A First Look at Accessibility Issues in Popular GitHub Projects

Tingting Bi   Xin Xia
Faculty of Information and Technology
Monash University
Melbourne, Australia
Email: tingting.bi, xin.xia@monash.edu

David Lo
School of Information Systems
Singapore Management University
Singapore, Singapore
Email: davidlo@smu.edu.sg

Aldeida Aleti
Faculty of Information and Technology
Monash University
Melbourne, Australia
Email: aldeida.aleti@monash.edu

***Abstract*—Accessibility design elements allow people to access software products and services independent of their different abilities. However, accessibility is challenging to handle and whether accessibility is widely considered in software projects is unclear. In this work, we aim to understand if accessibility is a prevalent consideration in practice, what accessibility issues are discussed in GitHub projects, what potential reasons cause accessibility issues, and what solutions (e.g., tools and standards) are applied for addressing accessibility issues. In this work, we collect 11,820 accessibility issues and their threads discussed by developers in popular GitHub projects. We manually analyzed and grouped the collected accessibility issues into seven categories. The results of our study uncover that accessibility is widely discussed in general projects, and the potential reasons that cause accessibility issues are because developers are not aware of the importance of accessibility and they lack knowledge about accessibility concerns, standards, and existing tools. Our results and findings can enhance and improve developers' knowledge and awareness when they conduct accessibility-relevant design or incorporate accessibility elements into their projects.**

***Index Terms*—Accessibility Issues, Empirical Study, Mining Repository**

I. Introduction

In recent years, accessibility concerns are increasingly receiving attention in the software engineering field [1]–[3]. Accessibility is the practice of making software applications more usable by as many people as possible independent of their different abilities [4]. According to ISO 9241 standard, it is essential to incorporate accessibility goals and features into the design. Accessibility can also benefit other groups of people, because designing human-system interaction to increase accessibility promotes effectiveness, efficiency, and satisfaction [5], such as those using mobile devices or those with slow network connections [6], and semantic HTML improves accessibility and makes websites more easy to find [7]. Although accessibility laws and standards [8]–[10] have been introduced in many countries, some companies and developers still find it challenging to incorporate accessibility into projects. Furthermore, a certain number of people with different abilities are struggling and have to spend extra effort and time using popular software applications [11]. Given that a lot of accessibility issues are reported during software development, we argue that a better understanding of accessibility issues, challenges, and other relevant discussions can help developers and organizations create more accessible software applications [12].

There are some attempts at accessibility development and design. For example, in our previous work, we have investigated developers' general opinions on accessibility and how accessibility is incorporated into different phases of software development [13]. Abdulaziz *et al.* have presented a study of understanding accessibility issues in Android apps from complementary perspectives [14]. In addition, there are numerous standards and guidelines available for basing accessibility design, development, and testing [15]–[17]. While previous works and resources are valuable for understanding accessibility, some gaps exist in the literature. For example, which accessibility issues do developers discuss in practice, what causes accessibility issues, what solutions developers applied for addressing accessibility issues. These gaps motivated us to investigate accessibility issues in real-world projects, and a comprehensive understanding of the characteristics of accessibility issues would provide better support for incorporating accessibility design elements into software applications.

GitHub is a large and popular open source platform, which hosts various projects. Developers use "Issue"[1] to report the concerns and bugs of projects [18] [19]. In this study, we took advantage of Issue discussions in GitHub projects as our data source to investigate and understand accessibility issues in practice. We manually collected 11,820 accessibility relevant issues in 1,000 GitHub projects; based on the analysis results and findings, we provide a set of practical lessons for developers to be aware of accessibility issues. We offer a complimentary analysis focusing on accessibility issues and threads discussed by developers. We analyzed the potential reasons that cause accessibility issues and proposed potential suggestions for improving accessibility in practice. With those new angles, this work makes the following key contributions:

- We found that accessibility issues are widely concerned in real-world projects, i.e., around 70% of selected projects in our data sample have reported and discussed accessibility issues.

[1]https://docs.github.com/en/github/managing-your-work-on-github/about-issues

- We grouped accessibility issues into seven categories, and most accessibility issues and discussions (i.e., around 62% of accessibility issues) were triggered by UI design (e.g., color and navigation). In addition, based on developers' discussions, we analyzed four reasons potentially causing accessibility issues.
- We structured the discussed knowledge for supporting accessibility design and development, i.e., standards, tools, and APIs. Furthermore, we offered a set of recommendations for developers to promote better accessibility design in practice.

The paper is organized as follows: Section II elaborates our methodology. The results of each research question are presented in Section III, and their implications and discussions are presented in Section IV. Section V examines the threats to validity, and Section VII concludes this work and outlines the directions for future research.

II. Methodology

In this work, we conducted an exploratory study [20] to investigate accessibility issues in GitHub projects. We elaborated our goals and research questions in Section II-A, and we described the data collection process in Section II-B, and the data analysis process is presented in Section II-C.

A. Goals and Research Questions

This study aims to understand what accessibility issues and relevant discussions are discussed by developers and how do developers address those accessibility issues. The Research Questions (RQs) we address in this study are as follows:

RQ1: What accessibility issues are discussed across GitHub projects?

Rationale: Accessibility design covers a wide range of elements [21] and accessibility requirements can come in a wide variety of forms [22]. It is vital to understand if accessibility is a general consideration across projects and what accessibility issues are discussed by developers. By answering this RQ, we provide insights into what accessibility issues and relevant considerations are discussed by developers in real-world projects.

RQ2: What are the causes of reporting accessibility issues from the perspective of developers?

Rationale: There is a set of requirements, standards, and goals to define accessibility [22] [23], e.g., perceivable, operable, and understanding. Developers design and develop software applications to achieve one or multiple of these accessibility goals. However, many reasons could hinder software applications from achieving these goals. Developers discuss and report potential issues, for example, bugs, design decision trade-offs, and improvement of accessibility. By answering this RQ, we identify the potential causes and consequences of accessibility issues in practice.

RQ3: What tools, standards, and APIs are applied to address accessibility issues in practice?

Rationale: A set of standards, tools, APIs, and other design elements for accessibility have been proposed and extensively investigated in the literature, yet it is not known whether developers use them. By answering this question, we provided a list of such accessibility practical design elements, which can help other developers make informed decisions when dealing with accessibility and inspire researchers to develop new approaches and tools.

RQ4: What are the practical recommendations for developers to handle and address accessibility issues?

Rationale: The ways to design and develop software applications vary from project to project, depending on their design contexts (e.g., development teams, financial budgets, and project domains) [24]. Therefore, accessibility design and development also would be addressed in different ways in practice. By analyzing potential solutions for addressing accessibility issues discussed between developers, we attempt to summarize and provide recommendations for assisting accessibility development and design in practice [25].

B. Data Collection

Step 1: Identifying the most popular GitHub repositories for analysis.

