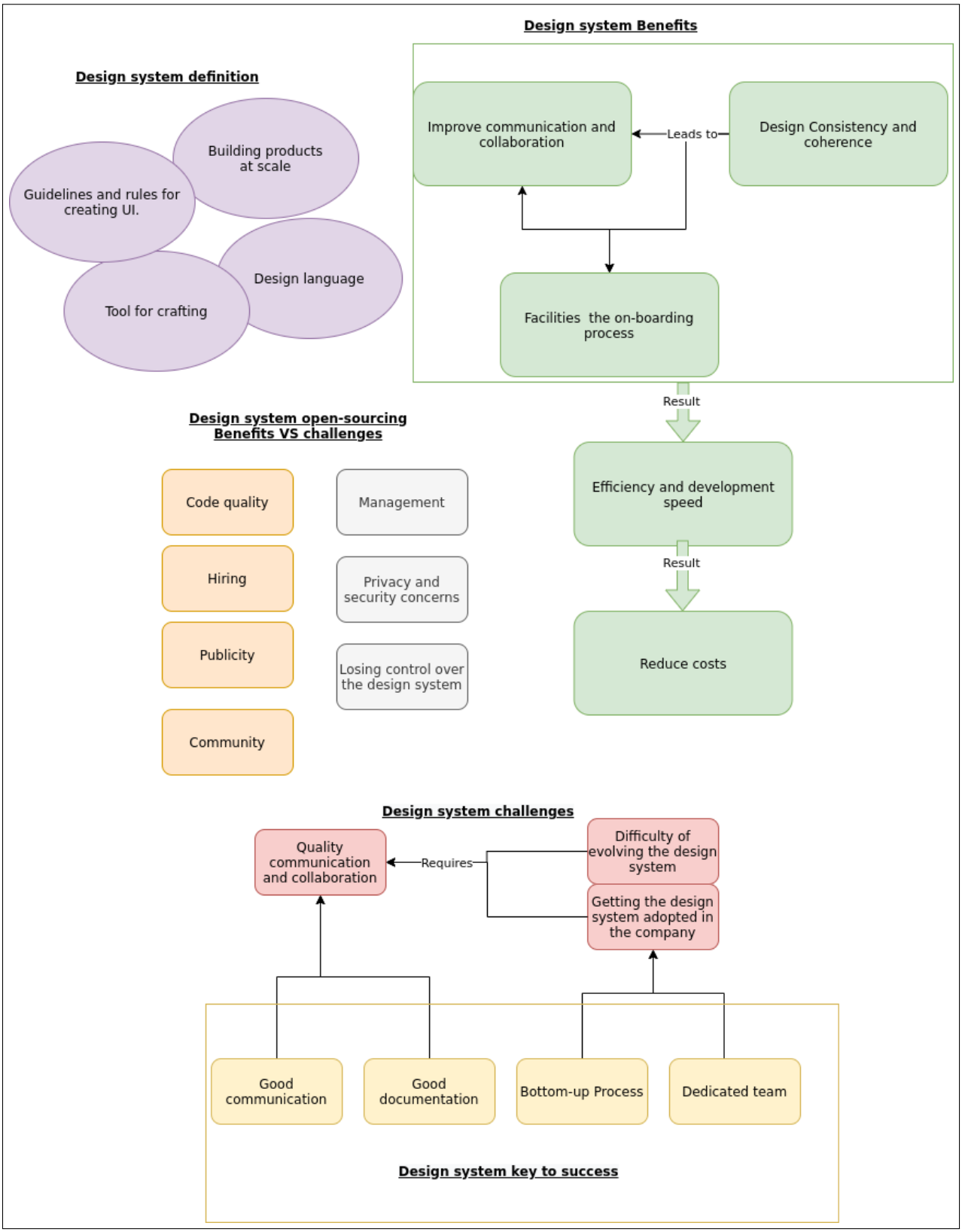


Figure 4.7 Interview study results summary

CHAPTER 5 DESIGN AND EVALUATION OF A TOOL PROTOTYPE FOR CREATING CUSTOM DESIGN SYSTEMS

To address the challenges design system practitioners face during the creation of design systems that we identified in the interview study, we designed a web-based application that generates custom design system in a few simple steps. In this section, we present the design guidelines we followed when creating this tool and an evaluation study of our tool by two groups of practitioners: (1) experienced practitioners that have worked on design systems and (2) novice users who do not have significant experience with design systems. The choice of these two groups of participants is to explore the usefulness and effectiveness of our application by users with different backgrounds and experiences about design systems. We aim in this study to further determine the features practitioners value in a tool design for efficient creation and customization of design systems (i.e RQ3).

5.1 Design principles and rationale

We learned from our previous studies that design systems are created to centralize the front-end development in software companies. Our observation and analysis of the OSS design systems issues revealed that design systems are best created when following a bottom-up and iterative process. Particularly, the bottom-up approach that extracts design system components from the existing products and applications of the company was identified by our interview participants as more effective than a top-down approach to create the design system from scratch. The iterative nature of design system creation was also highlighted by the number of requests to add new components in the issues we analyzed and was confirmed by our interview participants when describing the challenge of evolving and maintaining the design system and the importance of having a dedicated team. In addition, the participants emphasized the coordination and communication between different stakeholders in the company to create and maintain a design system.

All of this shows that design systems can be expensive and resource-consuming to create, particularly for small companies or organizations. An easier and effective approach to creating a design system is to tweak and customize existing open-source design systems based on the current products and applications of the company. This possibility was also mentioned by several interview participants; for example, P7 said: *(i.e "At least follow it in some way that brings consistency, but it doesn't break the product itself or the way interacting that we have the time or it doesn't break the design principles that we have built or the universal principles*

that we have brought into the system. Basically, in our design system, we have brought all the same principles that already exist in the universal principles."). Therefore, we're proposing a solution that consists of a web-based application that allows us to customize a design system in an efficient way. We aim to make our solution beneficial to both expert design system creators and novice practitioners while supporting both designers and developers. Particularly, we focus on the following principles when designing the tool.

Flexibility. We aim to design functionalities that will make the customization and the use of the design system a faster process. The users of the tool should be able to load any website and extract the component's styles of choice. Synthesizing these styles should also be flexible, allowing the application of styles to multiple and varied components. To support both designers and developers, users of the tool should be able to customize the design system either with a graphic user interface or using code. We believe that allowing designers and developers to use the same tool will help close the communication gap between these two software practitioners.

Supporting fast prototyping Our application will include a feature to extract styles from existing websites, and the ability to apply them to the UI components interactively, meaning that the component will be updated in real-time when applying a new style. This feature will be very helpful for designers to test whether their designs are technically feasible, at the same time it will help them improve their technical skills in CSS. In addition, fast prototyping will come in handy during the iteration process of reviewing components by designers and developers, these two practitioners will be able to carry out their reviewing and discussion activities while looking at implemented components instead of just designs.

Supporting design innovation. Likewise, developers can benefit from the features we offer without feeling that they are limited, we added the ability to either modify the style of the components using an options menu or by writing CSS code in an embedded text editor.

Familiarity and consistency. In order for our tool to appeal to our target user categories, our interface design and workflow are also inspired by popular tools that designers and developers are familiar with. Figma [1] is a web-based vector graphics editor and prototyping tool that is widely used by designers. Although our tool is very different from Figma in terms of functionally, the workflow is, to some extent similar. Another tool we inspired by is bootstrap studio [2], a desktop application for building UI components using the bootstrap framework [3], this powerful desktop app is commonly used by front-end developers to create

[1] https://www.figma.com/
[2] https://bootstrapstudio.io/
[3] https://getbootstrap.com/

websites; it provides an interactive display of created components with options to modify their code and styles. Figure 5.1 and Figure 5.2 presents the similar workflows of both applications, Figma and BootstrapStudio.

