DESIGN SYSTEMS FOR ACCESSIBILITY

Creating a sustainable methodology for workplaces within web development

Ingrid Berglund

Department of Applied Physics and Electronics
Umeå University, Sweden
Supervisor: Ole Norberg
Examiner: Thomas Mejtoft

Master thesis, 30 credits
Master of Science in Interaction Technology and Design, 300 credits
Spring 2023
Abstract

Although accessibility benefits all users of the web, irrespective of their abilities and disabilities, many web applications fail to fulfill basic accessibility requirements. Designers and developers need to improve their work with accessibility in order for digital inclusion to be possible. This thesis investigated how a design system can support them in producing accessible applications. It used a design approach to formulate an accessibility methodology for creating and maintaining a design system. Interviews and ideation workshops were held to understand the problem and generate solutions. A methodology was put together and evaluated through expert reviews, focus grouping, and user testing. The thesis resulted in 10 guidelines for how to promote accessibility through a design system. Central to the guidelines was that companies adapt the work with accessibility to their specific situation, by setting their own goals and creating tools for the different stages of the employees’ work processes. The thesis concluded a design system can help designers and developers not feel overwhelmed by accessibility requirements, and focus on what is relevant to the task they are currently performing. A design system that is merely a storage for components can only assist designers and developers with a fraction of the accessibility requirements. Greater potential is held by a design system that is made into a platform for accessibility, by providing guidance on how to use components and being referenced in testing procedures.

Keywords

Design systems, Accessibility testing, WCAG, Designer-developer collaboration, Digital inclusion, Inclusive design
Sammanfattning


Nyckelord
Designsystem, Tillgänglighetstestning, WCAG, Samarbete mellan designer och utvecklare, Digital inkludering, Inkluderande design
Contents

1 Introduction .......................................................... 1
   1.1 External partner ................................................. 1

2 Objective ..................................................................... 3
   2.1 Definitions .......................................................... 3
   2.2 Delimitations ......................................................... 4

3 Theory ....................................................................... 5
   3.1 Who needs accessibility? .......................................... 5
   3.2 Accessibility in the Swedish law .................................. 5
   3.3 WCAG ................................................................ 6
      3.3.1 Principles ....................................................... 6
      3.3.2 Guidelines and success criteria .......................... 6
      3.3.3 Levels ............................................................ 7
      3.3.4 Difficult to apply ............................................. 7
      3.3.5 Webbriktlinjer.se ............................................. 7
   3.4 Automatic accessibility evaluation tools ......................... 8
   3.5 Previous research on accessible web ............................ 9
   3.6 Design Systems ..................................................... 10
   3.7 Issue tracking systems ........................................... 11

4 Method ..................................................................... 13
   4.1 Phase 1: Discover .................................................. 15
      4.1.1 Interviews ....................................................... 15
      4.1.2 Affinity diagramming ...................................... 16
      4.1.3 Inspection of existing design systems ................. 16
   4.2 Phase 2: Define ...................................................... 17
      4.2.1 Reflecting exercise ......................................... 17
      4.2.2 5 Ws ............................................................... 17
      4.2.3 User Need Statements ..................................... 17
      4.2.4 Formulation of problem statement .................... 18
   4.3 Phase 3: Ideate ....................................................... 18
      4.3.1 Designer workshop ......................................... 18
      4.3.2 Stakeholder workshop ..................................... 21
      4.3.3 Sorting ideas ................................................ 23
      4.3.4 Conversion into methodology ......................... 23
   4.4 Phase 4: Develop .................................................. 23
      4.4.1 Four samples ................................................ 23
      4.4.2 First drafts of deliverables ............................... 23
   4.5 Phase 5: Evaluate and Phase 6: Modify ......................... 24
      4.5.1 Expert reviews .............................................. 24
      4.5.2 Focus group .................................................. 25
1 Introduction

Digital accessibility is concerned with making websites and applications usable for the widest possible spectrum of people with different abilities [1]. It is estimated that 16% of the world’s population has a severe disability [2]. A national study covering difficulties related to vision, hearing, speech, concentration and mobility, found 42% of the Swedish population have a disability [3]. Disabilities are not always permanent, they can also be temporary or situational [4, p. 17] and therefore everyone benefits from digital accessibility.

Inclusion at work for people with disabilities is a target within the UN’s Sustainable Development Goals [5]. In Sweden, 85% of all workers use computers in their profession [6, p. 24]. To achieve inclusion, digital services and tools used in the working environment must be accessible.

At the time of writing, software companies are performing remarkably poorly on the issue of accessibility. A 2022 audit of the one million most visited websites, showed that 84% failed to even meet the basic requirement of putting enough contrast between text and background [7]. In the US, companies behind inaccessible websites risk being sued. The amount of lawsuits has increased dramatically in recent years and in 2022 more than 100 lawsuits were filed every week [8, p. 2]. Clearly, better methodology is needed to assist designers and developers in achieving accessibility.

Design systems are a possible starting point for achieving greater accessibility. A design system can be employed by a software company to collect reusable design and code components for user interfaces in one place. It can contain buttons, icons, forms, typography and colors. The purpose is to improve time efficiency [p. 15], consistency and communication [p. 72] between employees [9]. This thesis explores the role of design systems in the work of making applications accessible. It is based on a case study in collaboration with an external partner.

1.1 External partner

Metria is a geodata company with its software development situated in Umeå and Luleå. They deliver solutions to customers in all of Sweden within city governance and service, construction, forestry, energy, telecom, banking and insurance. The customers use Metria’s products for management and decision-making. A couple of the products are pure information, such as maps, while the majority are complete applications letting the customer interact with the data and add their own. Most of Metria’s users use the products at work, but a smaller portion are private individuals.

The data itself is sourced from other actors and then packaged and refined by Metria. It is loaded into applications, visualized and made ready for use. Metria applications help the users to, for example, estimate the worth of a property based on factors such as size and vegetation, analyze alternative paths for new transmission lines, or display locations of
healthcare centers in a map embedded on a website. In total, there are over 20 different applications.

Metria strives to improve their work with accessibility in the development process. Their first incentive for this is the legal aspect. Swedish law requires all websites within the public sector to be accessible. Although Metria is not targeted by this law, it might be targeted in the future. The second incentive for Metria is that accessible applications are beneficial for all users.

Metria is in 2023 in the early stages of introducing a design system. Their ambition is to use this opportunity to integrate accessibility into all their applications. As a first step in this process, the company decided to partner with a student.
2 Objective

The aim of the thesis is to develop an accessibility methodology for creating and maintaining a design system. The methodology is to support designers and developers who make, as well as use, resources from the design system, in order to ensure all developing teams produce accessible applications. Applications that are built based on the design system are to inherit the accessibility measures of the components and other resources in the design system. The purpose of the thesis is to contribute to making applications accessible to the end users.

The research question of the thesis is: **How can a design system support designers and developers in producing accessible applications?**

The target group of the thesis is designers and developers, because they are the users of the design system. Their work processes are studied and improvements are proposed. Since end users of the produced applications do not interact with a design system directly, they are not central for the execution of the case study. They are however, very central to the purpose of the thesis.

2.1 Definitions

A number of words and abbreviations are, in the thesis, used in a different or narrower sense than how they are commonly understood, and therefore require clarification. The definitions used in the thesis are as follows.

A **methodology** is in the thesis understood as a set of methods and tools in a certain field. *Accessibility* here refers to digital accessibility and is defined as fulfillment of W3C’s standard WCAG 2.1 level AA, see explanation in Section 3.3.

The word **application** is used to refer to web-based services including websites, systems and products. It is not synonymous with mobile applications specifically. The device the applications are used on is irrelevant to the objective.

The word **company** is used to refer to a workplace where applications are produced, regardless of the type of the organization; if it is commercial or public.

The abbreviation **UX** stands for User Experience.

The verb **develop** may, outside of the thesis, sometimes refer to the combined process of designing and implementing an application in code, and other times refer to the coding part only. To avoid confusion in this thesis, which discusses activities performed by designers and developers together and separately, **produce** or **build** is used for the prior sense and implement is used for the latter sense. The compound words, **web development** and **development teams** are however still used because they are such well-recognized, and then refer to both design and implementation.
2.2 Delimitations

There are many roles involved in creating and maintaining an accessible digital product. This thesis is targeted towards UX designers primarily and front-end developers secondarily. Back-end developers and content creators such as authors and editors of websites, are not included in the scope.

The thesis regards accessibility on the web specifically. Digital accessibility in general also includes desktop apps, native phone apps and other systems, but these are disregarded in this thesis.

Design systems are pieces of software themselves, and they can be made accessible or inaccessible to potential employees with disabilities. This angle on the topic is not studied in this thesis. It studies how to create and maintain design systems that promote accessibility, not how to make design systems accessible.
3 Theory

The thesis is founded on a number of circumstances surrounding digital accessibility such as laws, standards and automated testing. Design systems and their role in the collaboration between designers and developers are also central to the thesis. These topics are introduced in this section, along with the term issue tracking systems, which is explained due to its relevance to the results of the thesis.

3.1 Who needs accessibility?

Defining disability is neither simple nor uncontroversial. This makes it hard to find statistics describing how many people have a disability, and also explains why the sources differ so much. Another aspect is that keeping registers of people’s disabilities infringes on their integrity [10]. Also, not all disabilities are relevant to digital accessibility, and there are conditions that cause barriers when using digital applications that are not necessarily considered disabilities. Additionally, experts argue accessible applications are beneficial for all users [11], [4, p. 21].

For the figure cited in the introduction, saying that 42% of Swedes have a disability, a complex definition was used [3]. The study was made by Statistics Sweden (Statistiska Centralbyrån) and is based on a nationwide survey. For respondents to be classified as having a disability, they answered they had some difficulties, severe difficulties or were unable to perform any one of six activities: (1) seeing, even when wearing glasses or contact lenses, (2) hearing, even using a hearing aid, (3) walking in staircases, (4) remembering or concentrating, (5) taking care of themselves such as washing and getting dressed and (6) communicating, understanding and making themselves understood. Respondents could also answer simply that they have a disability — physical, mental or intellectual — that causes them some difficulties, much difficulty or inability to do something, and thereby be classified as having a disability.

Listing the prevalence of some permanent difficulties relevant to digital accessibility, around 19% have difficulties with memory and concentration [3], 13% have a vision disability [3], 12% have trouble hearing [3], 5-8% have dyslexia [12] and 4.5% have difficulties with communicating [3]. Situational and temporary disabilities can be added to this list, such as carrying a baby, being in a loud environment, and as wearing a cast due to injury [4].

3.2 Accessibility in the Swedish law

At the time of writing, websites and mobile applications in the public sector in Sweden are bound by law to be accessible. In practice this means they are to follow the European Standard called EN 301 549 V3.2.1 [11], which is based on a different standard called WCAG 2.1. These demands can be expected to spread to other areas. The EU’s European Accessibility Act concerns customer products and services, and has a clear deadline in June 2025. It covers all e-commerce, banking and travel services [13], [14].
In the current Swedish law, there is an exemption for maps that are not intended for navigation — they do not have to conform to the accessibility standard [15]. There is a similar exemption in the European Accessibility Act — its requirements do not apply for "maps and mapping services, if essential information is provided in an accessible digital manner for maps intended for navigational use" [14].

3.3 WCAG

The Web Content Accessibility Guidelines (WCAG), are published by the World Wide Web Consortium (W3C). W3C has member organizations from the industry all over the world [16] and is behind standards for web technology such as CSS, HTML and XML alongside WCAG [17]. WCAG is an international standard that explains how to make web content more accessible to people with disabilities. It is a technical standard made for professionals within web development. WCAG 2.0 was published in 2008, and WCAG 2.1, published in 2018 [18], added guidelines on mobile accessibility, low vision and cognitive disabilities [19]. WCAG 2.2 is in draft and scheduled to be published in April 2023. WCAG 2.0, 2.1 and 2.2 are backwards compatible, which means that fulfillment of a newer version always implies fulfillment of a previous version [18]. WCAG 3.0 on the other hand, is expected in a few years time and has a different structure than the previous versions. It aims to be easier to understand and include a broader scope beyond the web, such as apps [20].

WCAG 2.1 is at the time of writing, the most recent, finished version. It is structured in principles, guidelines, success criteria and levels [21 pp. 65–73].

3.3.1 Principles

The WCAG principles are [21 p. 65]:

1. Perceivable - Users can process content by seeing, hearing, or touching. A user that is missing one sense can perceive information through a different sense.

2. Operable - Users can control the application using alternative input devices such as keyboard, voice, switch or head-controlled mouse.

3. Understandable - The content is presented in a way that is consistent, predictable and readable for a wide range of users.

4. Robust - The application functions and can be understood across a wide range of technologies, including evolving technologies.

3.3.2 Guidelines and success criteria

WCAG 2.1 has 13 guidelines grouped under the four principles. Every guideline has a set of success criteria and there are 78 in total. The success criteria are testable, unlike the guidelines, however, they are also technology-neutral. Specifics to different technologies and examples about how to meet the guidelines are listed in a separate W3C asset called Techniques for WCAG 2.1 [21 p. 72].
For example, guideline 1.2 is "Provide alternatives for time-based media." and success criterion 1.2.2 is "Captions are provided for all prerecorded audio content in synchronized media, except when the media is a media alternative for text and is clearly labeled as such."

### 3.3.3 Levels

Another dimension to categorizing WCAG is the levels A, AA and AAA. If level A is not met, the content is completely inaccessible to certain users. For instance, if there is no alternative text for images, blind users cannot determine what they are supposed to show or if they are at all important. If level A is met but not level AA, certain users need to use extra effort and workarounds to access the content. For example, if there are no focus states on buttons, keyboard users can still use trial and error to determine where they are currently at. AAA is the highest level and addresses usability rather than barriers. For instance, writing text in in high-school level English is recommended because it makes more people able to easily read them, but it is not suitable for all applications, for example an online medical science book [21, pp. 67–69].

Each success criterion is labeled with a level: 30 are level A, 20 are level AA and 18 are level AAA. For an application to conform with WCAG 2.1 level AA, all level A and level AA criteria must be satisfied, which totals in 50 success criteria [22].

### 3.3.4 Difficult to apply

WCAG has been critiqued for being confusing, hard to locate and using too general terms. From version 1.0 to 2.0, the guidelines were rewritten to apply to everything and abandoned terms specific to the HTML, CSS and Javascript the web is still mainly built on. This, according to the critique, has made WCAG much harder to apply to "the real things developers work on every day" [23]. Two studies have shown that beginners cannot test the WCAG success criteria in a reliable and valid way. The participants were told to test 15 success criteria and for only 2 of them did a clear majority assess them correctly [24, p. 8].

