A11yExtensions: Accessibility Extensions to Augment Mobile AI Assistive Technology In-Situ

Jaylin Herskovitz
Computer Science and Engineering
University of Michigan
Ann Arbor, Michigan, USA
jayhersk@umich.edu

Ellie Seehorn
University of Michigan
Ann Arbor, Michigan, USA
seehorn@umich.edu

Ather Jammoa
Independent Consultant
Detroit, Michigan, USA
atherjammoa@yahoo.com

Jason Meddaugh
A.T. Guys
Kalamazoo, Michigan, USA
jj@atguys.com

Anhong Guo
Computer Science and Engineering
University of Michigan
Ann Arbor, Michigan, USA
anhong@umich.edu

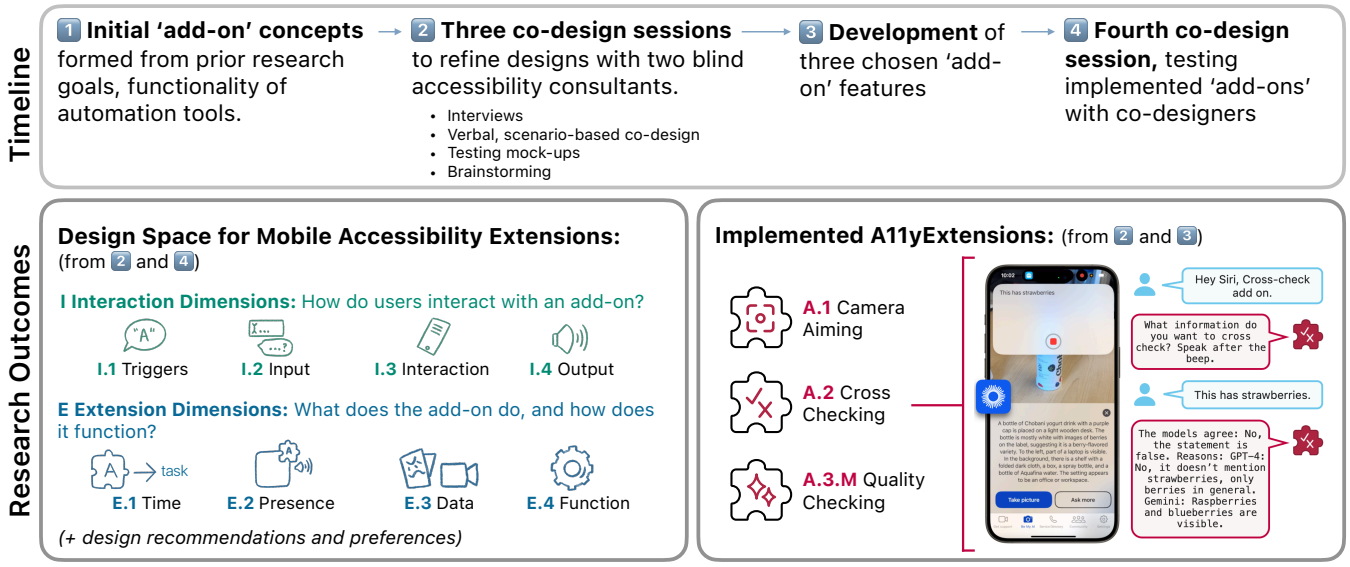

Figure 1: We present A11yExtensions, a design space and set of implemented 'add-ons' to mobile assistive technology powered by existing automation tools. A11yExtensions was generated through co-design using functional prototypes as design probes.

Abstract

Existing visual AI assistive technologies have usability gaps, and may need additional adaptations and features to serve users' needs. We propose A11yExtensions, in-situ interventions that augment existing mobile AI assistive technology with add-on services. Add-ons include features that have been researched but are not yet deployed (e.g., cross-checking AI results), or that are only available in certain applications (e.g., camera aiming assistance). Through co-design sessions with two blind accessibility professionals, we designed and implemented three exemplar extensions, leveraging mobile automation tools to invoke add-ons, enabling just-in-time interventions for adaptability. We found that A11yExtensions provide opportunities to test new features and a new degree of flexibility and customization, though they introduce additional onboarding and communication challenges. We also derived a design space of accessibility extensions as a basis for future extension designs. Overall, A11yExtensions is a demonstration of the effectiveness of deploying new features in-situ via automation, with the technologies people actually use in their day-to-day lives.

CHI '26, Barcelona, Spain

ACM ISBN 979-8-4007-2278-3/26/04
https://doi.org/10.1145/3772318.3791559

CCS Concepts

• **Human-centered computing** → **Accessibility systems and tools**; *Interactive systems and tools.*

Keywords

Accessibility, blind, visual impairment, AI assistive technology, mobile assistive apps, automation, co-design, research tools

ACM Reference Format:
Jaylin Herskovitz, Ellie Seehorn, Ather Jammoa, Jason Meddaugh, and Anhong Guo. 2026. A11yExtensions: Accessibility Extensions to Augment Mobile AI Assistive Technology In-Situ. In *Proceedings of the 2026 CHI Conference on Human Factors in Computing Systems (CHI '26), April 13–17, 2026, Barcelona, Spain.* ACM, New York, NY, USA, 23 pages. https://doi.org/10.1145/3772318.3791559

1 Introduction

Mobile, camera-based assistive applications such as Seeing AI [59], Lookout [30], and Be My AI [26] are used regularly by blind people for everyday tasks. While these AI assistive technologies are useful, they are imperfect. From a design perspective, these applications are created to be general purpose and to fit simple and common use cases, and thus are not suited to address a broad range of specific user needs [35]. Yet, the research community has designed and developed a variety of new features to improve AI assistive technologies. These range from better blind photography techniques [48], to techniques for verifying image content [40, 47], to techniques for helping blind people inspect objects further and receive specific information rather than general descriptions [17, 36, 65]. However, these features are largely siloed from the commercial applications that blind people use every day. While accessibility research may eventually trickle into commercial or open-source products, or research teams may try to maintain their own standalone assistive applications, these measures ultimately lead to a limited expression of research findings or hard-to-maintain software. This phenomenon of siloing also exists between corporate applications, whose features may be complementary to each other in theory, but cumbersome, if not impossible, to use concurrently in practice.

In this work, we present A11yExtensions, a design space and set of in-situ extensions to mobile assistive technology. Inspired by HCI research that has used various forms of extensions (browser extensions [46], bots [31], or scripts) to modify existing software, we aim to understand how the same concept could be applied in the space of mobile AI assistive technologies. While we know *what* features may be useful to add from prior HCI research, this work aims to answer the question of *how* to do so (Figure 2): **How can we integrate new features seamlessly into people's existing, familiar assistive technology workflows?**

To answer this question, we engaged in a Research through Design (RtD) approach [81], creating A11yExtensions through a longitudinal co-design process with two blind professional accessibility consultants and assistive technology experts, who are listed as coauthors (timeline shown in Figure 1). Based on existing applications, prior accessibility research, and mobile automation tool affordances, we developed initial conceptual designs of possible extensions for accessibility. We then worked with co-designers in three formative design sessions, using these initial sketches as starting points to collaboratively refine the concepts and guide implementation. Using interviews, scenario-based design brainstorming, mock-ups, and prototype testing, co-designers ensured that A11yExtensions

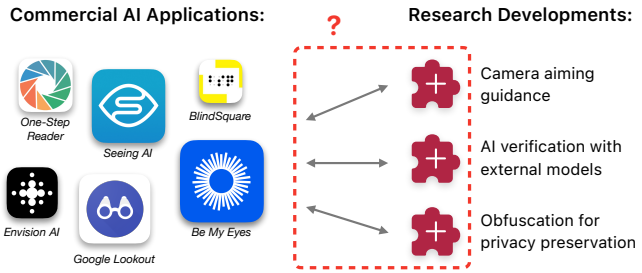

Figure 2: Research developments are siloed from assistive technology in practice. How can we design ways to integrate these developments into people's existing workflows?

address real-world needs, and are understandable and unobtrusive by iteratively testing for usability and suggesting adjustments. These sessions revealed important findings about the desirability of mobile automation for accessibility, specifically around the important balance between automation and user autonomy and control, which is critical for screen-reader users. We made design changes that prioritized efficiency and simplicity, transparency, and privacy preservation, characteristics that our co-designers highly valued.

We then implemented three extension features on iOS: camera aiming guidance, an AI cross-checking service, and an image quality check (Figure 1, Right). This implementation leverages the mobile automation tool Shortcuts [8] to create a framework that provides the additional extension features as 'add-ons' to mobile assistive applications. For example, while using an assistive AI application that takes static photos, such as Be My Eyes [26], a user would trigger the 'camera-aiming' add-on using Siri, be automatically redirected to a companion app offering verbal aiming guidance, and once a desired object is properly framed, be redirected again back to their original app to take a photo. This framework allows users to resume their normal patterns of use after interventions, aiming to be minimally disruptive to existing routines. A11yExtensions is implemented as an iOS application with add-ons included as independent end-points via Siri Intents, using a set of iOS Shortcut programs to automate their use and manage transitions between applications.

Afterwards, we conducted a final co-design session using the implemented extensions, allowing for analysis of previously hypothetical or mock-up designs now made concrete. Through user testing, co-designers reflected on both the practical implementation of extensions and the concept as a whole after the design process. We found that while co-designers initially had concerns about practicality and privacy, these concerns could be reduced through specific design changes such as prioritizing user initiation and better audio feedback. Our co-designers also saw benefits to add-on features in terms of increasing flexibility with how features can be accessed, and creating a faster testing and deployment cycle for new features. Finally, we analyzed the designs and concepts discussed in the four co-design sessions to create a design space of mobile accessibility extensions, characterizing user interaction and extension functionality (Figure 1, Left).

A11yExtensions is a design probe aiming to understand how we might adapt and extend commercial assistive technologies with

new and useful features. It demonstrates practical ways for augmenting assistive technology, presenting mobile automation as a new interaction paradigm to do so. We reflect on A11yExtensions' potential as a method for future user customization, and as a platform for assistive technology research. Overall, this work points to how deployed assistive technologies could be modified, customized, and improved in practice. We contribute:

(1) The concept of and a design space for mobile extensions to assistive applications.
(2) A11yExtensions, a set of implemented add-on features leveraging mobile automation tools, as a co-designed artifact.
(3) Findings from a longitudinal co-design process illustrating design goals and preferences, and envisioned add-on use and challenges.

2 Related Work

First, we review existing assistive technologies, as A11yExtensions aims to address gaps in commercial and research applications. Then, we review how extensions and automation tools have been used in HCI research, as A11yExtensions leverages these methods to extend assistive technologies.

2.1 Research and Practice in AI Assistive Technology

Commercial AI-powered visual assistive technologies have been commonplace for nearly a decade, performing tasks such as reading printed text, recognizing currency, describing light levels, and identifying people [29, 59, 68]. While these tools have been immensely helpful for blind people, they are also straightforward and simple: they take in camera input and directly verbalize model results, and they have straightforward feature sets and interfaces. For instance, SeeingAI provides an easy to navigate bar of 'channels' as options to users, where each channel verbalizes the result from a different AI model [59]. This design approach is not without good reason, as commercial applications are designed to appeal to the broadest range of people, including older adults or users with cognitive or motor disabilities, who need simple and clear interfaces [21]. However, due to this one-size-fits-all approach, these technologies have limitations in their features or functionality that prevent them from being universally useful [1]. Prior work has found that there are a range of needs unmet by existing tools, forming a 'long tail' of unique cases [35].

HCI and accessibility research often takes a contrasting approach. Without the constraints of creating cohesive products for broad populations, researchers are able to develop new solutions to address gaps. For instance, research has pointed to AI accuracy and understanding being a significant barrier in the accessibility space [5, 38, 44, 76], and AI verification and education features have been developed to address such challenges [40, 47]. Privacy has been a significant concern as well [73, 74], and research has investigated data obfuscation approaches to censor private information before it leaves a device [4, 78]. For one-size-fits-all challenges, approaches for customizing AI functionality have been developed [36, 43, 64]. Without the constraints of supporting a large user base, researchers create new and experimental solutions that address key needs, but may not yet be suited to all users or situations.

As a result, research developments are integrated into commercial technologies to varying degrees of success. For instance, although years of accessibility research has focused on blind photography [41, 48], this feature is often absent from commercial applications, including Be My AI and some Seeing AI modes [26, 59]. This is representative of a significant design challenge, a tension between adding beneficial but potentially complex features, and maintaining usability and approachability for novices. On the other hand, research is often done in small teams that lack the resources to deploy and upkeep fully fledged applications. A11yExtensions aims to bridge these two worlds by bringing new features to existing workflows on-demand, enabling people to access what they need while maintaining the structure and familiarity of existing applications.

2.2 Interventions, Extensions, and Deployments in HCI Research

In practice, extensions are an important part of customizing technology. As of 2025, the Chrome Web Store hosts about 160,000 extensions that customize all aspects of the user experience [19]. In accessibility, screen reader extensions are commonplace; the NVDA Community Add-Ons Website hosts over 120 extensions for modifying screen reader output [67]. These public extension ecosystems allow users to drive customization of systems to fit their unique needs, and are thus critical for accessibility.

Additionally, extensions are an important deployment method in HCI research. Researchers have long used extensions or 'workarounds' of various types to study features in the context of existing, deployed systems without necessarily having access to create direct modifications. For instance, 'Piggyback Prototyping' introduced the 'hack' of using social media bots' replies as a feature, allowing researchers to collect interaction metrics without deploying an entirely new system [25, 31]. Other areas of HCI research have similarly used 'hacks' to extend existing systems in unintended ways. For instance, Prefab presents a reverse-engineering approach to implementing advanced interactions with existing toolkits [22], with the specific goal of bridging research and practice. InteractOut leverages touch input manipulation to inhibit social media use without changing interfaces [54]. These approaches allow researchers to directly study how people would interact with a new technology or feature in-the-wild, overcoming constraints of lab studies. Similarly in accessibility, researchers often aim to address accessibility challenges of already deployed software in hardware, and extensions, proxies, and 'hacks' have been used as methods to do so. For instance, Co11ab and CollabAlly both provide Chrome extensions that modify the user experience of Google Docs to make it more accessible for collaborative writing [20, 46], and BrushLens provides a hardware add-on for standard smartphones that can actuate external touchscreens [51].

The ability to develop extensions depends heavily on the openness of the platform. Twitter's early API platform made developing bots possible and easy, sparking the Piggyback Prototyping approach [31, 57]. The web and Android platforms (which are open source) stand in contrast to iOS, a fairly closed ecosystem. Yet, iOS is a key platform for developing for accessibility, as it is known for robust accessibility settings and wider adoption among people

Session	Methods	Presented In
Session 1	Semi-structured interview and brainstorming introducing add-ons conceptually and discussing four initial add-on ideas (A.1 to A.4).	Sections 4 and 5
Session 2	Testing three add-on mock-up prototypes: two refined from feedback (A.1 and A.2), and one new concept based on challenges raised in Session 1 (A.5).	Sections 4 and 5
Session 3	Verbal, scenario-based co-design centered on personal experiences in order to discuss real-world add-on use in depth, with three existing concepts (A.1, A.2, and A.3.M) and two new ones (A.6 and A.7).	Sections 4 and 5
Session 4	Testing three fully implemented add-ons (A.1, A.2, and A.3.M), both individually and in one combined workflow simulating the real-world task of preparing a photo for social media. Interview reflecting on the entire co-design process.	Section 7

Table 1: Overview of co-design methods used. Each session focused on a slightly different subset of add-ons as ideal functionalities emerged over the course of design.

with disabilities than other smartphone operating systems [42, 60]. Nonetheless, researchers have still been able to leverage extensions on iOS. Notably, Grüning et al. and Haliburton et al. have studied 'one sec', an iOS app leveraging the native automation platform Shortcuts to create interventions for social media use [8, 32, 34]. 'One sec' includes pre-made Shortcuts that activate a visual timer intervention on the screen whenever specific applications are opened. Thus, 'one sec' acts more like an extension, adding a feature on top of other applications, rather than a standalone app. A11yExtensions, inspired by these approaches, aims to use a similar Shortcuts-based approach to deploy new and necessary accessibility features for iOS applications that can be adopted as-needed.