workspace.png workspace.png workspace.png workspace.png

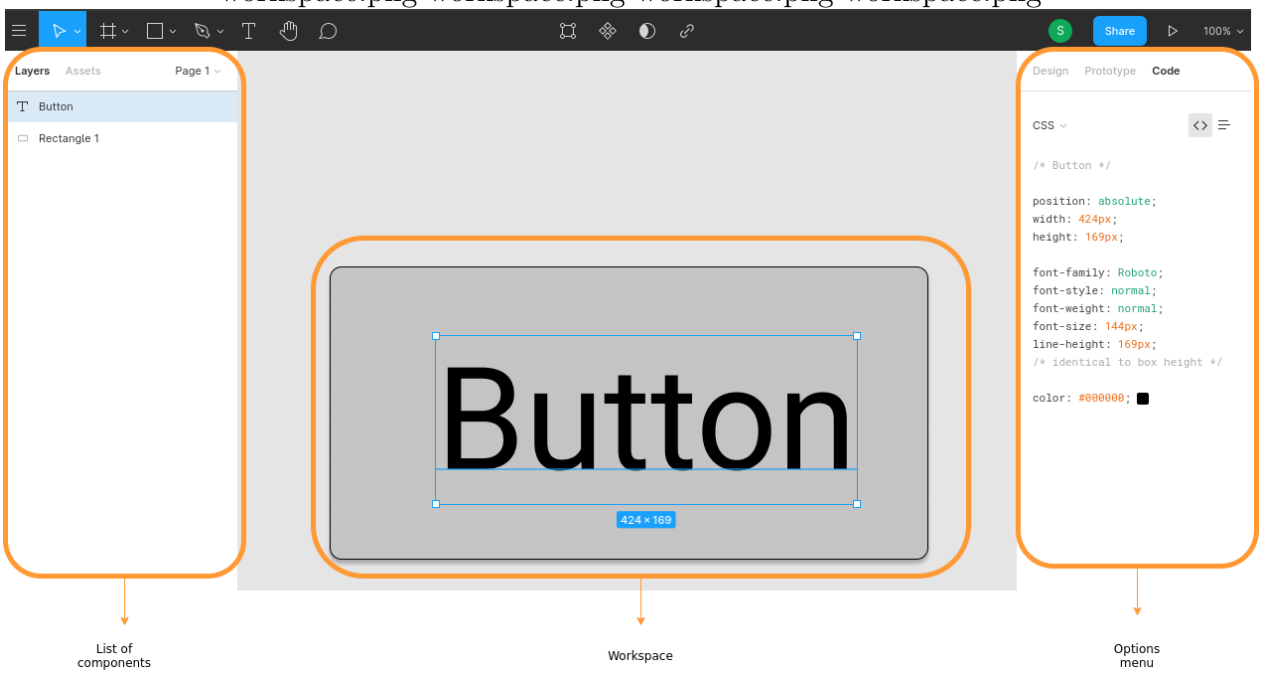


Figure 5.1 Bootstrap studio's user interface

workplace.png workplace.png workplace.png workplace.png

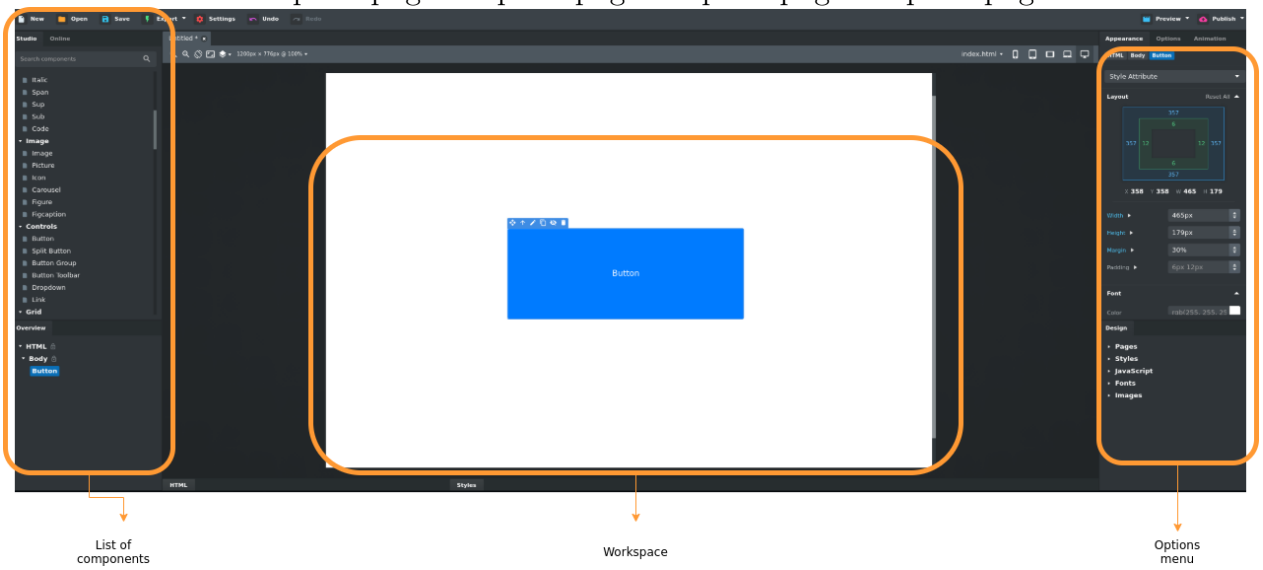


Figure 5.2 Figma's user interface

Simplicity. Our feature for extracting styles from websites is similar to the inspect tool that is present in all current browsers, however, we aim to simplify and implement a more efficient and user-friendly solution that extracts the styles that related to the component's visual design only and not its position and its relation with other components in the website.

Resource efficiency. Our application will come with an integrated open-source design system with components that are frequently used in UI design (Content display, Navigation, Input components and Information display components), which limits the work of designers and developers to customization only. This will help reduce the resources needed to develop and maintain the design system.

5.2 Prototypes and formative studies

We started the development of our prototype by creating wireframes and mock-ups (see Figure 5.3) that illustrate the core ideas of our system, the initial layout, and the features that will be included. We then shared it with two experienced practitioners who are experts in design systems from our interview study (i.e P1 and P8) in order to get their initial feedback about our proposed solution. We designed a formative study to get the initial feedback from these experienced individuals about our concept and to evaluate its potential. We conducted these two formative studies via Zoom; each took about 20 minutes. During each study, we started by explaining the concept of our tool and presenting the mock-ups to the participants. Then we asked our participants about their initial impressions of the tool, the potential usefulness of the tool inside their companies, the effects of the tool on creating design systems, the expected reaction from the larger design system community, as well as their feedbacks and recommendations for improving the tool design. The complete formative interview guide is included in Appendix D.

Overall, the feedback of our participants was positive. Both participants found the concept of the tool interesting. They thought that the tool is a good starting point for introducing the concept of design systems to individuals and companies who have no previous experience with this approach. Other design tools were mentioned by both participants. P8 suggested that we develop an interface that is intuitive and more similar to existing tools like Figma, specifically having a workspace to visualize the UI components built alongside with an options menu as a sidebar where the style attributes can be modified Figure 5.3. Additionally, we gained some perspective on the technical challenges that we might face during the implementation process. Manipulating the front-end code of websites can be problematic when security measures are present to prevent such practice, some websites block third-party applications from accessing their front-end code to eliminate any security risque.

Additionally, P1 pointed out the importance of having support for multiple UI libraries and frameworks (e.g., React, Angular, etc.) as the main feature. He suggested that this will allow our tool to be successful in promoting the use of the design system due to the fact that companies use different technologies to implement their front-end. We also got some insight about the community reaction to our tool, both participants think that the community of design systems will be interested in using our tool, especially developers that use open-source design systems in their personal projects.

The positive feedback we received from these two participants encouraged us to retain the core functionality and the design principles. We also iterated the interaction design of the tool based on the feedback given by the participants.