### 3.3.5 Webbriktlinjer.se

The Swedish Agency for Digital Government (Myndigheten för digital förvaltning) is responsible for the official guidelines on how to work with websites in the public sector in Sweden. The guidelines are posted on the website webbriktlinjer.se. The first version of the guidelines was posted in 2002, and has since then been reworked by several organizations. Since 2012, they are based on WCAG, level AA [25]. The agency writes that the original WCAG standard can be experienced hard to interpret and that they decided to publish simplified descriptions of the criteria with explanations, illustrations and examples [26].
3.4 Automatic accessibility evaluation tools

An alternative or supplementary method to let experts and disabled users perform manual evaluations of accessibility, is to use an automatic tool. When triggered on an interface, the tool generates a report, which flags the accessibility issues and describes them, and ideally also gives guidance on how to fix them [27, p. 9]. Automatic tools inspect the source code and determine the compliance level of the website with a specific set of guidelines, such as WCAG. There are both free tools and commercial subscription-based ones. Some are browser-based, some are located within other software such as IDE’s (Integrated Development Environments) and some are their own software [28, p. 20]. Lighthouse, Siteimprove, AChecker, WAVE, TAW, Deque’s Axe, Total Validator and ARC Toolkit are all examples of automatic accessibility evaluation tools [27, p. 3], [28, p. 18], [29, p. 1]. Two examples of accessibility reports in Lighthouse are shown in Fig. 1.

Figure 1: Lighthouse is built into the browser developer tools in Chrome [30]. To the left is a report of Umeå University’s website, which has the maximum score of 100. Further down in the report the 22 audits that the website passed are listed. To the right is the corresponding report of Metria’s website, which scores 59 in accessibility. The audits that have not been passed are marked with red triangles. They are listed under categories and the top one is Aria-labels.
There have been attempts to measure how effective automatic evaluation tools are at detecting accessibility issues on websites. It is trivial neither to define in which way this effectiveness should be measured nor to perform such a measurement. A 2013 paper by Vigo et al. [31, p. 3] differentiated between coverage and completeness and later studies have copied or extended this method. Coverage indicates how many of the WCAG-success criteria a tool is able to detect. Completeness indicates how many out of the total number of errors in a website the tool can detect, regardless of the distribution across the different success criteria. The sampling of websites and assessment of the true amount of violated success criteria complicates the measurement further.

Regardless of whether coverage or completeness is used to measure effectiveness, tools are far from satisfying 100%. Vigo et al. [31, pp. 5–6] tested tools against WCAG 2.0 and found the coverage to be 23-50% and completeness to vary between 0% and 48%. A 2017 study [28, p. 26] tested tools against WCAG 2.0 also and found the coverage to be 12-33% for individual tools and 55% when combining eight tools together. A 2021 study [29, p. 11] tested tools against the newer WCAG 2.1, level AA, and found the coverage to be 38-60% and completeness to be 15-40%. All three studies performed manual expert checks to use for comparison. However, the automatic tools found errors the experts did not, and therefore "the total number of violations" was defined to be the errors found in all the automatic tests and the manual checks combined. It is worth noting that the experts found far from all the errors.

Other studies have considered the consistency and usability of the tools. A 2020 study [32, p. 14] tested four free tools and found the reports across them to be inconsistent. Comparing two reports at a time, the proportion of matching issues was 88% at the highest, and 28% at the lowest. This study was directed towards library websites, for which content managers without technical background are crucial, and concluded that "for automated and manual assessments, knowledge of the WCAG success criteria, web languages, and disabilities was required" and "if the user did not hold the necessary knowledge or skill set to read or edit technical reports or languages, analyses and corrections would be practically impossible." A 2019 study [33, p. 90] tested five tools and also found the number of errors to differ vastly between tools. It pointed out that different tools categorized the errors differently and discussed that some ways are better for developers while others are better for evaluators.

3.5 Previous research on accessible web

Ara et al. [34] and Paiva et al. [35] analyzed the state of the web accessibility research through systematic literature studies. They investigated which topics have been covered by the research in 2010-2021 and 2011-2019 respectively. Both studies independently found that automated testing has been the focus in the research. They both call for frameworks and methods to correct issues found in testing. They also agree there is a research gap regarding engineering for disabilities other than visual impairment. Both studies concluded there is a need for reference architectures for referring to accessible web design, development, and evaluation processes. Paiva et al. wrote that such an architecture "would be useful to help accelerate and ensure affordable software." They
also stated that there are few articles on software maintenance and also pointed out the importance of including accessibility in the agile methodology [p. 14]. Ara et al. specifically asked for “easy methods to understand and ensure accessibility requirements [...] during the development phase” [p. 30].

3.6 Design Systems

Companies within software production employ design systems to improve time efficiency [p. 15], consistency and communication [p. 72] between designers and developers [9]. Simply described, a design system is a collection of reusable components that can be assembled into digital applications. Components range from small elements such as buttons and icons to larger components such as forms and navigation areas, to general styling such as typography and colors. Each component commonly has a name, description, example, and code included in the design system [9 p. 70]. The major advantages of a design system are that it enables a single source of truth, and that it provides concrete assets of designs linked to code [36 p. 3]. Companies have the option of adopting an existing design system [37 p. 2] or making their own, based on their particular context and needs [9 p. 154]. Google’s Material Design, see Fig. 2 is a prominent example of a design system, and it is used by many companies other than Google themselves [37 p. 2]

Figure 2: Google’s design system Material design [38]. It has sections called “Foundations” with patterns, “Styles” and “Components”. In the components overview, one can scroll through a catalog of components. Here visible are checkbox, chips and date pickers. They can be clicked to view examples, read documentation and access design and code files.
Because designers and developers often work independently, a hand-off phase is required for turning designs into code. This transition is prone to miscommunication, due to the different backgrounds of the designers and developers [39, p. 630]. Designers tend to use terms that reflect how the user sees an object, while developers use terms borrowed from programming language. Also, designers are often not aware of the technical constraints of building interfaces. This causes both redundancy and missing information in the hand-offs [39, pp. 632–636]. A way of bridging the gap between designers and developers is to work modularly and reuse components [39, p. 639]. When designers use components from a design system, the hand-off is thus simplified, because the components have already been technically verified, and each component has a name that both designers and developers use.

Worth noting is that the users of design systems are the designers and developers, not the end users of the applications. Unless the company adopts an existing design system, their own designers and developers are also the creators of the design system. Previous research has shown that many artifacts delivered by designers to developers, such as personas and scenarios, are not actually used by developers [40, pp. 1, 32]. Even prototypes can be seen as "owned" by the designers and require their expertise to operate and interpret [41, p. 357]. A potential strength of design systems is hence that they are co-created by designers and developers.

Working with a design system involves a number of activities. A designer or developer with a given case in a user interface project scans the design system to see which component or pattern is suitable for their case. If none is suitable, they modify an existing pattern or create something new. If their modification or creation can be useful for other projects, it is submitted to the design system. There is typically a dedicated team that supports processes surrounding the design system. In some companies, the design system team is also responsible for creating components and patterns, while in other companies the project teams also contribute [36, pp. 64–65].

### 3.7 Issue tracking systems

An issue tracking system, ITS, is a piece of software where the development tasks of a software company are managed. These can be new features, bug fixes and other maintenance tasks. Each issue has a description, a state such as opened or fixed, a priority and comments made by the employees working with it [42, p. 480]. The ITS is used throughout a software product’s life, for both production, releases and maintenance work. It is a database for tracking issues but also a focal point for communication and coordination. The ITS serves as a shared to-do list and also an archive of completed work. It helps with issue reporting, assignment, tracking and resolution. The issues’ states and attributes are typically manipulated by a number of stakeholders within and beyond the software team. Issues can be assigned back and forth among project managers, developers, and quality assurance staff. Therefore issue tracking can be seen as a social process [43, p. 1]. Issues are commonly displayed in a board, see Fig. 3.
Figure 3: An example of a board within an ITS. Two issues have the status "To do", two have the status "In progress" and three have the status "Done". Five of the issues have been assigned to an employee. The issues can be dragged and dropped across the columns and clicked to view more information and edit them.
4 Method

The method to develop an accessibility methodology for design systems consisted of a case study of the external partner company Metria. The case study was designed to acquire results useful for Metria in particular. In other words the case study was both intrinsic and instrumental [45, p. 165].

A design approach was applied in the case study and therefore it will hereon be referred to as a project. Techniques used for designing concepts and UX prototypes were used to create an accessibility methodology. An approach inspired by the Double Diamond’s view [46, p. 6] of widening and narrowing the scope, see Fig. 4, was used to find a solution for Metria. The phases were:

1. Discover — Gain an understanding of the context.
2. Define — Based on the new knowledge set a goal for the case study.
3. Ideate — Produce plenty of ideas and choose which to proceed with.
4. Develop — Create the concrete methodology.
5. Evaluate — Test the methodology with users.
6. Modify — Change the methodology according to feedback and possibly add new elements to the methodology if requested by users.

![The triple diamond method](image)

Figure 4: Graphical representation of project phases. The scope is widened in the phase Discover, then narrowed in the phase Define. In the phase Ideate the scope is widened again, until an idea is chosen and the phase Develop beings, which narrows down the scope again. In the phase Evaluate the score is widened a third time, and in the phase Modify it is narrowed again. The last two phases are iterated on until an end result is delivered.
The phases were not completely sequential. Occasionally activities in two phases were performed in parallel, as one phase was being finished and a new one was beginning. Discover was given three weeks, Define one week, Ideate two weeks, Develop three weeks, Evaluate one week and Modify one week. The last two phases were repeated twice. In total, the project and thesis writing lasted 15 weeks.

A total number of 26 people participated in the project, including the author. The steps of each method will be described in the following sections of the thesis. Fig. 5 shows an overview of the steps and which participants were involved in which steps. From the target group, three designers and 10 front-end developers participated. Target group members participated at five occasions during the project: initial interviews, workshop,
survey, focus group and user tests. A person responsible for the design system at Metria participated in a workshop and an evaluating interview. Another seven Metria employees participated in a workshop, survey and focus group. People from outside Metria were invited for two special occasions; a design student was invited to a workshop and three accessibility experts performed an expert review.

4.1 Phase 1: Discover

The first step in the case study was to explore the context in which the methodology was to be applied. Understanding of the users, which were designers and front-end developers at Metria, was gained and knowledge of the setting of their workplace was collected. The exploration revolved around the two topics design systems and accessibility in parallel, but it was intentionally problem-oriented rather than solutions-oriented. This way of finding the problem first and then solving it is a part of the Double Diamond way of designing [47, p. 15].

4.1.1 Interviews

Interviews were chosen as the first step in the project because they help build an understanding of the users’ needs, goals, practices, concerns and attitudes in the initial exploration of a design project [45, p. 189]. Five interviews were conducted. Each interview lasted 25 minutes. Two were held in an office meeting room and three using a video conference tool. Two interviewees were developers and three were designers. The interviews followed a semi-structured format with seven open-ended questions and, to confirm some details, a few closed questions. A semi-structured format was chosen because it permits elaborating on unforeseen topics of interest that are mentioned by the interviewee, which can provide additional insight [45, pp. 198–199]. The format was tested in one pilot test to verify timing and phrasing.

The seven questions were:

• How do you work with accessibility at Metria today?
• What are some concrete activities you perform regarding accessibility?
• What do you miss to do a better job regarding accessibility?
• In what phase of the production does accessibility need to be included?
• Do you have any thoughts on how the cooperation between designers and developers can be improved? (only developers were asked this)
• How does Metria work with the design system today?
• How can the design system support you regarding accessibility?

The interviewee steered the direction and disposition of each interview. The interviews were recorded and transcribed.
4.1.2 Affinity diagramming

Affinity diagramming is a method used in UX design to sort research findings into clusters using sticky notes [49]. It was chosen to analyze the interview data because it suits the designer approach chosen and a qualitative method was desired. Of interest was not how many people mentioned an idea but whether it was brought up by more than one person and whether it was expressed on both the designer and the developer side.

Affinity diagramming is often performed by a group of people [48] but in this project, it was performed by the author alone. Quotes from the interview transcripts were written on sticky notes. A number was written on them to keep track of which interviewee said what. The notes were thereafter attached to a wall and sorted in categories and subcategories, see Fig. 6. As more and more notes were added, the notes were regrouped and divided into smaller categories. The subcategories were then turned into findings phrased as 1–3 sentences. Key findings were chosen using a set number of 15 stickers. Lastly, all the findings were digitally recorded in a bulleted list with key findings highlighted.

4.1.3 Inspection of existing design systems

A small-scale and unstructured inspection of five existing design systems was made. One of them was the design system Metria had begun to make before the project began and the other four were public design systems from other companies. Two of them were from Swedish companies, one was Google’s Material design and the last from another large global company. Notes were taken on which content the design systems have and how the contents are organized. Extra attention was put on how accessibility was included in the design systems.

Figure 6: Category structure of affinity diagram of interview data. Quotes are written on yellow notes. Each column with a blue note above it is a subcategory. Some categories only have one yellow note. The blue notes describe the categories. Orange notes show top-level categories.
4.2 Phase 2: Define

The purpose of the second phase was to arrive at a problem statement. A reflecting exercise was carried out and then two techniques called 5 Ws and User Need Statements were performed, before a final problem statement was formulated.

4.2.1 Reflecting exercise

The reflecting exercise was performed in a comfortable, undisturbed environment using the findings from the phase Discover as a basis. The reflection circled around these questions:

- What is the problem and what is not the problem?
- Which phenomenon/concepts are opposing each other?
- What are chains of causes and consequences happening?

4.2.2 5 Ws

The technique 5 Ws was chosen because it has been proposed to gather facts for a problem statement [49]. In it, five questions are answered:

- Who is affected by the problem?
- What is the problem?
- Where does this problem occur?
- When does the problem occur?
- Why does the problem occur? Why is the problem important?

The technique was repeated twice, once with the topic "accessibility" and once with the topic "design system". Only later commonalities between the two topics were searched for.

4.2.3 User Need Statements

User Need Statements were performed because they have been called "a primary tool" in the define stage of a design process [50]. The statements traditionally have three components combined in the pattern "[User] needs to [need] in order to [goal]". It is important that the need and the goal are verbs, not nouns, to keep away from solution-based thinking and focus solely on the problem. The User Need Statements are not supposed to be quotes from users, but are rather to be derived by the designer from the combined research. The technique works by generating several statements and rephrasing them, rather than trying to find the perfect statement right away [50].