We used star ratings and popularity to identify popular GitHub projects to collect data in our study. This search was done as follows:

1) *Execute project search*: We used the search function provided by GitHub to rank the projects by star rating. The number of stars of a repository works like an easily accessible and reliable proxy to its popularity [26]. In addition, to ensure that the GitHub projects we investigated are non-trivial (i.e., the projects are active and the repositories of the projects are continually being updated), we selected the projects that were launched at least two years ago and keep being updated.
2) *Identify potential GitHub repositories*: We excluded the GitHub repositories, which are related to documentation, as those repositories provide books, code examples, and other textual materials [27]. In addition, Issue is a function provided by GitHub, and each repository has its own section for Issues [28]. Please note that, in this step, we excluded repositories with no issues information. We finally selected the top 1,000 GitHub projects with the highest star rating for the next step of the analysis.

Step 2: Identifying accessibility-related issues in the selected GitHub repositories.

In this step, we collected data related to accessibility issues. Before the formal data collection, we defined a set of search terms based on the topic (i.e., accessibility issues). Specially, we defined two sets of search terms. Set A (specific terms): "*Adaptability, Disabilities Act, Alternative Text, Assistive Technology, Audio Browsers, Captions, Screen reader, Switch Control, Usability, Web Accessibility, Universal Design, Color, Navigation*" [14]. Set B (general term): "*Accessibility*". We conducted a pilot search for ***accessibility*** issues in randomly selected 50 GitHub projects. We then checked the sample data from the returned results, and determined their relevance to the topic "Accessibility".

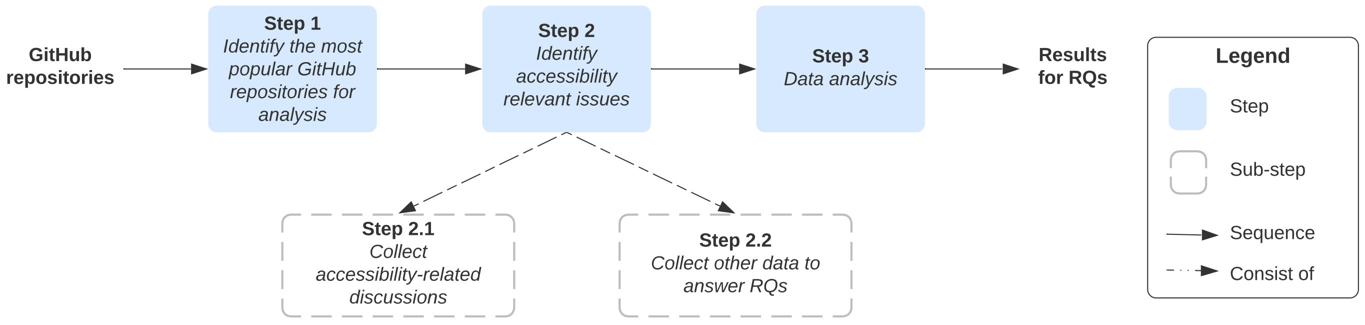
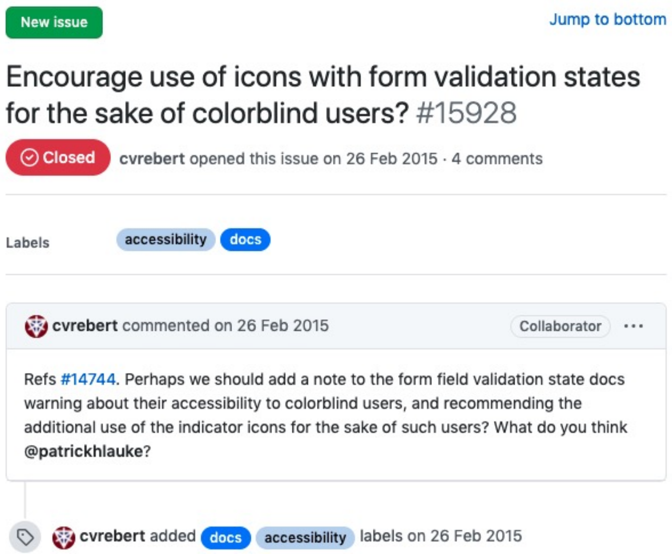

GitHub repositories
Step 1
Identify the most popular GitHub repositories for analysis
Step 2
Identify accessibility relevant issues
Step 3
Data analysis
Results for RQs
Step 2.1
Collect accessibility-related discussions
Step 2.2
Collect other data to answer RQs
Legend
Step
Sub-step
Sequence
Consist of

Fig. 1. Overview of the research process.


New issue
Jump to bottom

Encourage use of icons with form validation states for the sake of colorblind users? #15928

Closed cvrebert opened this issue on 26 Feb 2015 · 4 comments

Labels accessibility docs

cvrebert commented on 26 Feb 2015 Collaborator

Refs #14744. Perhaps we should add a note to the form field validation state docs warning about their accessibility to colorblind users, and recommending the additional use of the indicator icons for the sake of such users? What do you think @patrickhlauke?

cvrebert added docs accessibility labels on 26 Feb 2015

Fig. 2. An accessibility issue example from Github repositories.

The pilot search results of Set A were 564 issues, and 478 relevant accessibility issues were finally confirmed. The pilot search results of Set B were 554 issues, and 501 relevant accessibility issues were finally confirmed. We found that accessibility issue search using Set B is more efficient and can cover the accessibility issues using the Set A search terms. In addition, we found that when developers report accessibility-relevant issues, they would use "accessibility" term or select the accessibility tag to label the issue (see an accessibility issue example in Fig 2). As such, we decided to use the general term "*accessibility*" to search data and then manually check whether the issues are actually relevant to accessibility.

In addition, for answering the RQs listed in Section II-A, we collected the relevant discussion threads in the selected GitHub repositories. In summary, we got two steps for data collection:

- **Step 2.1**: Collecting issues relevant to accessibility. As we discussed earlier, to identify the most suitable terms for capturing issues relevant to accessibility, we experimented with several terms within the pilot search, and then we decided to use the general search term "*accessibility*". We manually checked and confirmed the discussed accessibility issues.
- **Step 2.2**: Collecting whole threads discussed between developers of the collected accessibility issues in Step 2.1. The collected data items are shown in Table I.

TABLE I
EXTRACTED DATA ITEMS AND THEIR DATA ANALYSIS METHODS.

No.	Data item	Data analysis method	Relevant RQ
D1	Description of accessibility issues.	Constant Comparison and Descriptive Statistics.	RQ1
D2	Description of accessibility relevant discussions.	Constant Comparison	RQ2
D3	The tools, standards, and APIs for accessibility that discussed by developers.	Descriptive Statistics	RQ3
D4	Description of potential solutions of the accessibility issues.	Constant Comparison	RQ4
D5	The domain of the GitHub projects.	Descriptive Statistics	RQ1 and RQ4

To reduce personal bias in data collection, we invited another Ph.D. student for the formal data collection. The first author and the Ph.D. student separately collected data from 500 projects. They then double-checked each other's collected data (i.e., whether the issues are related to accessibility), and any disagreements were discussed and solved with the second author. We calculated the Cohen's Kappa coefficient [29] of the data collection between two inspectors, and the result achieved an agreement of 0.79. Finally, we collected 11,820 accessibility issues and their threads from the selected 1,000 popular repositories. Please note that, some GitHub projects do not include any accessibility issues and discussions, and we discuss the detailed results in Section IV.