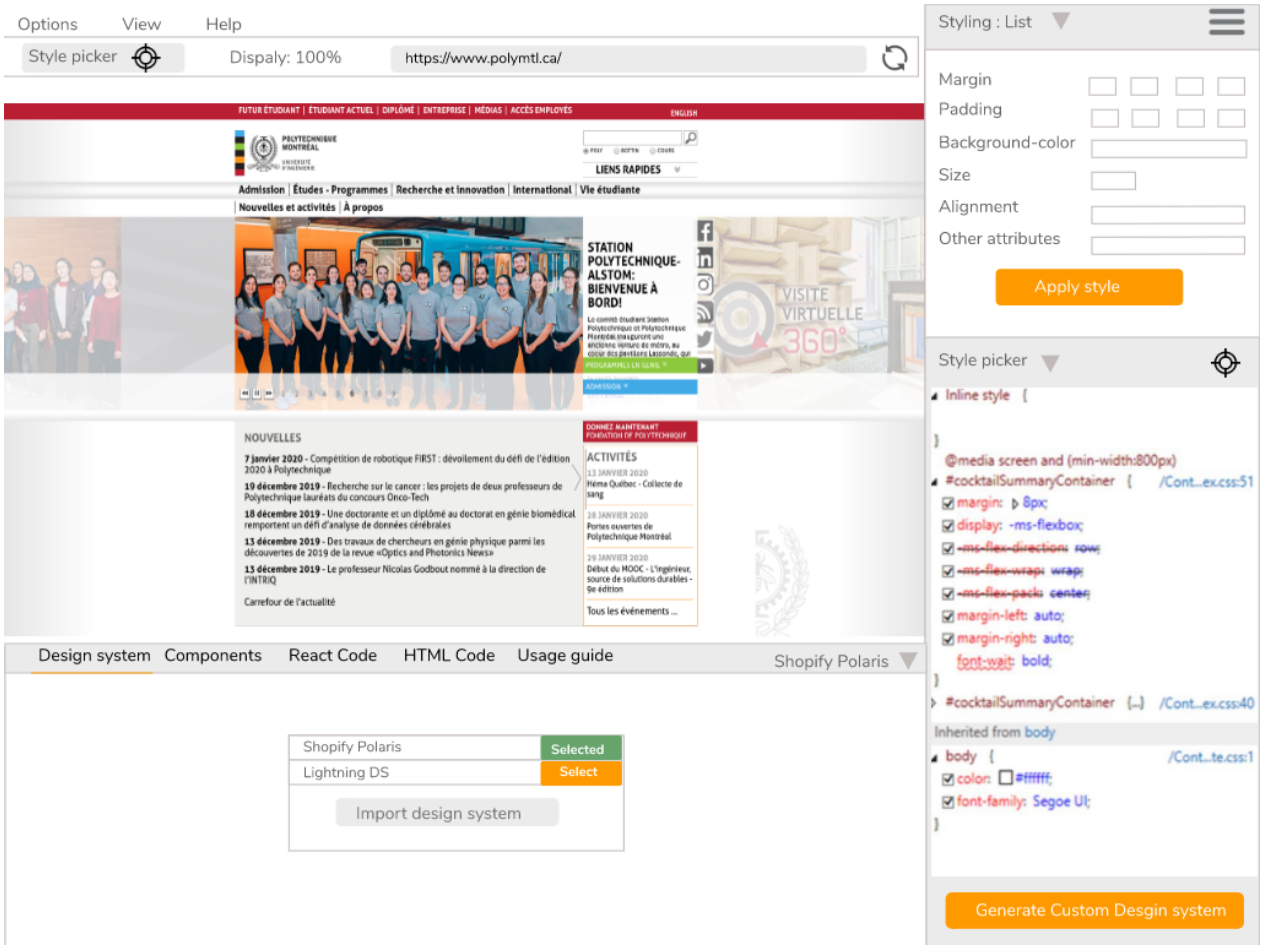


Figure 5.3 Initial tool mockup

5.3 Final prototype design

We iterated the tool design by incorporating the participants' initial feedback and reviewing the user interaction within our research team. We then developed a working prototype of the tool. In this section, we present the main features and the workflow of the final prototype we created as a first step to evaluate features design system practitioners' values in tools designed to promote and ease design system customization and use.

Features

In the following we describe each feature we implemented as illustrated in Figure 5.4.

- **A:** This panel displays a list of components that are included in the design system. Initial components can be imported from an existing design system or initiated from a generic template.
- **B:** This panel displays the HTML rendering of the selected component from the components list.
- **C:** This sidebar contains all the style attributes of the selected component of the design system. Users can change them manually by modifying the values of the attribute or by checking an extracted style from panel C. The information in panel **E** will be updated automatically when a new component is selected from the panel **D**.
- **D:** By activating this tab, the panel **C** will be changed to display a code editor that represents the entire CSS code of the selected component. The code is synchronized with the attributes sidebar when either of them was modified. This way, more flexibility will be given to practitioners who are familiar with CSS code, and more intuitive modification will be provided to those who do not.
- **E:** A button for applying styles on the selected component. Once the button is clicked, the styles represented in panel **F** are saved.
- **F:** A search bar that allows us to load websites of choice by typing the domain name. The website will be displayed in the panel **B**.
- **G:** A panel that displays the loaded website and allows the user to navigate thought its components, the components will be highlighted upon mouse hover. When the user clicks on a component, the component's styles will be extracted and displayed in the panel **C**.

- **H:** This section is for displaying the extracted styles. Each style will have a side checkbox; when clicked, the style attribute will be applied to the selected component in the panel **B**.
- **I:** A button for generating the design system when finishing customization. Once the button is clicked, an archive file containing the design system files will be download, the user then can import it into his project, and start using its components.

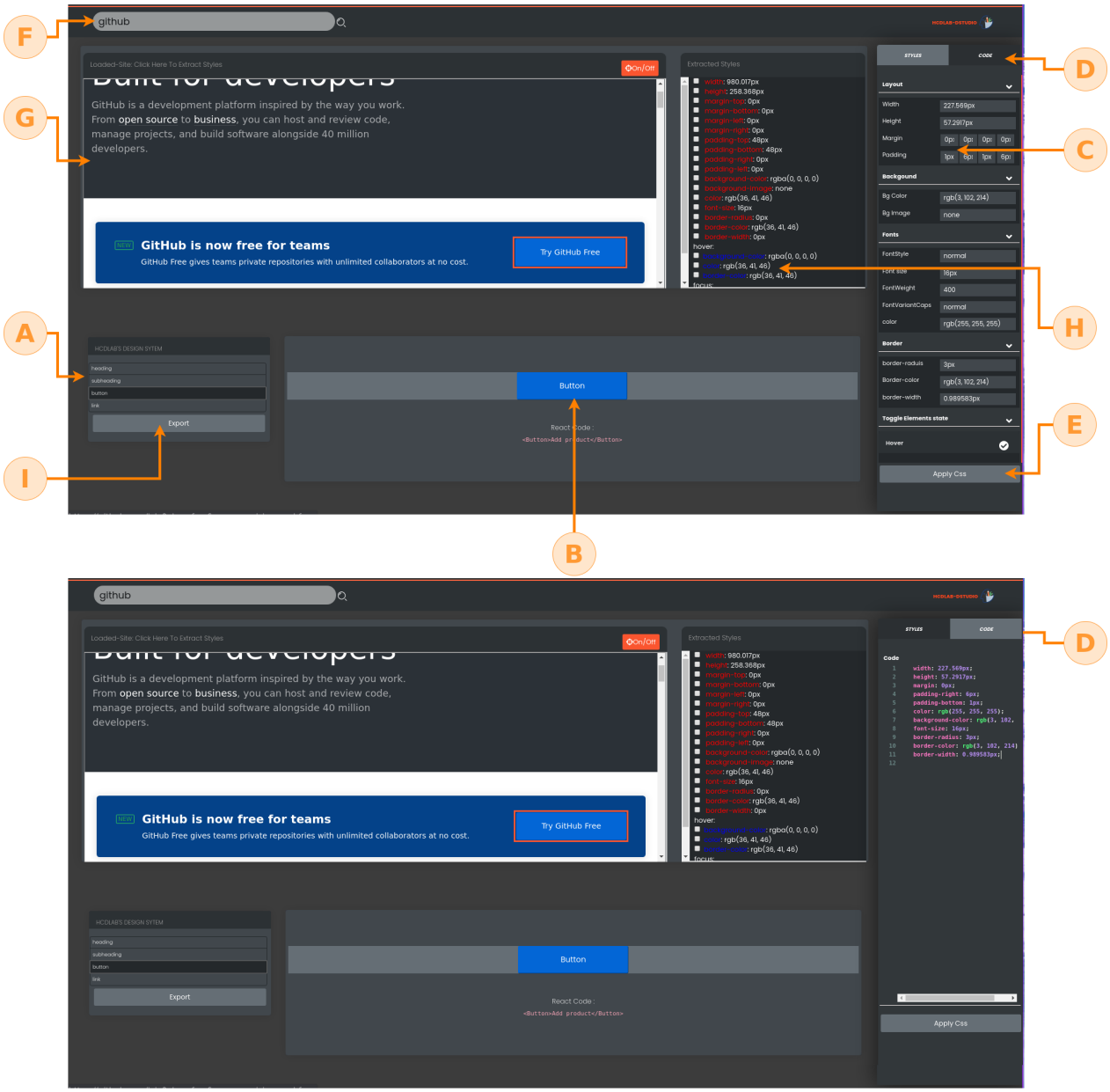


Figure 5.4 The interface of the developed prototype

Workflow

Figure 5.5 presents the flow diagram of the prototype. It describes the sequence of tasks using the different features to customize the design system, this sequence starts with selecting a specific component from the list of the design system components, then proceed to customize

it using the style extraction feature, the sidebar style options or by modifying the CSS code, then the user can save his changes to the components and repeat the process for other components. In the end, a design system can be generated and downloaded to be used in projects.