In this project, a template with three columns and 24 rows was printed, see Fig. 7. It was then filled in in a brainstorming manner. Not until all rows were filled, an assessment decided which statements best represented the problem.
4.2.4 Formulation of problem statement

A problem statement is defined by Rosala as "a concise description of the problem that needs to be solved" [49]. It should include the background of the problem, the people affected by the problem and the impact of the problem on the company. It is important that the statement does not contain a solution. It should be brief, about three sentences. A problem statement can be negative or positive. In the latter case, it captures an opportunity and highlights the gap between where the present state and the desired state. It can be formulated at the beginning of the discover-phase of the Double Diamond [49], but in this project, it was instead used as a bridge between the phases Discover and Ideate. The problem statement was formulated by the author at the end of the phase Define. It was then verified by a designer at Metria.

4.3 Phase 3: Ideate

In the third phase, the goal was to find ideas of solutions to the problem statement that can be proceeded with in the fourth phase Develop. Two workshops were held to generate as many ideas as possible. In both workshops, the author acted as facilitator and participant. The ideas from both workshops were later sorted by the author. A third step was thereafter necessary to convert abstract ideas into deliverable elements of a methodology.

4.3.1 Designer workshop

The first workshop had two participants: the author and another master's student in interaction technology, with insight into the case at Metria. Ideation is benefited by an
inspiring location outside of the office [51, p. 44], and therefore the workshop was held in the lounge of a movie theater, see Fig. 8. It lasted two hours.

Two exercises were performed. They were chosen because they spark creativity, result in many ideas being produced and complement each other well.

The first exercise was called Random Words Associating [51, pp. 111–113]. The participants took turns saying the first word that came to mind, freely associating from the previous word, to form a chain of random words. For example "Frankenstein" lead to "revenge", which in turn lead to "opposite roles". In the next step, the actual problem statement of the project was presented, and the participants associated solutions to the problem with regard to the words in the word chain. "How can we solve the problem with regards to Frankenstein?". These ideas were first written very briefly on sticky notes, to keep the energy of the ideation high. Toward the end of the exercise, some ideas were chosen and elaborated on to give them more substance, see Fig. 9. This exercise requires the participants to be highly comfortable with expressing their thoughts to each other [pp. 64–65], and it was chosen for this workshop because the author and the other students knew each other well.

The second exercise was called The Lotus Blossom [51, p. 114]. The problem statement of the project was written in the center of a 9x9 grid on a large sheet of paper. Eight ideas were then thought out, written down briefly on sticky notes and placed around the middle square. These ideas were then moved out to the rims of the grid. These eight ideas were thereafter in turn surrounded by new sticky notes. This time the new sticky notes were not new ideas. Instead, they were supposed to answer how the idea in the center was supposed to be implemented. The layout is shown in Fig. 10.
Figure 9: Notes produced during the technique "Random words Associating". In the middle written with a blue marker is the chain of random words. Ideas on sticky notes have been placed in unordered piles around it. The ideas in the bottom right corner have been placed to the side at the end of the exercise.

Figure 10: Structure of Lotus Blossom exercise, step by step. To the left, eight initial ideas have been written down. In the middle, the eight ideas have been moved out to an outer ring. To the right, more ideas have been thought out inspired by each of the eight initial ideas.
4.3.2 Stakeholder workshop

The second workshop had four participants: the author, two employees at Metria with responsibility for the design system and a third Metria employee without connection to the design system, but with a creative skill. An ideation workshop can benefit from having participants with little knowledge of the problem as long as they have the ability to think innovatively [51, p. 43]. Three employees who are in the target group of the project had to decline the invitation. One of the employees responsible for the design system is a designer, hence designers were represented, but unfortunately no developers were present.

The stakeholder workshop took place in the office in an open environment, see Fig. 12, and lasted 1.5 hours. Firstly, the participants were instructed to produce as many ideas as possible, bad or good did not matter, and not to assess their own or each other’s ideas. An icebreaker exercise was done to get the participants into this mindset. It was a brainstorm on the topic "Impossible things to do with this clothes hanger that are actually possible.” It lasted about 10 minutes.

Then, the actual topic of the workshop was presented. The problem statement from the phase Define was read out loud to the participants and put up where everyone could see it. Two ideation techniques were used to think of solutions for this problem. They were chosen because they spark creativity, complement each other well and do not require high maturity of the group or experience by the facilitator.

The first technique is in the thesis called "No limits" and uses lateral thinking. In the opposite of linear thinking, which bases solutions on experience and logic, lateral thinking is disruptive. It challenges assumptions and seeks alternatives [52]. Two common activities used in lateral thinking are provocation and movement [53], and these were performed in the workshop. A format proposed in a blog post [54] was followed. First, a list of ideas that are unrealistic, but solve the problem, was written on a board. These were the provocations. Then, the underlying principle of each provocation was formulated on a separate board next to the provocations. Lastly, realistic solutions were derived and written down on a third board. This last step is what is called movement. An example is [54]:

- Problem: A car runs out of gas every 50 km and needs constant refueling.
- Provocation: Move everything closer that the car needs to drive to.
- Principle: If you drive shorter distances, you do not need to refuel as often.
- Solution: Plan optimal routes for the errands of the whole week.

The boards used are shown in Fig. 11. Different colored markers were used for provocations, principles and solutions.
The second technique is referred as "Brainwalking". Brainwalking is a variant of brainstorming in which participants walk around the room and build on each other’s ideas [53]. The format used in the workshop was also inspired by a technique called 6-3-5 [51, p. 130]. Four sheets of paper with 4x3 squares, see Fig. [12] were placed on four stations in the room. The participants started at a station each and wrote down three ideas in the first row of squares. After four minutes, they swapped station and kept ideating based on the notes left by the previous participant. The participants were allowed to either associate freely based on the previous row or to think of entirely new ideas. They were encouraged to mix both realistic (linear thinking) and crazy ideas (lateral thinking). After four rounds, the group gathered the sheets, discussed and circled the best ideas.
4.3.3 Sorting ideas

The ideas were filtered after the workshops by the author. First, notes that answer why the problem should be solved, rather than how to solve it, were put to the side. Negative ideas and duplicates were removed. Next, the ideas were grouped with the question "What problem do they actually solve — the problem statement or a different, related problem?" in mind. The group of ideas addressing the relevant problem statement was analyzed more closely. Some ideas were excluded but documented with a motivation why. The remaining ideas were summarized in a mind map.

4.3.4 Conversion into methodology

The ideas in the mind map were partly abstract. Therefore an extra step was necessary before a methodology, consisting of methods and tools, could be developed. In this step, the mind map was converted into a list of possible deliverables. The deliverables were documents describing methods and tools that can be delivered to Metria and possibly other companies in a similar situation.

4.4 Phase 4: Develop

The goal of the fourth phase was to implement one or several ideas from the third phase Ideate into a methodology consisting of a set of deliverable methods and tools. After the ideation, four ideas of deliverables were defined closer by producing pictures that would represent each deliverable. Surveys were then used in order to collect feedback on the ideas from a large number of employees. Then, two ideas were developed one step further into drafts of deliverables.

4.4.1 Four samples

Four ideas from the list of possible deliverables from the previous phase were elaborated on. A small picture was created for each deliverable to demonstrate what it could look like. The pictures were made either using a word processor or a UX prototyping tool when interactive elements were part of the idea. The pictures were used in a survey to audit the demand for each deliverable among the Metria employees. The respondents were asked "Which of the following options would help you the most in your work?" and got to rank them from most useful to least useful. The options were assigned a score: 4 points for every time it was ranked the highest, 3 points for the second-highest and so on. The survey was filled in online. The number of respondents was 18. All respondents had experience with either design or front-end development, although some mainly worked with back-end.

4.4.2 First drafts of deliverables

The highest-scoring deliverable from the survey and another deliverable from the phase Ideate were developed further. They were created in a word processor. One document manifested methods and two manifested tools. To create the tools, webbriktlinjer.se and
WCAG 2.1, as well as WCAG 2.1 Techniques were used as sources for the content. The format of the tools was based on the findings from the phase Discover and the problem statement from the phase Define. To create the methods, the findings from the phases Discover and Ideate were used.

4.5 Phase 5: Evaluate and Phase 6: Modify

The evaluating phase was performed in two iterations. The purpose of both iterations was to answer whether the methodology is a solution to the problem statement from the second phase Define. The aim was to evaluate the methodology compliance and not the outcome. Compliance in this case refers to whether the methodology is being followed, is easy to understand and fits in the work process. The outcome in this case corresponds to improved accessibility in the end products.

In the first iteration, the deliverables created in the fourth phase Develop were evaluated. Two expert reviews and a focus group were used in the first iteration. In the second iteration, the deliverables had been modified and were evaluated with an expert review, four user tests and an evaluating interview.

The motivation behind using expert reviews was to get an outside perspective. The invited experts had much experience with methodical accessibility work. Since experts were chosen due to their expertise in accessibility, the method deviates from the norm of letting UX experts review a design prototype [55]. Combining user tests with expert reviews provides the best results. It is recommended to perform expert reviews before user testing [45, p. 269], and iteratively [55], which was also done in the project. Performing the expert reviews first allows for repairing small problems in the material before showing it to the users to let them focus on other aspects.

The decision to hold a focus group was made to stimulate participants to new thoughts and let them support each other in understanding abstract concepts and imagining hypothetical situations. Because focus groups are dynamic, they can raise ideas that participants would not have thought of in one-to-one interviews [45, p. 204].

What people say and what people do are often different [56]. Another way to put it is that even the users themselves struggle with identifying their needs [45, p. 271]. Therefore user tests were performed to add a behavioral perspective to the attitudinal perspective provided by the interviews and focus group. It is a well-established method in UX design [57] as well as the Double Diamond [58]. The goal is to improve the design by investigating what causes problems for the user as well as what works well [45, p. 264].

4.5.1 Expert reviews

The expert reviews had the form of one-hour meetings between the author and an expert. They followed a loose structure where the experts first commented on strengths in the methodology, to make sure they were not removed in the next modification. They then commented on problems and gave recommendations and rated their severity "high",
“medium” or “low”. The structure was inspired by a guide by the Nielsen Norman Group \cite{55}. Two out of three expert reviews were held online, and one in an office meeting room.

4.5.2 Focus group

In the focus group, the main object of evaluation was the tools of the methodology, although the methods were also mentioned. It was believed that more employees would be able to form an opinion about concrete tools rather than abstract methods.

The focus group had seven participants, which is within the lower range of the recommended span \cite[p. 204]{45}. All the participants were employees at Metria. One was a designer and one was a front-end developer. The other five have experience with front-end, but worked mainly with back-end or testing. Two more front-end developers and a designer were invited but were unable to attend. The absent designer got to leave feedback in a complementary interview after the focus group.

An undisturbed environment was chosen for the focus group: an office meeting room. To remove the associations with the usual office meetings and to stimulate a collaborative discussion, the chairs were put in a semicircle, facing a large screen. In the middle, printed copies of the deliverables were placed.

The focus group lasted one hour. In the beginning, it was emphasized that the methodology is meant to be improved and the participants were encouraged to disagree with one another. Then, an ice-breaker question was discussed: "Would you prefer the ability to travel 10 minutes or a thousand years into the future?". Thereafter the deliverables were briefly presented to the participants for the first time. The whole introduction lasted 25 minutes.

The discussion was initiated with the question "What do you believe is the purpose of making these documents?". Further discussion was opened with the question "What do you think about the material?". When the group was silent, it was followed up with questions such as "Do you think the material is overwhelming or easy to overview?" and "Is it clear or unclear?". The participants were asked what they thought about the format of the tools. Thereafter an unstructured discussion followed. Towards the end, they were asked if they thought the tools would be used in practice, and what features could make them more likely to be used.

The absent designer got to leave feedback in a complementary interview after the focus group. This decision was made since designers were the main target group of the project. The interview followed a similar scheme as the focus group and the interviewee got to respond to opinions that had been brought up in the focus group. It lasted 50 minutes and took place online.

4.5.3 User test of tools

Four user tests were performed. Two users were designers and two were developers. Three tests took place online and one in the office.
The users were given two tools and instructed to try them out for 30 minutes. They used their own computers at their own desk. They were told to choose a graphical interface component, either one they are working on now or have worked on earlier, and use the tools to support them in accessibility quality assurance. The idea behind this setup was to simulate a situation were an employee is creating a component for the design system and wants to refine it to be accessible. The participants were instructed to try both tools but could choose one to work with more. During this time, they were not being observed, but the author was present nearby and they could ask questions or make remarks if they wanted to.

When the users had tried the tools, they were interviewed about the experience. They were asked these questions:

1. What did you think about the tools?
2. Which tool did you prefer?
3. Do you think the tool would be used in practice? Why or why not?
4. What do you think is the biggest risk that the tool will be forgotten about? What can be changed in the tool to minimize that risk?

At the end of the interview, the users were shown an example of a tool from a different company. They were asked what they thought of this solution compared to the tools they had tested. The interviews were recorded and later notes were made, but they were never transcribed.

4.5.4 Evaluating interview

It was decided to additionally evaluate the methods with one of the two employees at Metria with responsibility for the design system. This was done to complement the reviews made by experts. They were instructed that the purpose of the interview was to evaluate if the methods are clear and applicable. The interviewee was given 30 minutes to read the document at their own desk, and the author was present nearby. Thereafter followed a 30-minute unstructured interview.

4.5.5 Modification process

Small changes were made to the deliverables after each expert interview. The changes were based on the problems provided and some conclusions made by the author based on the conversation that had taken place. Which changes to make at this point was decided mostly based on how little time they required. After both evaluations were finished more and larger modifications were made. This time around, the experts’ severity ratings of the problems were regarded, but time was still the major factor. The lists of strengths were also used to make sure positive aspects of the methodology did not get lost in the modification process. When the experts disagreed and said the same aspect was a strength and a problem respectively, either a compromise was made or the majority vote decided.


5 Results

The results of each of the six project phases are presented one after the other. The phase Discover resulted in findings about the Metria case. The phase Define resulted in a problem statement. The phase Ideate resulted in ideas. The phase Develop resulted in a draft methodology. The phase Evaluate resulted in feedback on that methodology and new findings regarding design systems and accessibility. Lastly, the phase Modify resulted in a final proposal of a methodology. The deliverable result of the whole project was a set of guidelines describing how to work with accessibility in and through a design system. The crafting of these guidelines, from the collection of information to drafting to testing, is the common theme throughout the whole thesis result. The end version of the guidelines, after two iterations of evaluation and modification, is summarized at the end of this section.

