Accessibility Evaluation of IoT Android Mobile Companion Apps

Faiza Tazi
Suleiman Saka
Faiza.Tazi@du.edu
Suleiman.Saka@du.edu
University of Denver
Denver, Colorado, USA

Griffin Opp
griffin.opp@colostate.edu
Colorado State University
Fort Collins, Colorado, USA

Shradha Neupane
sneupane@wpi.edu
Worcester Polytechnic Institute
Worcester, Massachusetts, USA

Sanchari Das
sanchari.das@du.edu
University of Denver
Denver, Colorado, USA

Lorenzo De Carli
lorenzo.decarli@ucalgary.ca
University of Calgary
Calgary, Alberta, Canada

Indrakshi Ray
indrakshi.ray@colostate.edu
Colorado State University
Fort Collins, Colorado, USA

ABSTRACT

Internet of Things (IoT) devices use mobile companion apps to configure, update, and proxy communications between devices, cloud endpoints, and users. However, to the best of our knowledge, their accessibility characteristics have received little study. Thus, we report the analysis results of 248 IoT companion apps. Our approach involves manual analysis based on the Accessibility Insights tool and reports on: the presence of contextual information (descriptions of controls, images, and text input), size of touch elements, and color contrast. Our primary findings are: (i) most apps have reasonable accessibility posture, but there exists a long tail of apps with significant problems, (ii) only two apps do not present any accessibility errors, and (iii) nearly 87% of apps in the corpus exhibit errors involving a lack of names and descriptions of elements and/or images. We further provide actionable recommendations to enhance the accessibility posture of the IoT android apps.

CCS CONCEPTS

• **Human-centered computing → Accessibility**; **Accessibility systems and tools**; **Accessibility design and evaluation methods**.

KEYWORDS

Internet of Things, IoT Devices, Mobile Apps, Accessibility Evaluation.

ACM Reference Format:
Faiza Tazi, Suleiman Saka, Griffin Opp, Shradha Neupane, Sanchari Das, Lorenzo De Carli, and Indrakshi Ray. 2023. Accessibility Evaluation of IoT Android Mobile Companion Apps. In *Extended Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems (CHI EA '23), April 23–28, 2023, Hamburg, Germany*. ACM, Hamburg, Germany, 7 pages. https://doi.org/10.1145/3544549.3585652

CHI EA '23, April 23–28, 2023, Hamburg, Germany

ACM ISBN 978-1-4503-9422-2/23/04.
https://doi.org/10.1145/3544549.3585652

1 INTRODUCTION

According to the center for disease control and prevention, 26% of adult Americans live with a disability [13]. The unique needs of these people and their right to participate and have access to different technologies cannot be overlooked [12]. For the disabled population, reduced physical mobility, or reduced vision, for example, can cause problems when it comes to daily tasks [9, 18]. These disabilities make it difficult and sometimes impossible to perform regular everyday tasks [25, 28, 36]. Previous work has underscored the potential for Internet of Things (IoT) devices to assist disabled people in their daily activities [16, 23]. Among other things, IoT can be important enablers of functionality such as health monitoring [7], home automation [33], and online education [21]. In order to realize these promises, the interfaces between IoTs and their users must be designed in a manner that makes the devices accessible to diverse users, including the disabled population. Accessibility implies technologies are designed and developed for everyone to use. Particularly, for users to be able to: identify, understand, interact, and contribute to these technologies [2].

To delve further into this, we perform a large-scale review of accessibility characteristics of mobile companion apps. We believe gaining this understanding is relevant, as severe User Interface (UI) accessibility issues have a broad impact, effectively gating the use of the technology for those who require accessibility accommodations. Performing such an assessment presents various challenges. Our process involves manual analysis of apps (248) within an Android emulator, using the industry-standard *Accessibility Insights* tool. While time-consuming and cumbersome, this approach enabled us to exhaustively collect issues falling within multiple categories: lack of names/descriptions for UI elements and images, lack of accessible text input content, size of UI elements, and color contrast. The results of our work, summarized in Section 4, reveal that on average the rate of errors per app is low (2.01 errors per app per page). However, most app exhibit accessibility errors (only 2 apps were error-free); and there is a long tail of apps with significant issues (almost 23% of the apps analyzed have 2 or more accessibility errors per page).

2 RELATED WORK

IoT research has focused primarily on the technical component of the implementation [14, 15, 19, 30, 31, 34]. Researchers have also worked on accessibility issues with regards to digital interaction including that of mobile apps [5, 26]. However, very few have focused on the accessibility issues of the IoT device mobile companion apps.