3 Method Overview

Our goal was to design techniques for integrating new features into existing mobile assistive technologies. We aimed to (1) evaluate and refine the concept of surfacing accessibility features with automation, and (2) design add-on features for assistive technologies that are desirable and usable. To do so, we worked with two blind assistive technology specialists (names denoted as IDs CA and CJ) over three months as co-designers. Working with a small number of co-designers is not uncommon in accessibility research [39, 63], as it can support deep design engagement with people with disabilities. We narrate this process with CA and CJ from a Research through Design (RtD) perspective, a method for generating new knowledge on interactive technology use through the creation of artifacts [81, 82]. Through this, we created A11yExtensions as a design artifact that embodies automation-driven customizations and for accessibility, which provided a medium for continuous iteration.

We met with each co-designer individually four times, using a mix of semi-structured interviews, scenario-based verbal co-design [14], system mock-ups, and prototype testing to spark discussion. An overview is shown in Table 1. The first three co-design sessions were formative, shaping A11yExtensions's design, with new design possibilities and limitations emerging over time (Sections 4 and 5). As we discussed different A11yExtensions concepts, co-designers indicated which ones were more promising or desirable, and we chose those concepts to prototype and later fully implement. The details of this selection are included in Sections 5.2.2 and 5.2.3. The fourth session tested three implemented A11yExtensions, reflecting on it as a fully realized artifact shaped by the previous sessions (Section 7). Each session lasted approximately one and a half to two hours each, for a total of 6.5 hours of meetings with each co-designer. Co-designers were compensated $50 per hour for their time and expertise. This study was approved by our Institutional Review Board (IRB). Complete study protocols are available in Appendix D.

3.1 Co-Designers

CA and CJ are both blind assistive technology consultants and specialists, who met the research team at a local accessibility conference. We reached out to them directly to involve them in this project. CA is an assistive technology trainer. He is an independent contractor, doing work for several state organizations for blind people, and offers both beginner and advanced assistive technology instruction. CJ is the owner of a technology sales, training, and consultancy company. His company has developed and distributed products and services for people who are blind and low vision, and has partnered with numerous businesses on accessibility and usability projects. Additional co-designer background information can be found in Appendix A.

CA and CJ were recruited as co-designers and co-authors for a variety of reasons. First, they both are early adopters of new technologies. They both stay up to date on advances in AI, and frequently try new services in their professional and personal lives. Second, CA and CJ can offer perspectives from their lived experiences being blind, and can also share their experiences gained from working with blind people. In particular, CA spends time conducting training for blind people, and can speak to difficulties faced by novice users. Finally, CA and CJ can speak to accessibility issues in technology, informed by their consulting.

3.2 Positionality Statement

The authors of this paper consist of CA and CJ, and three academic researchers. Jaylin, Ellie, and Anhong are all sighted, with six, two, and over ten years of working with blind and low vision communities in research respectively. We approached this research with the core value of centering disabled people as technology creators and designers. The academic authors analyzed the data collected and ultimately made implementation decisions, and as such aimed to transparently report critique of the approach presented in this work, though sighted biases may still be present. Additionally, we acknowledge that academic research often prioritizes technical and design innovation, which can lead to some of the limitations we explore further in Sections 9.1 and 9.2.

3.3 Data Collection and Analysis

We conducted interviews over Zoom and took audio recordings of each session. Typically, two researchers were present during each interview, with the lead author conducting the session and another researcher taking notes. When testing prototype features, we did not ask co-designers to share their screen over Zoom due to it occasionally causing lag. Instead, their microphones captured their interactions with VoiceOver, and we asked them to verbalize specific actions, which we noted. We then transcribed the interviews for analysis.

After each session, two researchers analyzed the transcripts and notes to create plans for system design iterations, and to create protocols for the next co-design session, which built off of participant feedback. After all of the sessions were completed, the lead author returned to qualitatively analyze the data, aiming to summarize key themes relating to the design process, as many topics were returned to repeatedly over the course of the design sessions, and themes relating to the co-designers overall reflections on the add-on approach, which is presented later in Section 7.

4 Co-Design Sessions 1-3 Methods

The first three co-design sessions were formative, refining concepts, designs, and functionality prior to implementation. From these sessions, we derived: (1) system design iterations that prioritize approachability, privacy, and control for extensions (Section 5), and (2) an initial set of 'add-ons' to prioritize in implementation, based on user needs (Section 5).

4.1 Initial Conceptualization

We began this project with the goal of developing ways to modify and customize already-deployed applications. This was drawn from prior findings demonstrating gaps in assistive technologies, and prior HCI research that has used bots, automation, and extensions to bridge research with deployed technologies.

First, we reviewed prior accessibility research to expand on this initial concept. While AI tools have evolved with technical capabilities, they are known to have issues [1, 35]. For instance, accuracy and trustworthiness is a known challenge with assistive AI tools [1, 5, 38]. Adnin and Das further characterize these as challenges with verifying results without visual ground truth, and understanding visual input used by an AI service [1]. Another commonly reported limitation is the lack of customizability in assistive AI tools: given that tools are general purpose, they may miss unique user needs [35, 55]. Interface accessibility or usability issues are also sometimes present [1, 35]. These gaps, while not exhaustive given the ongoing nature of research in this area, remain central to understanding and improving assistive AI. We also note that large amounts of HCI research have also addressed related challenges, in areas such as human–AI collaboration and explainability [11, 12, 23, 56], nonvisual photography [41, 48], and adaptive or customizable software [36, 43, 65]. The combination of these pressing gaps in commercial technologies, and substantial related research addressing similar issues, led us to consider these directions as strong starting points for extensions to assistive technology.

Next, we reviewed existing mobile automation tools to understand their current and potential functionalities in surfacing new accessibility features. We focused on iOS as a platform, as it is popular among blind and low vision users, and thus also focused on Apple's automation service Shortcuts [8]. Though this does introduce some limitations, which we discuss in Section 9.3. We were inspired by communities of 'power users' who frequently make and share advanced automations using shortcuts on online forums [7]. We noted common capabilities, such as opening or switching apps, surfacing notifications or background audio, taking screenshots or photos, copying and pasting text, and calling APIs, which we used to inform our initial designs.

From these informal reviews, we developed four initial designs for mobile accessibility extensions, or 'add-ons'. These designs are not comprehensive, and while we aimed to represent a variety of functions and automation workflows, they served as a starting point for the co-design process:

A.1 Camera Aiming: An add-on to surface verbal camera aiming feedback, helping to center objects in frame for photos [41, 48]. Despite years of research focusing on this challenge, many assistive AI services lack appropriate verbal aiming feedback. For example, while using an AI service that lacks this feature, users could trigger the add-on by saying 'Hey Siri, help me aim the camera'. A Shortcut automation would then open a separate app that gives live feedback for camera aiming, and once the object is centered, it would automatically switch back to the original app to take the photo.

A.2 Cross Checking: An add-on to cross-check the results of one AI service with other AI models and compare results for validity. Cross-checking between AI services is a common strategy used by blind people to attempt to catch AI errors, although it requires a high degree of manual interaction [18, 40]. To automate cross-checking, when getting a confusing result from an AI service, users could ask, 'Hey Siri, can you cross-check this result?'. A shortcut automation could then take a screenshot of the current output, send it to different services, and have Siri read out other responses for users to compare.

A.3 Image Quality: An add-on for detecting issues in image quality that blind users may be unaware of [1, 48]. Blind users often attribute AI errors to their own photo taking abilities, and wonder if their images are too dark or blurry [5]. A Shortcut could surface a quality checking feature automatically in the background when specified apps are opened (e.g., the Camera app), periodically take and analyze screenshots when the user is about to take a photo, and alert the user with a pop-up notification if there's an issue.

A.4 Routines: An add-on to automate frequent tasks or questions. For example, if users always use a specific service for one type of task [5], a Shortcut could switch the app based on a users location or what objects are detected in frame. Alternatively, a Shortcut could add preferred prompts to the clipboard based on similar triggers.

4.2 Session 1: Current AI Use, Add-On Concept, and Initial Scenarios

In our first co-design session, we focused broadly on the concept of augmenting existing AI applications with extensions. First, we conducted a semi-structured interview about the co-designer's current uses of assistive technology and AI, and challenges or limitations that they experienced. We aimed to understand both their personal

experiences with technology, and their perceptions of common issues from their consulting and training work. Next, we discussed the idea of using automation to address gaps. We conducted verbal co-design around the four initial designs (**A.1** to **A.4**), discussing benefits, concerns, and ideal functionality, as well as reflecting on the add-on approach overall.

4.3 Session 2: Testing System Mock-Ups

In the second session, we tested different ways of triggering add-on features, and mock-ups of add-on functionality. Our goal was to test different add-on workflows so co-designers could reflect on how they could function.

First, we compared three different methods of triggering add-ons: (a) per-feature **trigger phrases** ('Hey Siri, aim the camera'), (b) a **verbal menu** listing all options ('Hey Siri, tell me the add-on menu', 'Choose an option: Aim camera, Cross-check, ...'), (c) an **on-screen menu** displaying an add-on menu to read through ('Hey Siri, show me the add-on menu').

Next, we tested three mock-ups of add on functionality, chosen based on co-designer feedback from Session 1. Co-designers first tested mock-ups of **A.1** and **A.2**. These mock-ups were partially implemented functional prototypes, which demonstrated the actual Shortcut workflows and automations, and had sample audio or partially implemented feedback in place of actual functionality. For instance, for **A.1**, users triggered the mock-up with Siri, which then triggered a Shortcut switching to a second app for camera aiming feedback, where fixed audio played instead of live verbal feedback. This method allowed co-designers to experience the novel automation interactions prior to fully implementing the functionality. The mock-up of **A.2** also asked co-designers a follow-up prompt ('What do you want to cross-check?') that they could respond to verbally. Co-designers also tested a mock-up of a new add-on concept for explaining sources of error, with the goal of helping users build understanding and AI familiarity, which was noted as a challenge in the Session 1 interviews:

A.5 Error Reasoning: An add-on describing possible sources of error in AI results [56]. Blind users occasionally will know an AI result is incorrect, but can only guess about what could have caused the error, leading to uncertain understanding of the services they rely on [5]. The add-on will attempt to provide reasons for the difference between the correct and incorrect result. For instance, if the original error was that a dog was confused for a cat, the add-on could output 'the dog has pointy ears and is small, so it looked like a cat'. For the mock-implementation, participants used the prompt 'Hey Siri, debug this', and were asked to provide a counterfactual (i.e., something wrong in another AI tool) by responding to 'What did you expect to find?'

4.4 Session 3: Functionality For Real-World Use

The third session focused on specifics of how add-ons would be used in the real world, to refine their functionality and implementation. We focused on five add-on concepts: three that seemed most promising from the prior sessions, and two additional add-on concepts designed to address challenges around AI accuracy and understanding, which co-designers indicated as an important area to address with new features. The add-ons discussed were **A.1 Camera Aiming** and **A.2 Cross Checking**, along with the following additions:

A.3.M Image Quality (modified): A modified version of the image quality add-on that only takes a single screenshot rather than multiple, and is triggered by user request rather than automatically.

A.6 Explain Results: An add-on that could give quantitative certainty for a response, inspired by explainability research [12] and aiming to address similar challenges to **A.5**. For instance: 'I'm 70% sure that there's a dog, but it has pointy ears so it could be a cat'.

A.7 Background Information: An add-on that could highlight background information that's important or relevant to the user (e.g., obstacles, safety risks, or time sensitive information), providing context on visual 'unknown unknowns' [13].

5 Formative Design Findings

Here, we present primary themes that we identified from our formative co-design sessions: advantages and disadvantages of the add-on approach, functionality preferences, methods for triggering add-ons, and methods for capturing data. Quotes are labeled with the co-designer's ID and session number (e.g., S1, S2).

5.1 Initial Impressions of Add-Ons

5.1.1 Advantages. CA and CJ both acknowledged the advantages of add-on features as a way to concretely make up for deficiencies in existing applications. CJ said: *"I don't think [add-ons] have to compete with [existing applications]. I think whatever can make it more accessible can be really useful in the long run. Maybe eventually Be My Eyes integrates it more directly, but in lieu of that, using the Shortcut totally makes sense."* (CJ, S1). Similarly, CA acknowledged that although native and centralized accessibility features are the ideal, many existing applications are incomplete. Comparing two applications, they explained: *"Seeing AI already has a lot of things that Envision AI doesn't, and Envision has a few other things that Seeing AI doesn't. So it would be great to combine them somehow. Every one of these apps has these great features, but they're all missing something."* (CA, S1). CA saw add-ons, particularly **A.4 Routines**, as a way to realize this concept of merging existing applications and features. They also noted that add-ons would allow a degree of convenience and automation currently available only in desktop settings, which are typically more extensible and flexible to multitasking. CA compared **A.2 Cross Checking** to 'Picture Smart AI', a new JAWS feature that uses multiple AI services to describe images [28]: *"I've always wanted this multi-AI verification like we have on the desktop, we don't really have anything like this on the iPhone yet. Without needing to go into multiple different apps to see [every result]."* (CA, S1).

CA and CJ imagined using and tinkering with add-on features themselves as technology experts. For instance, CJ imagined creating new a add-on for navigation that would combinine GPS and Oko (a streetlight recognition app), automatically opening Oko when approaching an intersection. CA likened experimenting with add-ons to how they currently use Shortcuts, wanting the ability to test new ideas and customize things for themselves: *"Honestly, I would find it interesting, and in some cases refreshing, like, I would love to play with it... if there was a Shortcut package, just to see how*

that would work." (CA, S1). As experts, CA and CJ saw value in both the add-on functionality, and the ability to 'play' with add-ons as a toolkit and a mechanism for customization.

5.1.2 Challenges and Approachability. However, CA was also skeptical of the approachability of add-on features. CA often thought about how their blind students would use and approach add-ons: *"I would definitely play around with it, but it may not be the ideal solution for everyday users."* (CA, S1). They expressed the ideal case as features centralized into single apps, as they put it *"The more centralization, the better"* (CA, S1), though they did also recognize that this is difficult to realize due to development challenges. In contrast, CJ is less involved in technology training in their work, and largely evaluated the idea of add-ons from their own perspective as a technology expert and someone who frequently tries new services.

While approachability challenges are due in part to the additional steps involved in installing and setting up a Shortcut, a larger barrier to approachability is that add-ons introduce a new interaction workflow. For instance, after walking through the idea for **A.1 Camera Aiming** which automatically switches the user to a new app, CA said: *"It would probably take more work, but I would centralize all of that into one app rather than you know, use the Shortcut, go into this app, then get out of that app and go into another app. It's gonna be a lot for people. I can definitely see where that will probably spark some confusion."* (CA, S1) Because of this, CA expressed a preference for add-ons that run in the background and don't introduce significantly novel interactions. When discussing **A.3 Image Quality**, which uses screenshots and runs in the background without switching apps, they said *"This sounds like it's a 1-stop thing like, you take a picture, is it good or bad? That one I don't mind so much."* (PA S1), This indicates that simpler add-on workflows, specifically those that run in the background and operate through familiar Siri-like interaction, could be more approachable.