5.1 Phase 1: Discover

The first data collected in the project was the interview responses from two designers and three developers at Metria. In total 51 findings were identified in the affinity mapping of this interview data. These were reduced to 15 key findings about the context at the workplace for designers and developers. The key findings are as follows:

1. The work that is currently performed regarding accessibility at Metria is at the initiative of the individual employee, in implementation as well as design.

2. There is an unofficial reasoning that it is unlikely that the users of the expert applications produced by Metria would have any (serious) disabilities. Accessibility is perceived to be more important for the smaller share of applications they produce for the public.

3. Developers think it is the designer who takes on the task of adapting for accessibility. The designers have also noticed this.

4. Testing for accessibility at the beginning or at the end of the production does not suffice. Even if the application is perfectly accessible at one point a change can affect that result.

5. The interviewees want to build accessibly independent of how probable it is that the user has a disability. It is seen as good design and it is enough that one user with the need appears. Additionally, most people have some sort of need for accessibility.

6. WCAG is thought to be overwhelming. It causes employees to rely on an inner instinct for what is accessible enough rather than actually checking.

7. The design system facilitates communication. It creates a middle ground for developers, designers and product owners. It makes it easier to ask people outside of one’s team for help.
8. The design system is a natural place to test the most important components. Today, the threshold for testing an application with which one has not been involved oneself is high, since one might not understand what it is used for. Another barrier is that it is difficult to get access to the test environment, which means receiving an account and the correct URL.

9. The user situation for the 20+ Metria applications varies much. Some are used on small screens out in the field, some are used on desktops in offices. Some are used seldom and for a short while, others are used daily for hours. The design system must provide freedom for different solutions.

10. The developers want to urge designers to reuse design components. It is a precondition to reuse code, which is also beneficial for accessibility compared to coding new components.

11. The interviewees are not unanimous on whether the design system should be a) a vast collection of different variants of components that can be retrieved or b) a lightweight register of the most important decisions that is to be followed as far as possible.

12. Guidelines regarding accessibility should be connected to the resource one is wanting to retrieve in the design system. It should not be its own tab that could easily be overlooked. An employee has written a page in Metria’s company wiki about accessibility and the colleagues reacted positively to it then but they have not looked at it since.

13. Metria’s applications are not concerned with all WCAG-guidelines, such as the ones regarding video. The interviewees want to see guidelines grouped and connected to functionality such as "menu".

14. The interviewees believe in using automatic accessibility evaluation tools, in general and inside the design system.

15. Metria employees think accessibility is important but the interviewees think they need to be reminded about it in the they-to-day work for it to be done.

Two additional findings were collected from the target group outside of the interviews.

- A Metria employee has called the The Swedish Agency for Digital Government to ask whether their applications are exempted by the coming law, since they contain maps, and received no clear answer. It has been difficult to decipher whether the map services provided by Metria are for navigational use or not.

- The Metria employees disagree on how the components in the design system should be organized. Some think they should be organized in directories, while others think they should be in a flat hierarchy and displayed in a grid-view to be easier to browse through.
The inspection of Metria’s existing design system revealed it has two main sections: Guidelines and Components. The guidelines are text pages describing how to use buttons, color, fonts and icons and they contain some advice regarding accessibility. The components are sorted into directories. The naming of the components and the contents in the component pages are inconsistent. Swedish and English is mixed. Some components have example data and others are empty. There is little to no documentation telling the user when and how it is suitable to use each component. The documentation that is there has been auto-generated from the comments in the code, such as descriptions of attributes. There also seems to be a lack of overview because many components that are there are very specific and basic components are not there. For example, to get a primary button one needs to navigate to "Spinner button" (the spinner is an animated circle that communicates that something is loading) and deactivate the spinner. The inconsistency and lack of overview can be explained by the fact that the design system has not been its own project at Metria. The components have been created and inserted by employees while they have been working in a project developing an application. There is a tab within each component page for automatic accessibility testing, which is based on Deque’s Axe, but it only tests few requirements and it fails to test contrast in most cases.

The inspection of other design systems showed they all contain components and styling guidelines. Some have separated components and patterns. Some have another main section describing company values and design principles. Accessibility has been included to some extent in all of the design systems. The information on accessibility is separated from the other resources — either it is its complete own section, a guideline, or it is a heading or tab in each component.

5.2 Phase 2: Define

Before presenting the problem statement, which is the main result of this phase, some notable results from the preceding activities of the phase are documented to provide more depth to the problem.

Some important user needs statements created were:

- Designers and developers need to streamline and weave the work with accessibility into the regular work in order to feel that they have time to do it properly.
- The employees need to set a common bar for accessibility in order to avoid making their own assumptions of how prioritized it is, and to have something to strive for.
- Designers and developers need to feel that they have approval from the company management in order to make applications thoroughly accessible.
The reflecting exercise provided boundaries of the problem that are presented in Table 1.

Table 1: Results of reflecting exercise

<table>
<thead>
<tr>
<th>The problem <strong>is not</strong> that...</th>
<th>The problem <strong>is</strong> that...</th>
</tr>
</thead>
<tbody>
<tr>
<td>• ... employees do not want to build accessibly.</td>
<td>• ... accessibility is not prioritized.</td>
</tr>
<tr>
<td>• ... employees know nothing about accessibility.</td>
<td>• ... there are no routines for building accessibly.</td>
</tr>
<tr>
<td>• ... there are so many disabilities that it is impossible to adapt for them all.</td>
<td>• ... adapting for one disability often automatically results in improvements that benefit others too.</td>
</tr>
<tr>
<td>• ... WCAG is hard to comprehend.</td>
<td>• ... WCAG criteria are logical to understand but the amount is overwhelming — it is seen as an unclimbable mountain.</td>
</tr>
</tbody>
</table>

The final result of the phase Define was the problem statement:

Accessibility is not prioritized in software development, especially not when there is time pressure to deliver new functionality. Many designers and developers are aware of the aspect of accessibility and find it important. Making the work with accessibility systematic would decrease the time and effort required for it. There is an opportunity to thereby decrease accessibility issues in the applications as well as increase work ethics.

Having reached the conclusion that the problem is to make accessibility measures systematic, it is easy to motivate the design system as an adequate setting for the solution. A such chain of reasoning arose in the reflection exercise and is shown in Fig. 13. Time deficiency is lessened by structure and cooperation which is what a design system is for.
5.3 Phase 3: Ideate

The workshops generated over 100 ideas, which were reduced to 10 topics in an idea mind map. These were in turn converted to a list of ideas of deliverables, which are presented at the end of this section. The deliverables are tools and descriptions of methods that make up the methodology and can be delivered to Metria. Before the idea mind map and list of deliverables, some examples from the workshop exercises are presented.

The random words associating in the designer workshop lead to these creative ideas:

- "How can the design system contribute to accessibility in the applications, related to the word Frankenstein?"
- Idea 1: Dr Frankenstein dug up body parts from a graveyard for his monster. The design system should utilize components with good accessibility measures from old Metria applications.
- Idea 2: Focus on getting the design system "alive” with basic functions, not perfect or beautiful.
- Idea 3: The design system has a drag-and-drop system where you pull together the accessibility code snippets you need.

In the stakeholder workshop, the lateral thinking technique resulted in this thought chain:

- Provocation: We give the users full control of the design system. We design based on what they have decided. They will make sure it is accessible for them.
- Principle: Leave the work to the users.
- Solution 1: Make it easy for users to leave feedback inside the interface. Shorten the communication chain from users to the team that makes the design system.
- Solution 2: Develop functions with settings that the user can adjust to their needs, such as line height.
The mind map, see Fig. 14 had the following topics:

- **Weave into**
  - So you do not need to be reminded
  - In the design system, use the word accessibility in a manner that teaches people it is relevant for everyone, not just people with disabilities.

- **Routines**
  - Onboarding in accessibility for new employees
  - Vocabulary of accessibility — how to refer to user groups

- **Own goals**
  - Spend less time interpreting WCAG and laws and instead set internal bar for accessibility
  - Lower the ambition - set a part-way goal, less overwhelming
  - Prioritized
  - Everyday language

- **Referring to use in context**
  - Do’s and don’ts
  - Utilize old applications
  - Refer to where a component has been used before

- **Get help from the user**
  - The user adjusts the interface themselves, such as line-height
  - Shorten the channel of contact from user to software development and let them influence the components in the design system
  - Contact information to users with different needs is listed in the design system

- **Get help from experts**
  - Links to what others have done
  - Contact information in the design system to an employee with special competence in accessibility
  - Let an external company revise the design system

- **Brief text**

- **Tests**
  - Automatic evaluation tools
- "Squint your eyes and step back" and other manual procedures
- Controlling and welcoming at the same time: it is annoying but after a while becomes your friend

• Checklist
• Rules and guidelines
  - Reserve time for accessibility
  - Deciding on the most basic may be a one-time-job

Figure 14: Idea mind map (in Swedish). The title in blue is "Solves streamlining and making systematic". The main ideas: "Weave into", "Routines", "Own goals" and so on are written in red markers and interconnected. Sticky notes are taken directly from the workshops.

Two other recurring themes in the workshops were gamification and storytelling. They were however disregarded since they solve the problem of lacking motivation or empathy rather than streamlining and systematization, but they are worth mentioning for future work. One innovative idea was to imagine the UI components as characters that get unhappy if not all users can use them. Another was to create five human characters with different needs that embody the design system; to interact with the design system you interact with the characters. Using a superhero narrative or promoting hero emotions was another idea. One workshop participant, who has experience in game design, suggested using particle effects such as confetti when checking off an accessibility test, to increase motivation.

The end result of the phase Ideate was a list of possible deliverables. They are potential elements of an accessibility methodology:

• List of things to consider when making a new design or code
• Checklist for testing whether a UI component is accessible
• Version of WCAG with a filter for role (designer or developer) and functionality
• Guide for writing Do’s and don’ts
• Vocabulary list with terms for different user groups with disabilities and other needs
• Guidelines for setting an accessibility goal at the company
• Small set of bare-minimum mandatory requirements for all applications
• General advice for accessibility in design systems

5.4 Phase 4: Develop

It was decided the project would proceed with developing one deliverable describing methods and one deliverable being a tool. The method deliverable was decided to be a document of guidelines with general advice for accessibility in design systems. A survey was performed to decide which tool to develop, of which the result showed that a checklist was the most desired tool. First drafts of guidelines and two checklists were created, which became the end result of the phase Develop. The documents were written in Swedish.

5.4.1 Four samples

The first thing that was developed in the phase was four picture samples used for the survey, as shown in Fig. 15, which demonstrated four tools. To give the respondents an idea of the volume of the contents of each tool, a number of points or items was estimated and written in a short description next to each picture.

The first idea (Fig. 15a) is a list of things to consider when making a new design or code. There is a separate list for designers and developers and each has about 20 points. The second idea (Fig. 15b) is a sorted version of WCAG with a filter for role (designer or developer) and functionality (e.g. menu) and has 41 items in total. The third idea (Fig. 15c) is an interactive checklist for testing for accessibility. The fourth idea (Fig. 15d) is a rules list of the 12 most important requirements that are mandatory for all applications.

The interactive checklist for testing scored the highest, with 64 points. The sorted WCAG scored second highest with 42 points. The list of things to consider scored 39 points and the 12 rules scored 35 points.

The interactive checklist with testing procedures was thus the winning option, and it was therefore decided it should be the second deliverable of the project, alongside the guidelines. During the development of the checklist, it was however understood, that two lines of text will not suffice to communicate the complexity of the requirements that need to be tested. Therefore the deliverable created much resembled the second most popular option in the survey, namely the sorted WCAG. The difference is that the items in the checklist did not mirror the WCAG structure one-to-one, and the focus was on how to test, not on the requirements themselves. When practical, two WCAG success criteria were combined in the same test, or one success criterion was broken into several tests.
(a) List of things to consider when designing or coding. A simple bullet point list with concrete requirements in informal language such as "Think about the hierarchy of headings on a page, using h1, h2, h3, etc."

(b) Sorted WCAG with filter option for role (designer or developer) and functionality (e.g. menu). The tool shows how many success criteria are applicable to the filter settings and lists them, with reference number in WCAG, title and link to read more. The first success criteria shown here says "Clearly show which element is in focus".

(c) Interactive checklist for testing. The first item is marked as done and it says "Check contrasts with All1ycolor". All1ycolor is a made-up name.

(d) Metria's 12 accessibility rules. A simple text document with points such as "All buttons and links have descriptive names".

Figure 15: Sample pictures of tools sketched up in a word processor and UX prototyping tool. They were used to demonstrate to survey respondents what different types of tools could offer, so that they could rank which one they wanted.
5.4.2 Draft of guidelines

The first deliverable made in the phase Develop was the guidelines. Their contents were based on findings from phases Discover and Ideate. Six guidelines were formulated. Each guideline had a description below it with 20-200 words. The draft descriptions have not been documented, but the final version of the guidelines including descriptions can be found in Appendix A.

The six guidelines were:

1. Set your own company accessibility goals.
2. Weave accessibility into all other resources.
3. Break down testing based on phase, level and functionality.
4. Set aside time for working on the design system.
5. Begin small and build onto the design system as you go.
6. Use language that is easy to grasp and explaining.

The guidelines attempted to solve several problems identified in the phase Discover, see Section 5.1 below referenced with key finding numbers in brackets. Some of the guidelines were traces of ideas from the phase Ideate.

Guideline 1 described that with clearly stated, official company goals, the work with accessibility no longer relies on initiatives from individuals (KF 1). With goals, the employees can stop reasoning about which products are more likely to have users with disabilities (KF 2). Goals can also make it clear that the company’s vision for accessible products concerns both designers and developers (KF 3) and narrow the whole concept of accessibility down to what is relevant to the company (KF 13).

Guideline 2 was a solution to the problem that accessibility resources are easily overlooked (KF 12). Guideline 3 made it evident that testing for accessibility is needed in the design phase and the code phase (KF 3). It also solved the problem of employees getting overwhelmed by WCAG (KF 6) by breaking down the requirements and directly adopts the idea of connecting requirements to functionality such as "menu" (KF 13). Guideline 4 and 5 were based on the results of the inspection of the existing Metria design system, which was halting in consistency and overview, together with the finding that the employees have not agreed on whether the design system should be light-weight or vast (KF 11). Guideline 6 ties back to results from the phase Ideate: to be brief, use everyday language and provide a vocabulary for employees to talk about accessibility.