Sohaib et al. identified and discussed how IoTs could improve the buying experience for people with disabilities in the context of e-commerce [29]. Specifically, the work focuses on modernizing systems by removing IoT accessibility issues. In regards to overall usability, Da Silva et al. proposed UXmood, a tool that performs sentiment analysis to evaluate usability [8]. UXmood compiles a combination of video, audio, and interaction logs, to evaluate usability. Bakiu and Guzman proposed to analyze user reviews to examine usability and user experience [3]. A similar approach was adopted by Tan et al. to evaluate the usability of disaster apps [32]. Billi et al. [6] proposed a general methodology for accessibility/usability analysis, however their method is task-based and difficult to apply to a large-scale study.

Abou-Zahra et al. motivate the need for accessibility in the context of the Web of Things (WoT) advocated by the World Wide Web Consortium [1]. WoT aims to make IoT platforms and apps more interoperable and secure. The authors provide examples illustrating the problems of IoT concerning people with disabilities. The paper has made the community aware of the need to address accessibility issues pertaining to IoT apps and environments. Along these lines, De Oliveira et al. [10] has analyzed six IoT apps using the Web Content Accessibility Guidelines (WCAG) 2.0 standards. Specifically, they investigate 11 specific categories, including *Non-text content*, *Sensory Characteristics*, and *Heading and Labels.* To add to this area of research, we investigate five broad categories, including *ActiveViewName*, *ImageViewName*, *TouchSizeWzag*, *EditTextValue*, and *ColorContrast.* We analyze 248 IoT apps and provide a methodology for selecting these apps. The work closer in spirit to ours is that of Ross et al. [27], who in 2018 analyzed 5753 apps, however their app dataset is older and non-IoT specific. Balaji and Kuppusamy analyzed the accessibility of e-governance mobile apps in India [4], an app domain which is orthogonal to ours. Finally, this work complements our past analysis of the security/privacy of IoT companions apps [24], by investigating accessibility.

3 METHODS

To avoid ambiguity, we build a definition of IoT device based on previous work: any device that can interact with the physical world through sensing or actuating and can transmit sensing/actuation-related data (directly or indirectly) over a network [17, 22]. The definition of a companion app follows from the above: *an app which connects to an IoT device to relay actuation commands or receive sensing data.* In other words, a companion app acts as a *gateway* between the IoT device on one side, and the user and cloud backends on the other [20].

First, we collected IoT companion apps from the Google Play Store. Since Android devices have a much bigger market share than iOS devices, we chose to study Android apps more extensively since they have a bigger impact on a larger population [1]. We plan to expand this work to study iOS apps as a future extension of this work. Our data collection methodology was inspired by Wang et al. [35], with some considerable modifications, consisting of the following steps:

- **Step 1: Manual Search**: We first manually downloaded IoT apps from Google Play store. We did so by looking through apps used in the context of smart home/sensing or collecting information about the physical environment and connectivity to the network. This formed our "Seed App Set ".
- **Step 2: App Scraping**: For each app in our Seed App set, we scraped app names and descriptions through the "Similar Apps" suggestions in the Play Store using play-scraper [2]. After this step we had a set of 2000 apps.
- **Step 3: Keyword-based Filtering**: We then performed keyword-based filtering to remove high false positives. The false positives in this case were apps that did not match our definition of IoT companion apps. We generated a set of keywords, empirically, based on the correlation of keywords to these false-positive apps. Then we removed apps that match specific keywords (e.g. *currency and compiler*).
- **Step 4: Naïve-Bayes Classification**: Using Machine Learning, we then refined the candidate set to classify IoT and non-IoT apps. Using Naïve-Bayes classifiers lead to better accuracy, although far from optimal (64.6%), on a small set of manually labelled data. We also experimented with the BERT algorithm [11] and Logistic Regression, which, led to lower accuracy. Step 3 and Step 4 reduced our set of 2000 apps to 1596 (20.2% reduction).
- **Step 5: Manual Filtering**: Lastly, we manually review all the apps classified by the Naïve-Bayes and only retain the ones which match our definition of companion apps. After manual inspection, we determined only 484 (30.3%) matched our definition of IoT apps and were relevant to our analysis.

Data collection results: Once we had the list of IoT apps, we downloaded the app packages using PlaystoreDownloader [3]. Out of the 484, we were able to retain only **455** APKs because the remaining could not be downloaded or were of incompatible format.