5.2 User Needs and Add-On Functionality

5.2.1 Challenges with Existing AI Tools. Both CA and CJ saw value in the core functionality of the add-ons that we discussed and felt that they addressed important challenges with existing assistive AI applications. AI accuracy was cited as the top issue for both co-designers, with CA saying: *"My biggest thing with AI is the need to fact check it all the time. Sometimes AI will lie or hallucinate, and then it will double down on that hallucination. So there has to be some kind of human verification."* (CA, S1). Efficiency was also regarded as an important consideration. CJ said, *"I guess efficiency... that's one of the huge ones. That's how Shortcuts can help with that. Because my goal is always to get what I need as quickly and as accurately as possible."* (CJ, S1) These challenges confirm findings from prior work [1, 35], and generally match our initial feature designs, which were aimed at improving AI understanding and automating complex actions.

5.2.2 Preferred Add-On Functionality. Overall, co-designers felt that **A.1 Camera Aiming**, **A.2 Cross Checking**, and **A.3** and **A.3.M Image Quality** were the most immediately promising.

A.1 Camera Aiming, and **A.3 Image Quality** were consistently described as broadly useful, with co-designers expressing a strong desire to access these functionalities faster and in new settings. Co-designers instead gave feedback on the specific ways they wanted these features to work. For example, with **A.1 Camera Aiming**, CJ suggested including an option for users to specify what they wanted centered in the frame, enabling different guidance algorithms depending on the context. While implementing multiple guidance algorithms is beyond the scope of this work, this example illustrates how co-designers were supportive of these functionalities as add-ons and envisioned ways to extend them further.

A.2 Cross Checking was also seen as a desirable feature, though with additional concerns about its implementation and accuracy. CA and CJ both described cross-checking AI results manually, though this is time consuming which makes it infeasible to do frequently. In our testing in Session 2, our mock-up of **A.2** ran too many prompts with too many models, and both co-designers commented that it was too slow to be immediately useful.

5.2.3 Secondary Add-On Functionality. The remaining add-on functionalities discussed with co-designers (**A.4** to **A.7**) were not as strongly preferred. While these functionalities typically weren't immediately dismissed or disliked, co-designers felt these functionalities would need more refining through research before attempting to deploy them as add-ons. For instance, co-designers were split on **A.4 Routines**. While CA was not opposed to the idea and saw some use in automating frequent tasks, CJ was skeptical. Since CJ already knows what apps they want to use in a given scenario, the time savings from automation would be minimal. For **A.5 Error Reasoning** and **A.6 Explain Results**, co-designers were more skeptical that AI would actually have the capabilities to deliver useful or correct results for that task. They thought that these may be useful add-on features in the future, but that with current AI capabilities they may end up causing more confusion. Due to these preferences, we did not focus on these features in our implementation. A full analysis of these secondary features is included in Appendix C.1 for reference.

5.3 Triggering Add-Ons

In our Session 1, co-designers noted that it was sometimes difficult to remember the specific phrases used to trigger an add-on with Siri. As CA put it: *"Especially for like elderly people. They will not, most of the time, say the exact words that you're supposed to say to trigger a shortcut, because they don't remember."* (CA, S1). While some of this could be mitigated on the operating system level with better Siri assistance, in Session 2 we aimed to design supplemental mechanisms to help users recall and trigger add-on features.

One option that we discussed with co-designers was a **verbal menu**, triggered with a single phrase, versus needing to remember specific phrases for each add-on. CA emphasized that Siri interactions are familiar to novice VoiceOver users, as they typically rely heavily on it as they are learning. CA suggested ways to make the verbal menu even more approachable for novice users, by shortening the trigger phrase and by making options easier to select by adding numbered choices. They said: *"What I would like to see for something like this, is put numbers before each option. They can say 'one' instead of 'aim the camera'. It's like the old phone style system. 'To aim the camera, one, to check photo quality, three.'"* (CA, S2). While this does increase approachability and familiarity, CA

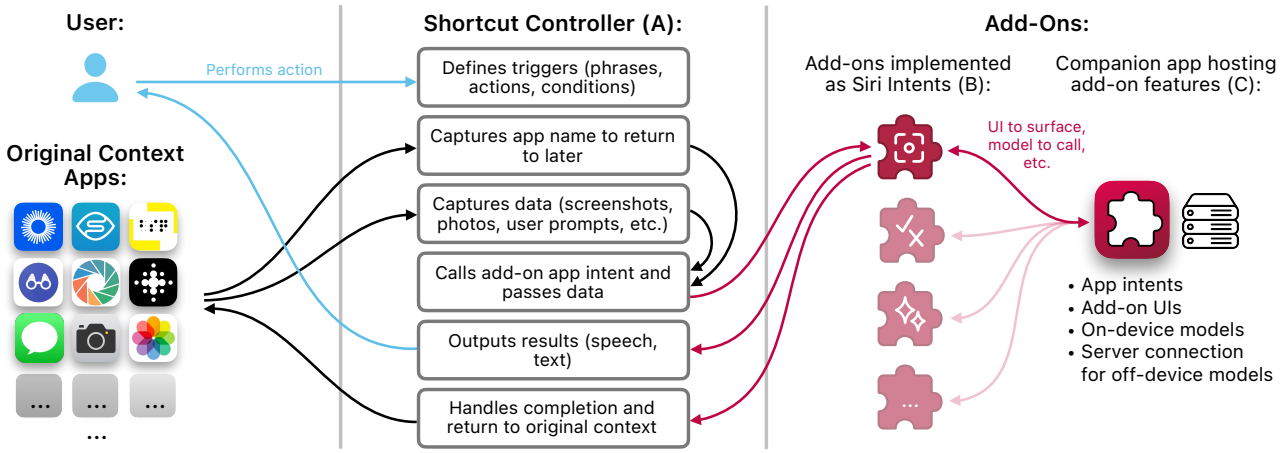

Figure 3: A11yExtensions system components. (A): A Shortcut associated with each add-on controls data, interface transitions, and output between the user's original app context and the add-on feature. (B): Add-on features are implemented as Siri intents that can be called from Shortcuts. (C): A companion app hosts the add-on endpoints including add-on UI, local models, and server connection infrastructure for off-device analysis.

also mentioned that speech interaction is not always ideal due to privacy or social comfort.

As an alternative to verbal interaction, we discussed the possibility of triggering add-ons with an **on-screen menu**. Selecting an item from the menu could trigger a Shortcut redirecting the user to their previous app to run the add-on, allowing the automated functionality of the add-on to be preserved. CJ appreciated the idea of on-screen menus, particularly as a backup for less-used options: *"I would strive to memorize the individual trigger phrases, or at least the ones I use the most. But if there was one I didn't use very often, I might be like 'Oh, what is that one called again?'"* (CJ, S2).

Co-designers also raised the idea of using **gesture interactions** or the **Share Sheet** for accessing add-ons in different contexts. Both co-designers mentioned a preference for gestures (e.g., holding the action button, or double tapping the back of the phone) as a fast and non-verbal way of triggering add-on features. Though, this does currently require some setup to link the intended Shortcut to the gesture. Overall, co-designers stressed the importance of providing a range of different mechanisms for triggering add-ons so that they can be as adaptable as possible to different users, settings, and tasks.

5.4 Capturing Data for Add-Ons

5.4.1 Capturing Visual Data. Many of the add-on concepts we discussed used visual data as input. As Shortcuts doesn't allow a way to automatically start screen recordings, we instead used screenshots of the user's current task, potentially similar to Apple's proposed On-Screen Awareness feature for Siri [77]. Both co-designers accepted this when it was for an add-on intentionally triggered by them. However, our initial concept for **A.3 Image Quality** explored a different approach, activating automatically when a photo-taking application opens, and taking screenshots periodically for continuous feedback. CJ said: *"It has to be very much opt-in and transparent to what it's doing and when... and having access to that information, I'm not sure I'd run it in the background, I want to initiate it."* (CJ, S1). If screenshots are only taken on user request and one at a time, the user is more in control. As CA described: *"As long as they're consciously doing it... they'll probably be more comfortable with it. It's the getting sent out in the background periodically... that will be the ultimate concern for a lot of other people."* (CA, S1). As a result, we redesigned **A.3** as **A.3.M Image Quality**, removing the method of capturing screenshots periodically in the background.

5.4.2 Capturing User Intent and Add-On Parameters. For some of the add-ons, co-designers noted that simply invoking a feature would not be specific enough.For instance, for **A.1 Camera Aiming**, CJ noted that guidance should vary based on intended subject and context (e.g., taking a photo to share versus taking a photo for AI). Later, CJ asked, *"Can you have a shortcut where you fill in part of it on the fly?"*, wanting to invoke an add-on by saying 'aim the camera at the plate' or similar, to receive contextual instructions. In Session 2, two of the mock add-ons we tested had a follow up question where users could provide a parameter (e.g., **A.2 Cross Checking** asked users what information to cross check). While this two-step process is not as streamlined, with some adjustments to the prompt language and audio cues it was acceptable to co-designers.

6 A11yExtensions Implementation

Based on preferences indicated in co-design Sessions 1-3 (Section 5.2), we implemented three add-ons: **A.1 Camera Aiming**, **A.2 Cross Checking**, and **A.3.M Image Quality**. Here, we describe the complete workflow and user experience of each add-on, and how add-ons are implemented.

A11yExtensions is implemented as three primary components (see Figure 3). First, each add-on feature has a corresponding Shortcut that controls the flow of information between the starting context and the companion app. Second, to connect shortcuts to the add-on functionality, the A11yExtensions companion app implements 'App Intents', an iOS feature that allows app functionality to be called by shortcuts or Siri. Finally, all add-ons are implemented

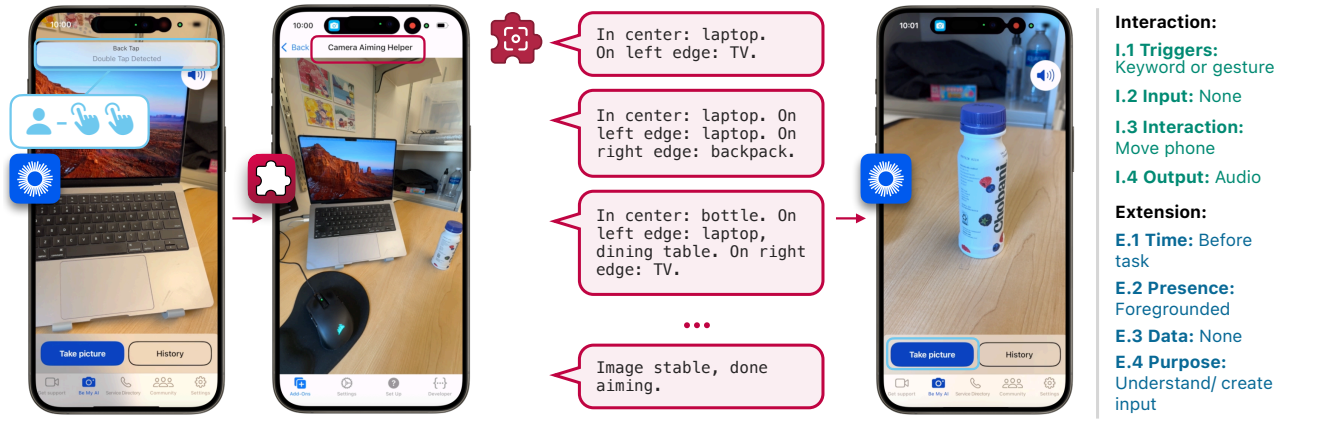

Figure 4: Summary of A.1 Camera Aiming. Users trigger the add-on by double tapping the back of the phone. They are then automatically redirected to the A11yExtensions companion app by the associated Shortcut, displaying the camera aiming interface. They receive verbal guidance on framing until the image is stable, then the Shortcut redirects them back to their original app to take the photo.

within a single iOS companion app, which functions as an add-on store and connects to the Shortcuts via App Intents.

6.1 Add-Ons

6.1.1 ***A.1 Camera Aiming.*** The camera aiming add-on is similar to our initial conceptualization (see Figure 4). Following co-designer interest in gesture triggers, the add-on is activated by double-tapping the back of the phone in photo-taking apps. This triggers the Shortcut associated with the add-on, which will then log the current app name and open the A11yExtensions companion app, navigating to the 'Aim Camera' sub-page in the interface automatically through a custom App Intent. The camera aiming feedback then starts automatically.

To focus on validating the idea of extensions, we opted for a simple implementation of aiming guidance, based on existing implementations and feedback from co-designers. Periodically, YOLO object detection is used to detect objects in frame. Once detections are stable, the current frame is described to the user, specifying which objects are centered in frame, and which objects are cut off at the edges (i.e., 'Laptop in center. Keyboard on left edge. Book on right edge.'). After three stable announcements of the center object, we assume the user has reached their target frame.

A final announcement states that the image is stable and aiming is complete. The Shortcut is notified of this completion by the App Intent, which switches the app back to the original source. The user can then keep their phone in the same position and take the photo. Although co-designers were interested in automating photo capture, this was not implemented due to feasibility constraints, as it would require external apps to support photo-taking intents.

6.1.2 ***A.2 Cross Checking.*** For cross-checking, users trigger the associated Shortcut with Siri by saying 'Hey Siri, cross check add-on' (see Figure 5). Once initiated, a single screenshot is taken, with an accompanying sound effect for awareness. The Shortcut will then prompt the user, saying 'What do you want to cross check? Speak after the beep.' Another sound effect plays, and the Shortcut will trigger a voice recording request, which automatically ends when the user stops speaking. The transcribed recording and screenshot are then passed from the Shortcut to the A11yExtensions companion app to be processed on an off-device server. The user remains in their current app while this is happening, and the Shortcut will play a continuous sound effect to let the user know it is still running.

Again with the goal of focusing on the add-on concept, we implemented cross-checking as a highly simplified and sped-up version of the verification question concept presented in GenAssist [40], which allows users to confirm visual details in AI generated images. The server first uses GPT-4o to process the request by breaking it down into a series of yes or no verification questions. Then, the verification questions and image are sent to GPT-4o and Gemini, which are prompted to give only brief answers that are easy to compare. If the models are in agreement, the user receives the response 'yes, that is correct'. If the models disagree, the user receives

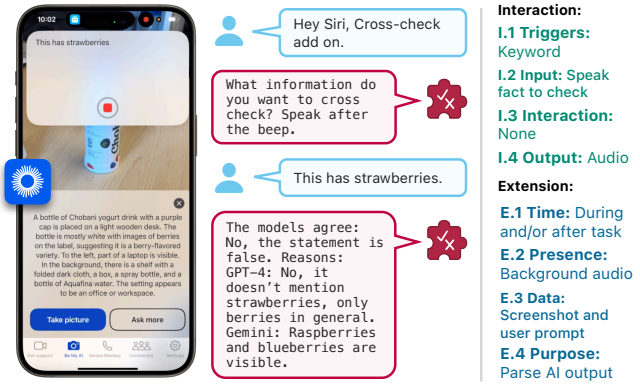

Figure 5: Summary of A.2 Cross Checking. Users trigger the add-on with Siri, and are then prompted by the associated Shortcut to verbally specify what information to check. The Shortcut then takes a screenshot, sending it to the associated app-intent which sends it to the server to analyze with multiple models. The results are then read by Siri.

an 'uncertain' response, along with a brief summary of the differences This summary is created by GPT-4o. Finally, once the result is returned from the server, it is spoken aloud by the Shortcut. We simplified this process to include fewer verification questions and models for speed. In Session 2 we originally included more questions and additional models, but we found that this increased the time to generate a response and the efficiency benefits of using an add-on were lost.