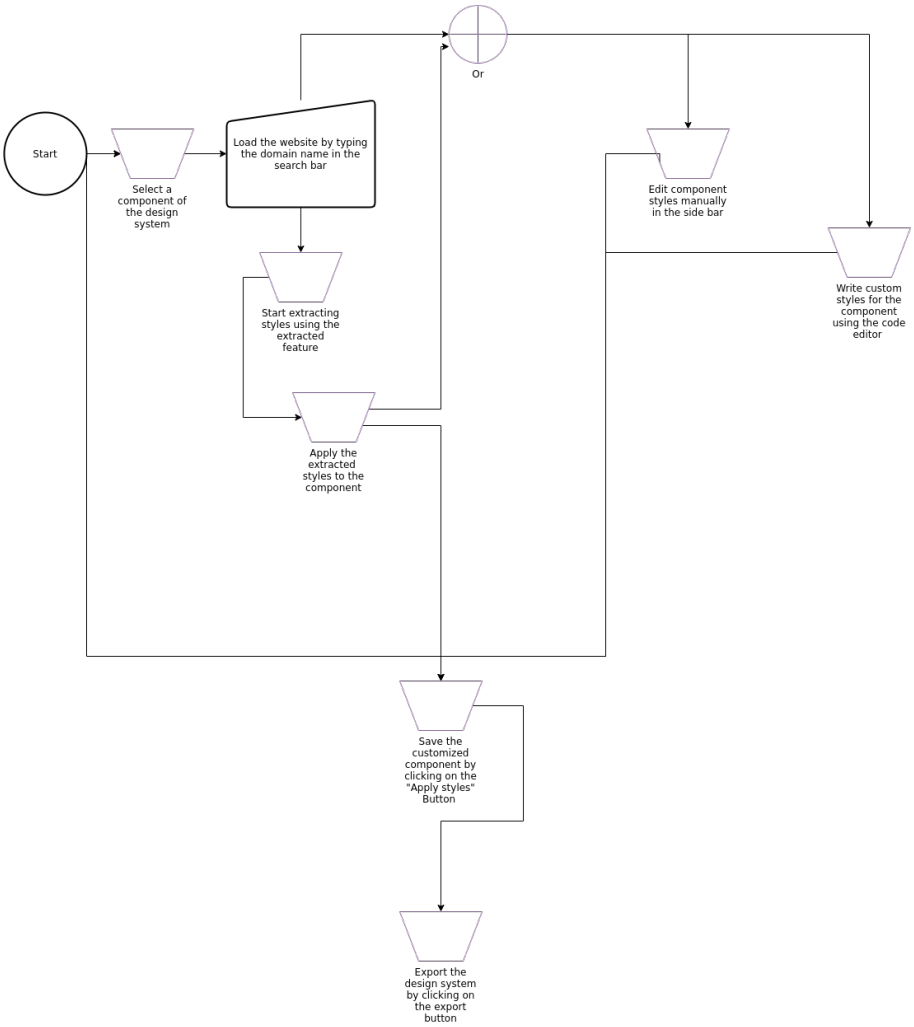


Figure 5.5 Interface flow diagram

5.4 User study methods

Once the prototype was developed, we conducted a user study to evaluate the effectiveness and user perception of the tool. Through this user study, our goal is also to better understand practitioners' perceptions of tools for creating and customizing design systems. In this section, we describe the user study methods we used.

5.4.1 Participants

We recruited two groups of participants [Figure 5.6]: (1) software practitioners with experiences using, developing or maintaining design systems who are affiliated with an organization that owns or uses a design system; and (2) software practitioners without experience in developing, maintaining, or using design systems, but are familiar with a front-end framework (e.g. React), CSS and HTML, or having basic knowledge about design systems. We chose such criteria to assess how both experienced and inexperienced users perceive our proposed solution and value the various features in such type to tools.

5.4.2 User study design

We start our user study by asking questions about the participant's experience with front-end design/development and design systems to assess their experience and categorize them into experienced or novice users (the user study protocol is included in Appendix E). We then continue our study by asking the user to perform the following tasks:

Task 1: Five-seconds test. The five-second test is a common UX testing method. It involves displaying visual components, in our case our prototype interface, for a duration of 5 seconds to the participants to collect their feedback about the overall impressions and the prominent aspects they recall. This test aims to understand what characterizes and stands out in the interface design of our prototype [52].

Task 2: Intuitiveness test. This test is for evaluating if our interface is intuitive and can be understood easily by both experienced and novice practitioners. During this task, the participants were given the mouse and keyboard controls and asked to explore the main interface and identify the functionalities of the application without clicking any component while thinking aloud (meaning the participant speaks out whatever comes into his or her mind).

Task 3: Basic usability test. The participants were given tasks to use all the application main features, including (1) selecting components from the components list, (2) loading two

Participants	Age	Gender	Country	Job title
P1	37	Male	Portugal	Lead designer
P2	28	Male	Uk	Product design manager
P3	32	Female	Canada	Software engineer
P4	25	Female	Mexico	Designer
P5	25	Female	Mexico	UX designer
P6	41	Male	Canada	UX consultant
P7	45	Male	Mexico	Web design Freelancer
P8	35	Male	Ecuador	Designer
P9	25	Male	Morocco	Fullstack web developer
P10	25	Male	Morocco	Android developer
P11	28	Male	France	Senior software developer
P12	25	Female	Canada	Software developer
P13	28	Male	Canada	Software developer
P14	26	Male	Canada	Software developer
P15	23	Male	Canada	Software developer

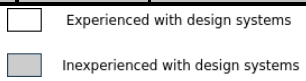


Figure 5.6 Evaluation study participants

websites and merge styles from both, , (3) manually modifying a certain style attribute of the selected components, and finally (4) applying the modified style to the component. The objective of this task is to evaluate each step of the application workflow. The think-aloud protocol was also used in this task.

Task 4: Component customization. The fourth task focuses on the customization of the button component. The participants were asked to merge the styles of two buttons from the GitHub and the GitLab website. Participants were asked to use two methods to complete this task: (1) by using the application, and (2) by using a text editor to modify the design system code and browser to inspect the component. The order of the methods

was counterbalanced among the participants. The goal of this task is to assess the entire workflow of the application and to compare the user experience of the application with the traditional design system customization method.

Task 5: Free use test. The final task consists of customizing the remaining design system components we provided in the template (i.e. heading, subheading, land a link). No direct instruction was given to the participants and the participants were free to find styles from different websites, merge them, and/or modify them manually. We aim to give the participants the freedom to use the application without any guidance for the possibility of getting extra comments about the design. Finally, we asked the participants to answer a questionnaire about the most useful features in our interface and how it will fit in their workflow using a like-rt scale, this questionnaire is attached in E.

Usability test After the participant finished all the tasks, we asked them to complete a questionnaire adapted from the System Usability Scale (SUS), a popular self-administrated measure for evaluating the usability of user interfaces [53]. The SUS is a 10 item questionnaire with 5 response LIKE-RT scale options (i.e 1 for Strongly Disagree and 5 for Strongly agree). They were also asked to answer a Subjective Mental Effort Question (SMEQ). It is basically a single answer question to about the mental effort required from the participant to perform a certain task [54].

More details about these two tests are provided in Appendix C.

5.5 Data collection and analysis

To conduct our evaluation study, we run our interface on our computer and use Zoom[4] share screen feature to allow users to use and interact with our prototype, we start our study by asking questions about the experience of our participants with design systems and front-end development, we then guide them through the evaluation tasks we defined. We video record the sessions using also Zoom.

To analyze the recorded data, we go through the recordings and summarize the comments of our participants and also take notes about the interaction of our users with the interface while performing the tasks.

[4]https://zoom.us/

5.6 User study results

In this section, we report the results of our evaluation with the recruited participants. We group these results in 6 categories, 1) First impressions, 2) Intuitiveness, 3) Suggestions for improvements, 4) Usefulness of the tool, 5) Usability, and 6) Experience and perception of the workflow.

5.6.1 First impressions

Participants had different impressions about the interface design from initial exposure during the five-second test (i.e Task 1), P2 for example commented on how it looks like a software more than a website, he said it is similar to Photoshop in terms of design. While P5 and P6 commented on how it has a developer tool feel. P4 on the other hand noticed how the application is a single page which she finds useful, she prefers to have all the information on the same page without the need to navigate to perform certain tasks. Other participants (i.e 6 participants), mostly the inexperienced group commented on the aesthetics, colors, and the theme of our design. They had different preferences, P7 for example finds our interface simple both not that ascetic in terms of design, P12 and P15 liked our dark theme design.

5.6.2 Intuitiveness

In the intuitiveness test (i.e Task 2). All of our participants (i.g experienced and inexperienced) found it difficult to identify all the functionalities of our application during this task. The main difficulty was to make association between different features of the application, particularly, the relation between the selected component and sidebar that has the option to modify its styles. This was a common thing with all participants we interviewed of both groups. Another thing worth mentioning is that participants' attention was drawn immediately toward the loaded website, with 5 participants (i.e P7, P5, P14, P15, and P13) starting to explore it as if it was the main interface. These results suggest that our interface has a certain degree of complexity. However, it's a common thing in most developer tools.

The improve the intuitiveness aspect, 5 participants suggested a guide introducing the features and the relation between different elements of the tool when opening the application.

5.6.3 Suggestions for improvements

During the third test (i.e basic usability test), the feedback of our participants consisted mainly of suggestions for improvements to our design and features. The first and the most common is providing a short tutorial to guide users during their initial use of this tool, 8 participants mentioned this (2 experienced and 6 inexperienced). As for missing features, an interesting thing one of our participants motioned is providing a pattern or guidelines to assure that there will be some consistency while customizing the components. Another suggestion by two experienced participants is being able to add new and build new components, a feature that was not included in this prototype. In addition, three experienced participants suggested changes to the workflow, specifically, changing the position of the export button and include it in the sidebar for more improved ergonomics. Finally, 2 participants, one experienced and another developer talked about how they would prefer to not starting customizing the design system from scratch, but rather generate a complete design system from the loaded website and then start removing and editing the components according to their needs. Other suggestions consisted of improvements to the UI including the color, fonts, spacing, etc.

5.6.4 Usefulness of the tool

From the fourth (i.g component customization and free use test) we received positive feedback from our participants about the usefulness of our tool Figure 5.8, two of them commented on its usefulness for novice users, our tool doesn't require a lot of experience in UI development to be used. Additionally, 4 participants (i.e 3 experienced and 1 inexperienced) commented on how this can be a very good solution for small companies because of low resources it can spare compared with a real design system. While 6 developers (i.e inexperienced group) talked about how our tool can reduce the time they put in developing UI components.