3.1 Analysis

To evaluate Accessibility, we used *Accessibility Insights for android*[4], which is a free, open-source tool that allows us to evaluate accessibility. This tool can detect common accessibility issues such as: contrast, missing names and descriptions, or inadequate touch target sizes, through UI analysis. Accessibility insights for android bases its rules on axe-android, which is an automated WCAG 2.0 and WCAG 2.1 Accessibility library for Android apps [2]. The tool works by taking screenshots of the interface of the app being evaluated and highlighting any instances that may relate to accessibility issues. The instances of the accessibility analysis include the following:

(1) *ActiveViewName*: Active views must have a name that is available to assistive technologies. Missing text results in a violation.
(2) *ImageViewName*: Meaningful images must have alternate text. Images without associated text result in a violation.

[1]https://www.statista.com/statistics/272698/global-market-share-held-by-mobile-operating-systems-since-2009/

[2]https://pypi.org/project/play-scraper
[3]https://github.com/ClaudiuGeorgiu/PlaystoreDownloader
[4]https://accessibilityinsights.io/docs/en/android/overview/

(3) *TouchSizeWcag*: Touch inputs must have a sufficient target size. The tool checks elements to have a minimum width or height of 44dp; elements smaller than 44dp result in a violation.
(4) *EditTextValue*: EditText elements (used to enter text) must expose entered text to assistive technologies. Failing to expose such text results in a violation.
(5) *ColorContrast*: Text elements must have sufficient contrast against the background. This category is different from the others as it requires the operator's discretion. For example, if a button was intended not to be selectable and was grayed out intentionally, the operator should not record a failed instance.

These instances all relate to accessibility because they ensure: (i) each button is easily selectable for all users (*TouchSizeWcag*); (ii) the text is easily readable to all users (*ColorContrast*); and (iii) visually-impaired people can still use the app by having all pertinent information conveyed to them through text descriptions (*ActiveViewName*, *ImageViewName*, *EditTextValue*). To apply Accessibility Insights to each app, we executed it simultaneously with the Android Studio emulator analyzing apps of interest. After testing every screen in a preliminary analysis on a sample of 15 apps, we found that a non-negligible fraction of apps only exhibit certain categories of errors on specific screens.

After running Accessibility Insights on the app, the tool returned a list of failed instances. A single page of an app can have multiple failed instances for any number of categories. When a previously used asset returns the same failed instance(s), we only recorded each failed instance once. This is due to software developers' tendency to reuse their assets throughout an app leading to double counting. Thus, we decided to de-duplicate these errors as they represent a repeated appearance of the same mistake rather than a new one. To eliminate duplicate screens of companion apps, we manually retained graphical records of previously tested screenshots. This de-duplication process was necessary to ensure an accurate analysis of each companion app's features and functionality.

We also identified several false positives – apparent violations of rules caused by assets that are either not visible or have height and/or width set to 0. Since the developer's intention is clearly for such assets not to be visible, we filtered out these instances from our calculation. After performing these adjustments, we then recorded any failed instances that the tool returned for all app pages. This process was repeated for every Android app. If a lock screen was encountered in an application, we attempted to find a demo within the app that would potentially display a better understanding of the errors within an app. This was successful on a portion of apps with a lock screen but when this was not possible, analysis was performed on the lock screen and any additional accessible page within the app that could provide better insight to potential errors beyond the lock screen. All APKs of the IoT mobile companion apps required for the project were downloaded and installed on the emulators in Android studio. During the app's installation process, we discovered that some of the emulators were not compatible with some apps. After investigating, we concluded that Nexus (Android 9.0) and Pixel (Android 8.1) were the most stable emulators for the installation and analysis.

4 RESULTS

During our data collection phase, we collected as many as 455 IoT companion apps APKs for accessibility analysis. Emulator-blocking features prevented the analysis of some companion apps in our corpus despite various attempts to bypass this restriction. We ultimately decided to exclude these apps from our final corpus which resulted in us analyzing only 248 apps. We considered the trend and usage frequencies of all the companion apps to ensure that we were analyzing the most commonly used IoT companion apps. The emulators-Nexus (Android 9.0) and Pixel (Android 8.1) were preferred as virtual devices because they seem more stable than other tested virtual devices.

After that, we launched Accessibility Insights for Android to analyze and evaluate the various pages of individual IoT apps. It took about 15 – 25 minutes to analyze each companion app, depending on the number of pages[5].