6.1.3 ***A.3.M Image Quality.*** The workflow of **A.3.M** is similar to that of **A.2 Cross Checking**, except it does not require any follow-up prompts from the user. Once triggered verbally, the associated Shortcut takes a screenshot which is processed off-device, using GPT-4o and a series of custom prompts designed to identify image quality issues, particularly those that are relevant to blind photography. This list was developed by heuristics and suggestions from CJ in Session 1. Finally, audio cues are played by the Shortcut when the screenshot is taken and when processing, and the result is spoken aloud.

7 Co-Design Session 4

After implementing the three add-ons (**A.1**, **A.2**, and **A.3.M**), we conducted a fourth co-design session with CA and CJ. We chose to do so as their involvement provided continuity of perspective, allowing them to assess how the implemented add-ons compared to earlier prototypes. We met with co-designers over a two-hour Zoom call to test the implemented features, and to reflect on the overall design process of A11yExtensions. Here, we present the methods used and the results from this session.

7.1 Method

In this final session, our goal was to test the fully-implemented prototypes in terms of their usability and utility. We aimed to understand if co-designers thought that the add-ons could help with core issues raised at the start of our co-design, namely AI understanding and efficiency, and if they met co-designer's goals of being efficient, controllable, and transparent.

To test each add-on individually, co-designers were asked to complete a simple task using each feature (e.g., photographing an object) and were encouraged to think aloud during the interaction. Next, co-designers were asked to integrate all three features in a more realistic workflow: preparing a hypothetical social media post. This was chosen as a task as it incorporates elements from each of the three add-ons, and was also previously mentioned by both co-designers as an area where they sometimes encounter challenges. Co-designers were informed that they did not have to follow a particular workflow and could choose when and how to access add-ons. Finally, we conducted a semi-structured interview to reflect on the testing and broader design process.

7.2 Post-Implementation Design Findings

7.2.1 Add-On Understanding and Workflows. Co-designers found the implemented add-ons fairly seamless, more so than in our initial sessions. CA said of **A.1**: *"It was definitely much smoother than before. I liked its responsiveness when it was switching, as soon as it was done aiming it put me right back into Be My Eyes."* (CA, S4). While shortcuts may have initially come across as a complex or clunky feature, co-designers appreciated the idea after implementation. As CA said *"Once you guys implemented that and I saw it in that Shortcut, I was like, well, this is actually really cool."* (CA, S4).

Co-designers appreciated additional audio cues and phrasing modifications that we made from prior feedback, and made suggestions for additional changes. For instance, CA mentioned wanting a cue announcing the name of the current app when automatic switching is performed: *"When it switched to Be My Eyes, it did announce that it was going back to Be My Eyes... So there was a small cue. It doesn't tell you that it's opening up A11yExtensions though... If it's possible to have, that would be great."* (CA, S4). Onboarding and training could continue to further address concerns about understanding add-ons. CA mentioned that as a technology trainer, they would narrate the Shortcut automation to users to explain what the add-ons do and how. Alternatively, CJ mentioned that over time, shorter prompts might be desirable for users that prioritize efficiency: *"I would definitely shorten the whole 'speak after the beep'. I mean once you use it the first time or two you could take it out. With some people if they've used this enough, you could make the prompts as short as, 'Verify what?'"* (CJ, S4). Different populations will have different needs in regards to feedback levels, and how to taper feedback over time.

Finally, co-designers felt that add-ons were extensible and could easily apply to new contexts. For instance, CA described how a modified version of **A.1 Camera Aiming** for centering documents would be useful: *"There are some text reading applications like Voice Dream Reader, it will only give you indications of how good the page is via a tone. And so, for my Deaf and hard of hearing blind students it's not a practical way of taking a picture. But if we were able to use this add-on to get some guidance [for framing text], that would make life a lot easier... There are a lot of apps that you could theoretically have this shortcut handshake with."* (CA, S4). When completing the social media post task, CJ mentioned that they would still bring additional tools into their workflow to finalize the post: *"I think [this] definitely would get me a lot closer. Depending on what I'm doing, I might still then bring in a tool to crop. It ties into everything... I would probably use your tools along with other tools."* (CJ, S4). This is the ideal case for add-ons: features that can seamlessly fit into new workflows with existing apps to complement their use quickly. Add-on features can act as building blocks: *"Each [add-on] was kind of sandboxed into itself, which is good. And each tries to do their own little thing. So I think that helps."* (CJ, S4).

7.2.2 Add-On Privacy. While we previously discussed with co-designers their privacy concerns regarding automating screenshots and data capture, testing the add-ons raised an additional privacy question. When asked what aspects of automatically switching between apps would cause concern, CA said: *"It depends how much of that gets shared with your servers. Like does any of it, the pictures... end up on your server? If the Shortcut is just used as a way to aim and nothing else, and everything else is [done] on the phone or through Be My Eyes, then the user just has the inherent risk of taking a picture through Be My Eyes."* (CA, S4). Users may know the risks of apps they are already familiar with and trained to use, but less so with new applications. Add-ons may introduce one or multiple additional apps into a single task workflow, making it harder to pinpoint where data is going at each stage.

7.2.3 Final Add-On Impressions and Concept Reflections. Throughout the design process, co-designers reflected on the tradeoffs between add-on or extension features, versus more ideal but often unrealized native accessibility features. After testing, co-designers reflected on this further. CA summarized their opinion as still somewhat split. On one hand, using Siri can be beneficial for people who are less familiar with VoiceOver, as they may use Siri more frequently; on the other, setup is still complex. In their words:

"It is a very interesting approach, and it requires a little bit more of, not necessarily configuration, but work on the part of the user. That work is not immensely complicated... On one hand, it's good because it helps automate a lot of these [features] through Siri. On the other hand, setup for less advanced users, they'll probably still need a little help in all honesty. But once they get it set up, and running, it can be more or less automated. I don't dislike the approach, but some users will be uncomfortable with it. Still, like ultimately seeing now more of a complete product, it's not a bad approach." (CA, S4).

CJ had a similar perspective, highlighting that although Shortcuts can be time consuming to set up, they provide opportunities in the form of new feature testing and versatility:

"I think [native] apps work a little more seamlessly? Because you don't have all the permission things happening. And it's the tradeoff, I think Shortcuts are kind of good for these quick actions. I think Shortcuts is a good testing bed, for sure. Try out new things to see what works, what doesn't... You give people the versatility because you can access Shortcuts from from Siri, or just from the app, or from the the back-tap, or whatever you want, so that can be interesting as well. You wouldn't have that flexibility if they were just an app feature." (CJ, S4)

Ultimately, co-designers saw advantages to the A11yExtensions approach, in terms of efficiency, ability to test new features, the familiarity of Siri and verbal interaction, and a new form of versatility in interaction. Especially after testing the polished workflows in the implemented add-ons, co-designers appreciated efficiency and workflow benefits more directly, with both commenting on the ability to complete tasks in one place, without jumping between applications or platforms. Setup costs are the primary concern, and co-designers hoped that as Apple continues to develop Siri intelligence, that setup can become more streamlined and approachable.

8 Design Space

From the co-design sessions, we reflect on the design possibilities of mobile extensions for accessibility by creating a design space. We analyzed the 8 distinct add-on features described in Section 4, and additional variations, add-on ideas, and tasks raised by co-designers by coding common features. The resulting design space that we propose includes a set of 8 dimensions centered on two themes, summarized in Figure 6:

(1) **Interaction dimensions:** How do users interact with an add-on?
(2) **Extension dimensions:** What do add-on features do, and how do they function?

Design spaces have long organized knowledge in HCI [15, 27], to both classify and generate a range of interactions, interfaces, and devices [37, 61, 69, 71, 79], including applications in accessibility [6, 62, 66]. Design spaces can be thought of as both descriptive, used as a method to analyze and compare design options, and generative, used to produce future designs though an enumeration of possibilities [24, 52, 72].

The design space proposed here aims to serve both a descriptive and generative role, in that it can explain how features of our implemented add-ons in A11yExtensions work and differ, and can spark ideas for new future extensions. To the latter point, some of the dimensions listed here include functionalities that are technically possible but that do not yet exist due to platform limitations. These are indicated, and can help to imagine how extensions may function in the future.

8.1 Interaction Dimensions

I.1. Triggers: How do users trigger or activate an add-on feature? Triggers can be *manual* with varying degrees of user control, for instance: using specific verbal commands, a verbal menu, Share Sheet, gesture, the menu within the Shortcuts app interface, or simply using the feature as a standalone app. Additionally, with developer support, add-ons could be accessed via buttons or links included within other applications, as Shortcuts can be triggered with the URL scheme: `shortcuts://open-shortcut?name=[name]` [9]. Triggers can also be fully *automatic*. iOS supports several automatic triggers for Shortcuts, including location, time, or opening specified applications. In the future, new automatic triggers could include the presence of certain visual content (e.g., activating an add-on if there is a specific object in frame). Both co-designers expressed a preference for manual triggers, while automatic triggers would need more onboarding support and transparency for acceptance.

I.2. Input Mechanisms: What data do users have to actively provide to the add-on, and how do they do so? In most of the add-ons we tested during co-design, input was limited to one screenshot (which was automated after the manual trigger) and optionally, a user prompt, which was collected via a verbal follow-up query. In some cases, co-designers imagined providing more specification, for instance with **A.2 Cross Checking**, CJ envisioned being able to pre-select specific or customized models, or to ask follow up queries in a conversation.

I.3. Interaction & Automation Level: Once an add-on has been initiated, what level of interaction is necessary by users to be able to complete their task or goal? This is separate from the level of automation of the *trigger* for the add-on. Add-ons we designed required different degrees of interaction. For instance, with **A.2 Cross Check** and **A.3.M Image Quality**, once the add-on is triggered and input is provided, the rest of the functionality will run without additional user interaction. However, this is not always the case. **A.1 Camera Aiming** has a higher degree of interaction, where users need to actively move around to frame their shot and manually take photos. Future add-ons could include more traditional interfaces for users to navigate, such as maps, menus, or elements to explore via touch (as in ImageExplorer [47] or VizLens [33]).

I.4. Output Types and Verbosity: How is information presented? The majority of our add-ons deliver verbal output through Siri, with audio cues played through background audio. Co-designers also imagined add-on results being copied to a clipboard or saved elsewhere so that they could export it to other contexts, for instance,

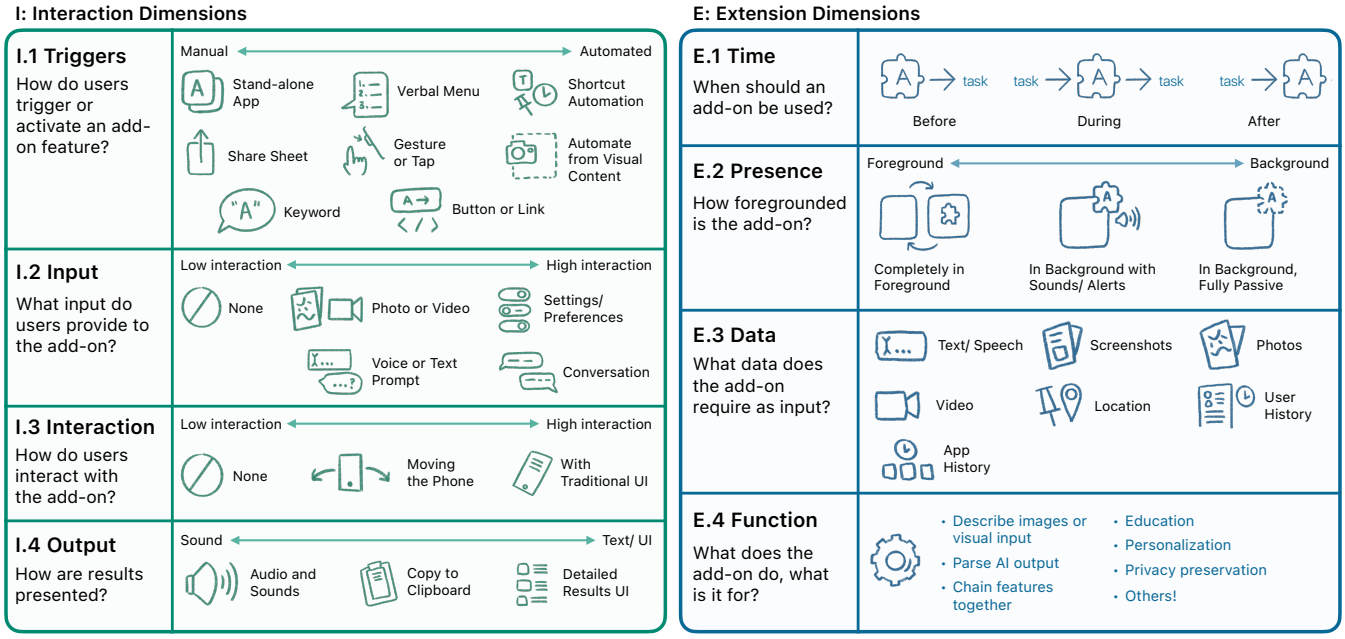

Figure 6: Proposed design space of mobile extensions for accessibility, centered on two themes: Interaction Dimensions (How do users interact with an add-on?) and Extension Dimensions (What do add-on features do, and how do they function?)

sending a description in a text message. Additionally, co-designers discussed the possibility of a smaller amount of output being verbalized, and a more detailed version presented in a separate interface for users to access with screenreaders.

8.2 Extension Dimensions

E.1. Time: When should an add-on be used, relevant to the users existing task? The add-ons that we discussed were frequently intended to be used at a specific interaction point relative to an existing application. This could be before, during, and/or after the existing interaction takes place. For instance, when using **A.1 Camera Aiming** to supplement Be My AI, it should be activated immediately before a photo is taken and analyzed. In contrast, **A.3.M Image Quality** can be used any time there is a photo on the screen: it can be used before taking the photo to check lighting conditions, while the user is getting a result from Be My AI to confirm details, or afterward on a photo from the camera roll.

E.2. Presence: How foregrounded is the add-on when running? When the user runs the add-on, how much are they redirected from their current state? **A.1 Camera Aiming** runs completely in the foreground, automatically redirecting the user from their current app, while **A.3.M Image Quality** runs completely in the background, and audio output is overlaid onto whatever the user is already doing.

E.3. Data Captured: What data is taken as input for the add-on? This could include any combination of text prompts, screenshot(s), photo(s), live camera video, or screen recording. Future data could include location, or a users prompt or app-use history. Finally, some add-ons take no input, like **A.1 Camera Aiming**. This is closely related to dimension I.3 Input Mechanisms, though we note that data captured by add-ons can be done automatically without additional user interaction.

E.4. Extension Purpose & Functionality: What is the add-on for, and what does it do? Addressing challenges around AI errors and understanding, many of the add-ons we focused on aimed to describe visual input and parse AI output. However, we also discussed add-ons for chaining existing features together (e.g., **A.4 Routines**), educating novices (e.g., **A.5 Error Reasoning**), or allowing for higher personalization (e.g., CJ suggested an add-on that would store frequent prompts in a geo-tagged library for retrieval). This is not a comprehensive list, and there are other possibilities.

9 Discussion and Future Work

In this work, we presented the design process of A11yExtensions, a set of add-on extensions for bridging new accessibility features with existing technology workflows. A11yExtensions was created with two blind co-designers, however, it is not finalized. Here, we discuss its limitations and avenues for future work.