C. Data Analysis

As shown in Table I, we used Descriptive Statistics and Constant Comparison methods to analyze the extracted data items [30]. Constant Comparative is a bottom-up open coding approach from Grounded Theory, and Constant Comparative is an inductive data coding process used for categorizing and comparing qualitative data for analysis purposes [31]. To support the coding and categorize the data, we employed a tool, MAXQDA[2], for qualitative data analysis.

We first performed a pilot data analysis exercise by randomly selecting 100 accessibility issues and their threads collected in Section II-B to mitigate personal bias in analysis and labeling data.

The formal data analysis comprises three steps: (1) open coding, executed by the first author and the invited Ph.D. student, split the data into separate parts, i.e., what accessibility issues are discussed, what the potential reasons cause accessibility issues, what tools, standards, and APIs are discussed, and what the solutions discussed. The two annotators separately label the accessibility issues and their threads of 500 projects, and they then checked each other's labeling results. Any disagreements on labeling accessibility issues were discussed and confirmed with the second author. (2) axial coding, executed by the first author and the Ph.D. student, was employed to identify categories generated in the open coding step. (3) reducing the personal bias in coding, we made a final reliability test and calculated Cohen's kappa reliability coefficient for data analysis between two annotators, and the mean value is 0.80 for all the RQs. The result indicates a strong agreement between the two annotators.

III. Results

We explain the results of four research questions that investigate accessibility issues from different perspectives in this section.

A. RQ1 - What accessibility issues have been discussed across GitHub projects?

We found that around 70% of selected projects (708 projects, see Table III) have discussed accessibility issues. As we discussed in Section II-B, when developers discuss accessibility relevant issues, they would label the issues with the "*Accessibility*" tag or use the term "*Accessibility*" in the Title/Summary to indicate. Besides, when crafting a title, the majority of the developers explained the situation and problems on accessibility issues that occurred (i.e., 71.2% of the collected accessibility issues), and some developers copy and paste the error message (i.e., 34.5% of the collected issues) in the description of the reported accessibility issues. However, we found that only a few developers provided details (e.g., release version) in the issue descriptions (i.e., 15.9% of the collected accessibility issues). In summary, based on the accessibility issues that we collected, most of the developers are able to report accessibility issues clearly. To answer RQ1, we applied open coding approach (i.e., Constant Comparison) to group the collected accessibility issues into seven categories, and we also presented some example issues for each category. Please note that an accessibility issue and their threads can include multiple categories; as such, the total percentages of all categories is greater than 100% (see Table II).

1. UI design - accessibility issues. This category ranks first in our data sample (i.e., 61.2%). In this category, developers often presented accessibility issues regarding UI design; for example, developers discuss colors, text sizes, and keyboard navigation issues. Accessibility in UI design leads to a better experience for all users, regardless of ability. This result is consistent with our previous work [13], i.e., in current software application development, accessibility has been incorporated somehow in a short-term goal and mostly focuses on UI design.

💬 "*After watching @mmatuzo's talk "Writing even more CSS with **accessibility** in mind", I wanted to suggest optional **smooth scroll**. I saw #24889 but it's kinda outdated, and with the support of prefers-reduced-motion it's more robust now. Very simple and **lightweight** PR so feel free to discard it if you think it's still not valuable.*"

💬 "*This led to unexpected results...This doesn't ensure that our dark **foreground color will cross the threshold**, neither than its contrast ratio with background will be better than white...*"

💬 "*I'm looking at ReDoc v2 (nice job by the way) but I'm running into problems where the interface isn't very accessible. Looking at the "Accessibility" tab in Firefox, it's mostly div and span tags with onClick events attached that are tripping me up. In order to be useful to more people (and for us to potentially adopt this at work)...*"

2. Accessibility improvement. This category ranks second in our data sample (i.e., 42.3%), and those issues are related to accessibility functionality improvement (e.g., adding voice assistant functionality). Accessibility is an essential consideration for any project. However, only a few projects consider accessibility at the beginning of the software design phase [32]. Many projects and developers have to improve accessibility elements and functionalities when the project requirements evolve.

💬 "*Improves **accessibility** on tables by adding captions documentation and scope="col" on thead #23755.*"

💬 "*I was wondering if we should change role="alert" to role="alertdialog" on dismissible alerts since W3C alert role specs read...This would mean wrapping the text with a div with an id and adding an aria-describedby which I don't quite like but on modals we have plenty of hooks to make it more **accessible**.*"

3. Overlapped with other quality attributes. This category ranks third in our data sample (i.e., 22.1%) that describes how accessibility interacts with other quality attributes, vice versa. We found that developers report and discuss accessibility issues along with other quality attributes, such as usability, performance, and user privacy. For example, developers report that accessibility services have been incorporated into Android to make devices running the mobile operating system easier to use for people with disabilities, but those services can provide a path for an attacker [33].

[2]https://www.maxqda.com/

TABLE II
A TYPOLOGY OF TOPICS ON ACCESSIBILITY ISSUES THAT DISCUSSED IN GITHUB PROJECTS.

Category	Excerpt title	Percentage
UI design - accessibility issues	*Title 1: "Bug: React roots are announced as clickable to screen readers." - screen reader* *Title 2: "DevTools: Check if accessibility regressions exist compared to old DevTools." - color for disabled* *Title 3: "Flow-root clearfix alternative proposal." - view focus* *Title 4: "Bootstrap Navigation: Radio button inside Label not activated." - navigation issue*	61.2%
Accessibility improvement	*Title 1: "Add a play/pause button; Focus the selected item; Disable prev/next controls when data-bs-wrap="false" and stopped at one end" - functionalities for accessibility improvement* *Title 2: "Improve accessibility for anchor tag." - accessibility improvement*	42.3%
Overlapped with other quality attributes	*Title 1: "Usability Testing? as a UX Researcher I get asked about Bootstrap user testing and general usability of elements or patterns." - usability level changes* *Title 2: "Remove the unnecessary newline and unused vars" - usability and performance* *Title 3: "Added the third parameter column for Table's row click event" - event addition*	22.1%
Solving accessibility relevant technical problems	*Title 1: "Significant change: upgrades calico/canal for security vulnerability" - updating issues* *Title 2: "Existing Calico user on clusters that were created prior to kops 1.8.0 need to be updated for the new "DefaultDeny" bSolving accessibility relevant technical problemsehavior for Kubernetes Network Policies" - hard coded issues* *Title 3: "Move kops-controller to use a yaml configuration file" - accessibility controller issue*	10.3%
Sharing knowledge for supporting accessibility	*Title 1: "QuickPick with Accessibility Support:on on type does not auto select first element if there is no string match" - accessibility knowledge sharing* *Title 2: "QuickPick with accessibility Support:on on type does not auto select first element if there is no string match." - accessibility knowledge sharing*	7.9%
Refactoring and reuse for incorporating accessibility	*Title: "I work as a Web Developer for accessibility and, to me, it felt as though there was a parent container with a WAI-ARIA role of "application" somewhere which was essentially stealing focus. There are other roles that engage Forms Mode like this, expecting the control to have coded all keyboard interactions itself." - refactoring for accessibility*	3.9%
Documentation update for accessibility	*Title 1: "Docs: improve/expand button documentation." - accessibility information documented* *Title 2*: "Lots of documentation have been polished" - *other software artifact update*	2.2%

💬 "*We rely on community feedback to make changes to components, which we usually get in larger quantities than we can handle as it is. We also have Patrick, who provides extensive* ***accessibility*** *and standards* ***usability*** *support. If you have any particular questions or suggestion, we'd be happy to hear them.*"

💬 "***Usability*** *Testing? as a UX Researcher I get asked about Bootstrap user testing and general usability of elements or patterns.*"

💬 "*Significant change: upgrades calico/canal for* ***security vulnerability***."