Noteworthy is that the word "levels" was added to guideline 3 after creating the checklists. Initially, it had been assumed in the project that if an application is built based on components that fulfill WCAG, it must also fulfill WCAG. When examining the requirements closer, however, it became evident that not all of them can be fulfilled, or even tested for, at the component level, but only at the page level. Two examples of requirements at the page level are "The navigation is consistent" or "The language of the page is
indicated in the code”. Other requirements such as ”Form fields have labels” and ”Menus have a skip link” can be incorporated into components.

In the description of guideline 3 it was thus recommended to use four different checklists in the testing routines:

- Checklist for design at the component level
- Checklist for code at the component level
- Checklist for design at page level
- Checklist for code at page level

The two checklists at the component level were drafted. The checklists at the page level were left out because it was considered it was only weakly related to the objective.

5.4.3 Drafts of checklists

The drafts the checklists were created using webbriktlinjer.se and WCAG as sources. As recommended in guideline 3, there was a separate one for design and code, and they were broken functionality. It was central in the idea of the checklists that they are digital and interactive, but in the phase Develop they were only drafted as text documents.

The structure of both checklists was the same, only the content differed. Each checklist item had

- Briefly stated requirement title
- Description that was sometimes explaining and sometimes mentioning examples and exceptions to the requirement
- Procedure how to test the requirement
- WCAG success criteria number linked to the requirement for further reading
- List of functionality for which the requirement is relevant

Because there were several pieces of information in every requirement, they were represented using a table instead of a list. For it to still give the impression of a checklist an empty checkbox was placed in front of the title. The format is shown in Fig. 16.

The requirements were loosely sorted based on how frequently they are violated. The WebAIM Million [7], which is a yearly report on the accessibility errors on the web, was used as a source for this.

The design checklist contained 20 requirements and the code checklist contained 13 requirements. The code checklist was, however, incomplete in the sense that not all relevant WCAG requirements were included. Since it was only a draft, that sufficed. Some of the requirements titles were identical in both checklists, but the description and/or test procedure was different. In the design checklist, it was assumed there was yet no implementation to be tested, only a design mock-up. The design checklist had links to access
5.5 Phase 5: Evaluate and Phase 6: Modify — first iteration

The results from the last two phases Evaluate and Modify are presented together, one iteration at a time. In the first iteration, feedback on the guidelines and checklists was collected through two expert reviews and a focus group, and then drafts were made into complete deliverables. First the results from the expert reviews and focus group are presented and beneath them are tables with the corresponding changes made. At the end of this section the state of the deliverables at the end of the first iteration is presented.

5.5.1 Expert opinions on guidelines draft

Both experts, here named "Expert 1" and "Expert 2", thought it was a strength to begin the document by talking about setting an internal goal in the company. Expert 1 said companies who have been successful in working continuously with accessibility have linked it to their brand values, not requirements from the outside. Both experts also found the guideline about setting aside time for the design system a strength.

The experts did not find it clear to who the guidelines were targeted. There was no information about that in the document.
The experts confirmed that accessibility measures are different on the component level and the page level. Expert 1 admitted they had a misconception when they started working with design systems, that general components could solve all accessibility issues. They said it matters not only how the components are made, but how teams use them together.

Both experts gave special attention to the sentence about going beyond engaged individuals and found it good. Expert 1 recommended emphasizing it even more and calling for ”a clear message from management, however boring it sounds, because it creates stability”. They talked about taking “ownership” of accessibility at the company. Expert 2 said the accessibility work will never be successful unless people from management are involved.

Expert 1 advised explicitly mentioning WCAG or a similar standard in the company goals. They said it sets a frame for the work and prevents discussion. ”If we have decided to follow WCAG, then a designer who wants to use color in a certain way can’t win the argument if it is inaccessible”. Expert 1 acknowledged that this sounds contradictory to the advice of setting own company goals, but implied the two can co-exist.

Expert 2 said it is important for companies to define what you mean when you say design system. What support should it give, they asked, and to which projects, both old and new? They said design systems can be very different, and suggested giving some examples from small to large in the guidelines. They also pointed out that a design system team can have a wider responsibility than filling the design system with accessible components. At Expert 2’s workplace, the design system team had become a support hub for all new software development and people come to them with their questions on accessibility. Expert 2 summarized that their work circled around processes and used the Swedish term ”förändringsarbete” (direct translation: ”change work”). Expert 1 said it is important to place resources somewhere that fits the way the employees already do their work.

Expert 1 revealed that the most used material they had made was a four-point definition of done. Its points were: follows the company’s best practices for design and code, passed all automatic tests, has been tested manually for keyboard access, and has been tested using a screen reader. Expert 2 also said they test with keyboards and screen readers in their routines.

Expert 1 also showed two component libraries they have been working with. The first was a website with an overview of all components, with still pictures and links to design files. The other was a library where developers get an overview of the properties of general components and can learn their behavior in different states, hands-on. The expert said their vision is to combine these two into one website with design files, interactive previews, code and documentation, including accessibility guidelines. That website would be the design system.

Expert 1 said it is common that companies make an effort to work with accessibility for six months, and as soon as training and talking about it stops, the work is paused for a year or two, until a new effort is made. They said it is typical that someone writes a document and puts it in the company wiki, but then it is forgotten about. Both experts mentioned the danger of relying on accessibility consultants who later leave the company. Expert 2
talked about being accepting towards doing accessibility partway — for example, asking for 75% percent fulfillment of requirements at launch, and with each iteration increasing that number. They also said primary user flows need to be prioritized when accessibility bugs are reported. They said these bugs need to be treated alongside other bugs in the development processes. They also said companies without experts to ask may need to stop at having done the best they can.

The expert reviews confirmed the overall outline of the guidelines. The feedback and comments resulted in adjustments being made to four of the guidelines. The modifications made to the guidelines after the expert reviews are presented in Table 2.

Table 2: Modifications made to guidelines after first two expert reviews

<table>
<thead>
<tr>
<th>Feedback or comment</th>
<th>Modification - where?</th>
<th>Modification - what?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Setting aside time is important.</td>
<td>Guideline about setting aside time</td>
<td>Moved the guideline higher up to signal its importance more.</td>
</tr>
<tr>
<td>For whom are the guidelines?</td>
<td>Beginning of document</td>
<td>Wrote out the target group in the beginning of the document.</td>
</tr>
<tr>
<td>People from management need to be involved.</td>
<td>Guideline about setting company goals</td>
<td>Explicitly mentioned the role of management in description.</td>
</tr>
<tr>
<td>Mention WCAG or other standard in your goals.</td>
<td>Guideline about setting company goals</td>
<td>Added advice to explicitly mention standard in the guideline about setting company goals.</td>
</tr>
<tr>
<td>My work is about processes.</td>
<td>Guideline about weaving accessibility into other resources</td>
<td>Included processes, not just resources, and mentioned some examples other than design system components.</td>
</tr>
<tr>
<td>Accessibility efforts often cease after some time.</td>
<td>Guideline about weaving accessibility into other resources</td>
<td>Added sentence about not pausing the accessibility work.</td>
</tr>
<tr>
<td>My company currently has two component libraries and wants to combine them to one.</td>
<td>Guideline about beginning small</td>
<td>Exemplified which parts a component page in the design system may contain and instructed the companies to decide which parts they want to include.</td>
</tr>
</tbody>
</table>
5.5.2 Expert opinions on checklist drafts

Both experts had created checklists themselves and talked about their experiences with that. One of them had created separate checklists for different roles such as designer and developer. The other also liked that the checklists they were evaluating separated on roles because “accessibility is teamwork and no one can do everything”.

Both experts found it a strength that there was the possibility to filter for functionality. One of them said the WCAG structure does not reflect how people talk about accessibility. The expert’s checklist instead had headings such as forms, screen reader interaction, dynamic content, automated testing, layout, clickability and motion. They said tests would be needed to find out if these categories match how designers and developers think about their work. The other expert said they wanted to create a small form at the beginning of their checklist that would ask ”What does your application contain?” instead of using headings.

An indication of which accessibility requirements that are the most important was missing, thought one of the experts. They said the requirements that, when ignored, cause severe consequences for the user are the most important. The expert said the companies want to know which issues are the most problematic when they order accessibility reviews from experts. Another problem with the checklists that was pointed out was the incompatibility between comprehensiveness and approachability. Making material that covers all aspects of accessibility something that is referred to on an everyday basis is very hard, the expert put it. The other expert said the threshold for teams needs to be very low for new processes to be used.

One of the experts said the checklist should refer back to the design system, for example, "If you have used the error message pattern from the design system this requirement is already fulfilled”. They said it is good to link to other websites, "do not reinvent the wheel”. They said they link to WCAG in their own checklists because some employees get nervous and want to confirm that they follow the law, however this seemed to be something more common among members of company management than designers and developers.

With the experience they had gained when working with accessibility and components, one of the experts said they themselves do not need a list of requirements to just quickly check a component for accessibility.

One of the experts had created a checklist in a spreadsheet but was not content with it. They said it is hard to update because there are so many copies lying around. They said its functions, such as filters, need to be taught to be understood, so they wanted to move it to the company’s design system and give it a self-explanatory interface instead. They also wanted to incorporate automated testing in it. The same expert also showed an example of an interactive checklist in a design system from a different company that is made public on the web. This checklist was used later in this case study to gain feedback from users.
The expert reviews did not lead to any concrete modifications made to the checklists but to two other changes, see Table 3.

Table 3: Modifications made to checklists after expert reviews

<table>
<thead>
<tr>
<th>Feedback or comment</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Comprehensive material is hard to make easy to approach and use every day.</td>
<td>Changed the intention with the checklist from covering all of WCAG to making it more likely to be used. Decided 20 checklist items should be the maximum.</td>
</tr>
<tr>
<td>Refer to the design system in the checklist. Write out which requirement it has solved.</td>
<td>Did not add such references to the checklist because Metria yet does not have an accessibility-tailored design system. Instead wrote it into the guideline about breaking down testing.</td>
</tr>
</tbody>
</table>

5.5.3 Focus group and complementing interview

The focus group participants thought the purpose of making the guidelines and checklists was "to reach consensus between projects", "to save time", "to avoid reinventing the wheel", "to make it easier to understand accessibility" or "to increase accessibility for the end users".

The format of a checklist was positively received by the focus group. One person found it easy to follow and said the requirements were clear, but that the test column was harder to understand. The participants drew parallels to work tasks they do not perform often, and said they would like a checklist then as support to feel certain they had thought of everything. One person compared it to preparing for a trip: "You always want a packing list but you never have one". One person added that applications are sometimes tested by other employees than designers and developers, and that many of the accessibility requirements also could be tested by people without experience in either coding or designing.

Much of the discussion revolved around the need to make the accessibility check into a task that is enforced in the work process. One person commented that a material that requires a person to sign their name and answer for that something has been done would be effective. They said it is a general problem in the workplace that there are general goals that everyone agrees they should strive for, that are not implemented in practice. An idea to implement the checklist inside the issue tracking system, ITS, was thought out and discussed. One participant mentioned there are plugins for checklists that automatically place checklists inside certain types of issues.

Two new features that were asked for were (1) priority based on importance (and not just frequency) and (2) grouping the features that are automatically tested at the beginning of the checklist. One participant said they would want to run tests in automatic evaluation.
tools first, then use other tools and lastly perform manual checks. One participant said they wanted to be able to sort the priority themselves, based on their own tendency to miss certain requirements or the relevancy to the project.

A need was expressed to note in the checklist that a requirement has been reflected on and will be returned to later. It was discussed whether a third state, in addition to "checked" and "unchecked" is suitable, or if a comment function better serves that purpose. One argument was that "maybe"-states lead to tasks being left on hold.

In order for the checklist to be used in practice by employees, the participants agreed that the possibility to filter and expand collapsed information is crucial. They said one would learn after a couple of times of use what each requirement means and only need to be reminded about some requirements. One participant said "I was overwhelmed by all the small text" and another "If you condense it more, it may be used." Collapsing parts of the information would allow for space for example images. Some participants expressed a clear preference for information being collapsed rather than being directed to new pages or popups. One person said, "I don't want to be tossed away somewhere else". One participant warned that links to external pages may become outdated.

How much information the checklist should contain was inconclusive. They agreed a person who uses it for the first time needs more explanations but were not certain if and how much can be explained in some other resource. One person asked for more text explaining why a requirement helps a user. One participant said the explanations should be there to remind the reader what the requirements mean, but "maybe not teach them from scratch".

The participants thought it was logical to have separate checklists for design and code. The group concluded that it is advantageous to have two checklists to benefit those who have a distinct role and those with mixed roles can simply use both. One developer commented that design is done before code even if it is done by the same person, which would still motivate two checklists. The designers explained that they not only check design in the mock-ups, but also go through the coded version and check for design requirements when it has been implemented. They said they expect changes to occur and feel a responsibility to check design-related requirements, such as contrast, all the way until the implementation is delivered to the users. The designers never examine code, they just look for errors in the interface.

It was inconclusive whether the participants found the separation of checklists for making general components and testing whole pages clear and meaningful. They answered that it makes sense but seemed to not fully understand the question. One participant also said checklists for testing complete applications would possibly have stronger requirements for public applications than expert applications.

The participants had the impression that the checklist should be used to check one’s own work. Only one person, who mainly works with testing, thought it would be used to check other people’s work. An argument for testing one’s own work was that in the process of going through the checklist, you also learn things that you can keep in mind in your future work. One participant expressed a reluctance towards telling colleagues
about specific problems in their work and said it would feel much easier to tell someone "Have you checked the checklist?".

The feedback and comments from the focus group largely shaped the modifications made to the checklists in this iteration, see Table 4. The main modification was adapting a new format inside the ITS.