9.1 Study Limitations and Approachability for Novice Users

A11yExtensions explores a new interaction paradigm that could influence how assistive technologies are researched and developed in the future. At the same time, we acknowledge that this vision is not yet complete, and this work has several limitations. Primarily, given the longitudinal engagement with two professional accessibility consultants as co-designers, A11yExtensions has not been validated with a broad audience. Assessing A11yExtensions with

a wide range of participants is a critical next step in this work. Given our co-designers' unique experience consulting with other blind people through their work, they were able to highlight where A11yExtensions may pose challenges for a broader audience. As implemented, add-ons suit users who are already comfortable with automation, extensions, programming, and AI. Beyond the difficulties of installation and setup, more substantial barriers to approachability relate to the concepts of automation and multi-app coordination. For instance, how do users understand when an add-on automatically directs them from one service to another when a condition is met? How can users be made aware of what those conditions are, and how a system is acting on their behalf? Future evaluation work should focus critically on evaluating users mental models of add-ons and automation, and work towards techniques that are easily understandable.

9.2 Extensions vs. Native App Improvements

Throughout the co-design process, we discussed the tension between the ideal of significantly improving native applications, and using potentially less-natural extensions to modify the user experience. This is a tension broadly present in the DIY assistive technology community, where it can sometimes feel like the burden is placed on users to create accessible experiences for themselves, rather than accessibility being rightfully provided as a default [16, 35]. A11yExtensions aims to address underlying accessibility issues in already deployed applications and user workflows, which users must put in additional effort to deal with [35]; yet, there is an additional setup cost. Our hope is that with well-designed onboarding and defaults, the benefits can outweigh the cost.

One future possibility is making add-ons that integrate more seamlessly with native applications, blurring the boundary between native app and extension. Since native apps risk bloating when overloaded with features, letting users embed only the add-ons they need could preserve usability. With LLMs now able to generate code, add-ons could instantly match an app's look and feel, enabling hyper-personalized software that moves beyond one-size-fits-all design.

Another avenue for future work would be to investigate accessible, end-user trigger-action programming for extensions, ideally minimizing setup costs as much as possible. Such methods already support DIY assistive technologies; for example, ProgramAlly's block-based interface lets users build visual filters with the structure '`find item on item`' [36]. Extending this paradigm, lightweight '`if this then that`' patterns could be used support more automation. Future work would need to consider how to make these logical statements easily understandable to a wide range of users, and how to allow users to specify triggers in a fast and robust way.

9.3 Generalizing to New Apps and Platforms

We chose to design A11yExtensions with iOS as it is historically widely used among the accessibility community [60]. However, as compared to other platforms, iOS is extremely conscious of platform consistency, user control, and privacy, which limits the add-ons we could implement and discuss. Each Shortcut needs to be manually installed by the user, with specific permissions requested on its first run, and actions such as screen sharing cannot be automated at all. While we aimed to elaborate future possibilities in our design space separately from implementation limitations, it is possible that new aspects could emerge as new affordances are created by the platform. For instance, on Android, user interactions can be easily captured in the background [10, 50], providing a very high-granularity 'user history' (E.3. Data Captured). Yet, we believe the design values presented in this work (e.g., transparency, user awareness, and privacy) remain applicable for add-ons across platforms, regardless of technical differences.

Additionally, our concept of add-ons focused on augmenting common existing assistive AI applications (e.g., Seeing AI [59], Be My AI [26]). These apps follow common AI design paradigms (e.g., operating on a live camera view or a static image and providing text feedback), so co-designers were able to generate examples of extending add-ons to use with other apps that also followed similar patterns. While our proposed design space fits well with apps of that design paradigm, it will need to shift as AI-based assistive technology evolves, potentially including tactile output [49, 75], wearables [58, 63, 80], or new hardware devices [53, 70].

9.4 Extension Privacy

Add-ons can introduce an additional space for data leaks, and potentially obfuscate where data is going, given that data can be automatically passed along through multiple apps and services. For instance, CA and CJ each occasionally expressed not being certain if data was being accessed or saved by Shortcuts, or by our companion app. Additionally, for add-ons that run fully in the background, they noted that it was not always clear what was processed locally or off-device. This mirrors privacy challenges of traditional web extensions [2, 3, 45], where users often lack awareness of what data extensions are capable of capturing due to the language and technical complexity of permission requests. Overall, the complexity of add-on workflows muddies the water on the control users feel over their data. In deployment, add-ons need to be transparent and specific about their data handling practices, moreso than typical AI assistive applications due to their automated nature.

9.5 Extensions as a Platform For Research

We also imagine A11yExtensions as a possible platform for future accessibility research. In this work, we were inspired by research leveraging 'one sec', which was later used as a platform for a large-scale study comparing intervention strategies [32, 34]. As it stands, researchers often need to deploy entire, complex applications to test a small number of novel features. Beyond easing technical barriers of deploying and maintaining applications, an add-on ecosystem could also broaden participation in accessibility research. Ideally, add-ons would enable researchers and blind users to more rapidly prototype and share ideas, with researchers creating add-on features, and users devising new and creative uses. Researchers would be able to test features within the context of use, running studies with participants' everyday devices rather than in constrained lab settings. Realizing this vision will be a long-term effort, and will require supporting and growing a community of users, and industry partnerships. At present, our add-ons are constrained by the limited set of actions that can be automated. Their potential could

be expanded through open APIs, broader adoption of App Intents, and platform-level support for distribution and setup.

10 Conclusion

We have presented A11yExtensions, a design space and set of in-situ extensions that augment existing mobile AI assistive technology with add-on services. A11yExtensions was developed through multiple co-design sessions with two blind accessibility consultants to be useful, transparent, and privacy preserving. The resulting 'add-ons' represent a new paradigm in mobile interaction, leveraging automation as a method to introduce new, targeted features for usability and utility, providing an additional avenue for future customization of mobile assistive technology experiences. Overall, A11yExtensions is a design probe working towards bridging HCI and accessibility research and practice to create more effective assistive AI applications for blind people.

Acknowledgments

First and foremost, we would like to thank our co-designers and co-authors Ather and J.J. for their time and commitment to this work. This research would not have been possible without them. We would also like to thank our reviewers for their time and feedback. This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE 2241144, and the National Science Foundation CAREER Award No. 2442243. Any opinion, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. This research was also supported in part by a Google Research Scholar Award.

References

[1] Rudaiba Adnin and Maitraye Das. 2024. "I look at it as the king of knowledge": How Blind People Use and Understand Generative AI Tools. In *Proceedings of the 26th International ACM SIGACCESS Conference on Computers and Accessibility*. 1–14. doi:10.1145/3663548.3675631

[2] Shubham Agarwal, Rafael Mrowczynski, Maria Hellenthal, and Ben Stock. 2025. "I have no idea how to make it safer": studying security and privacy mindsets of browser extension developers. In *Proceedings of the 34th USENIX Conference on Security Symposium* (Seattle, WA, USA) *(SEC '25)*. USENIX Association, USA, Article 151, 20 pages.

[3] Mohammad M Ahmadpanah, Matías F Gobbi, Daniel Hedin, Johannes Kinder, and Andrei Sabelfeld. 2024. CodeX: Contextual Flow Tracking for Browser Extensions. In *Proceedings of the Fifteenth ACM Conference on Data and Application Security and Privacy*. 6–17.

[4] Rahaf Alharbi, Angela D. Cheong, Jaylin Herskovitz, Robin N. Brewer, and Sarita Schoenebeck. 2025. "Trying to Piece It Together": Exploring Accessible Error Detection in Emerging Privacy Techniques With Blind People.

[5] Rahaf Alharbi, Pa Lor, Jaylin Herskovitz, Sarita Schoenebeck, and Robin N. Brewer. 2024. Misfitting With AI: How Blind People Verify and Contest AI Errors. In *Proceedings of the 26th International ACM SIGACCESS Conference on Computers and Accessibility*. 1–17.

[6] Oliver Alonzo, Sooyeon Lee, Akhter Al Amin, Mounica Maddela, Wei Xu, and Matt Huenerfauth. 2024. Design and Evaluation of an Automatic Text Simplification Prototype with Deaf and Hard-of-hearing Readers. In *Proceedings of the 26th International ACM SIGACCESS Conference on Computers and Accessibility*. 1–18.

[7] Anon. 2025. r/shortcuts. https://www.reddit.com/r/shortcuts/

[8] Apple. 2022. Shortcuts User Guide. https://support.apple.com/guide/shortcuts/welcome/ios

[9] Apple. 2025. Shortcuts User Guide: Open and Create a Shortcut Using a URL Scheme on iPhone or iPad. https://support.apple.com/guide/shortcuts/open-create-and-run-a-shortcut-apda283236d7/ios

[10] Deniz Arsan, Carl Guo, Muhammad Rizky Wellyanto, Erik R Ji, Jerry O Talton, and Ranjitha Kumar. 2025. On-Device Interaction Mining. *Proceedings of the ACM on Human-Computer Interaction* 9, 5 (2025), 1–18.

[11] Gagan Bansal, Besmira Nushi, Ece Kamar, Walter S. Lasecki, Daniel S. Weld, and Eric Horvitz. 2019. Beyond Accuracy: The Role of Mental Models in Human-AI Team Performance. In *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, Vol. 7. 2–11.

[12] Umang Bhatt, Javier Antorán, Yunfeng Zhang, Q Vera Liao, Prasanna Sattigeri, Riccardo Fogliato, Gabrielle Melançon, Ranganath Krishnan, Jason Stanley, Omesh Tickoo, et al. 2021. Uncertainty as a Form of Transparency: Measuring, Communicating, and Using Uncertainty. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*. 401–413.

[13] Jeffrey P. Bigham, Irene Lin, and Saiph Savage. 2017. The Effects of "Not Knowing What You Don't Know" on Web Accessibility for Blind Web Users. In *Proceedings of the 19th International ACM SIGACCESS Conference on Computers and Accessibility*. 101–109.

[14] Robin N Brewer. 2018. Facilitating discussion and shared meaning: Rethinking co-design sessions with people with vision impairments. In *Proceedings of the 12th EAI International Conference on Pervasive Computing Technologies for Healthcare*. 258–262.

[15] Stuart K. Card, Jock D. Mackinlay, and George G. Robertson. 1990. The Design Space of Input Devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 117–124.

[16] Yoonha Cha, Victoria Jackson, Karina Kohl, Rafael Prikladnicki, André van der Hoek, and Stacy Branham. 2025. The Dilemma of Building Do-It-Yourself (DIY) Solutions For Workplace Accessibility. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*. 1–17.

[17] Ruei-Che Chang, Yuxuan Liu, and Anhong Guo. 2024. WorldScribe: Towards Context-Aware Live Visual Descriptions. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology*. 1–18.

[18] Meng Chen, Akhil Iyer, and Amy Pavel. 2025. Surfacing Variations to Calibrate Perceived Reliability of MLLM-generated Image Descriptions. *arXiv preprint arXiv:2507.15692* (2025).

[19] Chrome-Stats. 2025. Chrome Extension Statistics (Jun. 2025). https://chrome-stats.com/chrome/stats

[20] Maitraye Das, Thomas Barlow McHugh, Anne Marie Piper, and Darren Gergle. 2022. Co11ab: Augmenting Accessibility in Synchronous Collaborative Writing for People with Vision Impairments. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*. 1–18.

[21] Melissa Dawe. 2006. Desperately Seeking Simplicity: How Young Adults with Cognitive Disabilities and Their Families Adopt Assistive Technologies. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 1143–1152. doi:10.1145/1124772.1124943

[22] Morgan Dixon and James Fogarty. 2010. Prefab: Implementing Advanced Behaviors Using Pixel-Based Reverse Engineering of Interface Structure. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 1525–1534.

[23] Finale Doshi-Velez and Been Kim. 2017. Towards a Rigorous Science of Interpretable Machine Learning. *arXiv preprint arXiv:1702.08608* (2017).

[24] Graham Dove, Nicolai Brodersen Hansen, and Kim Halskov. 2016. An Argument for Design Space Reflection. In *Proceedings of the 9th Nordic Conference on Human-Computer Interaction*. 1–10.

[25] Daniel A. Epstein, Fannie Liu, Andrés Monroy-Hernández, and Dennis Wang. 2022. Revisiting Piggyback Prototyping: Examining Benefits and Tradeoffs in Extending Existing Social Computing Systems. *Proceedings of the ACM on Human-Computer Interaction* 6, CSCW2 (2022), 1–28.

[26] Be My Eyes. 2024. Introducing: Be My AI. https://www.bemyeyes.com/blog/introducing-be-my-ai

[27] James D. Foley, Victor L. Wallace, and Peggy Chan. 1984. The Human Factors of Computer Graphics Interaction Techniques. *IEEE Computer Graphics and Applications* 4, 11 (1984), 13–48.

[28] Freedom Scientific. 2025. New and Improved Features in JAWS - Picture Smart AI. https://www.freedomscientific.com/training/jaws/new-and-improved-features/

[29] James Gashel. 2014. A New Era in Mobile Reading Begins: Introducing the KNFB Reader for iOS. https://nfb.org/sites/default/files/images/nfb/publications/bm/bm14/bm1411/bm141107.htm

[30] Google. 2022. Lookout - Assisted Vision. https://play.google.com/store/apps/details?id=com.google.android.apps.accessibility.reveal

[31] Catherine Grevet and Eric Gilbert. 2015. Piggyback Prototyping: Using Existing, Large-Scale Social Computing Systems to Prototype New Ones. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. 4047–4056.

[32] David J. Grüning, Frederik Riedel, and Philipp Lorenz-Spreen. 2023. Directing Smartphone Use Through the Self-Nudge App one sec. *Proceedings of the National Academy of Sciences* 120, 8 (2023), e2213114120.

[33] Anhong Guo, Xiang 'Anthony' Chen, Haoran Qi, Samuel White, Suman Ghosh, Chieko Asakawa, and Jeffrey P. Bigham. 2016. VizLens: A Robust and Interactive Screen Reader for Interfaces in the Real World. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*. 651–664.

[34] Luke Haliburton, David Joachim Grüning, Frederik Riedel, Albrecht Schmidt, and Nađa Terzimehić. 2024. A Longitudinal In-the-Wild Investigation of Design Frictions to Prevent Smartphone Overuse. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*. 1–16.

[35] Jaylin Herskovitz, Andi Xu, Rahaf Alharbi, and Anhong Guo. 2023. Hacking, Switching, Combining: Understanding and Supporting DIY Assistive Technology Design by Blind People. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. 1–17.

[36] Jaylin Herskovitz, Andi Xu, Rahaf Alharbi, and Anhong Guo. 2024. ProgramAlly: Creating Custom Visual Access Programs via Multi-Modal End-User Programming. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology*. 1–15.

[37] Teresa Hirzle, Jan Gugenheimer, Florian Geiselhart, Andreas Bulling, and Enrico Rukzio. 2019. A Design Space for Gaze Interaction on Head-Mounted Displays. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–12.

[38] Jonggi Hong and Hernisa Kacorri. 2024. Understanding How Blind Users Handle Object Recognition Errors: Strategies and Challenges. In *Proceedings of the 26th International ACM SIGACCESS Conference on Computers and Accessibility*. 1–15.

[39] Jeremy Zhengqi Huang, Jaylin Herskovitz, Liang-Yuan Wu, Cecily Morrison, and Dhruv Jain. 2025. Weaving Sound Information to Support Real-Time Sensemaking of Auditory Environments: Co-Designing with a DHH User. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*. 1–18.