4. Solving accessibility relevant technical issues. This category ranks fourth in our data sample (i.e., 10.3%) that reports accessibility design and development technical issues. For example, developers discuss the coding or programming bugs regarding accessibility functionalities (e.g., hard coding issues).

💬 "*Button shadow is* ***hard coded.*** *We have 3 variables for button shadow: $btn-box-shadow$btn-focus-box-shadow btn-active-box-shadow. Issues here: If i set btn-box-shadow value to "none", button mixin generates incorrect box-shadow for :focus/.focus states.*"

💬 "*Existing Calico user on clusters that were created prior to kops 1.8.0 need to be updated for the new "DefaultDeny" bSolving* ***accessibility relevant technical*** *problemsehavior for Kubernetes Network Policies" - policy document update.*"

5. Sharing knowledge for supporting accessibility. This category ranks fifth in our data sample (i.e., 7.9%) that describes developers sharing resources (e.g., API and tools), knowledge, and their personal experience on accessibility development and design when having new functionalities for accessibility or fixing specific issues.

💬 "*This particular use is fairly non-standard, and* ***from an accessibility point of view this impedes adopting a "proper" tabbed interface pattern with correct role*** *"..." etc. As such, I'd like to get opinion from implementers about whether or not they find the current tab with dropdowns useful, or if dropdowns can be removed (which would then allow a proper ARIA tab implementation).*"

💬 "***QuickPick with Accessibility Support****: on type does not auto select first element if there is no string match.*"

6. Refactoring and reuse for incorporating accessibility. This category accounts for 3.9% of our data sample describing the techniques for refactoring projects or components to incorporate accessibility. Developers discuss the information about components or the software refactoring for accessibility functionalities. Accessibility requirements need to be separately designed for low coupling and reuse. In addition, refactoring is a downside when projects need to incorporate accessibility elements into a system, as many systems do not consider accessibility at the beginning.

💬 "*An* ***accessibility refactor*** *will improve your product and your code base more effectively than a code-only refactor. Refactoring as an essential technique to incrementally improve the accessibility and usability of a web.*"

💬 "*I work as a **Web Developer for accessibility** and, to me, it felt as though there was a parent container with a WAI-ARIA role of "application" somewhere which was essentially stealing focus. There are other roles that engage Forms Mode like this, expecting the control to have coded all keyboard interactions itself.*"

7. Documentation update for accessibility. This category ranks the least in our data sample (i.e., 2.2%). Essentially, in this category, developers discuss documentation for checklists about accessibility concerns and issues.

💬 "*Assorted **accessibility** (and some consistency) fixes for **documentation**.*"

💬 "*Update toast **documentation on accessibility,** maybe need a review for english because it's not my native langage. Also add a question about default toast timeout actually defined to 500ms maybe most of users will let it as is by copy/paste... don't know what's the usual timer on toaster but maybe increase it by default at 3seconds or 5seconds.*"

We then further analyzed what accessibility issues have been discussed across different project domains. As mentioned in Section II, we collected the domain information of GitHub projects during the data collection phase (i.e., D5 in Table I). GitHub does not include information about the domain of a project; however, According to Borges *et al.*, GitHub projects could be classified into six domains (i.e., *Application Software, System Software, Web Libraries and Frameworks, Non-web Libraries and Frameworks, Software Tool, and Documentation*) [34]. The first author and the invited Ph.D. student manually checked the domains of selected GitHub projects. As we mentioned in Section II-B, we did not include projects in the documentation domain in the data collection phase. We listed the top three accessibility issues in the five domains in Table III. The results show that **UI design relevant accessibility** issues are most frequently discussed in *Software Application* domain. A potential reason is that projects in this domain provide a range of functionalities for end-users (e.g., Google Docs), which need to consider accessibility elements in their applications. In contrast, in *System Software* and *Non-web Libraries and Framework* domains, the most frequently discussed accessibility issues are related to **Refactoring and reuse for incorporating accessibility**. Reasons could be the most users of those projects tend to be developers, and the projects are mostly at lower design (e.g., provide frameworks for developers).

***Summary for RQ1**: Accessibility issues are widely concerned in GitHub projects, and in our data set, around 70% of selected projects have reported accessibility issues. Most accessibility issues are related to **UI design** (i.e., color usage and layout navigation). This is especially true in the **Application Software** domain(i.e., 72.7% of accessibility issues are related to **UI design**).*

TABLE III
ACCESSIBILITY ISSUE CATEGORIES REPORTED IN DIFFERENT PROJECT DOMAINS. WE REPORTED THE TOP THREE DISCUSSED ACCESSIBILITY ISSUES IN THE FIVE DOMAINS OF GITHUB PROJECTS.

Project domain	**Top three accessibility issue categories**
Application Software (312 projects)	**UI design** - 52.7% Improvement for accessibility - 35.9% Sharing knowledge for supporting accessibility - 11.4%
System Software (129 projects)	**Refactoring and reuse for incorporating accessibility** - 34.2% Overlapped with other quality - 34.5% Sharing knowledge for supporting accessibility - 31.3%
Web-libraries and Framework (102 projects)	**Sharing knowledge for supporting accessibility** - 65.7% Overlapped with other quality - 23.4% Sharing knowledge for supporting accessibility - 10.9%
Non-web Libraries and Framework (87 projects)	**Refactoring and reuse for incorporating accessibility fixed** - 54.1% Sharing knowledge for supporting accessibility features - 23.1% Overlapped with other quality attributes - 22.8%
Software Tools (76 projects)	**Solving accessibility relevant technical issues** - 65.7% Sharing knowledge for supporting accessibility - 26.3% Overlapped with other quality attributes - 8.0%

B. RQ2 - What are the causes of reporting accessibility issues from the perspective of developers?

As we discussed in Section II, we applied the Constant Comparison method to analyze the potential reasons why developer report accessibility issues (i.e., why developer report them), and we got the four reasons and named them as *No awareness of accessibility*, *Experience accessibility problems*, *Perceived negative consequences*, and *Development practice causing accessibility issues* (see Table IV).