For the useful features, 14 participants including 5 experienced described the CSS extraction feature as the most useful, merging styles from multiple websites while staying on the same platform is very efficient for our participants, However, one experienced design system practitioner (i.e P2) and a developer (i.e P13) commented on how this approach was somehow limiting, it lacks the flexibility that coding offers using a text editor and the browser. Another useful feature most of the participants focused on (11 non-experienced and 2 experienced) is the interactive way of customizing components, the ability to see the changes in the selected component in real-time is something these participants value.

5.6.5 Usability

In terms of usability, the average task completion rate is **81%**, the majority of the participants managed to perform the tasks successfully except for a few who either skipped the task and didn't comply with the provided protocol or participants who had experienced technical difficulties, and finally participants that explained how they're work is consisted mainly of design work and they weren't comfortable writing CSS and using the text editor. Figure 5.1 present the task completion for each participant.

The average SUS score for all participants was **56.16** which suggests that our application's usability is fairly acceptable or "Ok" according to Bangor et al. evaluation of the system usability scale [55]. Table 5.2 presents the average SUS score of our participants.

Participant	Task 1	Task2	Task 3	Task4	Task 5
p1	1	1	0	1	1
p2	0	1	0	1	0
p3	1	0	1	1	1
p4	0	0	1	1	1
p5	1	0	0	1	1
p6	1	1	1	0	0
P7	1	1	1	0	0
P8	1	1	1	0	1
P9	1	1	1	1	1
P10	1	1	1	1	1
P11	1	1	1	1	1
P12	1	1	1	1	1
P13	1	1	1	1	1
P14	1	1	1	1	1
P15	1	1	1	1	1
average	0.86	0.8	0.8	0.8	0.8

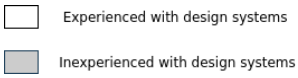


Table 5.1 Task completion for each participant (i.e 1 for completed task and 0 for uncompleted task)

We concluded our study by asking one question about the mental effort it took to perform the tasks following the SMEQ questionnaire. The average score in our case was **20.8**. This value shows that the tasks we asked the participants to perform did not require a lot of mental effort.

5.6.6 Experience and perception of the workflow

Our participants had different ideas about how this application will fit in their workflow, the most interesting was that four participants (3 experienced and 1 inexperienced), all designers considered this tool as a perfect solution for prototyping and improving the collaboration between developers and designers, its seems that they have problems when they push their design to the developers without considering the technical aspect which creates a conflict

Participant	SUS score
P1	45
P2	52.5
P3	72.5
P4	67.5
P5	60
P6	35
P7	65
P8	52.5
P9	47.5
P10	55
P11	60
P12	52.5
P13	62.5
P14	52.5
P15	62.5

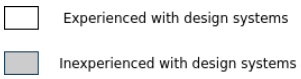


Table 5.2 Average SUS score of participants

between these parties, however, according to them, this tool will allow them to test their designs without understanding the code, then if they can build it, there will be no excuse for the developers to reject their designs because the implementation is feasible. Other participants, 9 to be exactly mentioned that the efficiency that this tool offers in customizing the design system component is enough to fit in their workflow, it offers the ability to customize UI components without the need to use other tools and to switch between different applications (Browser, Text editor end Inspect tool).

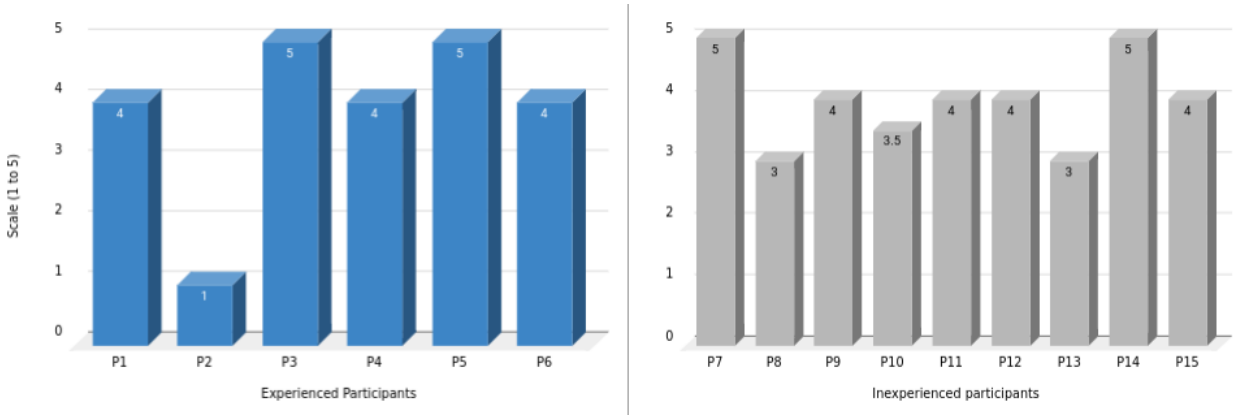


Figure 5.7 How the tool fit in the participants workflow from a scale (1 (would not fit at all) to 5 (a perfect fit))

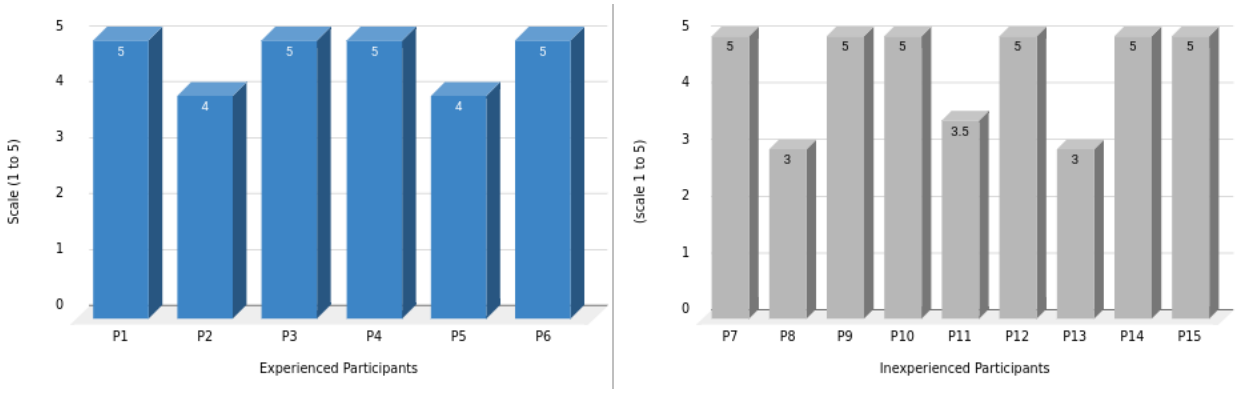


Figure 5.8 The usefulness of the application on a scale of (1 (not useful at all) to 5 (very useful))

5.7 Discussion

The evaluation of our proposed solution by the two group of participants (i.e. practitioners experienced and inexperienced with design systems) revealed issues in terms of intuitiveness of our interface. Our participants failed to identify or figure out the functionalities of the system from initial trial which suggest a degree of complexity that can be reduced by a use guide explaining the features our solution offers. This phenomenon can also be explained by the fact that no tools currently exist to support the creation of design systems.

As for the usability of our system, the result of SUS scores from our participants suggests that, once familiarized with the tool features, both experienced and inexperienced participants perceived adequate usability of our system. Interesting, the use cases of our tool are perceived differently by the experienced and inexperienced participants. The inexperienced participants explained how they would use it for its original propose, for customizing the design system UI components. However, experienced participants with design system thought that the tool fits more for introducing the design system concept in small companies with limited resources and also for small projects use. Additionally, prototyping was a common use case for designers, but not developers. Our designer participants mentioned how they could use our tool to efficiently test the implementation of their designs before pushing them to developers. According to them, this could help resolve the collaboration problems between these two disciplines that is sometimes caused by the poor understanding of technical aspect of implementing UI components by designers as we mentioned in our literature review.

To conclude, we believe that our solution achieved the goals we aimed for. The positive feedback we received about the features we included (i.e. style extraction from websites, integrated design system, and options menu and text editor for editing styles) suggest that a full fledged application incorporating the suggestions by our participants can help promote the use of design systems and facilitate UI development for web applications.

CHAPTER 6 CONCLUSION

6.1 Summary

This research aimed to understand various aspects related to design systems, particularly, their open-sourcing, development, and benefits. We conducted three qualitative analysis studies to address each of these topics. We started our research project by a content analysis study of GitHub ITS data of 41 open-source design system, this data consisted of 4714 issues. We discovered UI visual design and behavior is the main focus in design systems which wasn't surprising considering the nature of the design system as a tool for UI development. However, our analysis also revealed discussions regarding community-related issues, which suggested that design system owners are interested in engaging their design systems with external contributors. In addition, we also encountered issues reporting software engineering and documentation problems.

Following these preliminary results, we conducted an interview study with active contributors to open-source design systems in order to explain to learn about the motivations behind open sourcing design systems, the challenges they face during their development and maintenance, how they mitigate these challenges, and finally the benefits of having a design system inside a company that produces software products.