[40] Mina Huh, Yi-Hao Peng, and Amy Pavel. 2023. GenAssist: Making Image Generation Accessible. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*. 1–17.

[41] Chandrika Jayant, Hanjie Ji, Samuel White, and Jeffrey P. Bigham. 2011. Supporting Blind Photography. In *The Proceedings of the 13th International ACM SIGACCESS Conference on Computers and Accessibility*. 203–210.

[42] Kevin Jones. 2023. How for Blind People, the iPhone Significantly Changed the World of Cell Phones Even More Than for the Sighted. https://accessibility-insights.com/2023/05/16/how-for-blind-people-the-iphone-significantly-changed-the-world-of-cell-phones-even-more-than-for-the-sighted/

[43] Hernisa Kacorri, Kris M. Kitani, Jeffrey P. Bigham, and Chieko Asakawa. 2017. People with Visual Impairment Training Personal Object Recognizers: Feasibility and Challenges. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 5839–5849.

[44] Rie Kamikubo, Lining Wang, Crystal Marte, Amnah Mahmood, and Hernisa Kacorri. 2022. Data Representativeness in Accessibility Datasets: A Meta-Analysis. In *Proceedings of the 24th International ACM SIGACCESS Conference on Computers and Accessibility*. 1–15.

[45] Ankit Kariryaa, Gian-Luca Savino, Carolin Stellmacher, and Johannes Schöning. 2021. Understanding users' knowledge about the privacy and security of browser extensions. In *seventeenth symposium on usable privacy and security (SOUPS 2021)*. 99–118.

[46] Cheuk Yin Phipson Lee, Zhuohao Zhang, Jaylin Herskovitz, JooYoung Seo, and Anhong Guo. 2022. CollabAlly: Accessible Collaboration Awareness in Document Editing. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*. 1–17.

[47] Jaewook Lee, Jaylin Herskovitz, Yi-Hao Peng, and Anhong Guo. 2022. ImageExplorer: Multi-Layered Touch Exploration to Encourage Skepticism Towards Imperfect AI-Generated Image Captions. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*. 1–15.

[48] Kyungjun Lee, Jonggi Hong, Simone Pimento, Ebrima Jarjue, and Hernisa Kacorri. 2019. Revisiting Blind Photography in the Context of Teachable Object Recognizers. In *The 21st International ACM SIGACCESS Conference on Computers and Accessibility*. 83–95.

[49] Wan-Chen Lee, Ching-Wen Hung, Chao-Hsien Ting, Peggy Chi, and Bing-Yu Chen. 2023. TacNote: Tactile and audio note-taking for non-visual access. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*. 1–14.

[50] Florian Lettner, Christian Grossauer, and Clemens Holzmann. 2014. Mobile interaction analysis: towards a novel concept for interaction sequence mining. In *Proceedings of the 16th international conference on Human-computer interaction with mobile devices & services*. 359–368.

[51] Chen Liang, Yasha Iravantchi, Thomas Krolikowski, Ruijie Geng, Alanson P Sample, and Anhong Guo. 2023. BrushLens: Hardware Interaction Proxies for Accessible Touchscreen Interface Actuation. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*. 1–17.

[52] Youn-Kyung Lim, Erik Stolterman, and Josh Tenenberg. 2008. The Anatomy of Prototypes: Prototypes as Filters, Prototypes as Manifestations of Design Ideas. *ACM Transactions on Computer-Human Interaction (TOCHI)* 15, 2 (2008), 1–27.

[53] Guanhong Liu, Tianyu Yu, Chun Yu, Haiqing Xu, Shuchang Xu, Ciyuan Yang, Feng Wang, Haipeng Mi, and Yuanchun Shi. 2021. Tactile compass: Enabling visually impaired people to follow a path with continuous directional feedback. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. 1–13.

[54] Tao Lu, Hongxiao Zheng, Tianying Zhang, Xuhai "Orson" Xu, and Anhong Guo. 2024. InteractOut: Leveraging Interaction Proxies as Input Manipulation Strategies for Reducing Smartphone Overuse. In *Proceedings of the 2024 CHI conference on human factors in computing systems*. 1–19.

[55] Kelly Avery Mack, Kate S. Glazko, Jamil Islam, Megan Hofmann, and Jennifer Mankoff. 2024. "It's like Goldilocks:" Bespoke Slides for Fluctuating Audience Access Needs. In *Proceedings of the 26th International ACM SIGACCESS Conference on Computers and Accessibility*. 1–15.

[56] Haley MacLeod, Cynthia L. Bennett, Meredith Ringel Morris, and Edward Cutrell. 2017. Understanding Blind People's Experiences with Computer-Generated Captions of Social Media Images. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 5988–5999.

[57] Richard MacManus. 2023. Twitter in 2007: The Open Platform That Wasn't. https://cybercultural.com/p/twitter-in-2007-the-open-platform/

[58] Florian Mathis and Johannes Schöning. 2025. LifeInsight: Design and Evaluation of an AI-Powered Assistive Wearable for Blind and Low Vision People Across Multiple Everyday Life Scenarios. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*. 1–25.

[59] Microsoft. 2021. Seeing AI. https://www.microsoft.com/en-us/ai/seeing-ai

[60] John Morris and James Mueller. 2014. Blind and Deaf Consumer Preferences for Android and iOS Smartphones. In *Inclusive Designing: Joining Usability, Accessibility, and Inclusion*. Springer, 69–79.

[61] Meredith Ringel Morris, Carrie J. Cai, Jess Holbrook, Chinmay Kulkarni, and Michael Terry. 2023. The Design Space of Generative Models. *arXiv preprint arXiv:2304.10547* (2023).

[62] Meredith Ringel Morris, Jazette Johnson, Cynthia L. Bennett, and Edward Cutrell. 2018. Rich Representations of Visual Content for Screen Reader Users. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 1–11.

[63] Cecily Morrison, Edward Cutrell, Martin Grayson, Anja Thieme, Alex Taylor, Geert Roumen, Camilla Longden, Sebastian Tschiatschek, Rita Faia Marques, and Abigail Sellen. 2021. Social Sensemaking with AI: Designing an Open-ended AI experience with a Blind Child. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. 1–14.

[64] Cecily Morrison, Martin Grayson, Rita Faia Marques, Daniela Massiceti, Camilla Longden, Linda Wen, and Edward Cutrell. 2023. Understanding Personalized Accessibility through Teachable AI: Designing and Evaluating Find My Things for People who are Blind or Low Vision. In *Proceedings of the 25th International ACM SIGACCESS Conference on Computers and Accessibility*. 1–12.

[65] Rosiana Natalie, Ruei-Che Chang, Smitha Sheshadri, Anhong Guo, and Kotaro Hara. 2024. Audio Description Customization. In *Proceedings of the 26th International ACM SIGACCESS Conference on Computers and Accessibility*. 1–19.

[66] Hugo Nicolau, André Rodrigues, André Santos, Tiago Guerreiro, Kyle Montague, and João Guerreiro. 2019. The Design Space of Nonvisual Word Completion. In *Proceedings of the 21st International ACM SIGACCESS Conference on Computers and Accessibility*. 249–261.

[67] NVDA Project. 2025. Welcome to the NVDA Community Add-Ons Website. https://addons.nvda-project.org/index.en.html

[68] National Federation of the Blind. 2022. KNFB Reader (OneStep Reader). https://nfb.org/programs-services/knfb-reader

[69] Payod Panda, Molly Jane Nicholas, David Nguyen, Eyal Ofek, Michel Pahud, Sean Rintel, Mar Gonzalez-Franco, Ken Hinckley, and Jaron Lanier. 2023. Beyond Audio: Towards a Design Space of Headphones as a Site for Interaction and Sensing. In *Proceedings of the 2023 ACM Designing Interactive Systems Conference*. 904–916.

[70] Adil Rahman, Md Aashikur Rahman Azim, and Seongkook Heo. 2023. Take my hand: Automated hand-based spatial guidance for the visually impaired. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. 1–16.

[71] Hans-Jörg Schulz, Thomas Nocke, Magnus Heitzler, and Heidrun Schumann. 2013. A Design Space of Visualization Tasks. *IEEE Transactions on Visualization and Computer Graphics* 19, 12 (2013), 2366–2375.

[72] Mary Shaw. 2011. The Role of Design Spaces. *IEEE Software* 29, 1 (2011), 46–50.

[73] Abigale Stangl, Emma Sadjo, Pardis Emami-Naeini, Yang Wang, Danna Gurari, and Leah Findlater. 2023. "Dump it, Destroy it, Send it to Data Heaven": Blind People's Expectations for Visual Privacy in Visual Assistance Technologies. In *Proceedings of the 20th International Web for All Conference*. 134–147.

[74] Abigale Stangl, Kristina Shiroma, Nathan Davis, Bo Xie, Kenneth R. Fleischmann, Leah Findlater, and Danna Gurari. 2022. Privacy Concerns for Visual Assistance Technologies. *ACM Transactions on Accessible Computing (TACCESS)* 15, 2 (2022), 1–43.

[75] Saiganesh Swaminathan, Thijs Roumen, Robert Kovacs, David Stangl, Stefanie Mueller, and Patrick Baudisch. 2016. Linespace: A sensemaking platform for the blind. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. 2175–2185.

[76] Xinru Tang, Ali Abdolrahmani, Darren Gergle, and Anne Marie Piper. 2025. Everyday Uncertainty: How Blind People Use GenAI Tools for Information Access. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing*

Systems. 1–17.
[77] Kristina Terech. 2025. Apple is Currently Levelling up Siri's 'Onscreen Awareness,' Enabling It to Interact With Your Screen. https://www.techradar.com/computing/artificial-intelligence/apple-is-currently-levelling-u-siris-onscreen-awareness-enabling-it-to-interact-with-your-screen
[78] Jingyi Xie, Rui Yu, He Zhang, Sooyeon Lee, Syed Masum Billah, and John M. Carroll. 2024. BubbleCam: Engaging Privacy in Remote Sighted Assistance. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*. 1–16.
[79] Amy X. Zhang, Michael S. Bernstein, David R. Karger, and Mark S. Ackerman. 2024. Form-From: A Design Space of Social Media Systems. *Proceedings of the ACM on Human-Computer Interaction* 8, CSCW1 (2024), 1–47.
[80] Yuhang Zhao, Michele Hu, Shafeka Hashash, and Shiri Azenkot. 2017. Understanding low vision people's visual perception on commercial augmented reality glasses. In *Proceedings of the 2017 CHI conference on human factors in computing systems*. 4170–4181.
[81] John Zimmerman, Jodi Forlizzi, and Shelley Evenson. 2007. Research Through Design as a Method for Interaction Design Research in HCI. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 493–502.
[82] John Zimmerman, Erik Stolterman, and Jodi Forlizzi. 2010. An Analysis and Critique of Research through Design: Towards a Formalization of a Research Approach. In *Proceedings of the 8th ACM Conference on Designing Interactive Systems*. 310–319.

A Additional Co-Designer Background

Co-designers describe their background here, in their own words:

CA: *CA is an assistive technology trainer. He is an independent contractor, doing work for several state organizations who assist blind people. He offers both beginner and advanced instruction for using many different pieces of hardware and software. Diagnosed with Retinitis Pigmentosa at birth, CA was legally blind when he was born, and his vision progressively became worse as he got older. From a young age, CA was determined to integrate himself into the sighted community. As a result, he started playing with different pieces of technology when he was young, always trying to figure out how to adapt things in a way that would work for him. Whether it be figuring out ways of using a piece of technology to help him accomplish certain tasks, or testing out new hardware and software with both himself and those he teaches in mind to make sure that it is usable by as many blind people as possible, he aims to ensure that the barrier between the blind and sighted communities is as easy to overcome as possible.*

CJ: *CJ is the owner of a technology sales, training, and consultancy company. Since 2008, his company has developed and distributed products and services to benefit people who are blind and low vision, and has partnered with numerous companies including Fortune 500 businesses on accessibility and usability projects. CJ is a screen reader and braille user and employs access technology in his day-to-day work. Other interests include artificial intelligence and its applications for blind and low vision users, audio production, and goalball.*

A.1 Familiarity With Shortcuts and Automation

CA and CJ were both familiar with Shortcuts, an application for programming time-saving automations on iOS. CA uses shortcuts frequently. They use them with the action button on iPhone, and with Siri: *"[I use them] all the time. A lot of the things that I've used shortcuts for are just to simplify things on my phone. Like if I wanted to take pictures of something on a tripod, I have a 'say cheese' shortcut implemented where I just kind of trigger that with Siri. And then I have a few other small ones that will take me to specific web pages, or that will set sleep timers for specific apps, things like that."* (CA, S1). CJ was less immediately familiar with Shortcuts, as they are primarily an Android phone user in their day-to-day life. CJ has used similar automation services on Android and expressed confidence that they could quickly understand the Shortcuts app.

B Additional Implementation Details

B.1 System Components

Here, we further describe the technical components that support the implementation of the three add-ons.

B.1.1 App Intents and Shortcut Infrastructure. Each add-on feature has a corresponding Shortcut that controls the flow of information between the starting context and the companion app. Shortcuts were pre-made by us, and can be saved to the user's library by clicking an iCloud link. Users can then change the name of the shortcut to adjust how they trigger it with Siri, or set up automations such as running the shortcut with the action button or back-tap. To connect Shortcuts to add-on functionality, the A11yExtensions companion app implements 'App Intents', an iOS feature that enables app functionality to be invoked programmatically by Shortcuts or Siri. Each add-on defines a corresponding app intent, which in turn can initiate actions such as launching the app or transmitting data to a remote server. We also implement additional helper intents such as 'play screenshot alert', 'play loading sound', or 'store app name', for playing audio cues or for storing the current app name for future reference respectively.

B.1.2 Companion App Structure. All add-ons are implemented within a single iOS companion app, which functions as an add-on store and connects to the Shortcuts via App Intents. The companion app also offers a centralized place for setting up and using add-ons as-needed. The app can store links to Shortcut options for users to save to their own libraries. It also provides a centralized place for user settings such as speaking rate. Finally, we added options for accessing add-ons 'on-demand' and through the Share Sheet via this companion app, as suggested by our co-designers. Although these are not a central aspect of our evaluation, this provides manual or semi-manual ways for users to access the same add-on functionality, helping to build familiarity.

B.2 Add-On Prompts

Prompt for **A.2 Cross Checking**: "`Please confirm that the following statement is true. Your answer must start with 'yes', 'no', or 'unsure'. If the answer is yes, do not say anything else after 'yes'. If the answer is 'no', say no and then give the correct answer as concisely as possible. If you are not sure, respond only with 'unsure'. Respond with no other text. The statement is: ...`"

Prompt for **A.3.M Image Quality**: "`Are there any problems with this image that would result in an AI model not being able to understand it? Is the image blurry, dark, or at an angle? Are the objects too small to understand? Is the camera too close to something? Is something covering the camera partially? Answer this as briefly as possible, within a sentence. If there are no issues say 'no quality issues'. If there are UI elements on`

`the screen, ignore them, and focus only on what's in view of the camera."`

B.3 Add-On Shortcuts

The Shortcuts used to trigger **A.1 Camera Aiming**, **A.2 Cross Checking**, and **A.3.M Image Quality** are included as screenshots in Figure 7.