No awareness of accessibility is the most frequent reason that is incurring accessibility issues in our data sample. We found that 85% of the accessibility issues occur because of developers' lack of accessibility relevant knowledge and are unaware of accessibility at the beginning of project development, and accessibility issues often happen in the design and implementation phases. In addition, some developers would share links about accessibility standards, knowledge, or tools for facilitating accessibility design to inform other developers in the threads. Official forums of accessibility design, research papers, and question and answer websites are the top three popular sources that developers pointed to for sharing accessibility knowledge.

💬 "*We **did not notice** that color is not suitable in that chat box, might be it's not **accessible** for users.*"

Experience accessibility issues. 56% of the reported accessibility issues due to projects are experiencing accessibility problems in a number of ways, e.g., their projects need to fix some specific accessibility UI design issues or need to refactor for incorporating accessibility design elements.

💬 "*The overall **accessibility** of any project built with Bootstrap depends in large part on the author's markup, additional styling, and scripting they've included. However, **provided that these have been implemented correctly**, it should be perfectly possible to create websites and applications with Bootstrap that fulfill.*"

Perceived negative consequences. Around 32% of accessibility issues were reported due to this reason in our data sample. Developers perceive and report that some potential accessibility issues would be barriers if the system project evolves (e.g., introducing new functionalities) or when they need to maintain the system.

💬 "*The overall **accessibility** of any project built with Bootstrap depends in large part on the author's markup, additional styling, and scripting they've included. However, **provided that these have been implemented correctly**, it should be perfectly possible to create websites and applications with Bootstrap that fulfill.*"

💬 "*We set our focus **styles using shadows**, and since we couldn't ensure anyone **disabling** those shadows would set proper focus styles, we don't provide a way to change this. Focus styles are mandatory, using box-shadow allows more theming than keeping outline.*"

Development practice causing accessibility issues. We found that architecture patterns and design patterns could cause around 28% of reported accessibility issues. Many projects fail to take accessibility into account initially, and many projects start with achieving some level of accessibility in the later design (i.e., accessibility is postponed to once projects are built). However, at the end of the project, time starts to decrease rapidly, and resources start to be assigned to other priorities. In addition, at the end of the project, accessibility is dropped or postponed for a later version.

💬 "*Hey there, as a UX Researcher I get asked about Bootstrap user testing and general accessibility of elements or patterns. Our design system leverages Bootstrap and we typically test the **elements and patterns that we customize**, so I wanted to reach out to see how various things make their way into a new release or recommendation?* "

> ***Summary for RQ2***: *Four reasons potentially cause accessibility issues that are identified in our data sample, and one most common reason is that developers lack accessibility knowledge (e.g., accessibility-related standard and commonly used tools) or are not aware of accessibility in the first place of project development.*

C. RQ3 - What tools, standards, and APIs are applied to address accessibility issues in practice?

To answer RQ3, we collected the tools, standards, and APIs employed by developers to address accessibility issues. We listed the top five most-discussed items and their counts in Table V.

Accessibility Standards

TABLE IV
ANALYSIS ON CAUSES OF ACCESSIBILITY ISSUES.

No.	Cause	Type
No awareness of accessibility		
1	Lacking of relevant accessibility knowledge.	Non-technical
2	Not fully understand requirement changes related to accessibility.	Non-technical.
3	Accessibility design is missing in the early stage.	Non-technical
4	No proper tools or checklists for accessibility design.	Non-technical
Experience accessibility issues		
5	Running through hoops of accessibility development.	Technical
6	Accessibility relevant bugs.	Technical
Perceived negative consequences		
7	Increasing accessibility for more users.	Non-technical
8	Poor accessibility would hurt development team reputation.	Non-technical
9	Involving bugs if accessibility is ignored.	Technical
10	Increasing more efforts or time to incorporating accessibility in the next step work.	Technical
Development practice causing accessibility issues		
11	Reported by external entity.	Technical
12	Development team mandates accessibility into project development.	Technical

There exist an overwhelming amount of accessibility standards. However, it is unclear whether developers employ them for accessibility design in practice. We identified the top five most-discussed accessibility standards. The World Wide Web Consortium (i.e., W3C) issues guidelines for accessing web content, and Web Content Accessibility Guidelines (WCAG) are most frequently discussed and used the standards (i.e., accounting for 86% of all discussed accessibility standards).

💬 "*Differentiate between the two **WCAG** links (minimum and enhanced). The two terminal.integrated.minimumContrastRatio example values 4.5 and 7 have the same text and link text but corresponds to two different levels of compliance (4.5 minimum, 7 enhanced). The description should call out the differences and have unique link text.*"

Accessibility Tools

There is a wide range of tools to support checking accessibility during development and assessment, which come in varying formats and varying purposes. We collected the top five most frequently mentioned tools, which support accessibility development and design. Some tools support checking single accessibility issues (e.g., Accessibility View), and other tools can provide a range of accessibility issues at one time (e.g., Accessibility Checker). In addition, some automated tools can be set to analyze the code for specific issues rapidly and generate reports on results (e.g., A11Y).

💬 "*Verify and fix our a11y support using accessibility checker verification. Steps to Reproduce: Launch Visual Studio Code Launch Accessibility Checker, select all verification routines, and click Run Verifications Check the result.*"

TABLE V
STANDARDS, TOOLS, AND APIS THAT DEVELOPERS APPLIED FOR ADDRESSING ACCESSIBILITY DESIGN.

No.	Name	Count
Accessibility standards		
1	W3C	654
2	WCAG	231
3	Authoring Tool Accessibility Guidelines (ATAG)	67
4	Accessibility Regulation	43
5	Disability Discrimination Act	29
Accessibility tool		
1	Accessibility Checker	102
2	Accessibility Checklist	65
3	Accessibility View	49
4	A11Y	34
5	Accessibility Scanner	12
Accessibility APIs		
1	MSAA/IAccessible (Windows)	89
2	Acceesibility Framework (Android)	71
3	NSAccessibility (Mac OS)	70
4	IAccessible2 (Windows)	54
5	UIAccessibility (IOS)	39

Accessibility APIs.

One of the key assisting technologies is accessibility API. Accessibility is a hard technical challenge, and a firm grasp of the technology is paramount to making informed decisions about accessible design. We listed the top five most-discussed APIs for accessibility design. The platform accessibility API, Microsoft Active Accessibility (MSAA), was made available in a 1997 update to Windows 95. MSAA provided information about the role and state of objects and some of their properties. Other discussed APIs are for different platforms, for example, Accessibility Framework is developed for Android, and UIAccessibility is designed for IOS operation system.

💬 "*vscode api: introduce accessibilityInformation. This PR is not yet tested. I might have missed some step in the endless conversion of objects. Alternative name to AccessibilityInformation is just Accessibility, but for some reason I prefer the longer version.*"

Summary for RQ3*: We identified the most discussed standards, tools, and APIs for supporting accessibility design and development, and such list could help developers, especially those who do not have much accessibility background knowledge, choose and take advantage of them better.*

D. RQ4 - What are the practical recommendations for developers to handle and address accessibility issues?

To answer RQ4, we first calculated the averaged thread length (i.e., the number of discussions per thread) for the categories listed in Table II. For the discussion (i.e., thread) length, there was a statistically significant difference between categories. **UI design - accessibility issues** and **Solving accessibility relevant technical issues** had the discussion that yielded the most active discussion (i.e., mean length is 6.7), with mean and median thread length much higher than the other categories in our data set. In contrast, the category **Sharing knowledge for supporting accessibility** had the shortest discussion per thread (mean length is 2.3) in our data set. The potential reason is that **UI design - accessibility issues** and **Solving accessibility relevant technical issues** topics potentially require developers to spend a longer time to fix and require certain knowledge. We then provided three general recommendations discussed by developers for addressing accessibility issues (i.e., the accessibility relevant threads). The recommendations are not meant to exhaustive. As we discussed in the rationale of RQ4, the ways to design and develop accessibility can vary from project to project. Therefore, there may be additional guidelines to address accessibility issues.