Analysis of the accessibility metrics reveals that *TouchSizeWcag* (95%) has the highest rate of violations and errors, where as *EditTextValue* (3.94%) has the least number of violations and errors. Furthermore, the majority of the apps exhibit some errors. There were only two apps that passed all the accessibility metrics, one of them has a total of six pages, while the other has 12 pages. In fact, the 248 IoT companion apps produced 5,349 violations and errors over a total of 3964 pages during accessibility analysis. The overall error counts are as follows: *TouchSizeWcag* = 2357, *Contrast* = 1200, *ActiveViewName* = 1322, *ImageViewName* = 515 and *EditTextValue* = 53. The data shows that *TouchSizeWcag* accounted for almost half of the total errors while the errors from *EditTextValue* are almost negligible compared to other evaluation metrics. Table 1 summarizes the standard deviation, mean, and range for each of the accessibility metrics under review. We took into account errors on the landing page, the total number of errors within an app, and the average number of errors per page. Results show how the metrics vary in violation of accessibility. In the following sections, we expand on the individual results of the accessibility errors.

4.1 Landing Page vs Other Pages

Results show that the number of available pages to be evaluated and analyzed varied based on the mobile companion app. The page variation was considered because apps with more pages tend to produce more opportunities for errors; this is further proven by the mean of the average number of accessibility violations per page over our corpus and the mean of the total number of errors per app. Furthermore, for almost all companion apps, an account was created on the landing page to access subsequent pages.

4.2 ActiveViewName

Assistive technologies are paramount to accessibility because they aid individuals with disabilities to perform tasks on their devices that they would otherwise not be able to perform. As such, different UI elements on apps should have a name accessible by these technologies for ease of navigation. Along these lines, *ActiveViewName* errors detect UI elements in apps with missing names. These errors represent about 25% (1322) of the total errors obtained during accessibility analysis of the IoT mobile companion apps. It is the second

[5]Our app dataset and detailed results are accessible upon request.

most violated feature across the accessibility metrics considered. Furthermore, 86.67% of the apps analyzed presented at least one *ActiveViewName* error throughout its pages, and only 33 apps avoided it successfully. The most *ActiveViewName* errors introduced within the same app were 35 errors in 23 pages, however, this was not the highest rate, as the highest rate was 6 errors for a one-page app.

4.3 ImageViewName

ContextDescription attribute is supposed to provide text alternatives to explain the meaning of an image to users. So, any meaningful image that does not have such a description to provide assistive technologies violates the accessibility standards. From our analysis, we realized that errors associated with *ImageViewName* were about 9.63% of the total errors. A total of 128 companion apps were without violation of *ImageViewName* and two apps have the highest number of errors (19 errors). Though, the highest rate of errors was detected for a one-page app (7 errors per page).

4.4 TouchSizeWcag

Sufficient target size of minimum width and height of 44dp is required for *touch inputs* to pass accessibility criteria determined by the WCAG. Carrying out the accessibility and producing the detailed results of all the violations will help developers to do better by following the recommendations and standards required for designing *touch inputs*. *TouchSizeWcag* accounted for about 25% of the total metric violations. From our analysis, we noticed that almost (95%) of our companion apps had *TouchSizeWcag's* errors on at least one of their pages. At the same time, only 13 apps did not present any errors pertaining to elements' size. One app presented 53 *TouchSizeWcag's* errors with an average of 2.21 errors per page; this rate, however, was not the highest. The highest rate is 10 errors per page for a one-page app.

4.5 EditTextValue

If an editable text object is not configured correctly, assistive technology may announce the type of the object rather than its text content. This, in turn, will give users who rely on assistive technology insufficient information to verify their input. *EditTextValue* was only present in 24 of the apps analyzed. This was the least prominent error, with only 3.94% errors of the total metric violations. The highest number of *EditTextValue* detected errors within one app is 7 errors for a 21-page app, while the highest rate is 0.667 errors per page for a 3-page app.

4.6 ColorContrast

Developers need to consistently implement elements, including text, and images or icons, that have acceptable contrast with their background because it is an essential step in meeting accessibility standards. This accessibility metric is the only one for which we perform manual error screening to determine the final result before recording. It is the third most violated metric with about 22.44% errors of the total metric violations. It also produced the highest error within the same app (72 errors) and the highest rate of errors per page within the same app (18 errors within a one-page app).