B.4 Platform Details and Generalizability

While we chose iOS as a platform due to it's prevalence in the accessibility community, it did result in some limitations. One limitation of our implementation is that each Shortcut must be manually saved to a user's library, and permissions for potentially invasive actions must be granted the first time they run. For example, **A.2 Cross Checking** requires approval for taking a screenshot, recording voice input, and sharing data with the A11yExtensions companion app. While these permissions are certainly important for transparency, they cannot be pre-approved at install time and instead must be granted in real time, which adds friction. We also noticed fairly severe VoiceOver bugs in this step, where these permission requests failed to receive VoiceOver focus and went unnoticed. These bugs impacted our testing in Session 2, making Shortcuts seem less feasible. After filing bug reports, most issues were resolved within a year, and co-designers responded more positively in Session 4. Our co-designers, particularly CA, noted frequently that the setup process would be much more approachable and accessible if users could install many shortcuts and accept permissions with one click.

Finally, iOS and Shortcuts restrict certain automations. For instance, screen recordings cannot be automated, so we relied on screenshots instead. Though, while screen recordings might better capture interactions, they may raise privacy concerns similarly to co-designers concerns over taking multiple screenshots. By comparison, Android platforms allow for much broader automation. The barriers to installing and granting permissions for add-on or extension software are lower, and would likely not require an intermediary app like Shortcuts to run automations. Lower third-party developer support is needed, as add-ons may more easily access application source code, automate interactions with third-party apps, or inject new functionality directly.

C Additional Results

Here, we include additional results from our study, primarily on additional add-on functionalities and usability details.

C.1 Secondary Add-On Functionality

We implemented **A.1**, **A.2**, and **A.3.M**. The remaining add-on functionalities discussed with co-designers (**A.4** to **A.7**) were not as strongly preferred. We include the complete findings on these functionalities here, as they could influence directions for future work.

For instance, co-designers were split on **A.4 Routines**, using automation to chain together existing applications based on triggers such as image contents. While CA was not opposed to the idea and saw some use in automating frequent tasks to merge functions from multiple apps, CJ was more skeptical of the utility. They said: *"I mean, it depends, right? Because for a lot of those things I'm going to know what I want. Like I know where I'm at, I'm gonna just grab the correct app. So is it worth the effort? Is it going to save time? Focusing on a verification service or a camera aiming service, those are features that do not exist currently. So if you focus on those areas, you're creating new useful stuff. Although I mean the efficiency thing is cool, but I would focus on the others."* (CJ, S1).

A.5 Error Reasoning and **A.6 Explain Results** were introduced in our later co-design sessions as a response to the accuracy challenges that co-designers highlighted in Session 1. While co-designers saw these as important areas for future work, they were more skeptical that AI would actually have the capabilities to deliver useful or correct results at present. For this reason, they preferred **A.2 Cross Checking** as a way to address error issues, which presents results side-by-side without offering definitive claims of correctness.

Similarly, for **A.7 Background Information**, co-designers worried that this information would often be distracting and irrelevant, excepting the case of severe safety issues that AI would likely notice to begin with. As CJ described: *"I guess it depends on how efficiently or quickly that can be done. And how good it is at identifying relevant information. If it starts going on about random stuff that detracts from the task at hand."* (CJ, S3). While co-designers were able to list some scenarios where they would want to be made aware of background 'unknown unknowns' automatically (finding a lost object, noting a discrepancy in oven temperature from intended setting), there is too much risk in presenting additional information. However, CA also noted that a similar add-on feature could help people construct prompts for additional information when they *do* want to request it: *"Many people have not gotten the detailed prompt feature down. They need hand-holding with regards to constructing a detailed prompt to get the information they want. Most people won't think to say tell me more about this object, or the color, or the patterns. So you have to have the bulk of the prompt constructed for them, which a Shortcut could do."* (CA, S3).

C.2 Implementation and Usability Feedback

In Session 4, co-designers provided usability feedback, which could influence future implementations.

A.1 Camera Aiming: Due to a bug, only CA was able to test the camera aiming feedback. We suspect this is due to CJ using an older device that may have had issues with running the models locally. CJ was only able to test the aiming workflow and app switching, without live feedback. However, CA felt that the feedback provided was useful for aiming and was a good first version for basic tasks. They said *"And it's good, it seems to be pretty solid. [The level of feedback] it's appropriate, because what I did like is that it also told me the other things that are in the background of that picture."* (CA, S4). However, CA also suggested some minor usability improvements to make the aiming feature more accessible to a range of users. After aiming, users still need to double tap to take the photo manually. CA noted this may be challenging for those with motor impairments, and wondered if there were ways to automate it.

A.2 Cross Checking: Co-designers liked that this add-on provided a quicker way to do something that would otherwise take much longer: *"It's kind of a quick way to ask two AIs the same question."* (CJ, S4). They also appreciated that we removed some of the

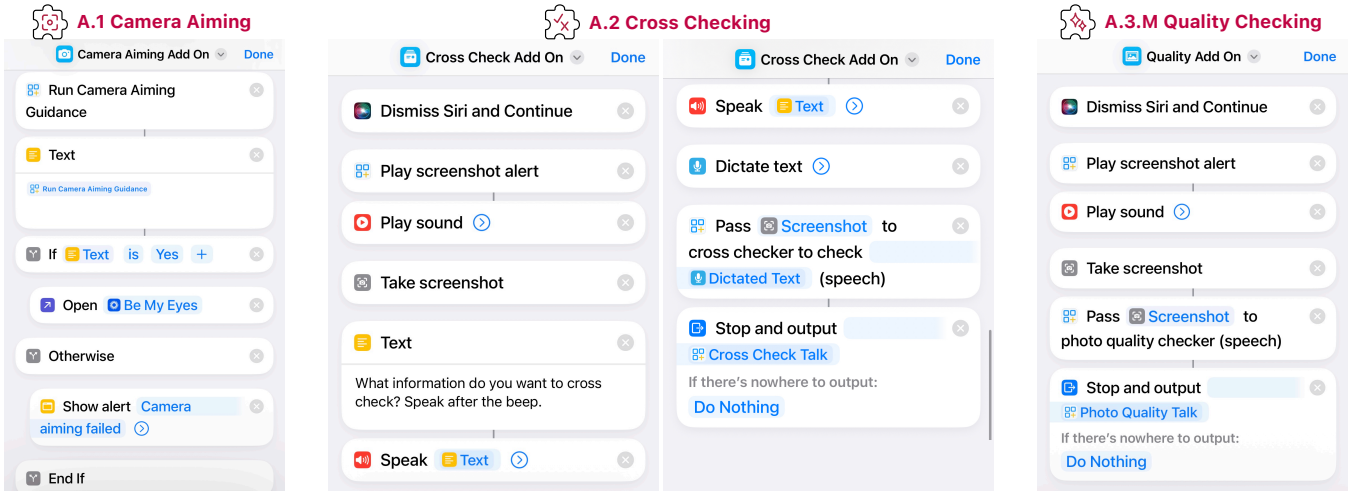

Figure 7: (1) Shortcut used to trigger A.1 Camera Aiming. The 'Run Camera Aiming Guidance' block returns the text 'yes' when aiming is successful. (2) Shortcut used to trigger A.2 Cross Checking. (3) Shortcut used to trigger A.3.M Image Quality.

lag from the previous version. We removed some of the verification questions and models from our intitial prototype, and were concerned that removing feature complexity would ultimately hurt the effectiveness of the add-on, however, this was not the case. CA said: *"It was easier to follow through it this time, and kind of stick with it. This time I felt like it was quicker, more responsive. Last time it was not, so this time it definitely seemed to go much smoother."* (CA, S4). One challenge in cross-checking is that our implementation preferred user prompts to be phrased as yes/no statements. For instance, 'This is red' versus 'What color is this?'. This is due to our understanding of the user context (we assume that users want to check information coming from an existing service), and the implementation complexity (it is easier to generate and summarize yes/no answers). However, both co-designers mentioned that this feels unnatural, and that they would not want to begin with a statement that they were uncertain of.

A.3.M Image Quality: Both co-designers appreciated this add-on, citing it as their favorite at the end of the testing sessions. CA said *"I think that is one of my favorite features, because it really kind of helps put into perspective the proper focus of the picture. Lighting levels, things like that."* (CA, S4). CJ mentioned that this add-on felt the 'most seamless' of the add-ons tested because it didn't have any follow up questions, and ran quickly in the background. As a standalone feature, both co-designers mentioned that they wished it included more actionable feedback. When CJ received the description: 'The image is blurry and the objects are too small to understand clearly.', they replied, *"But without it knowing which object I'm trying to... So what object does it think is too small, I wonder?"* (CJ, S4). Listing possible issues is not sufficient for some tasks, and co-designers wanted to know the severity and how it actually affected the image content and purpose. This is also dependent on the task context: *"It depends what I'm doing. If I'm doing a product photo for my website, my standard is higher than if I just want a picture for AI."* (CJ, S4). Finally, actionable feedback could also indicate how users should fix an image, as CJ described *"It's like, all right, what is the quick tip for someone to try to fix whatever it is... I think that would be really useful. What they should do, and you know, use non-visual language to explain to the user what to do."* (CJ S4).

D Complete Co-Design Protocols

Here, we provide approximate protocols for our co-design sessions. In detailing each session, we denote scripts from the study facilitator with italics. As these were co-design sessions rather than structured interviews, the questions and topics listed here are our initial plans, and are not fully representative of the resulting conversations. Sessions frequently touched on new topics and ideas introduced by co-designers.

D.1 Co-Design Session Overview

(1) **Session 1:** Interview on limitations and challenges of assistive AI, experiences with Shortcuts and automation tools. Verbal co-design introducing add-ons conceptually, and discussing four initial add-on ideas.
(2) **Session 2:** Testing three add-on mock-ups: two refined from feedback, and one new concept based on challenges raised in Session 1.
(3) **Session 3:** Verbal, scenario-based co-design centered on personal experiences in order to discuss real-world add-on use in depth. Discussed five add-on features, three from previous sessions, and two new concepts based on feedback and challenges raised in previous sessions. Brainstorming future add-on features and applications based on different key challenge areas with assistive AI applications.
(4) **Session 4:** Testing three fully implemented add-ons, both separately, and in one combined workflow simulating the real-world task of preparing a photo for social media. Interview reflecting on the entire co-design process.

D.2 Session 1

D.2.1 Topic Introduction. "We're interested in developing a project around AI assistive applications. We thought you would be interested

given your personal and professional experiences using and sharing these sorts of apps. In general, as we mentioned over email, we're interested in learning more about the limitations of AI. We're also interested in ways that they could be improved, and we have some ideas about this that would be great to get your feedback on and maybe do some brainstorming.

One direction that we're thinking of is trying to try to develop new features as add-ons for existing apps, and then use Shortcuts (the automation tools on iOS) to automatically switch to those add-ons as needed. So for today we can chat about those things, and we also have some very basic prototypes that you could try out if we have time. Does all of that sound okay?"

D.2.2 Limitations of Assistive AI Applications. "Let's start by discussing some challenges or limitations of current AI applications. We have some ideas from our conversations with other participants, but I'm curious about your impressions too."

(1) Do you use any AI apps personally?
(2) What do you think are the biggest challenges or limitations with current AI applications?
(3) From our prior studies with blind people, we've noticed common issues emerging: accuracy, so a model producing errors; usability, when there's not good live feedback for things like aiming the camera and getting live feedback; and also to a lesser degree personalization, so people wanting more or less information, or information in a different way. What do you think? Does that seem representative?

D.2.3 Initial Opinions on Concept. "For many of these issues, we've noticed researchers working to address them in different ways. For example, research around explaining AI results. One thing we're interested in is making the research on these types of things more available in people's day to day workflows."

(1) I'm curious if there are any new features you could imagine as 'add-ons'?
(2) Have you used Shortcuts or other automation or scripting platforms? If so, for what?

"One idea would be to use Shortcuts to to automatically switch to an app with an 'add-on' feature, and then automatically switch back to the original app when you're done."

(1) What is your first impression of that idea?

D.2.4 Reflections on Initial Add-Ons. **A.1 Camera Aiming:** *"One thing that we imagined was camera aiming assistance, which I think Seeing AI has for document mode, but Be My AI doesn't have. I noticed sometimes it's hard to get something fully in frame. So one idea was to trigger a shortcut by saying something like, 'Hey Siri, help me aim the camera'. Then, the app would switch to a different app that can give live feedback for centering an object in frame. Once the object is centered, it could automatically switch back to Be My AI, so you could keep the phone in the same place and then just take the picture."*

(1) Does this sound useful?
(2) What do you think of the workflow of asking Siri for something, switching to another app, and then switching back later?

A.2 Cross Checking: *"Another thing would be the ability to cross-check a result with a different service. So imagine using Be My AI and it gives a confusing result. Sometimes if you ask Be My AI a follow-up question, it says something completely different, and then there might be uncertainty about what's correct. So you could trigger a shortcut by saying, 'Hey Siri, can you cross-check this result?' The shortcut could then take a screenshot of what you're currently doing, send it to different services, then have Siri read out some of the other responses."*

(1) Does this sound useful?
(2) What do you think of the workflow of asking Siri for something, taking a screenshot and then analyzing it in the background?

A.3 Image Quality: *"Another thing could be automatically checking image quality. So shortcuts can start automatically in the background whenever you open specific apps, and then run in the background. So we were imagining a Shortcut that would periodically take and analyze screenshots as you're using an app, and then it could alert you with a pop-up if there's a quality issue like if the camera is too dark or it looks blurry."*

(1) Does this sound useful?
(2) What do you think of the workflow of a Shortcut running automatically in the background, capturing what you're doing, and then trying to give information in a pop-up?

A.4 Routines: *"One thing that we've heard from people we've worked with is that they sometimes have a lot of assistive applications downloaded that they might use in different situations. So for instance someone told me that they would read English text with SeeingAI and Arabic text with Envision AI. So another Shortcut idea would be to set up automations where the app could switch depending on what language is detected in the frame. Another example is with Be My AI, people may have common questions that they use in different situations, maybe the user can save questions and then they could be automatically filled in or given in a pop-up notification in similar situations later."*

(1) Do these sound useful?
(2) What do you think of each of these examples?
(3) Can you think of any other routines like these?

D.2.5 Open Brainstorming. "Now that we've gone over some different ideas for how shortcuts could be used to add new features to existing apps, I'm curious if this sparked any thoughts or ideas?"

D.3 Session 2

D.3.1 Introduction. "I wanted to start by recapping a bit of what we talked about last time. Last time, we talked about wanting to develop 'add-ons' for existing AI assistive technology, sort of like a companion app or companion features for existing apps. And this would hopefully address some common issues with assistive AI apps, like we talked about accuracy being a big thing, and being able to figure out and work around errors.

We talked about a couple of different add-on features, like better camera aiming feedback or the ability to double check results from an AI to confirm them. And we talked about some possible ways these add-ons could be accessed, like triggering them with Siri or having them run in the background.

Our main takeaways from our conversation were that the add ons or new features sounded useful, and running them with automations seemed interesting, but there was concern about running things from

the background from a privacy perspective, if this companion app were to run in the background and try to analyze what the AI was doing, it would feel more like a spy than a helper.

Does this sound like a good summary? Is there anything big that I'm missing?

For the design of this AI companion app, we have some pretty basic prototypes we can go over together. It's definitely rough around the edges, but the goal would be exploring different possibilities as we kind of design and develop it together."

D.3.2 Comparing Triggering Approaches. "The first thing that we want to test more of is different ways that people could bring up this companion app, or bring up specific add-on features. So we came up with a couple of different options to try out and discuss. These are mostly just mock ups to show what it could be like, they aren't fully implemented."