1. Considering accessibility at earlier phases. Organizational factors tend to impact accessibility development and design in practice; for example, developers lack knowledge on accessibility or agile development. Relatively larger development teams tend to have more resources for addressing accessibility issues and often prioritize them. In contrast, small-size projects find it more challenging to address accessibility, including lack of expertise, tight development schedule, or lack of support from management. However, through the discussion "Sharing knowledge on accessibility", we found that developers discuss that small steps of accessibility design can make a big difference. All projects need to improve their practice to handle and address accessibility during software design and development. Based on analysis of accessibility relevant threads, we listed several recommendations for developers to be aware of accessibility when they design and develop projects: (1) Resource and guidelines are essential for developers to understand common accessibility practice; (2) Having a checklist for accessibility; and (3) Evaluating accessibility internal the development team.

💬 "*[Accessibility] tree items should be visible in 400% zoom, Steps to Reproduce: set to screen display resolution to 1280 x 1024 .... some items in the tree view are not completely visible. Low Vision users will not be able to see what is on the screen.*"

2. Communicating inside the development team. The second recommendation is that reinforcing communication between development teams for addressing accessibility issues. From the very beginning of development, ensure accessibility is a requirement of the projects, and all team members reach an agreement on accessibility design and development. For example, before making any decisions about "Accessibility": (1) All developers in a project should reach a consensus on accessibility-relevant design elements; (2) What the exact accessibility requirements are; (3) How to address these accessibility-related requirements; (4) Put accessibility requirements as a high priority to address; (5) When making decisions about accessibility, some design decisions may be invalid in the first place, and some trade-offs might exist between design decisions; and (6) Accessibility requirements need to be well-documented.

💬 "*I had a look a few weeks ago but I'm not a frontender and it wasn't clear to me how to build a bleeding-edge or other-branch version of the thing I include in my html page to render specs (or is the CLI ready enough for an unskilled* *person to use at this point? ... **let me know if we should track this in a separate issue or other location for discussion about accessibility**?"*

3. Applying practical standards and tools for supporting accessibility. Some standards and guidelines have been widely discussed and applied. For example, (1) the Web Content Accessibility Guidelines (WCAG) Developers would take advantage of those standards and tools for self-assessment in terms of accessibility. Although technology evolves quickly and the standards needed to catch up to ensure that people or disabilities can access today's systems, those standards and tools still valuable for developers who lack accessibility knowledge. In addition, correctly reporting problematic accessibility requirements is recommended; (2) Have an iterative process to fit accessibility into the whole life cycle; and (3) Use a proper approach (e.g., the accessibility framework) and appropriate supporting tools or APIs;

💬 "*Setting sync: Sign in **dialogs not accessibility friendly**. This dialog is very unnacessible. Problems: Once the dialog is created the "Please sign in to synchronize" part is not being read out by screen reader. I recommend to use a placeholder since that would nicely be read.*"

***Summary for RQ4**: We provided three general recommendations for addressing accessibility issues in practice; In addition, we recommend that development teams consider accessibility in the first place during the project development.*

IV. Discussion

We reflect on our findings of research questions in this section. We also highlight the avenues of future research on accessibility for researchers and practitioners.

Be aware of a variety of accessibility issues. Based on the results of RQ1 (see Section), we found that a set of different accessibility issues widely discussed by developers. Like most non-functional quality attributes, software accessibility should begin as early as possible in the development process. Design flaws often are the root cause of software accessibility issues; these flaws can be difficult and expensive to remedy late in the development process. In addition, accessibility is often overlapped with other quality attributes, for example, usability, security, and privacy. The frequent occurrence of accessibility issues in software development may result in software refactoring with more effort. As such, we advocate that accessibility could be treated as a functional requirement to ensure people use software applications more efficiently. Such accessibility design drives innovation in general products, and it can be more accessible for disabled people. This could motivate researchers to spend extra efforts on accessibility management, design, and development.

Documenting accessibility issues. The identified accessibility issues are discussed and reported along with other defects of projects, such as configuration and testing bugs. The impact of an isolated occurrence of an accessibility issue is not very high, but together these issues create a broad and consistent impact for an entire project or the majority of primary functions, effectively blocking the application usage. We encourage that developers pay attention to accessibility and document accessibility issues in more detail and timely.

Applying standards, tools, and APIs for accessibility. Although many standards and guidelines that have been widely introduced in the literature, based on the results of RQ3, only several standards, tools, and other APIs are discussed between developers frequently. Those standards and tools have been used for decades; for example, the Web Content Accessibility Guidelines (WCAG) have been a part of digital accessibility since 1999. Technology evolves quickly, and the standards needed to catch up to ensure that people or disabilities can access today's systems. The accessibility community (e.g., Mobile Application development) has been eagerly awaiting the release of new standards. As such, we encourage researchers to craft new success and comprehensive criteria based on different accessibility levels.

Evaluation strategies for accessibility. Based on the results of RQ2 and RQ4, we summarized four reasons that potentially cause accessibility issues; some developers perceive, identify, and raise accessibility issues when developing projects. Those accessibility issues are vital for the project's continuous development and success. As such, three general solutions could be taken for accessibility development and design. Firstly, engaging communication with different stakeholders of projects. The different classes of different ability constitute the usage types, and different requirements are depending on them. In the evaluation, representatives for each group provide feedback on whether the product supports their specific disability type. Secondly, to apply practical standards and tools to supporting accessibility design and development. In addition, our results also suggest that accessibility needs to be concerned in the early stage of development and should be included in all phases of software development.

V. Threats to Validity

The threats to the validity of this study are discussed by following the guidelines proposed by Wohlin *et al.* [35]. Internal validity is not considered since this study does not address any causal relationships between variables.

Construct validity refers to whether the theoretical and conceptual constructs are correctly interpreted and measured. In this study, there are two key threats. The first one is the search process related to data collection of accessibility issues. To mitigate this threat, we leveraged the search function provided by GitHub to collect the top 1,000 popular (i.e., by star rating) projects. It is possible that this filtering omitted accessibility issues of both small projects and outside data sources. More accessibility issues from various size projects and information from external data sources are required to be included for a better and more comprehensive analysis. We leave this as future work. The second threat lies in the process of manually extracting and analyzing the collected data. To partially mitigate this threat, we did a pilot execution of data filtering, extraction, and coding by the first two inspectors to agree about all the data used

External validity concerns the generality of the study results in other settings. This depends on the sampling methods we employed and the representativeness of the data used. A threat is that if the accessibility issues we collected are representative. To reduce this threat, we used a large number of representative GitHub repositories and extracted a large number of accessibility issues to analyze. Another threat exists is that if the keyword for searching and collecting accessibility was suitable. We reviewed the accessibility relevant literature papers, and conducted a pilot search to decide the suitable search terms to derive our search string. Further studies with more accessibility discussions in other resources, for example, Stack Overflow, would need to be done to demonstrate the generality of our results.