Table 4: Modifications made to checklist after focus group

<table>
<thead>
<tr>
<th>Feedback or comment</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>There is a plugin for checklists in our ITS that could be useful.</td>
<td>Loaded checklist into ITS-plugin.</td>
</tr>
<tr>
<td>We want to see priority, not just frequency.</td>
<td>Added filter for three levels of priority.</td>
</tr>
<tr>
<td>I would want to run automatic tests first, then do the manual checks.</td>
<td>Placed requirements that get tested by automatic tools on top of the list.</td>
</tr>
<tr>
<td>You learn things as you go through the checklist to keep in mind in your future work.</td>
<td>Kept requirements that can be tested by automatic tools in the checklist to make them more visible to employees.</td>
</tr>
<tr>
<td>How do I note that a requirement has been reflected on and will be returned to later?</td>
<td>Added column for making comments.</td>
</tr>
<tr>
<td>The text is overwhelming and needs to be condensed.</td>
<td>Shorter text and added &quot;read more&quot;-link for employees who want details.</td>
</tr>
<tr>
<td>Collapsible descriptions.</td>
<td></td>
</tr>
<tr>
<td>The description should remind, not teach from scratch.</td>
<td>Shorter text and &quot;read-more&quot;-links for novices.</td>
</tr>
<tr>
<td>Add text explaining why a requirement helps a user.</td>
<td>Mentioned the users in descriptions when that helps understanding what the requirement is about.</td>
</tr>
<tr>
<td>As a designer I feel responsible for testing design-related requirements in the implementation as well, but I never examine the code.</td>
<td>Rewrote design checklist to support testing implemented applications, not just mock-ups.</td>
</tr>
</tbody>
</table>

5.5.4 Modified guidelines after first iteration

At the end of the first iteration of the phases Evaluate and Modify, there were seven guidelines. The descriptions below each guideline were still 20-200 words, except guideline 4 which had about 450 words.
The seven guidelines were:

1. Set your own company accessibility goals.
2. Prioritize and sort out irrelevant requirements.
3. Set aside time for working on the design system.
4. Break down the work and put in the right resource at the right phase.
5. Weave accessibility into resources and processes that are already being used.
6. Begin small and build onto the design system as you go.
7. Use language that is easy to grasp and explaining.

Guideline 2 had also in the draft been included in the document, but as part of the description of guideline 1. When more text was added to the first guideline, according to feedback, it was decided to break the guideline down into two.

A "glasses"-metaphor was introduced in the description of guideline 4 to explain that different accessibility requirements are relevant to consider in different stages of the work. In the draft version, the description (of guideline 3) had jumped straight to checklists, but it was modified to instead focus on the different situations when different resources, or "glasses", are needed:

- When you design general components
- When you code general components
- When a team uses the general components in a context in a design
- When a team uses the general components in a context in code
- When an implemented page is tested

Below this list it was described how and why checklists and documentation could be seen as "glasses".

5.5.5 Modified checklists after first iteration

The checklists were made interactive. The previous format which separated description and test method was abandoned, to allow for collapsing all the details in one field. The checklists were then loaded into two different interactive versions: one in the ITS, as suggested by the focus group, and one in a spreadsheet. For an overview of the features see Fig. [17] The reason why two versions were made is that the ITS plugin did not have support for all the features that had been wished for, so the spreadsheet version was made to be able to test those.

Altogether there were four checklists, an ITS-checklist for design, a spreadsheet checklist for design, an ITS-checklist for code and a spreadsheet checklist for code. Both versions for design and code respectively contained the same checklist items and descriptions.
Seven new requirements were added to the checklist for code, which resulted in it having 20 requirements in total.

In the spreadsheet versions checked items got the color green and the text was crossed through, as seen in Fig. 18. It was also possible to filter based on functionality, but since spreadsheets do not support filtering rows and changing row data in the same sheet, the filtered-out rows were colored dark grey instead of hidden, see Fig. 19. There was a column for adding comments next to each requirement.

In the ITS checklists the descriptions were collapsible, as seen in Fig. 20. It had no support for filters or comments for specific items. Some new features that came with the plugin were rearranging the items, adding separators between them and a progress bar on top. Less relevant features that also came with the plugin were marking an item as optional, editing the items and deleting them. The ITS-plugin was set to add a checklist by default to every new issue created in the ITS. Inside an issue, it was also possible to click a button to quickly create a subtask out of an individual checklist item.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Spreadsheet</th>
<th>TMS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interactive checkboxes</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>&quot;Read more&quot;-links</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>Grouping automatic tests</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>Filter for functionality</td>
<td>✔️</td>
<td></td>
</tr>
<tr>
<td>Sort on priority</td>
<td>✔️</td>
<td></td>
</tr>
<tr>
<td>Collapsed descriptions</td>
<td></td>
<td>✔️</td>
</tr>
<tr>
<td>Rearranging the items</td>
<td>possible but not intuitive</td>
<td>✔️</td>
</tr>
<tr>
<td>Commenting each item</td>
<td>✔️</td>
<td></td>
</tr>
<tr>
<td>Commenting the whole checklist</td>
<td>✔️</td>
<td></td>
</tr>
<tr>
<td>Progress bar</td>
<td></td>
<td>✔️</td>
</tr>
</tbody>
</table>

Figure 17: A table showing which of the two checklist formats Spreadsheet and TMS has which features. The features include a filter for functionality, collapsed descriptions and comments.
Figure 18: The top three items of the spreadsheet checklist for design. When the checkbox is checked in, the whole row turns green and the title of the requirement is crossed through. To the right of the descriptions, there is a link to read more and a column for writing comments. The two columns to the far right are used for filtering and sorting. In the far left column, the names of automated testing software that can be utilized are noted.
Figure 19: Above the checklist inside the spreadsheet there was a title, an instruction and filter controls. Irrelevant functionalities to the component the user is currently wanting to test are checked out, and based on this selection irrelevant rows of the checklist turn dark grey.
Figure 20: The top of the ITS-checklist for design. Checked-in items are crossed out and the progress bar above the checklist automatically updates, showing the proportion of items checked in. The checklist has a number of separators, the first one says "gets caught in automated testing" and the second one says "high priority". Descriptions can be shown or hidden by clicking the blue text. The buttons next to the item that is being hovered over are from left to right: set as optional, edit, create subtask, delete.
5.6 Phase 5: Evaluate and Phase 6: Modify — second iteration

In the second iteration, feedback on the guidelines and checklists was collected through an expert review and they were tested using an evaluating interview and user tests. First the results from the expert review, evaluating interview and user tests are presented and beneath them are tables with the corresponding changes made. At the end of this section, the final state of the deliverables at the end of the project is presented.

5.6.1 Expert opinions on guidelines

The expert consulted in the second iteration, here named ”Expert 3”, said guidelines are good to have a documented common vision and also for onboarding new employees. They said general guidelines do not become outdated as fast but also advised making them company-specific.

Expert 3 liked the tone and order of the guidelines. They said goals and prioritization is the most important thing, which was also the first two guidelines. They thought the glasses metaphor was good for companies who want to get started with working with accessibility. They wanted to add to the glasses metaphor that there are occasions when no glasses are needed. They explained that one has to realize that some accessibility aspects need to be ignored sometimes, otherwise the methodology is not rooted in reality. The important thing, they said, is to understand when one deviates from accessibility requirements. They implied it is unnecessary to make prototypes for quick user tests accessible. They said that if a disabled user cannot interact with the prototype, feedback can still be gained by explaining things to them.

Expert 3 thought something was missing in the guidelines, namely the strategy to locate driven employees and start from there. They exemplified that you can be driven by work pride, a conviction about an including society or knowing someone with a disability. They thought this strategy was central and said ”You need to create momentum. It is hard to get everyone on board at once”. They advised collecting them and making them ambassadors for accessibility. In some companies, it starts with developers and in some, it starts with designers, they said. They used the term ”förändringsarbete” (”change work”) to describe their work with accessibility at companies.

Expert 3 was skeptical about the guideline about setting aside time for the design system. They said it is very problematic to talk about accessibility as something that takes time. ”Everything takes time”, they said. Instead, they wanted the guidelines to talk about advocating for what you gain and how users benefit from it, also people without disabilities. They also stressed it is possible to do an accessibility test in as little as 10 minutes.

The expert review confirmed the new order of the guidelines and the introduction of the ”glasses”-metaphor. The feedback and comments resulted in one new guideline and adjustments being made to two guidelines, see Table 5.
Table 5: Modifications made to guidelines after third expert review

<table>
<thead>
<tr>
<th>Feedback or comment</th>
<th>Modification - where?</th>
<th>Modification - what?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Some accessibility aspects need to be ignored sometimes.</td>
<td>Guideline about breaking down testing</td>
<td>A point was added to the list of glasses, saying it is okay not to wear accessibility glasses sometimes, for example when making a prototype.</td>
</tr>
<tr>
<td>Collect the people who have a drive.</td>
<td>New guideline</td>
<td>Added guideline about getting ambassadors for accessibility.</td>
</tr>
<tr>
<td>Problematic to talk about accessibility as something that takes time.</td>
<td>Guideline about setting company goals</td>
<td>Explicitly mentioned the role of management in description.</td>
</tr>
</tbody>
</table>

5.6.2 Expert opinions on checklists

Expert 3 found the simplicity in the language and the way the checklists speak to one target group at a time a strength. They also said it was positive to refer to automatic evaluation tools in the checklist.

Expert 3 found it a major problem to link to WCAG in the checklist. They said reading WCAG "does not make anyone engaged to start working with accessibility". They strongly recommended linking to blog posts and other external resources, even at the risk that those links become outdated. They said there are many knowledgeable and inspiring people who are posting online and that that should be utilized. They also found it problematic to expect companies to write their own articles about every single requirement. The expert tied this back to the guidelines and said that the goals and decisions of what is important need to be company-specific but that the knowledge is general.

The expert doubted that the format of the checklist would be appealing to designers. They suggested making it more visually appealing, or even better letting employees own the checklist themselves. They said it is better to let employees create their own resources, and as an accessibility expert fill it with content for them.

The expert was shown the checklist in the spreadsheet version, because the checklist for the issue tracking system, ITS, was not ready yet at the time. When told about the idea to integrate the checklist into the ITS, the expert reacted that this would be powerful. They said that would make accessibility a part of the process and stop excuses such as "We did not think about contrasts". They added that if the checklist requires too much effort, however, it would be negatively received by the employees.
The expert review confirmed the central concepts of the checklists, being separated on design and code and the text being simple. It also lead to two modifications related to checklists, see Table 6.

Table 6: Modifications made to checklists after expert reviews

<table>
<thead>
<tr>
<th>Feedback or comment</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Problematic to link to WCAG.</td>
<td>Removed all links to WCAG. Used blogs or webbriktlinjer.se, when they had good examples, in the &quot;read more&quot;-links.</td>
</tr>
<tr>
<td>Make the checklist visually appealing or let employees change it themselves.</td>
<td>Attempted to make the checklists more visually appealing, but the environments in which they were created had very limited support for changing the styling. Instead, recommended in the guideline about weaving accessibility into existing processes, to let employees control the format themselves.</td>
</tr>
</tbody>
</table>

5.6.3 User tests

Four users tested two checklists; one placed inside their issue tracking system, ITS, and one in a spreadsheet. Both checklists contained the same list items but had different appearances and functions, for more details see Section 5.5.5. The designers were given checklists with items relevant to design and the developers received checklists with items regarding code. The ITS checklist was configured to automatically appear on all the issues in the board of a fake project.

All four users preferred the ITS checklist over the spreadsheet checklist. All of them talked about the placement inside the ITS as more important than the appearance or functions. One person said, "I don’t believe in documents because we’ve had them and they haven’t been used.” One person experienced the ITS checklist more as a service compared to the spreadsheet which demands more of the user. One person said the amount of visible text in the spreadsheet made it harder to concentrate. Two users said the ITS checklist was easier to navigate or that it required less scrolling. Two users appreciated the progress bar in the ITS checklist. Two users explicitly mentioned that they liked that the descriptions are collapsed in the ITS checklist.

The users thought the descriptions were well-written and easy to understand, with the exception of one item in the design checklist and two items in the code checklist. Two users liked the links and clicked them during the test. One person said they did not click the links because they thought the descriptions were enough to understand what to test. One person said, "It does not feel like a flood of things to check”. Two users mentioned they believed product owners could also use the checklists in this format.

One user remarked that one might check the boxes to signal "I have tested this” regardless of whether the requirement was fulfilled or not. Two users tried converting an item into
a subtask and said they appreciated that feature. Two users expressed they would want to be able to comment on each item in the ITS checklist but did not seem to have noticed the ability to do this in the spreadsheet.

One user liked that they could add and edit checklist items themselves, while a different user thought it would be better if only an admin could do that, and referred to them as "the person who owns the checklist". One user said they really liked the ability to rearrange the items in the ITS checklist.

None of the users used an automatic evaluation tool during the test, but one person commented that it was good to put the automatically testable requirements on top. One person shared an anecdote that they had participated in a challenge to build the most inaccessible website that scored perfectly in Lighthouse, and that they since then do not trust Lighthouse.

The users were convinced the ITS checklist would be used in practice. The reasons they expressed for this were that it is in front of them every day, that it is clearly assigned to a person, and that the colleagues can see the progress. One user said, "I think it would be very useful and it would raise the quality of our work". Two others said "For my part, I am very positive it will be used" and "My team would definitely use it". The fourth user said the daily standup meetings would become a forum for talking about accessibility if this checklist was implemented. One of the users said "We've had documents for testing, but we never use them, we only say we should think about it" and added that if it is in the ITS, that would be different.

Three users found accessibility errors in applications they are working on during the test and expressed excitement about learning new things while going through the checklist. One of the users did an effort to improve keyboard access in a project they are working on the next day.

Although the participants were instructed to go through the checklist on a component, three still chose to test a whole page. The fourth started on a component but because it took little time to go through it, they went on and checked the surrounding page. This result was seen as an indication that checklists are not suitable for testing components, and therefore the guidelines were changed to only recommend checklists for testing whole pages.

At the end of the user test, the participants were shown an interactive checklist from a different company. The interesting aspect of this checklist was that it was placed inside the other company's design system. It had a filter for role and functionality, collapsed descriptions and 96 items in total. The participants were asked what they thought about this solution compared to the two solutions they had just tested. All four participants said they would prefer a checklist to be inside the ITS rather than the design system. Two users said they thought an extensive list of requirements in a design system would be useful for other purposes than testing. One of them said it could be used to learn about accessibility, the other said it would be useful when setting requirements for projects.
The user tests results contain many suggestions for improving the checklists. It was however decided not to proceed with developing them in this project, as explained in Section 5.6.5. The suggestions of improvement could be used to develop the ITS checklists further in a different project. The user tests also produced findings that resulted in modifications to the guidelines and those are presented in Table 7.