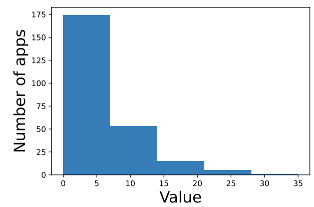

Figure 1: ActiveViewName errors

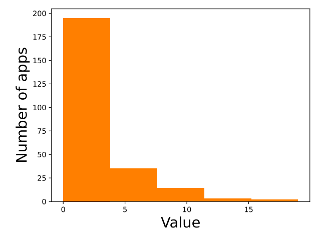

Figure 2: ImageViewName errors

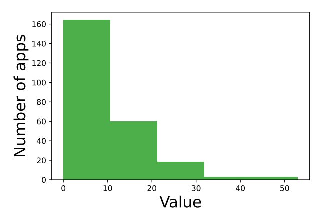

Figure 3: TouchSizeWcag errors

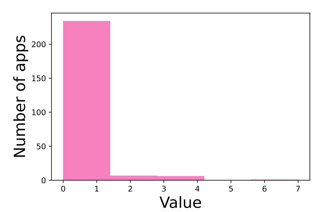

Figure 4: EditTextValue errors

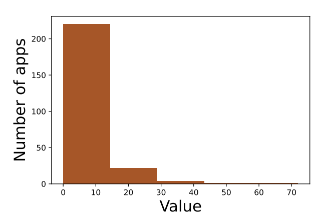

Figure 5: ColorContrast errors

4.7 Correlation between metrics

A question relevant to our analysis is whether accessibility errors correlate across error categories. In exploring this question, we use Spearman correlation as the distributions of error per app appear decidedly non-normal. When considering correlation, the effect of

Table 1: Standard Deviation, Mean, Max and Min for accessibility violations

Category	STD	Mean	Max	Min
NumPages	13.1	16.0	68	1
ActiveViewName Landing Page	1.92	0.90	21	0
ActiveViewName Total # of Errors	5.62	5.33	35	0
ImageViewName Landing Page	0.851	0.355	6	0
ImageViewName Total # of Errors	3.23	2.08	19	0
TouchSizeWcag Landing Page	2.67	1.84	20	0
TouchSizeWcag Total # of Errors	9.06	9.50	53	0
EditTextValue Landing Page	0.0635	0.00403	1	0
EditTextValue Total # of Errors	0.799	0.214	7	0
Contrast Landing Page	1.47	0.851	10	0
Contrast Total # of Errors	8.79	4.84	72	0
ActiveViewName Average Errors Across Pages	0.696	0.461	6.0	0
ImageViewName Average Errors Across Pages	0.532	0.171	7.0	0
TouchSizeWcag Average Errors Across Pages	1.22	0.848	10.0	0
EditTextValue Average Errors Across Pages	0.0753	0.0166	0.667	0
Contrast Average Errors Across Pages	1.49	0.517	18.0	0

Table 2: r_s Between Accessibility Metrics (AVN: ActiveViewName, IVN: ImageViewName, TSW: TouchSIzeWCAG, ETV: EditTextValue, CC: Contrast)

	AVN	IVN	TSW	ETV	CC
AVN	1.00	-	-	-	-
IVN	0.402509	1.00	-	-	-
TSW	0.499606	0.180339	1.00	-	-
ETV	-0.16903	-0.12007	-.05973	1.00	-
CC	0.2057	0.2025	0.4739	0.01932	1.00

app size – in terms of the number of screens – on the value of each metric must be taken into account. Generally speaking, a companion app consisting of more screens creates more opportunities for errors. Based on this consideration, when computing the Spearman correlation matrix, we used the average number of errors per page rather than the total number of errors. Results are presented in Table 2. The most relevant finding is that *ActiveViewName* violations exhibit moderate correlation with *ImageViewName* as well as *TouchSizeWCAG*.

5 DISCUSSION AND IMPLICATIONS

Accessibility is critical for IoT companion apps because an IoT app that is inaccessible not only limits the user from an app standpoint but also limits their ability to use their IoT devices and their knowledge of the output of these devices.

5.1 Impact of Accessibility Violations

Our analysis found 5, 349 violations and errors across a total of 248 apps and 3964 app pages. We were able to ascertain that appropriate UI element (*TouchSizeWCAG*) size is a significant issue across most IoT companion apps. Conversely, lack of accessible text input control *EditTextValue* rarely occurs. The impact of these non-accessible mobile companion apps on the disabled community affects a significant portion of the potential users of the apps. Not testing the mobile apps for accessibility compatibility before market roll-out makes users with disabilities experience exclusion which is a violation of the standards and guidelines set forth by *WCAG*.