By analyzing the interviews data, we identified multiple benefits of making design systems available to the public, external contributors are ideal candidates for hiring as described by our participants, the active ones are usually familiar with the code and the structure of the design system which makes their integration in the company very efficient. Additionally, the code quality of software projects improves after open-sourcing because developers pay more attention to their published work. Other benefits of open-sourcing our participants mentioned consisted of gaining publicity and a community that can help test and detects bugs in the design system. However, open sourcing comes with some drawbacks inclusive of extra management resources for the online repository to avoid sensitive information leaks. As for development challenges of design systems, communication and collaboration during the creation or the evolution of the design system came out first due to the need for continuous discussions while deciding the UI components it will include, in order to guarantee its adoption by the organization.

To mitigate these challenges, good communication among all the parties that interact and involved in the design system during development and maintenance is an essential element for

success the same as having its own dedicated team. On top of that building, the design system from existing products will minimize resources of development. Having a design system leads to more efficiency and development speed due to the consistency and the coherence of its UI components that can also help improve interactions between the consumer of UI components and the front-end developers and designers inside the company. These benefits help reduce costs significantly according to our participants.

Finally, to address some of the challenges our participants expressed, specifically the difficulty of developing and evolving the design system, we proposed a prototype solution that generates a full custom design system according to the user's need in very simple steps, our tool is a web-based application with a variety of features that offer flexibility, support fast prototyping, usage simplicity and resource-efficient for creating or using a design system. We conducted a user study to evaluate our solution and we received overall positive feedback from 15 participants that found our solution useful as a prototyping tool and an efficient way of introducing a design system in companies without enough resources for developing one from scratch.

6.2 Limitations and threats to validity

Although our results answer multiple questions about design systems, our study still has some limitations. For example, in our issue analysis study, we didn't include proprietary design systems or open-source design systems that are hosted on other repositories. While GitHub is considered as the biggest open source platform hosting many design system projects, our findings may not apply to all design systems.

Also, we only interviewed highly experienced contributors. The perspective of novice practitioners on how they perceive the benefits and the challenges of the use of the design system approach is still to be explored. Additionally, our experienced participants have all worked on established design systems. Our results thus may not apply to design system projects that are at their very early stages.

Additionally, in our interview study we managed to recruit 9 participants, while in our user study for the evaluation of our tool, we only interviewed 6 experienced participants. Although these numbers are comparable to other interview studies, our participant sample may have certain characteristics that have biased our results. Thus a larger-scale study with more participants would be useful to further validate our findings and evaluate the usability and effectiveness of our proposed solution.

Further, although inspired from real-world tasks, our evaluation study was conducted un-

der a controlled environment. Practitioners may work differently in real-world settings in which they face practical issues and constraints. A contextual field study would be useful to strengthen our findings.

Finally, our proposed solution is not a full-fledged application. It is only a prototype for demonstrating and evaluating certain features that can help design system practitioners. Other factors, such as scalability and compatibility with existing tools, should also be considered and evaluated when designing a full-fledged application.

6.3 Future work

For our future work, we would like to continue improving our prototype and eventually implement a beta version of our tool to actually benefit practitioners. More comprehensive evaluations with a larger number of design system practitioners, front-end developers, and designers with different backgrounds and experiences are also needed to assess the usability and the effectiveness of the tool.

REFERENCES

[1] D. Mounter, M. Suarez, R. Stanfield, K. Sylor-Miller, and J. Anne, *Design Systems Handbook*. InVision, July 26, 2019.

[2] C. Alexander, S. Ishikawa, and M. Silverstein, *A Pattern Language: Towns, Buildings, Construction*. New York: Oxford University Press, 1977.

[3] T. Erickson, "Lingua francas for design: Sacred places and pattern languages," in *Proceedings of the 3rd Conference on Designing Interactive Systems: Processes, Practices, Methods, and Techniques*, ser. DIS '00. New York, NY, USA: ACM, 2000, pp. 357–368. [Online]. Available: http://doi.acm.org/10.1145/347642.347794

[4] J. Tidwell, *Designing Interfaces*. O'Reilly Media, Inc., 2010.

[5] D. K. V. Duyne, J. Landay, and J. I. Hong, *The Design of Sites: Patterns, Principles, and Processes for Crafting a Customer-Centered Web Experience*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2002.

[6] B. Scott and T. Neil, *Designing Web Interfaces: Principles and Patterns for Rich Interactions*, 1st ed. O'Reilly Media, Inc., 2009.

[7] T. Neil, *Mobile Design Pattern Gallery: UI Patterns for Smartphone Apps*. O'Reilly Media, Inc., 2014.

[8] P. Morville and J. Callender, *Search Patterns: Design for Discovery*, 1st ed. O'Reilly Media, Inc., 2010.

[9] E. S. Chung, J. I. Hong, J. Lin, M. K. Prabaker, J. A. Landay, and A. L. Liu, "Development and evaluation of emerging design patterns for ubiquitous computing," in *Proceedings of the 5th Conference on Designing Interactive Systems: Processes, Practices, Methods, and Techniques*, ser. DIS '04. New York, NY, USA: ACM, 2004, pp. 233–242. [Online]. Available: http://doi.acm.org/10.1145/1013115.1013148

[10] J. Cheng, D. Anderson, C. Putnam, and J. Guo, "Leveraging design patterns to support designer-therapist collaboration when ideating brain injury therapy games," in *Proceedings of the Annual Symposium on Computer-Human Interaction in Play*, ser. CHI PLAY '17. New York, NY, USA: ACM, 2017, pp. 291–303. [Online]. Available: http://doi.acm.org/10.1145/3116595.3116600

[11] S. Bjork and J. Holopainen, *Patterns in Game Design (Game Development Series)*. Rockland, MA, USA: Charles River Media, Inc., 2004.

[12] R. S. Gutzwiller, S. H. Espinosa, C. Kenny, and D. S. Lange, "A design pattern for working agreements in human-autonomy teaming," in *Advances in Human Factors in Simulation and Modeling*, D. N. Cassenti, Ed. Cham: Springer International Publishing, 2018, pp. 12–24.

[13] K. Ma and J. Cao, "Design pattern as a practical tool for designing adaptive interactions connecting human and social robots," in *Intelligent Human Systems Integration 2019*, W. Karwowski and T. Ahram, Eds. Cham: Springer International Publishing, 2019, pp. 613–617.

[14] A. Dearden, J. Finlay, E. Allgar, and B. McManus, "Using pattern languages in participatory design," *Proceedings of the Participatory Design Conference (PDC 2002)*, pp. 104–113, 2002. [Online]. Available: http://eprints.leedsbeckett.ac.uk/666/

[15] R. Saxena, "A beginner's guide to css front-end frameworks," Apr 2019. [Online]. Available: https://dzone.com/articles/a-beginners-guide-to-css-front-end-frameworks

[16] D. A. Schön, *The reflective practitioner: How professionals think in action*. Basic Books, 1984.

[17] N. Cross, "Designerly ways of knowing: Design discipline versus design science," *Design Issues*, vol. 17, no. 3, pp. 49–55, 2001.

[18] J. D. Gould and C. Lewis, "Designing for usability: Key principles and what designers think," *Communications of the ACM*, vol. 28, no. 3, pp. 300–311, Mar. 1985. [Online]. Available: http://doi.acm.org/10.1145/3166.3170

[19] E. Goodman, E. Stolterman, and R. Wakkary, "Understanding interaction design practices," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '11. New York, NY, USA: ACM, 2011, pp. 1061–1070. [Online]. Available: http://doi.acm.org/10.1145/1978942.1979100

[20] C. M. Gray, "Evolution of design competence in ux practice," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '14. New York, NY, USA: ACM, 2014, pp. 1645–1654. [Online]. Available: http://doi.acm.org/10.1145/2556288.2557264

[21] E. Stolterman and J. Pierce, “Design tools in practice: Studying the designer-tool relationship in interaction design,” in *Proceedings of the Designing Interactive Systems Conference*, ser. DIS ’12. New York, NY, USA: ACM, 2012, pp. 25–28. [Online]. Available: http://doi.acm.org/10.1145/2317956.2317961

[22] C. M. Gray, “"it’s more of a mindset than a method": Ux practitioners’ conception of design methods,” in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, ser. CHI ’16. New York, NY, USA: ACM, 2016, pp. 4044–4055. [Online]. Available: http://doi.acm.org/10.1145/2858036.2858410

[23] X. Zhang and R. Wakkary, “Understanding the role of designers’ personal experiences in interaction design practice,” in *Proceedings of the 2014 Conference on Designing Interactive Systems*, ser. DIS ’14. New York, NY, USA: ACM, 2014, pp. 895–904. [Online]. Available: http://doi.acm.org/10.1145/2598510.2598556

[24] M. Mose Biskjaer, P. Dalsgaard, and K. Halskov, “Understanding creativity methods in design,” in *Proceedings of the 2017 Conference on Designing Interactive Systems*, ser. DIS ’17. New York, NY, USA: ACM, 2017, pp. 839–851. [Online]. Available: http://doi.acm.org/10.1145/3064663.3064692

[25] J. Ferreira, J. Noble, and R. Biddle, “Agile development iterations and ui design,” in *Agile 2007 (AGILE 2007)*, 2007, pp. 50–58.