Reliability refers to whether the study yields the same results if other researchers replicate this study, which is related to data collection and analysis in GitHub projects. To alleviate this threat, we made explicit the process of our study design, and we believe that this work can be replicated.

VI. Related Work

This section introduces the related works from two aspects, i.e., accessibility development and design and empirical studies of mining repositories.

A. Accessibility Development and Design

Recently, studies have investigated accessibility issues in mobile apps. For example, Alshayban *et al.* [12] presented a large-scale empirical study, which aimed at understanding accessibility issues of Android apps from three complementary perspectives. The authors analyzed the prevalence of accessibility issues in over 1,000 Android apps. The authors then investigated the developer sentiments through a survey, which aimed at understanding the root causes of accessibility issues. The authors presented the findings of a survey involving 66 practitioners. The results of the survey show that developers are generally unaware of the accessibility principles, and the existing analysis tools are not sufficiently sophisticated to be used. Finally, the authors investigated the user ratings and comments on app stores. Our study is similar to their work, but we not only focus on accessibility issues in Android apps but also on other projects (e.g., web applications). In addition, we provide empirical evidence that shows what accessibility issues have been discussed across different software domains. We also provide a set of recommendations for addressing and handling accessibility issues. Our previous work *et al.* [13] conducted 15 interviews and an online survey to investigate how developers perceive accessibility and how accessibility incorporates into general projects. In that work, we also reported the critical challenges and benefits of incorporating accessibility into software development and design.

Paiva *et al.* [14] conducted a literature review on accessibility in software engineering processes. The authors presented 94 relevant publications from 2011 to 2019. Their study showed the distributions of publications on different phases of the software development life cycle, mainly focus on design and testing phases. In addition, the results of their study show that most of the studies focus on complete or partial visual impairment in terms of accessibility development, and few papers discuss other disabilities, such as hearing and cognitive disabilities. The authors found challenges discussed in the literature for addressing accessibility in software development; for example, studies focus on requirements and vocabulary, and the actual demand is to evolve from designing development and test of accessible games considering different disabilities. Their work presented the incorporation of accessibility to agile methodologies and open source development. This idea motivated us to investigate accessibility issues in OSS projects and potential solutions discussed by developers to address accessibility issues. Bai *et al.* [36] presented an evaluation of nine accessibility testing methods that fit in the agile software process, and during the evaluation, the authors have investigated different accessibility testing. The authors also discussed the benefit and cost of the accessibility software development process and recommended methods of applying accessibility testing. Nganji *et al.* proposed a disability-aware software engineering process model that considers the needs of people with disabilities. The model can be used for improving accessibility and usability of the designed system. Shinohara *et al.* [37] conducted a survey to investigate the prevalence of higher education teaching about accessibility and faculties' opinions on the barriers to teaching accessibility. The results show that teaching accessibility is prevalent but shallow among the U.S. faculty as broad. The authors also reported that the most critical barrier to clear and discipline-specific accessibility learning objectives is the lack of faculty knowledge about accessibility.

B. Mining and Analyzing Developers' Discussions in GitHub

Millions of software developers and GitHub repositories are active, and developers provide an abundance of valuable discussions about different development aspects. Such knowledge has been extensively utilized to conduct studies in the field of software engineering. For example, Rahman *et al.* [38] provided a comparative study between successful and unsuccessful pull requests in 78 GitHub projects. The authors analyzed pull request discussion texts (e.g., domain and maturity). Eight topics are labeled discussed in the texts of pull request discussion. The results of their study would help developers overcome the issues with pull requests in GitHub and project administrators with informed decision making.

Tsay *et al.* [39] presented a study of how developers in open work environments evaluate and discuss pull requests. The authors also conducted a set of interviews with GitHub project developers about pull requests. Their study got several conclusions; for example, the authors found that when developers raised issues in pull requests, either the problem was attempting to solve or the solutions implemented, and full requests provide an occasion to discuss alternative solutions or negotiate requirements. In addition, the results of their study also inform the design of notification and discussion mechanisms for large-scale collaboration where a wide variety of stakeholders participate in evaluation discussion around code contributions. Casalnuovo *et al.* [40] investigated how well do people work together in GitHub teams, and the authors explored the evidence for socialization as a precursor to joining a project. The results of their study show that the presence of past social connections combined with prior experience languages dominate in the project leads to higher productivity both initially and cumulatively.

VII. Conclusion

In this study, we collected 11,820 accessibility issues that developers discussed in popular GitHub projects, and we grouped the accessibility issues into seven topics. We have provided empirical evidence to confirm that accessibility issues are frequently discussed between developers in practice, and we analyzed the reasons that potentially cause accessibility issues and provided general recommendations for addressing accessibility issues. In this paper, we also provided practical lessons about applying standards, tools, and APIs to ensure and facilitate accessibility design and development.

Further studies could put more effort into investigating the different accessibility design elements in different domains and generalizing existing tools, standards, or APIs, which could be helpful for developers in accessibility design, development, and improvement.

VIII. Acknowledgment

The authors would like to thank John Grundy for his great help on this work.

References

[1] J. T. Nganji and S. H. Nggada, "Disability-aware software engineering for improved system accessibility and usability," *International Journal of Software Engineering and Its Applications*, vol. 5, no. 3, pp. 47–62, 2011.

[2] S. Sanchez-Gordon, M. Sánchez-Gordón, M. Yilmaz, and R. V. O'Connor, "Integration of accessibility design patterns with the software implementation process of iso/iec 29110," *Journal of Software: Evolution and Process*, vol. 31, no. 1, p. e1987, 2019.

[3] P. Acosta-Vargas, L. Salvador-Ullauri, J. Jadán-Guerrero, C. Guevara, S. Sanchez-Gordon, T. Calle-Jimenez, P. Lara-Alvarez, A. Medina, and I. L. Nunes, "Accessibility assessment in mobile applications for android," in *International Conference on Applied Human Factors and Ergonomics*. Springer, 2019, pp. 279–288.

[4] D. Hoffman, E. Grivel, and L. Battle, "Designing software architectures to facilitate accessible web applications," *IBM Systems Journal*, vol. 44, no. 3, pp. 467–483, 2005.

[5] T. Jokela, N. Iivari, J. Matero, and M. Karukka, "The standard of user-centered design and the standard definition of usability: analyzing iso 13407 against iso 9241-11," in *Proceedings of the Latin American conference on Human-computer interaction*, 2003, pp. 53–60.