5.2 Inclusion in Accessibility Design

Our results show that a significant number of IoT companion apps in the market violates accessibility. This is because they do not comply with the standards and guidelines laid out by *WCAG*. One of the most important findings was that a large percentage of the overall companion apps analyzed lack sufficient target size of width and height required for *touch inputs*. While examining and analyzing these IoT companion apps, it became clear that many of them may not be accessible to disabled populations because of their level of violations. Creating this awareness about *WCAG* violations are part of what we set out to achieve through this work.

5.3 Mandatory Accessibility Testing

Designing accessible mobile companion apps for the vulnerable population has been largely ignored by developers. At the same time, usability and accessibility testing is concerned with determining how easy it is for users to understand the operation of technology and remember it at a later time. However, many technologies that pass broad accessibility assessments turn out to be inaccessible to users with impairments [3]. So, we believe there is a need to design better accessibility features to assist people with impairments to utilize mobile companion apps seamlessly with mandated accessibility testing. For example, developers must ensure companion apps' content must be presented in a way that is independent of its underlying structure [6].

[6]https://www.w3.org/WAI/tips/designing/

6 FUTURE WORK AND LIMITATIONS

This work examines the accessibility of the mobile companion apps required to access IoT devices. In this analysis, we focus solely on the accessibility of android companion applications. However, as discussed in the paper, due to technical limitations of our analysis environment we could only analyze 248 apps out of the 455 we collected. Future work will include analyzing a larger app dataset through employing physical devices in order to get around emulator-blocking features' restrictions. To guarantee that mobile applications are accessible to all users, we plan to extend our work beyond Android apps by investigating iOS apps as well.

7 CONCLUSION

Mobile accessibility was introduced to ensure that apps are designed to be easily accessible by the disabled populations. Along these lines, this paper aims to explore the level and type of accessibility violations on the most commonly used IoT companion apps. To do so, we performed an accessibility analysis of 248 IoT companion apps. We presented the accessibility violations based on the adopted metrics, including errors pertaining to names and descriptions of elements and images, sizes of touchable elements, and color contrasts. Our analysis reveals that only two apps from our corpus successfully implemented the WCAG's guidelines and did not produce any accessibility failures in the pages we were able to analyze successfully; the remaining 246 apps had at least one accessibility error. This analysis will provide recommendations for developers to produce IoT mobile apps designed for better accessibility.

ACKNOWLEDGMENTS

This work was supported in part by NSF under the award CNS 1822118 and by NIST, ARL, Statnett, AMI, Cyber Risk Research, NewPush, State of Colorado Cybersecurity Center, a gift from Google, and the Inclusive Security and Privacy focused Innovative Research in Information Technology (InSPIRIT) Laboratory at the University of Denver. We would like to also thank Upakar Paudel and Harrison Whittaker for their help with this project. Any opinions, findings, conclusions, or recommendations expressed in this material are solely those of the authors.

REFERENCES

[1] Shadi Abou-Zahra, Judy Brewer, and Michael Cooper. 2017. Web Standards to Enable an Accessible and Inclusive Internet of Things (IoT). In *Proceedings of the 14th Web for All Conference*. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3058555.3058568

[2] Andrew Kirkpatrick, Joshue O Connor, Alastair Campbell, and Michael Cooper. 2018. Web Content Accessibility Guidelines (WCAG) 2.1. https://www.w3.org/TR/WCAG21/

[3] Elsa Bakiu and Emitza Guzman. 2017. Which feature is unusable? Detecting usability and user experience issues from user reviews. In *2017 IEEE 25th International Requirements Engineering Conference Workshops (REW)*. IEEE, Lisbon, Portugal, 182–187.

[4] V. Balaji and K.S. Kuppusamy. 2016. Accessibility analysis of e-governance oriented mobile applications. In *2016 International Conference on Accessibility to Digital World (ICADW)*. IEEE, Guwahati, India, 141–144.

[5] Mars Ballantyne, Archit Jha, Anna Jacobsen, J Scott Hawker, and Yasmine N El-Glaly. 2018. Study of accessibility guidelines of mobile applications. In *Proceedings of the 17th international conference on mobile and ubiquitous multimedia*. Association for Computing Machinery, New York, NY, USA, 305–315.