[26] J. M. Brown, G. Lindgaard, and R. Biddle, “Joint implicit alignment work of interaction designers and software developers,” in *Proceedings of the 7th Nordic Conference on Human-Computer Interaction: Making Sense Through Design*, ser. NordiCHI ’12. New York, NY, USA: Association for Computing Machinery, 2012, p. 693–702. [Online]. Available: https://doi.org/10.1145/2399016.2399121

[27] S. Chamberlain, H. Sharp, and N. Maiden, “Towards a framework for integrating agile development and user-centred design,” in *Extreme Programming and Agile Processes in Software Engineering*, P. Abrahamsson, M. Marchesi, and G. Succi, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 143–153.

[28] M. Lundström, J. Åberg, and J. Blomkvist, “Perceptions of software developers’ empathy with designers,” in *Proceedings of the 2015 British HCI Conference*, ser. British HCI ’15. New York, NY, USA: Association for Computing Machinery, 2015, p. 239–246. [Online]. Available: https://doi.org/10.1145/2783446.2783563

[29] G. Leiva, N. Maudet, W. Mackay, and M. Beaudouin-Lafon, "Enact: Reducing designer–developer breakdowns when prototyping custom interactions," *ACM Trans. Comput.-Hum. Interact.*, vol. 26, no. 3, May 2019. [Online]. Available: https://doi.org/10.1145/3310276

[30] A. Y. Wale-Kolade, "Integrating usability work into a large inter-organisational agile development project: Tactics developed by usability designers," *Journal of Systems and Software*, vol. 100, pp. 54 – 66, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0164121214002337

[31] K. Crowston, K. Wei, J. Howison, and A. Wiggins, "Free/libre open-source software development: What we know and what we do not know," *ACM Comput. Surv.*, vol. 44, no. 2, pp. 7:1–7:35, Mar. 2008. [Online]. Available: http://doi.acm.org/10.1145/2089125.2089127

[32] J.-F. Schrape, "Open-source projects as incubators of innovation: From niche phenomenon to integral part of the industry," *Convergence*, vol. 25, no. 3, pp. 409–427, 2019. [Online]. Available: https://doi.org/10.1177/1354856517735795

[33] J. Cheng and J. L. C. Guo, "Activity-based analysis of open source software contributors: Roles and dynamics," in *Proceedings of the 12th International Workshop on Cooperative and Human Aspects of Software Engineering*, ser. CHASE '19. Piscataway, NJ, USA: IEEE Press, 2019, pp. 11–18. [Online]. Available: https://doi.org/10.1109/CHASE.2019.00011

[34] K. Nakakoji, Y. Yamamoto, Y. Nishinaka, K. Kishida, and Y. Ye, "Evolution patterns of open-source software systems and communities," in *Proceedings of the International Workshop on Principles of Software Evolution*. New York, New York, USA: ACM Press, 2002, p. 76.

[35] Y. Ye and K. Kishida, "Toward an understanding of the motivation open source software developers," in *Proceedings of the 25th International Conference on Software Engineering*, ser. ICSE '03. Washington, DC, USA: IEEE Computer Society, 2003, pp. 419–429. [Online]. Available: http://dl.acm.org/citation.cfm?id=776816.776867

[36] M. Joblin, S. Apel, and W. Mauerer, "Evolutionary trends of developer coordination: a network approach," *Empirical Software Engineering*, vol. 22, no. 4, pp. 2050–2094, Aug 2017. [Online]. Available: https://doi.org/10.1007/s10664-016-9478-9

[37] C. Cheng, B. Li, Z.-Y. Li, Y.-Q. Zhao, and F.-L. Liao, "Developer Role Evolution in Open Source Software Ecosystem: An Explanatory Study on GNOME," *J. Comput. Sci. Technol.*, vol. 32, no. 2, pp. 396–414, mar 2017.

[38] M. Gharehyazie, D. Posnett, B. Vasilescu, and V. Filkov, "Developer initiation and social interactions in oss: A case study of the apache software foundation," *Empirical Software Engineering*, vol. 20, no. 5, pp. 1318–1353, Oct 2015. [Online]. Available: https://doi.org/10.1007/s10664-014-9332-x

[39] C. Casalnuovo, B. Vasilescu, P. Devanbu, and V. Filkov, "Developer onboarding in github: The role of prior social links and language experience," in *Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering*, ser. ESEC/FSE 2015. New York, NY, USA: ACM, 2015, pp. 817–828. [Online]. Available: http://doi.acm.org/10.1145/2786805.2786854

[40] G. Hertel, S. Niedner, and S. Herrmann, "Motivation of software developers in open source projects: an internet-based survey of contributors to the linux kernel," *Research Policy*, vol. 32, no. 7, pp. 1159 – 1177, 2003, open Source Software Development. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0048733303000477

[41] S. O. Alexander Hars, "Working for free? motivations for participating in open-source projects," *International Journal of Electronic Commerce*, vol. 6, no. 3, pp. 25–39, 2002.

[42] A. Lee, J. C. Carver, and A. Bosu, "Understanding the impressions, motivations, and barriers of one time code contributors to floss projects: A survey," in *Proceedings of the 39th International Conference on Software Engineering*, ser. ICSE '17. Piscataway, NJ, USA: IEEE Press, 2017, pp. 187–197. [Online]. Available: https://doi.org/10.1109/ICSE.2017.25

[43] I. Steinmacher, M. Gerosa, T. U. Conte, and D. F. Redmiles, "Overcoming social barriers when contributing to open source software projects," *Computer Supported Cooperative Work (CSCW)*, vol. 28, no. 1, pp. 247–290, Apr 2019. [Online]. Available: https://doi.org/10.1007/s10606-018-9335-z

[44] S. Elo and H. Kyngäs, "The qualitative content analysis process," *Journal of Advanced Nursing*, vol. 62, no. 1, pp. 107–115, 2008. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1365-2648.2007.04569.x

[45] J. Saldaña, *The Coding Manual for Qualitative Researchers*, 2016.

[46] T. F. Bissyandé, D. Lo, L. Jiang, L. Réveillère, J. Klein, and Y. L. Traon, "Got issues? who cares about it? a large scale investigation of issue trackers from github," in *2013 IEEE 24th International Symposium on Software Reliability Engineering (ISSRE)*, 2013, pp. 188–197.

[47] D. Bertram, A. Voida, S. Greenberg, and R. Walker, "Communication, collaboration, and bugs: The social nature of issue tracking in small, collocated teams," in *Proceedings of the 2010 ACM Conference on Computer Supported Cooperative Work*, ser. CSCW '10. New York, NY, USA: Association for Computing Machinery, 2010, p. 291–300. [Online]. Available: https://doi.org/10.1145/1718918.1718972

[48] W. Wang, J. Cheng, and J. L. C. Guo, "How do open source software contributors perceive and address usability? valued factors, practices, and challenges," 2020.

[49] M. Ml, "Inter-rater reliability: the kappa statistic," pp. 276–286, 2012.

[50] L. Morgan and P. Finnegan, "Benefits and drawbacks of open source software: An exploratory study of secondary software firms," in *Open Source Development, Adoption and Innovation*, J. Feller, B. Fitzgerald, W. Scacchi, and A. Sillitti, Eds. Boston, MA: Springer US, 2007, pp. 307–312.

[51] K. R. Lakhani and R. G. Wolf, "Why hackers do what they do: Understanding motivation and effort in free/open source software projects," 2003. [Online]. Available: https://ssrn.com/abstract=443040orhttp://dx.doi.org/10.2139/ssrn.443040

[52] P. Doncaster, *The UX Five-Second Rules: Guidelines for User Experience Design's Simplest Testing Technique*, 1st ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2014.

[53] A. I. Martins, A. F. Rosa, A. Queirós, A. Silva, and N. P. Rocha, "European portuguese validation of the system usability scale (sus)," *Procedia Computer Science*, vol. 67, pp. 293 – 300, 2015, proceedings of the 6th International Conference on Software Development and Technologies for Enhancing Accessibility and Fighting Info-exclusion. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877050915031191

[54] J. Sauro and J. S. Dumas, "Comparison of three one-question, post-task usability questionnaires," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '09. New York, NY, USA: Association

for Computing Machinery, 2009, p. 1599–1608. [Online]. Available: https://doi.org/10.1145/1518701.1518946

[55] A. Bangor, P. T. Kortum, and J. T. Miller, "An empirical evaluation of the system usability scale," *International Journal of Human–Computer Interaction*, vol. 24, no. 6, pp. 574–594, 2008. [Online]. Available: https://doi.org/10.1080/10447310802205776

APPENDIX A LIST OF OF THE ISSUE NATURE THEMES IDENTIFIED IN CONTENT ANALYSIS

Theme	Description	Frequency	Sub-themes	Description	Frequency
Bugs	Reported bugs in the design system	59.6%	UI bugs	Bugs related the UI behavior, functionality, style, etc.	84%
			Software engineering bugs	Issues in testing, deployment and third-party libraries.	9%
			documentation issues	Bugs in the design system documentation.	5%
Requests for new elements	Requests for adding components and features	29.1%	-	-	-
Suggestions for improvements	Requests or suggestions for improvements to existing component or functionality of the design system	8%	-	-	-
Questions	questions about various aspects of the design system (e.g components, development, features, etc)	3.4%	-	-	-

APPENDIX B LIST OF OF THE PROJECT ASPECT THEMES IDENTIFIED IN CONTENT ANALYSIS

Theme	Description	Frequency
Behavior of UI components	e.g color or position change, animation, size change, etc.	61.8%
The visual design of UI components	The appearance of component	14.7%
Documentation	The documentation of the design system	9.8%
Software engineering	Testing, deployment, package managers, etc.	9.5%
Community	The repository contributors and external users of the design system.	4.1%

APPENDIX C INTERVIEW STUDY QUESTIONS

Experience and Roles

- What is your current job title?
- Please briefly describe your job responsibility.
- How many years have you been working as a designer?
- How many years have you been working around design systems?
- We know that you are actively contributing to xxxx. Have you been working on other design systems? If so, what are they?