Triggering with Siri: *"One of the examples we talked about last time was having a camera aiming assistant for apps that don't include it like Be My AI, then saying something like, 'Hey Siri, help me aim the camera'. Then, the app would switch to a different app that can give live feedback for centering an object in frame."*

(1) I know the main concern here was just remembering specific names. Like if there are a lot of add-on features that you might want to have eventually, it would be hard to remember specific wording and Siri is picky about that. Is that an accurate summary?
(2) Do you have any other thoughts to share about this approach?

Verbal Menu: *"Next, instead of using Siri to trigger one feature, we set up a Shortcut that has Siri read out different options."*

- Co-designers install a Shortcut prototype
- Starting in app, say to Siri: 'Tell me the add-on menu.'
- The Shortcut will ask: 'Do you want to aim camera, check photo quality, cross check result, or debug result?'
- Co-designers then speak the option they want to choose, and it is recorded.

After testing:

(1) What was your first impression of this option?
(2) How does it compare to having different prompts to Siri for different features?
(3) What are the positives and negatives of this approach?

In-App Menu: *"Next, we have a Shortcut that switches to the companion app to display a menu of different options that someone could read through."*

- Co-designers install a Shortcut prototype
- Starting in app, say to Siri: 'Open the add-on menu.'
- The Shortcut will switch to the companion app, opening a menu page.

After testing:

(1) What was your first impression of this option?
(2) How does it compare to having different prompts to Siri for different features?
(3) How does it compare to Siri reading the menu?
(4) What are the positives and negatives of this approach?

Standalone App: *"Users could also open this app manually and use it whenever they choose."*

(1) What was your first impression of this option?
(2) How does it compare to the other approaches, where you access an add-on feature while you're already in another app?
(3) What are the positives and negatives of this approach?

Overall Comparison:

(1) Would you have a personal preference?
(2) What would be a good default?
(3) For the second and third options we tried, where Siri either reads a menu or goes to the app to show the menu, you said 'add-on menu' to trigger this. Another option would be to setup a non-verbal trigger to do the same thing, for instance, using the action button, or tapping the back of the phone. Would you prefer this method?
(4) Do you think it's useful to let people choose how to set up these triggers?

D.3.3 Testing Add-On Mock-Ups. Co-Designers test semi-functional mock-ups of three add-ons that seemed to be the most useful:

For **A.1 Camera Aiming**:

(1) What are your thoughts on this feature?
(2) What are the pros and cons of this feature?

For **A.2 Cross Checking**:

(1) What are your thoughts on this feature?
(2) I know you previously mentioned that cross checking results could be tedious, do you think this would help?
(3) This automatically took a screenshot of what you were doing and analyzed it in the background, what did you think of that part?
(4) What are the pros and cons of this feature?

For **A.5 Error Reasoning**:

(1) What are your thoughts on this feature?
(2) I know we had talked about accuracy being a big challenge with AI, and although we can't really improve that, this feature is aimed at trying to explain why something might not be correct. Do you think this would be helpful?
(3) What are the pros and cons of this feature?

D.3.4 Open Brainstorming. "I'm curious, now that we've tested the prototype together and you've experienced some of these add-on features beyond just hypothetical descriptions, is there anything else that comes to mind as a useful add-on?"

D.4 Session 3

D.4.1 Introduction. "Last time, we tried some basic prototypes for how add-ons could be accessed, so there was a verbal menu with Siri, and an in-app menu. We also tried running a shortcut for cross-checking AI results, where you triggered it with Siri and then took a screenshot of Be My AI to analyze. We got a lot of good feedback on how to make this more user friendly and fix some of the prototype issues.

We've been focusing a bit on the overall idea and usability of this companion app. Today I want to talk more about the actual functionality and how it might be used in the real world. So I want to

go over each of the add-ons that we've talked about so far, and discuss in a bit more detail what the functionality would be like in different scenarios. We also want to know which scenarios would be the most critical for us to support, so we can use those as benchmarks in a way, to make sure we're helping with things that are actually important. So today I'm going to ask you to think about specific situations where you were using AI, or you can use examples from your students, because I know you have a lot of experience with different people's individual needs. And we'll do some brainstorming together.

My hope is that after today we will have a solid understanding of what we should be aiming to build, and then we can take a couple of months to focus on turning these prototypes into things that are functional and accurate, and check back in at some point then down the line. Does all of that sound okay or do you have any questions for me?

D.4.2 ***A.2 Cross Checking.***

(1) Can you think of any specific scenarios where this would be particularly useful? I know you mentioned that the need to fact check AI is one of the biggest challenges, so I'm wondering if you can think of any specific examples, either for yourself or for your students? These could be personal experiences that you've had, or examples that you've just observed from working or talking with other people.
(2) Are there any scenarios where you want to use AI, but accuracy is super critical so maybe you're hesitant to? Can you give specific examples?
  (a) Would cross-checking results change any of those situations?
  (b) What would make you willing to try using AI in these situations?
(3) For each scenario described above, walk through the add-on prompt/ trigger, ideal information, and any concerns. How would you imagine cross checking in this scenario?
  (a) Would you want to use Siri, a share-sheet, or type into an app directly?
  (b) What would be the ideal output? A yes/ no answer? Knowing how many models 'voted' on the fact? Any other reasoning or information?
(4) What are the pros and cons of this feature?
  (a) What concerns would you have about using something like this out in the real world?
  (b) Any accuracy concerns? How do you weigh conflicting responses against each other?
  (c) Any privacy concerns?
  (d) On the other hand, what would the benefits of this approach be over the current processes of manually checking or checking with a human?
(5) What are the open challenges still in the space of accuracy and feeling like you can rely on the results? Are there other add-on features that could help?

D.4.3 ***A.6 Explain Results.*** *"I want to talk about having an add on that tries to explain the results of an AI. So rather than trying to verify something, having an add on that would try to give the reasoning behind something. So for example something that might say, 'I'm 70% sure that there's a dog, but it has pointy ears so it could be a cat.'"*

(1) Can you think of any specific scenarios where this would be particularly useful? Any specific examples, either for yourself or for your students?
(2) We've noticed that sometimes people, especially people who are newer to AI, try to guess why a certain result was given and understand what must have happened. I'm wondering if you can think of any examples of this? Again either for yourself or your clients?
(3) For each scenario described above, walk through the add-on prompt/ trigger, ideal information, and any concerns. How would you imagine using the add-on in this scenario?
  (a) Would you want to use Siri, a share-sheet, or type into an app directly?
  (b) What would be the ideal output?
(4) What are the pros and cons of this feature?
  (a) What concerns would you have about using something like this out in the real world?
  (b) Any accuracy concerns? How do you weigh conflicting responses against each other?
  (c) Any privacy concerns?
  (d) On the other hand, what would the benefits of this approach be over the current processes of manually checking or checking with a human?
(5) What are the open challenges still in the space of helping people understand how AI works? Are there other add-on features that could help?

D.4.4 ***A.7 Background Information.*** *"I want to talk about having an add on that would try to make someone aware of relevant information in the background. I might have given this example before, but if I'm taking a picture of a sign at the bus stop to know what routes are coming, the add-on might say, 'Hey, there are other signs in the background that might list different routes'. Or, I think last time you gave the example of checking time on your oven, so it might say, 'Hey, the time on your microwave is different.'*

(1) Can you think of any specific scenarios where this would be particularly useful? Any specific examples, either for yourself or for your students?
(2) We've noticed that sometimes the description from an AI can be way too broad, but on the other hand maybe it can sometimes be too specific. I'm wondering if you can think of any examples of this? Again either for yourself or your clients?
(3) For each scenario described above, walk through the add-on prompt/ trigger, ideal information, and any concerns. How would you imagine using the add-on in this scenario?
  (a) Would you want to use Siri, a share-sheet, or type into an app directly?
  (b) What would be the ideal output?
(4) What are the pros and cons of this feature?
  (a) What concerns would you have about using something like this out in the real world?
  (b) Any accuracy concerns? How do you weigh conflicting responses against each other?
  (c) Any privacy concerns?

(d) On the other hand, what would the benefits of this approach be over the current processes of manually checking or checking with a human?

(5) Are there other challenges in helping people get the right amount of information at the right time? Are there other add-on features that could help?

D.4.5 ***A.1 Camera Aiming.***

(1) Can you think of any specific scenarios where this would be particularly useful? Any specific examples, either for yourself or for your students?
(2) We've noticed that sometimes it's hard to take a good picture for something like Be My AI, but on the other hand, the type of picture that you might take for Be My AI might be different than the type of picture you would like send to a friend or post on social media. I'm curious if there are different strategies for photo taking that you use, or times where you think an aiming assistant would need to give different levels of guidance?
(3) For each scenario described above, walk through the add-on prompt/ trigger, ideal information, and any concerns. How would you imagine using the add-on in this scenario?
   (a) Would you want to use Siri, a share-sheet, or type into an app directly?
   (b) What would be the ideal output?
(4) What are the pros and cons of this feature?
   (a) What concerns would you have about using something like this out in the real world?
   (b) Any accuracy concerns? How do you weigh conflicting responses against each other?
   (c) Any privacy concerns?
   (d) On the other hand, what would the benefits of this approach be over the current processes of manually checking or checking with a human?
(5) Are there other challenges in understanding what is in view of the camera? Are there other add-on features that could help?

D.4.6 ***A.3 Image Quality.***

(1) Can you think of any specific scenarios where this would be particularly useful? Any specific examples, either for yourself or for your students?
(2) For each scenario described above, walk through the add-on prompt/ trigger, ideal information, and any concerns. How would you imagine using the add-on in this scenario?
   (a) Would you want to use Siri, a share-sheet, or type into an app directly?
   (b) What would be the ideal output?
(3) What are the pros and cons of this feature?
   (a) What concerns would you have about using something like this out in the real world?
   (b) Any accuracy concerns? How do you weigh conflicting responses against each other?
   (c) Any privacy concerns?
   (d) On the other hand, what would the benefits of this approach be over the current processes of manually checking or checking with a human?
(4) Are there other challenges in understanding what is in view of the camera? Are there other add-on features that could help?

D.4.7 Open Brainstorming. For each co-designer, we collected a list of ideas or scenarios from previous sessions. We then discussed each idea more specifically to understand the desired features. These included ideas such as:

- Video descriptions
- Commonly repeated prompts or questions
- UI layouts and understanding buttons
- Streetlight recognition while navigating
- Location-based prompts

We then brainstormed more broadly:

(1) If we had all of the features that we talked about today, Do these solve all of the problems with Mobile AI AT? Think about all of the interactions and challenges from you and your clients?
   (a) What's missing still?
   (b) For you and your clients?
   (c) Is it ready to be put in the hands of you and your clients?
   (d) For each extension, what is not enough? Is it perfect or imperfect?
(2) With everything that we've talked about today, what other scenarios still seem hard or impossible?

D.4.8 Impressions of Customizability.

(1) Overall in the past couple of sessions, we have talked a lot about different ways of interacting with these add ons or the companion app. Last time we talked about how Shortcuts are just one of these options, but that it's important to have others so that people with different levels of experience can use them in a way that doesn't seem super advanced. I'm curious if this level of personalization would be beneficial? Or would it be overwhelming?
(2) Another thing we talked about was maybe using the add ons in different ways depending on what you're doing. So if you get an image message from a friend, the share sheet might be most convenient. Or if you're walking around in public, using Siri might not be ideal. I'm wondering if you can think of any other factors or situations that would influence how you want to use something like this?
(3) Are there any other ways that you think people would want to personalize or customize these add ons and how they work?
   (a) For you as an expert? If you could have any setting or option in these extensions or companion apps, what would it be?
   (b) For your clients? What do your clients need and what would you customize for them? What is most helpful for them? What tools would support them?

D.5 Session 4

D.5.1 Introduction. Now we want to test some of the features we have discussed more fully. Since our last session, we have implemented three of the different add-on features that we have talked about: camera aiming, cross-checking results, and checking image quality. We chose

these because from our discussions they seemed like they would be the most promising or immediately useful things to prioritize. We had discussed other possible add-on features like explaining errors, finding background information in an image, or saving common prompts/ questions in a library, but those either seemed less urgent or there were concerns about how useful they would be.

D.5.2 Add-On Testing. First, co-designers tested each add-on with a simple home prop, such as a can, or items that they kept on their desk. Then, co-designers tested all of the add-ons together in a full workflow.

A.1 Camera Aiming: *"I want you to act as if you were using this feature for a real task, so thinking that we want to use Be My AI to describe some information and we want to make sure an object is centered in the frame. So I want you to open Be My AI, back-tap to trigger the camera aiming, and then once you're back in Be My AI, take the picture to submit it."*

(1) What are your thoughts as you go through this task?
(2) Is the feedback useful?
(3) Anything you would like to add, change, customize from what we tried today?
(4) What did you think trying this workflow in full, where you were in one app, triggered the add on and swapped apps, and then had it switch back for you?
(5) Was it easy or hard to tell what was happening?

A.2 Cross Checking: *"You'll trigger the feature with Siri, saying, 'cross check add on'. It will then take a screenshot of what the camera sees, and will ask what piece of information you want to check. Finally it will run in the background and speak results when it's done."*

(1) What are your thoughts as you go through this task?
(2) Is the feedback useful?
(3) Anything you would like to add, change, customize from what we tried today?
(4) What did you think trying this workflow in full, where you were in one app, triggered the add on and swapped apps, and then had it switch back for you?
(5) Was it easy or hard to tell what was happening?

A.3.M Image Quality: *"You'll trigger the feature with Siri, saying, 'quality add on'. It will then take a screenshot of what the camera sees, and will process that in the background."*

(1) What are your thoughts as you go through this task?
(2) Is the feedback useful?
(3) Anything you would like to add, change, customize from what we tried today?
(4) What did you think trying this workflow in full, where you were in one app, triggered the add on and swapped apps, and then had it switch back for you?
(5) Was it easy or hard to tell what was happening?

Full Workflow & Social Media Post Task: *"Finally, I want to try using all three of these features together for a real task. So I want to leave the specific approach or exactly how you would go about it up to you, but I want you to pretend like you were using these features to prepare a social media post. We talked about this before as an area where it takes some work with existing apps to get things right, and where there are consequences if you miss something or the models get something wrong.*

So with the features we tried today, we can: use camera aiming to frame a good shot, check photo quality, and cross-check key visual details before posting.

You don't have to use them in any specific order, and you can use them either from the shortcuts, or directly from the app itself, or a mix of both. Feel free to try things, and ask questions, I can remind you of the phrases that we used for Siri as we go if needed. But just thinking about how you might approach this task of preparing a post for social media with these tools."

D.5.3 Design Reflection. Ended the session with a semi-structured interview.

Reflections on conceptual evolution:

(1) We discussed a lot of potential add-ons throughout the process, but chose to focus on these three as they seemed the most important. Does this seem accurate? What would you do next?
(2) What's one surprising thing you realized only after using the system today?
(3) If you could make the perfect extension, what would it do?

System reflection:

(1) Are there high-priority scenarios where you still wouldn't reach for this system?
(2) How much of the system feels general-purpose versus specific to certain tasks or apps?
(3) Do we meet the goal of efficiency?
(4) When things went wrong (bad feedback, disagreement, unclear output), did you feel you could recover easily?

Automation and mental models:

(1) Does this meet goals of privacy and control?
(2) Was it clear to you what was happening when an add-on ran automatically or switched apps? How would you want this process to be communicated better?

DIY and extensions versus native accessibility:

(1) This project has been about recognizing gaps in assistive apps, but also where we as researchers are limited (can't develop full products). I want to discuss a bit about this, what are your thoughts?
(2) What are the tradeoffs?
(3) What do you see as a vision for the future here?