[6] Marco Billi, Laura Burzagli, Tiziana Catarci, Giuseppe Santucci, Enrico Bertini, Francesco Gabbanini, and Enrico Palchetti. 2010. A unified methodology for the evaluation of accessibility and usability of mobile applications. *Universal Access in the Information Society* 9 (2010), 337–356.

[7] Luca Catarinucci, Danilo de Donno, Luca Mainetti, Luca Palano, Luigi Patrono, Maria Laura Stefanizzi, and Luciano Tarricone. 2015. An IoT-Aware Architecture for Smart Healthcare Systems. *IEEE Internet of Things Journal* 2, 6 (Dec. 2015), 515–526.

[8] Roberto Yuri da Silva Franco, Rodrigo Santos do Amor Divino Lima, Monte Paixão, Carlos Gustavo Resque dos Santos, Bianchi Serique Meiguins, et al. 2019. UXmood—A Sentiment Analysis and Information Visualization Tool to Support the Evaluation of Usability and User Experience. *Information* 10, 12 (2019), 366.

[9] Sanchari Das et al. 2022. SoK: a proposal for incorporating accessible gamified cybersecurity awareness training informed by a systematic literature review. In *Proceedings of the workshop on usable security and privacy (USEC)*. USEC, USA.

[10] Gabriela Amaral Araújo de Oliveira, Raphael Winckler de Bettio, and André Pimenta Freire. 2016. Accessibility of the Smart Home for Users with Visual Disabilities: An Evaluation of Open Source Mobile Applications for Home Automation. In *Proceedings of the 15th Brazilian Symposium on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 1–10. https://doi.org/10.1145/3033701.3033730

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* 2, 2 (2018), 4171–4186.

[12] Delia Ferri and Silvia Favalli. 2018. Web Accessibility for People with Disabilities in the European Union: Paving the Road to Social Inclusion. *Societies* 8, 2 (2018), 40.

[13] Centers for Disease Control, Prevention, et al. 2021. Disability Impacts All of Us Infographic. https://www.cdc.gov/ncbddd/disabilityandhealth/infographic-disability-impacts-all.html

[14] Shakthidhar Gopavaram, Jayati Dev, Sanchari Das, and L Jean Camp. 2021. Iot marketplace: Willingness-to-pay vs. willingness-to-accept. In *Proceedings of the 20th Annual Workshop on the Economics of Information Security (WEIS 2021)*. SSRN Electronic Journal, USA.

[15] Hilda Hadan, Nicolas Serrano, Sanchari Das, and L Jean Camp. 2019. Making iot worthy of human trust. In *TPRC47: The 47th Research Conference on Communication, Information and Internet Policy*. SSRN Electronic Journal, USA.

[16] Aamir Hussain, Rao Wenbi, Aristides Lopes da Silva, Muhammad Nadher, and Muhammad Mudhish. 2015. Health and emergency-care platform for the elderly and disabled people in the Smart City. *Journal of Systems and Software* 110 (2015), 253–263.

[17] 9241-11 ISO. 2018. Ergonomics of human-system interaction — Part 11: Usability: Definitions and concepts. https://www.iso.org/obp/ui/#iso:std:iso:9241:-11:ed-2:v1:en.

[18] Susanne Iwarsson and Agneta Ståhl. 2003. Accessibility, usability and universal design—positioning and definition of concepts describing person-environment relationships. *Disability and rehabilitation* 25, 2 (2003), 57–66.

[19] Treffyn Lynch Koreshoff, Toni Robertson, and Tuck Wah Leong. 2013. Internet of things: a review of literature and products. In *Proceedings of the 25th Australian Computer-Human Interaction Conference: Augmentation, Application, Innovation, Collaboration*. Australasian Computer system, Australia, 335–344.

[20] Hui Liu, Juanru Li, and Dawu Gu. 2020. Understanding the Security of App-in-the-Middle IoT. *Computers & Security* 97 (Oct. 2020), 102000.

[21] Jack Marquez, Jhorman Villanueva, Zeida Solarte, and Alexander Garcia. 2016. IoT in Education: Integration of Objects with Virtual Academic Communities. In *New Advances in Information Systems and Technologies*. Vol. 444. Springer International Publishing, Cham, 201–212.

[22] Sara N. Matheu, José L. Hernández-Ramos, Antonio F. Skarmeta, and Gianmarco Baldini. 2021. A Survey of Cybersecurity Certification for the Internet of Things. *Comput. Surveys* 53, 6 (Feb. 2021), 1–36.