Definition

- How do you define design systems?
- In your opinion, what are the biggest benefits of design systems?
- What are the main drawbacks of design systems, if any?

For the following questions, please consider the current design system you work on.

Basic Information

- How did you get involved?
- How big is the team involved in developing and maintaining this design system?
- How big is the audience who use this design system?

Developing the Design System

- How did you develop this design system? (What are the general approach?)
- What types of tools do you use to help you develop this design system?
- What would you consider are the biggest challenges in developing design systems?
- In this project, how did you address them?

Maintaining the Design System

- How do you maintain this design system? (What are the general approach?)
- What components tended to undergo frequent change?
- What types of tools do you use to help you maintain this design system?
- What would you consider are the biggest challenges in maintaining design systems?
- In this project, how did you address them?

Open-sourcing the Design System

- When was this design system put open source? (Was it open source from the development starting point or was it put on github after the first release?)
- What was the motivation to make this design system open source?
- What are the advantages and disadvantages?
- What are the most common issues and bugs people report?
- What are the criteria to approve contributors commits?
- What are the most common reasons you reject contributors commits?

Using the Design System

- How is the design system used in your organization?
- What parts are most frequently used?
- How do you support the use of design systems?
- What would you consider are the biggest challenges in using design systems?
- In this project, how did you address them?

Wrap Up:

- Do you have any question for me? Is there anything you would like to add?

APPENDIX D FORMATIVE STUDY QUESTIONS

What was your first impression about this application?

How do you think a tool like this aligns with the philosophy of design systems? (losing consistency and control over the design system)

How do you think this tool can affect the creation of design systems inside a company?

How would you think it will be used?

Who would use it? Why?

What are the negative effects of using it inside a company?

What are the positive effects of using it inside a company?

If not going to be used at all:

Why is that?

What kind of features should be included to be used?

How do you think this tool can affect the use of design systems inside a company?

How do you think this tool can affect the community of design systems? Why?

How do you think this tool can affect the use of design systems? Why?

What do you think about the current features of the tool based on the mockup?

Are there any other features you want to include?

What do you think about the user interface?

What kind of technologies would you recommend building this tool considering the fact that it will be open source and we would like to have contributors from the design system community?

Do you consider yourself involved in the design system community?

If yes, How do you think the community will react to this tool?

APPENDIX E USER STUDY PROTOCOL

[The following is the script that the researcher will use to guide his/her interaction with the participant.]

Introduction:

Thank you very much for being able to participate in this research study. As you have read in the consent form, we are focused on evaluating the usability and efficiency of a web-based application we developed to address the challenges of design systems development and usage. The process is estimated to take approximately 90 minutes. During the study, we will ask you to customize a design system with the support of the design system tool.

Before we get started, I would like to remind you that this study will be video recorded. We will strive to keep confidentiality about your identity and the data. You have the right to end your participation at any point. Once you give us the notice, we will remove all the data related to your participation.

I would also like to remind you that there is no right or wrong way to complete the tasks or answer the questions. We are not testing your ability to complete the tasks; instead, we want to understand how you approach the tasks.

Do you have any questions before we proceed?

[Answer participants' questions and gather consent if not yet.]

Pre-test questions:

We will start by asking you a few questions:

On a scale of one to five, one being very unfamiliar and five being very familiar, how familiar are you with front-end web application development? Please briefly describe your most recent web application project.

On a scale of one to five, one being very unfamiliar and five being very familiar, how familiar are you with design systems? In case of no-experience, The researcher will proceed explaining what a design system is.

Have you used design systems? If so, please briefly describe your experience.

Have you created design systems? If so, please briefly describe your experience.

Usability tests :

[The tasks below are for testing the application workflow and if it is intuitive, meaning that

participants can have basic understanding of the application's main features without any documentation or presentation].

Taks 1 : Five-seconds test :

I would like you to click on the display button in this web page [link] and pay close attention to the image that will be displayed.

What is your overall impression on the displayed design ?

Any particular things you want to point out?

Task 2 : Usability test:

Feel free to explore the main interface and try to identify the functionalities of the application without clicking any component.

When completing the tasks, I would like to ask you to think aloud, meaning to say whatever comes into your mind. I will try not to interrupt while you complete the tasks. I may, however, remind you to think aloud so that we could understand your cognitive process.

[After having basic understanding of the features of the application the participants will be asked to perform the following tasks]

Task 3: Basic usability.

I will ask you to perform the following tasks.

1.Load two websites and extract styles from both.

2.Select components from the components list.

3.Manually modify a certain style attribute of the selected components.

4.Apply the style by clicking on the "apply style" button.

[When completed, the participants will be asked the following questions]

What do you think about the features?

What do you think about the workflow? Is there anything you would like to add or change?

Task 4 : Component customization

Select the button from the component list, try to combine button styles from github and gitlab as shown in the example below. When completed, click "apply style" button to save your component.

[For the participants who starts with this task, they will be provided with a picture of the desired customization of the button and where they can find the wanted styles (github and gitlab)]

I would like you to produce the same component from task 2 the way you prefer.

[Our objective from this task is to compare the two methods, When completing the task, the participants will be asked to answer the following question]

Between the two approaches of creating customizing components, which one worked better for completing the task? Why?

Tell us two things you liked about our application in comparison with the manual way of customising the style.

Tell us two things you did not like about our application in comparison with the manual way of customising the style.]

Task 4: Customize all the components [I decided to remove this task because is the same as the basic usability] .

Customize the remaining components based on your preference and export your work.

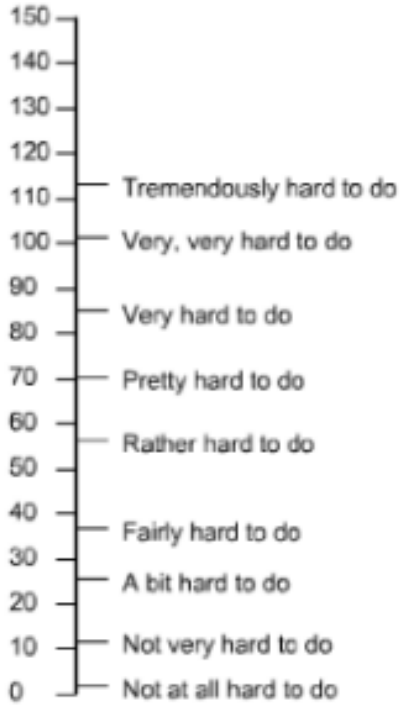
[we want to give the participants the freedom to use the application without any guidance, we might get interesting comments that may be considered in the design ,The new components will be displayed in a new window for the participants to see].

After the participant finished all the tasks, we will ask them to complete the following questionnaire adapted from the System Usability Scale (SUS) and Subjective Mental Effort Question (SMEQ) instruments.

For each of the following statements, mark one box that best describes your reactions to this representation of the Issue.

	Strongly Disagree				Strongly Agree
I think that I would like to use this approach/system frequently.	☐	☐	☐	☐	☐
I found this approach/system unnecessarily complex.	☐	☐	☐	☐	☐
I thought this approach/system was easy to use.	☐	☐	☐	☐	☐
I think that I would need assistance to be able to use this approach/system.	☐	☐	☐	☐	☐
I found the various functions in this approach/system were well integrated.	☐	☐	☐	☐	☐
I thought there was too much inconsistency in this approach/system.	☐	☐	☐	☐	☐
I would imagine that most people would learn to use this approach/system very quickly.	☐	☐	☐	☐	☐
I found this approach/system very cumbersome/awkward to use.	☐	☐	☐	☐	☐
I felt very confident using this approach/system.	☐	☐	☐	☐	☐
I needed to learn a lot of things before I could get going with this approach/system.	☐	☐	☐	☐	☐

The graph displays the amount of effort it took you to customize the design style using the application Please score the amount of effort by drawing a horizontal line on the graph.



[Ask the following question after the tasks are finished.]

In all the features, which ones were the most useful? Why?

In all the features, which ones were the least useful? Why?

Are there any features you would like to see but are missing in the application? If so, what are they?

How well would this application fit in your workflow?

Please rate it on a scale from 1 (would not fit at all) to 5 (a perfect fit).

Why did you rate it that way?

Overall, how useful do you think the application is?

Please rate it on a scale from 1 (not useful) to 5 (very useful).

Why did you rate it that way?

Do you have any comments about this approach of creating custom design systems?

Thank you so much for your participation!