Table 7: Modifications made to guidelines after user tests

<table>
<thead>
<tr>
<th>Result</th>
<th>Modification - where?</th>
<th>Modification - what?</th>
</tr>
</thead>
<tbody>
<tr>
<td>The users tested whole pages, although the instruction was to test components.</td>
<td>Guideline about breaking down testing</td>
<td>Recommendation to test components using checklists was removed.</td>
</tr>
<tr>
<td>The placement of the checklist inside the ITS was received very positively.</td>
<td>Guideline about weaving accessibility into existing processes</td>
<td>Wrote example that if employees get to control support materials themselves, they may want to put them in the ITS.</td>
</tr>
<tr>
<td>The users were not overwhelmed by the ITS-checklist and inspired to work more with accessibility. They were convinced the ITS-checklist would be used if introduced in their work process.</td>
<td>Guideline about breaking down the work</td>
<td>Recommended that checklists should be concrete and maximum 20 points. Explained that material that is easy to approach is better than material that covers everything.</td>
</tr>
<tr>
<td>The users did not request a filter in the ITS-checklist.</td>
<td>Guideline about breaking down the work</td>
<td>Removed recommendation to break down checklists on functionality or content of applications.</td>
</tr>
</tbody>
</table>

5.6.4 Evaluating interview

The interviewee, who was a person responsible for the design system at Metria, first commented on the order of the guidelines. They thought the last two guidelines about language and starting small had a higher hierarchy than the previous ones and wanted to move them higher up in the list.

In the first guideline about setting company goals, they read the sentence about finding out what needs the target group has, and wanted the text to proceed with explaining how this should be done. They conveyed doing as the customer says is not always doing what the customer needs. The interviewee said they see it this way: "Find out which are the needs but own the decision based on your circumstances [as a company]". They had
a similar opinion about the guideline about weaving accessibility into other resources. They agreed it was important but wondered what it means in practice to “weave it into other resources”.

The interviewee thought the guideline about setting aside time was great but also said it was stating the obvious. The description of that same guideline made them reflect on Metria’s own weakness in working with small parts and missing the non-technical overview.

In the guideline about breaking down the work, the interviewee interpreted the glasses as steps to take one after the other. They thought the last statement about removing the glasses for user tests was good. They commented however, that it is more than “okay” (which was the wording used in the text) not to refine a prototype, because perfecting it would actually be wrong. They said that that is the whole point of prototypes.

In the guideline about breaking down the work, the interviewee found the section about documentation for components difficult. They read the phrase “Components are not self-explanatory” and reflected on the design system that is under construction at Metria. They

<table>
<thead>
<tr>
<th>Feedback or comment</th>
<th>Modification - where?</th>
<th>Modification - what?</th>
</tr>
</thead>
<tbody>
<tr>
<td>The guidelines about language and starting small has a higher hierarchy than the rest and should not be at the end.</td>
<td>The order of the guidelines</td>
<td>Moved those two guidelines higher up, but not to the very top because the experts had said other guidelines were the most important.</td>
</tr>
<tr>
<td>It is obvious that time needs to be set aside for the design system (together with Expert 3’s disliking for saying accessibility takes time).</td>
<td>Guideline about setting aside time</td>
<td>Made it focus on defining the design system as its own project in order to achieve the overview, instead of stressing the word “time”.</td>
</tr>
<tr>
<td>The part about when not to wear the glasses is good. The whole point of prototypes is that they are not refined.</td>
<td>New guideline</td>
<td>Created guideline about postponing accessibility when it is motivated.</td>
</tr>
<tr>
<td>I believe automatic evaluation tools will improve at a high pace (together with comments from experts).</td>
<td>New guideline</td>
<td>The text about automatic evaluation tools in the description of the guideline about breaking down testing was made into its own guideline to emphasize the tools’ limitations more.</td>
</tr>
</tbody>
</table>

Table 8: Modifications made to guidelines after evaluating interview
concluded that it does not contain anything like the documentation that is recommended. They expressed doubt and said documentation would make it hard to maintain and split things up, and talked about keeping the design system together. The interviewee also made a comment about automatic evaluation tools. "Many people dream about them. I believe they will improve at a high pace.”

Summarizing, the interviewee said the guidelines were easy to read. They thought that the document was clear and they liked the format. They pointed out a section they thought was hard to understand and a sentence they thought was ambiguous.

The evaluating interview resulted in the order of the guidelines being changed, one guideline being adjusted and two new guidelines, see Table 8.

5.6.5 Modified deliverables after second iteration

The guidelines were changed and the final result had 10 guidelines. The whole document in Swedish, including descriptions, are in Appendix A. The guidelines were:

1. Set your own company accessibility goals to base the design system on.
2. Prioritize and sort out irrelevant requirements.
3. Make the design system into its own project and define its responsibility.
4. Use language that is easy to grasp and explaining.
5. Begin small and build onto the design system as you go.
6. Get ambassadors for accessibility who give support.
7. Break down the support on the different stages of the work.
8. Weave the support into resources and processes that are already being used.
9. Utilize automated testing for what it is capable of.
10. Postpone accessibility measures when motivated.

The checklists were not modified in this iteration. The user tests showed the connection to the objective is weak for two reasons. Firstly, they are more suitable when used to test whole pages rather than design system components. Secondly, it is better to place them in the ITS rather than in the design system. This is further discussed in Section 6.7. The user test results contain many suggestions for improving the checklists’ features and they could be used to develop the ITS checklists further in a different project.

The project, therefore, does not deliver any tool, only methods, as part of the accessibility methodology. Instead the methodology consists of making own tools at the company. In the guideline 7, Break down the support on the different stages of the work, it is recommended to make checklists for testing whole pages that refer back to the design system, and a template for a component page.
6 Discussion

The project resulted in an accessibility methodology for design systems consisting of 10 guidelines. Throughout the project, several topics have appeared in more than one of the six project phases Discover, Define, Ideate, Develop, Evaluate and Modify. These findings are discussed at the beginning of this section. Then the proposed accessibility methodology for design systems is presented as a whole, and the research question is answered. At the end of this section, the strengths and weaknesses of the method used in the thesis are discussed.

6.1 It is about processes

In the beginning of the project, my intention was to investigate what a design system should contain to contribute to accessibility. I wanted to figure out with what it should be filled, what properties it should have and how to best create this content. What I have realized since then, is that a resource is useless if it does not have a place in a process. In the interviews, expert reviews and user tests, see Section 5.1, 5.5.1 and 5.6.3, a similar story was repeated over and over. An employee puts effort into writing a document about accessibility, and the colleagues react very positively at first, but then the document is quickly forgotten about. Accessibility needs to be present in more than a document in a corner somewhere. Employees need to be reminded about it regularly. Getting development teams to work with accessibility is thus primarily about processes and only secondarily about resources.

I am not the only one who has shifted focus from working with making accessible design system components to designing processes. One of the experts had done a similar journey as me, see Section 5.5.1. The other two experts described their work using the Swedish term förändringsarbete which directly translates to "change work". What they do in their role as accessibility experts is that they strive to change how the business works and how the employees think and act.

I think accessibility measures and design systems are both, in isolation of each other, about förändringsarbete. Introducing a design system creates friction, just like starting to work with accessibility, because it requires employees to change their routines. Similar to accessibility, a design system is beneficial in the long run, but takes time in the beginning.

Shifting the perspective from resources to processes lead to this project not delivering any tool, and instead describing how to produce tools that are tailored for the company and including them in the workflow. The guidelines 7. Break down the support on the different stages of the work and 8. Weave the support into resources and processes that are already being used are the result of this shift.
6.2 The importance of management

At the beginning of the project, when the focus was on the content and the properties of the design system, I decided organizational themes were outside of the project scope. My target group was designers and front-end developers, and what people from the company management thought was irrelevant, I reasoned. It did, however, become impossible to ignore this aspect, because it reappeared despite my efforts to exclude it. It first came up in the interviews of the phase Discover. The first key finding, see Section 5.1, alludes to management when saying the work with accessibility that is performed is at the initiative of individual employees. Management was mentioned explicitly by interviewees but the quotes were not included in the affinity mapping.

Although management was not addressed in the first draft of the guidelines, the experts still commented on it and put much stress on that. One of them even said it is impossible to succeed without involving people from management, see Section 5.5.1. The focus group brought up that accessibility requirements and testing need to be enforced, see Section 5.5.3, which also relates to management. All of this motivated putting the guideline 1: Set your own company accessibility goals to base the design system on first.

If Metria had company goals clearly stating their bare minimum of accessibility, it would spare the employees having to reason about how likely it is that a product has users with disabilities and also pondering about whether their map applications are exempted by the law or not. In the interviews of the phase Discover they said accessibility is treated as more important for applications for the public than expert users, but also that they thought it should not matter, see Section 5.1. Official goals could hence raise work ethics, because employees feel they have the approval to work on the quality of the applications and make them including.

6.3 Approach to automatic evaluation tools

The topic of automation was raised in the interviews, the ideation workshops, the expert reviews, the focus groups, the user tests and the evaluating interview, see Sections 5.1, 5.3, 5.5.1, 5.6.2, 5.5.3, 5.6.3 and 5.1. It can therefore be concluded that any methodology for digital accessibility needs to include automatic evaluation tools. The question is how to approach this, considering people’s attitudes towards them, their limitations and the opportunities they create.

This project did not evaluate the effectiveness of the existing automatic evaluation tools on the market, but other studies cited in Section 3.4 did. They revealed that the tools cover 60% of the requirements at the highest, possibly as little as 12%, and find no more than 40% of the total amount of accessibility issues. The result also varies much between tools. I gained a deeper understanding of this in the phase Develop while making the checklists. When going through the requirements in WCAG, I realized many of them need to be interpreted by humans. This is because they are complex, relate to how we perceive things with our senses, or encompass large flows or a longer experience, not just single elements that can be technically analyzed.
The limitations of the tools also surfaced in the user tests when a participant shared an anecdote about intentionally building inaccessible websites that score perfectly, see Section 5.6.3. This experience had caused the participant to distrust. Interestingly, their distrust was directed towards that particular tool, and not automated accessibility testing in general. Since the tools’ fundamental principle is to evaluate against a standard, see Section 3.4, they can miss accessibility issues that would be obvious in a user test, and that is a general problem. The rest of the Metria-employees expressed a strong belief in automatic evaluation tools. This was evident in the interviews of the phases Discover and Evaluate, see Section 5.1 and 5.6.4. They were not aware of the tools’ limitations, or at least not the extent of them.

The three experts were all positive about using automatic evaluation tools as a part of the accessibility work and include them in their own processes at their workplaces. They were, however, very aware of the limitations. Two of the experts complement automated testing by testing with keyboards and screen readers, see Section 5.5.1. The third expert stressed it is better to test what you can with 10 minutes than to do nothing, see Section 5.6.1. Having that mindset, automatic evaluation tools are naturally a substantial asset. Also worth noting is that in the studies of automatic evaluation tools cited in Section 3.4, the manual checks found far from all the errors. The researchers approximated the total number of errors by combining the results of manual checks and several tools.

Another aspect to consider is the idea of learning while testing. If a resource such as a checklist, condenses all the issues automatic testing can discover into one sentence such as "passes automatic testing", those requirements are not visible to the employee. The results from the focus group and the user tests, see Section 5.5.3 and 5.6.3, show there is more to win through a checklist than just quality-assuring applications: the employees also increase their competence. This motivates writing out requirements as their own checklist items, for example, "all form fields have labels" together with an indication that they can be tested automatically, instead of condensing them together or removing them entirely from the checklist.

My conclusion is that automatic evaluation tools should definitely be used, because it makes the accessibility work systematic, but is important that they are used as part of a larger process. Each company needs to take into account what the tools they use are capable of. Also, employees need to understand what the tests are actually checking, in terms of what it means to a user, and not just a list of anonymous green check marks. Compare a checkmark saying "em or percentages are used to set the size of objects” to a requirement explaining that "container sizes set in pixels cause problems for people who enlarge the text". The error would be fixed through automated testing regardless of whether or not it appears in the checklist, but the results of this project suggest it is more likely that the employee will learn and work proactively with the requirement in the future if it is visible to them. It is as important to communicate what the tools test as what they do not test. This resulted in guideline 9. Utilize automated testing for what it is capable of.
6.4 Fitting the day-to-day work

The results of this project confirm that WCAG does not fit the day-to-day work of development teams, as others have previously pointed out, see Section 3.3.4. Already in the phase Discover, interviewees said WCAG is overwhelming, see Section 5.1. During the phase Develop, I struggled with extracting the concrete actions the guidelines ask for, and with summarizing the most important messages. In the phase Evaluate one of the experts even stated that nobody gets engaged to work with accessibility from reading WCAG, see Section 5.6.2. Another expert implied it is impossible to create material that covers everything within accessibility, that also gets referred to in the day-to-day work, see Section 5.5.2. Also in the phase Evaluate, the user tests, see Section 5.6.3, proved how condensed material, such as a 20-item long checklist with collapsed descriptions, can get employees excited to learn in a way WCAG has not.

The creators of WCAG write it is a technical standard made for professionals within web development, see Section 3.3. I would argue they do not succeed at what they have set out for. WCAG does not speak to designers and developers. I believe, however, that WCAG serves a different purpose, namely setting a definition of accessibility for legislation in the whole world. This project never set out to replace WCAG. The experts behind WCAG are doing an extensive job at attempting to cover all the corners of accessibility needs. Instead, I wanted to develop a methodology that works in practice and actually gets applications closer to being accessible to everyone.

A methodology that works in practice needs to acknowledge several things WCAG does not. My methodology firstly acknowledges accessibility does not mean the same to all companies. Secondly, it uses a language that is easy to understand. Thirdly, it states different roles, such as designers and developers, can keep track of different accessibility requirements. Lastly, it acknowledges that accessibility can be postponed when motivated. It does not have to be 100% at all times, that is not realistic. It is more important that accessibility measures are done at all, than that they cater to all accessibility needs. Too comprehensive material risks being ignored.

This theme of fitting to the day-to-day work can be traced in all the guidelines, but in particular in 2. Prioritize and sort out irrelevant requirements, 4. Use language that is easy to grasp and explaining, 7. Break down the support on the different stages of the work and 10. Postpone accessibility measures when motivated.

6.5 Teaching and reminding

It is typical that an effort is made to increase accessibility, and after six months, work goes back to normal, according to one of the experts, see Section 5.5.1. Regular training and talking about it is needed to keep the effort up over time. In the interviews of the phase Discover, the Metria employees also said they think accessibility is important, but that they need to be reminded about it in their day-to-day work for it to be done, see 5.1. One expert said accessibility issues should be introduced to the same workflow as other bugs and solved alongside them, not in a separate workflow, see 5.5.1.
Another topic that was discussed is how extensively accessibility requirements should be explained where. It is important not to overwhelm the employees, neither when first learning about accessibility nor when performing a task in the production of an application. The focus group discussed that a checklist is primary for being reminded, like a packing list, and does not need to teach the user from scratch, see Section 5.5.3. It can assume the user already has some level of knowledge. In the user tests of the checklist, see Section 5.6.3 some of the users thought the short collapsible descriptions were enough to understand the requirements, while others needed to click the links to external websites and read more.