[23] Ingunn Moser. 2006. Disability and the promises of technology: Technology, subjectivity and embodiment within an order of the normal. *Information, Communication & Society* 9, 3 (2006), 373–395.

[24] Shradha Neupane, Faiza Tazi, Upakar Paudel, Freddy Veloz Baez, Merzia Adamjee, Lorenzo De Carli, Sanchari Das, and Indrakshi Ray. 2022. On the Data Privacy, Security, and Risk Postures of IoT Mobile Companion Apps. In *Data and Applications Security and Privacy XXXVI*. Springer International Publishing, Cham, 162–182.

[25] Alisha Pradhan, Kanika Mehta, and Leah Findlater. 2018. " Accessibility Came by Accident" Use of Voice-Controlled Intelligent Personal Assistants by People with Disabilities. In *Proceedings of the 2018 CHI Conference on human factors in computing systems*. Association for Computing Machinery, New York, NY, USA, 1–13.

[26] Anne Spencer Ross, Xiaoyi Zhang, James Fogarty, and Jacob O Wobbrock. 2017. Epidemiology as a framework for large-scale mobile application accessibility assessment. In *Proceedings of the 19th international ACM SIGACCESS conference on computers and accessibility*. Association for Computing Machinery, New York, NY, USA, 2–11.

[27] Anne Spencer Ross, Xiaoyi Zhang, James Fogarty, and Jacob O. Wobbrock. 2018. Examining Image-Based Button Labeling for Accessibility in Android Apps

through Large-Scale Analysis. In *Proceedings of the 20th International ACM SIGACCESS Conference on Computers and Accessibility*. Association for Computing Machinery, New York, NY, USA, 119–130. https://doi.org/10.1145/3234695.3236364

[28] Sunny Shrestha, David Thomas, and Sanchari Das. 2022. SecureLD: Secure And Accessible Learning for Students with Disabilities. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, Vol. 66. SAGE Publications Sage CA, Los Angeles, CA, 465–469.

[29] Osama Sohaib, Haiyan Lu, and Walayat Hussain. 2017. Internet of Things (IoT) in E-commerce: For people with disabilities. In *2017 12th IEEE Conference on Industrial Electronics and Applications (ICIEA)*. IEEE, Siem Reap, Cambodia, 419–423. https://doi.org/10.1109/ICIEA.2017.8282881

[30] Alessandro Soro, Margot Brereton, Paul Roe, Peta Wyeth, Daniel Johnson, Aloha Hufana Ambe, Ann Morrison, Shaowen Bardzell, Tuck Wah Leong, Wendy Ju, et al. 2017. Designing the social internet of things. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 617–623.

[31] Joshua Streiff, Naheem Noah, and Sanchari Das. 2022. A Call for a New Privacy & Security Regime for IoT Smart Toys. In *2022 IEEE Conference on Dependable and Secure Computing (DSC)*. IEEE, Edinburgh, United Kingdom, 1–8.

[32] Marion Lara Tan, Raj Prasanna, Kristin Stock, Emma EH Doyle, Graham Leonard, and David Johnston. 2020. Modified usability framework for disaster apps: a qualitative thematic analysis of user reviews. *International Journal of Disaster Risk Science* 11 (2020), 615–629.

[33] Guilherme Mussi Toschi, Leonardo Barreto Campos, and Carlos Eduardo Cugnasca. 2017. Home automation networks: A survey. *Computer Standards & Interfaces* 50 (2017), 42–54.

[34] Otily Toutsop, Sanchari Das, and Kevin Kornegay. 2021. Exploring The Security Issues in Home-Based IoT Devices Through Denial of Service Attacks. In *2021 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/IOP/SCI)*. IEEE, Atlanta, GA, USA, 407–415.

[35] Xueqiang Wang, Yuqiong Sun, Susanta Nanda, and XiaoFeng Wang. 2019. Looking from the Mirror: Evaluating IoT Device Security through Mobile Companion Apps. In *USENIX Security*. USENIX Association, Santa Clara, CA, 1151–1167.

[36] Alisa Zezulak, Faiza Tazi, and Sanchari Das. 2023. SoK: Evaluating Privacy and Security Concerns of Using Web Services for the Disabled Population. In *7th Workshop on Technology and Consumer Protection (ConPro'23)*. Cyber Situational Awareness Techniques and Human Factors, USA, 329–357.