[6] C. Siebra, M. Anjos, F. Florentin, T. Gouveia, A. Filho, W. Correia, M. Penha, F. Q. Silva, and A. L. Santos, "Accessibility devices for mobile interfaces extensions: A survey," in *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct*, 2015, pp. 644–651.

[7] M. Elias, S. Lohmann, and S. Auer, "Fostering accessibility of opencourseware with semantic technologies–a literature review," in *International Conference on Knowledge Engineering and the Semantic Web*. Springer, 2016, pp. 241–256.

[8] C. M. Baker, Y. N. El-Glaly, and K. Shinohara, "A systematic analysis of accessibility in computing education research," in *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, 2020, pp. 107–113.

[9] J. Lazar, D. F. Goldstein, and A. Taylor, *Ensuring digital accessibility through process and policy*. Morgan kaufmann, 2015.

[10] D. Buhalis and E. Michopoulou, "Information-enabled tourism destination marketing: addressing the accessibility market," *Current Issues in Tourism*, vol. 14, no. 2, pp. 145–168, 2011.

[11] A. Holzinger, G. Searle, and M. Wernbacher, "The effect of previous exposure to technology on acceptance and its importance in usability and accessibility engineering," *Universal Access in the Information Society*, vol. 10, no. 3, pp. 245–260, 2011.

[12] A. Alshayban, I. Ahmed, and S. Malek, "Accessibility issues in android apps: state of affairs, sentiments, and ways forward," in *2020 IEEE/ACM 42nd International Conference on Software Engineering (ICSE)*. IEEE, 2020, pp. 1323–1334.

[13] T. Bi, X. Xia, D. Lo, J. Grundy, T. Zimmermann, and D. Ford, "Accessibility in software practice: A practitioner's perspective," *arXiv preprint arXiv:2103.08778*, 2021.

[14] D. M. B. Paiva, A. P. Freire, and R. P. de Mattos Fortes, "Accessibility and software engineering processes: A systematic literature review," *Journal of Systems and Software*, p. 110819, 2020.

[15] J. Gunderson, "W3c user agent accessibility guidelines 1.0 for graphical web browsers," *Universal Access in the Information Society*, vol. 3, no. 1, pp. 38–47, 2004.

[16] S. Harper and A. Q. Chen, "Web accessibility guidelines," *World Wide Web*, vol. 15, no. 1, pp. 61–88, 2012.

[17] M. Vigo, A. Kobsa, M. Arrue, and J. Abascal, "User-tailored web accessibility evaluations," in *Proceedings of the eighteenth conference on Hypertext and hypermedia*, 2007, pp. 95–104.

[18] J. Han, S. Deng, X. Xia, D. Wang, and J. Yin, "Characterization and prediction of popular projects on github," in *2019 IEEE 43rd annual computer software and applications conference (COMPSAC)*, vol. 1. IEEE, 2019, pp. 21–26.

[19] D. Arya, W. Wang, J. L. Guo, and J. Cheng, "Analysis and detection of information types of open source software issue discussions," in *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)*. IEEE, 2019, pp. 454–464.

[20] S. Easterbrook, J. Singer, M.-A. Storey, and D. Damian, "Selecting empirical methods for software engineering research," in *Guide to advanced empirical software engineering*. Springer, 2008, pp. 285–311.

[21] G. Rodriguez, J. Pérez, S. Cueva, and R. Torres, "A framework for improving web accessibility and usability of open course ware sites," *Computers & education*, vol. 109, pp. 197–215, 2017.

[22] R. F. de Oliveira, A. M. da Mota Moura, and J. C. S. P. Leite, "Reengineering for accessibility: A strategy based on software awareness," in *Proceedings of the 17th Brazilian Symposium on Software Quality*, 2018, pp. 180–189.

[23] N. Kesswani and S. Kumar, "Accessibility analysis of websites of educational institutions," *Perspectives in Science*, vol. 8, pp. 210–212, 2016.

[24] T. Bi, P. Liang, and A. Tang, "Architecture patterns, quality attributes, and design contexts: How developers design with them," in *2018 25th Asia-Pacific Software Engineering Conference (APSEC)*. IEEE, 2018, pp. 49–58.

[25] C. Putnam, M. Dahman, E. Rose, J. Cheng, and G. Bradford, "Best practices for teaching accessibility in university classrooms: cultivating awareness, understanding, and appreciation for diverse users," *ACM Transactions on Accessible Computing (TACCESS)*, vol. 8, no. 4, pp. 1–26, 2016.

[26] O. Jarczyk, B. Gruszka, S. Jaroszewicz, L. Bukowski, and A. Wierzbicki, "Github projects. quality analysis of open-source software," in *International Conference on Social Informatics*. Springer, 2014, pp. 80–94.

[27] T. Bi, X. Xia, D. Lo, J. Grundy, and T. Zimmermann, "An empirical study of release note production and usage in practice," *IEEE Transactions on Software Engineering*, 2020.

[28] T. F. Bissyandé, D. Lo, L. Jiang, L. Réveillere, J. Klein, and Y. Le 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)*. IEEE, 2013, pp. 188–197.

[29] J. Cohen, "A coefficient of agreement for nominal scales," *Educational and psychological measurement*, vol. 20, no. 1, pp. 37–46, 1960.

[30] S. Adolph, W. Hall, and P. Kruchten, "Using grounded theory to study the experience of software development," *Empirical Software Engineering*, vol. 16, no. 4, pp. 487–513, 2011.

[31] D. Walker and F. Myrick, "Grounded theory: An exploration of process and procedure," *Qualitative health research*, vol. 16, no. 4, pp. 547–559, 2006.

[32] M.-L. Sánchez-Gordón and L. Moreno, "Toward an integration of web accessibility into testing processes," *Procedia Computer Science*, vol. 27, pp. 281–291, 2014.

[33] R. Ismailova, "Web site accessibility, usability and security: a survey of government web sites in kyrgyz republic," *Universal Access in the Information Society*, vol. 16, no. 1, pp. 257–264, 2017.

[34] H. Borges, A. Hora, and M. T. Valente, "Understanding the factors that impact the popularity of github repositories," in *2016 IEEE International Conference on Software Maintenance and Evolution (ICSME)*. IEEE, 2016, pp. 334–344.

[35] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in software engineering*. Springer Science & Business Media, 2012.

[36] A. Bai, H. C. Mork, and V. Stray, "A cost-benefit analysis of accessibility testing in agile software development: results from a multiple case study," *International Journal on Advances in Software*, vol. 10, no. 1&2, pp. 96–107, 2017.

[37] K. Shinohara, S. Kawas, A. J. Ko, and R. E. Ladner, "Who teaches accessibility? a survey of us computing faculty," in *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, 2018, pp. 197–202.

[38] M. M. Rahman and C. K. Roy, "An insight into the pull requests of github," in *Proceedings of the 11th Working Conference on Mining Software Repositories*, 2014, pp. 364–367.

[39] J. Tsay, L. Dabbish, and J. Herbsleb, "Let's talk about it: evaluating contributions through discussion in github," in *Proceedings of the 22nd ACM SIGSOFT international symposium on foundations of software engineering*, 2014, pp. 144–154.

[40] 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*, 2015, pp. 817–828.