When talking to the experts, crafting the checklists and testing them with users, I made my own reflection about telling the user "for whom" an accessibility requirement exists. I decided employees do not necessarily need to be motivated about who they are designing or developing for, but that a mention of the user helps understand what the requirement means. For example "Use language tags on foreign words. Without them screen readers pronounce them incorrectly which can be very confusing to a user" or "Support autocomplete. If you were to fill in the form using your chin, you would be thankful for it." The same is valid for the example "Container sizes set in pixels cause problems for people who enlarge the text" mentioned earlier. These types of short explanations are valuable both when teaching and reminding.

Summarizing these findings, routines are needed for teaching as well as reminding. Separate material is necessary for this. A checklist for regular testing should be condensed, but it does not suit for introducing employees to accessibility.

### 6.6 Ownership

The concept of owning a decision or resource has been brought up repeatedly throughout the project by different people regarding different things. An interviewee advised to "own the decision" in the context of asking customers about their needs, see Section 5.6.4. As a representative from a software company, one should ask, but then not just do as they say. Instead, one should take this information and make a thought-out decision based on all circumstances as a whole: the customer's and one's own.

One of the experts talked about taking ownership of accessibility in the company, see Section 5.5.1. This idea encompasses two important aspects that were emphasized in the guidelines. The first is to not only abide by laws from the outside but to instead make accessibility an internal matter that is part of the identity of the company. The second is not letting individuals "own" the matter of accessibility in the company. Relying on individuals is usually not enough, and even if it works, one risks being too dependent on them which is devastating if they leave the company. Two of the experts also talked about the design system team as owners of the accessibility topic at the company. At their workplace, members of the design system team are knowledgeable and provide support surrounding accessibility in general, not only regarding the design system, see Section 5.5.1.
Taking ownership of accessibility is proactive, instead of waiting for customers to expect it or laws to be enforced. As described in Section 3.2, legislation demanding digital accessibility is spreading from the public sector to e-commerce and other areas. It is unclear how Metria is affected by the next coming laws. Metria has an e-commerce website, mostly targeted towards companies, not private individuals. Some of their customers are in the public sector on one hand, but on the other hand, if they use Metria’s services to embed maps on their websites, many maps are exempted by the law. Instead of pondering the law, I recommend they use the energy to take ownership because building accessibly is good quality, and that they decide for themselves what their level of accessibility should be.

Previous research has shown it is important whether employees feel they own a resource or not, see Section 3.6. Design systems are co-owned by designers and developers, which is a strength compared to, for instance personas, that are owned only by designers. A similar phenomenon is applicable to checklists, suggested one of the experts, see Section 5.6.2. The expert meant the checklists need to be visually appealing to be used by designers, and that this is best done by the designers themselves. By letting employees “own” material that supports them in the accessibility work, such as a checklist, the likelihood that it will be used increases. This can be done by providing them with the content, for example checklist items, and letting them decide the shape of the material themselves. When user testing the checklist, which was automatically generated, but could be edited by the user, the users reacted differently. Two users liked that they could rework it, but one user was confused and thought only admins who “own” the checklist should be able to edit it, see Section 5.6.3.

I believe achieving a sense of ownership among employees is key to getting accessibility measures to work long-term. They should feel they own the process, not as individuals, but in their professional roles as designers and developers. The ideas expressed by the interviewee and experts about owning decisions and resources were therefore written into the guidelines 1: Set your own company accessibility goals to base the design system on and 8. Weave the support into resources and processes that are already being used.

6.7 Checklists and design systems

The idea of a checklist first appeared in the project in the phase Ideate, see Section 5.3. It also gained the highest ranking in the survey of the phase Develop, see Section 5.4.1. During the expert reviews of the phase Evaluate, it became evident that other companies have had the same idea, see Section 5.5.2. Different versions of checklists were made and evaluated during the project, but the project resulted in no checklist being delivered. There are several reasons for this.

The original plan from the phase Develop was to have four checklists, see Section 5.4.3. A decision was made to proceed with developing the two checklists for components and leave the checklists for whole pages out of the project. The motivation for this then was that making components is connected to design systems, unlike testing whole pages, and hence in line with the objective of the thesis.
The concept of having checklists on both component level and page level seemed to confuse the target group. In the focus group, when asked if the separation was meaningful, the participants seemed to not understand the question, see Section 5.5.3. In the user tests, the participants tested on whole pages although the checklist was adapted for testing components, see Section 5.6.3. This result suggested there should only be one checklist for each target group, after all.

The project’s iterations steered the checklists in a certain direction: to adapt them to be easy to approach, ultimately avoid being ignored, instead of covering all aspects of accessibility. A checklist that covers all of WCAG in theory, but is not used in practice, does not do any good either for designers, developers or end users of applications. The project shows checklists that are brief, visually pleasing and placed in the issue tracking system, ITS, are easy to approach. They received very positive reactions in the user tests, see Section 5.6.3 suggesting that in this format, they are a powerful tool that can make a difference for accessibility.

That direction of the checklists, I concluded, is not suitable for design system teams. They, according to the guidelines I made, have a wider responsibility than just making general components and testing them for accessibility. They also write the documentation for components with advice on how to apply the components in an accessible way, and therefore they need a more comprehensive tool and understanding than a condensed checklist. This lead the project to dismiss the idea of checklists for testing components.

Meanwhile, the assumption that checklists for whole pages was not in line with the objective, was revised. The reasoning that testing whole pages was not connected to design systems proved to be wrong during the phase Evaluate, see Section 5.5.2. One of the experts showed two ways checklists for testing complete applications can be linked to the design system. Firstly they demonstrated that such a checklist can refer to the design system, for example saying "If you use the design system buttons, this requirement is already fulfilled". Secondly, they showed an example of a company’s design system that contains an interactive checklist about accessibility.

The idea to reference the design system was not investigated in practice in the project, because there were no drafts for checklists at the page level and also no accessibility-tailored Metria design system to refer to. The idea to place the checklist in the design system was evaluated in the user tests, but the participants unanimously preferred the checklist to be placed in the ITS, see Section 5.6.3.

In the methodology that the project resulted in, checklists do have a role. Making checklists that refer to which requirements the design system has already solved is mentioned in the guideline 7. Break down the support on the different stages of the work. Those checklists need to be made by the company themselves after they have set their accessibility goals and built the first set of accessible components and patterns. Therefore they could not be presented as a deliverable result of this thesis.
6.8 Proposed methodology

The aim of the thesis was to develop an accessibility methodology for creating and maintaining a design system. The methodology developed in the project is summarized by the 10 guidelines listed in Section 5.6.5.

6.8.1 Core concepts

Central to the methodology is adapting the work with accessibility to the company at hand. This includes setting own accessibility goals, prioritizing requirements and defining the responsibilities and shape of the design system. Based on those decisions, tools adapted for designers and developers can be created that help them make and use content from the design system in an accessible way. Those tools are best designed by the people who are going to use them, even though experts can contribute to the content with their knowledge of the actual requirements.

6.8.2 Different accessibility glasses

In the methodology, tools that support designers and developers with accessibility are described as "accessibility-glasses". There are different glasses to wear at different stages of a project. Designers and developers also need different sets of glasses. One and the same WCAG success criteria can contain design aspects and technical aspects. This is impractical because designers may not understand technical aspects and even if they do, they do not influence the coded implementation. Developers respectively do not influence how problems related to the shape and interactions in the applications are solved. Company-specific tools should distinguish between those two fields.

The methodology identifies three separate stages and proposes what can serve as glasses in each stage, see Fig. 21. The glasses proposed are:

- For creating components, patterns and documentation: resources outside the company for education and a catalog of requirements together with a component page template for the company-specific aspects.
- For using components and patterns: documentation that is placed alongside the component or pattern in the design system. The documentation needs to be written by the creator of the component or pattern.
- For testing an implemented page: checklists that refer back to which accessibility requirements the design system has already solved.

That every component should be associated with documentation is, according to the project results, far from evident to companies. Neither the design system that was shown by one of the experts, see Section 5.5.1 nor Metria’s existing design system, see Section 5.1 and 5.6.4 has any documentation describing when and how to use a component. I concluded that documentation is the right context to fulfill many of the success criteria in WCAG. In the phase Develop, I realized that many requirements cannot be built into general components, because they relate to the specifics of how a component is made in
Figure 21: A table of the recommended tools for different stages. Depending on the situation, either education, a catalog of requirements, a template page, documentation or a checklist can act as "accessibility glasses".

context. The best place to remind employees about these requirements, in my belief, is in association with the component. Designers and developers want to know what they need to consider at the moment, not everything accessibility is about. Therefore the guidelines emphasize that you can only do so much about accessibility by handing out general components, and that writing documentation gets you much further on the way. Components are not self-explanatory. As an expert stressed, much can go wrong when perfect components are applied, see Section 5.5.1.

The methodology also acknowledges that accessibility can be postponed when there is reason for that, as already discussed in Section 6.4. Keeping applications 100% accessible at all times is not realistic. Striving to get a part of the way there is motivating. The guidelines therefore recommend making checklists short to make them easy to approach for an employee instead of trying to cover everything, as discussed in Section 6.4. The guidelines exemplify two situations when the accessibility glasses do not need to be worn: when doing user tests with prototypes, and for functions that are used by few users and have alternative ways, such as contacting customer support. Disabled users can provide feedback on prototypes if the test facilitator explains things to them that they cannot access themselves. Spending time on making prototypes accessible is unwise, since they may not even make it to release.
6.8.3 Accessibility ambassadors

The methodology is based on making use of engaged individuals in the company. By letting them educate themselves, and then forward their competence through the design system and surrounding support, they can lift the accessibility level of the whole company. They work closely with the creation of the design system; the making of the components and the writing of the documentation. They also provide the checklist items for the checklists for testing complete applications.

6.9 Answering the research question

The research question of the thesis is: How can a design system support designers and developers in producing accessible applications? The project has given various perspectives to answer it.

6.9.1 Make it an internal matter

The project discovered that the introduction of a design system is an opportunity for a company to take ownership of the topic of accessibility. It is done by setting internal goals to be accessible instead of waiting for demands to come from outside the company. This has been further discussed in Section 6.2 and Section 6.6. Companies can relate accessibility to the brand identity and also increase work ethics among employees.

Taking ownership of accessibility helps limit WCAG down to what is relevant for the company depending on the content of their applications. Some have video, others have maps, others present large amounts of information in text and tables, others perform transactions such as purchases and making agreements. All of those types of content have specific WCAG success criteria tied to them. If a company’s applications do not contain any one of those things, they can ignore those requirements.

6.9.2 Fit it into a process

The effort and time required to build accessible applications can be decreased by making the work with accessibility systematic. The design system is a context for doing so. The project showed designers and developers need to be reminded about accessibility in their day-to-day work, and the accessibility measures need to fit into existing processes, as discussed in Sections 6.5 and 6.4. The goal of a design system is to get employees to use it regularly, and if that is achieved, and accessibility is a natural part of the design system, it does become a part of the process.

6.9.3 Components only get you so far

The results of the project show that the design system alone cannot ensure that applications built based on it become completely accessible. Several WCAG-requirements cannot be incorporated into individual components. Those requirements need to be tested either on an applied, in-context version of a component, or on a holistic level, on complete applications, pages and flows. There are however methods to support for accessibility
at a smaller or greater length. Building accessibility aspects into the components is the minimum level. The next level is reached by writing documentation on how to apply the components in context. The third and last level is to provide support material for testing at the application-level, and refer back to which requirements the design system has solved in these materials. An example of such material is an interactive checklist placed in the development teams’ issue tracking system. Such a checklist was tested in the project and received positive results, see Section 5.6.3.

6.9.4 Focus on fewer requirements at a time

Unlike WCAG, a design system and support associated with it is company-specific. It directs the employees’ attention to what accessibility aspects are relevant to what they are building at the moment, and thereby avoids them getting overwhelmed.

6.10 Method strengths and weaknesses

The project used a UX design approach and was based on a case study. It can be debated whether this method suits the objective. It could have been fulfilled with a theoretical approach, such as a literature review, or by surveying a large number of workplaces. The chosen method’s strengths are that it was centered on the users, in this case designers and developers, that it was problem-oriented rather than solution-oriented, and that it was iterative. This allowed the project to change direction throughout the six phases and also have a holistic view rather than being strictly limited to the contents of the design system.

An example of when the project took a turn is when the focus group suggested incorporating a checklist into the issue tracking system. Thereafter, the focus was shifted more to the format of the tool, rather than the content. Prior to that, in the phase Develop, it was attempted to make the checklists comprehensive, but after the evaluation by the experts and focus group, they were instead made concise. The checklist in the issue tracking system generated unexpectedly good reactions in the user tests. It showed how important it is to place resources where employees see them every day, and also not overwhelm them with accessibility requirements. I believe this finding was possible due to the method being iterative and user-centered.

The project investigated certain parts of the process of making and using an accessibility-supportive design system and others not. The resulting methodology describes what to consider regarding accessibility before making the design system and how to benefit the most from it when it is in place, but it leaves many questions unanswered regarding how to make the content in the design system. A possible explanation for this is that the Metria-employees have little hands-on experience with making a design system. On the other hand, this may also have been a strength for the project, because it could investigate the perspective of employees that are not directly involved with making the design system, but are consumers of it.
As demonstrated in Fig. 5 the method was complex with many techniques and participants, which may introduce many sources of error. One source of error is in the survey. The different styling of the samples may have caused the vote to be impacted by the appearance and not just the content of the alternatives. Another source of error is that convenience sampling was mostly used: participants were chosen based on who was available and who had prior knowledge of design systems and/or accessibility. More care could have been taken to make sure participants represent the whole target group. The fact that a large number of different techniques were used remedies such sources of errors in individual techniques.

Only four users participated in the user tests, which has been deemed insufficient by researchers in design [45, p. 275]. However, those assertions assume the goal is to identify a certain proportion of the usability flaws of an interface, for instance 80%. In this project, the user tests functioned only as a compliment to other methods, and the goal was to test the applicability, attitude and general experience of using an interactive checklist, not identifying detailed flaws. In total 13 members of the target group contributed to the creation of the methodology.

The experts provided an immense amount of insights to the project, which was a strength of the method. Their participation also mitigated the risk that the accessibility methodology would become tailored for Metria specifically. That the experts are unrelated to each other and were consulted separately, also raised the reliability of the thesis. Their high-level knowledge and experiences could however have been better utilized if they had been introduced earlier in the project. During the expert reviews, the experts made many general remarks about accessibility processes rather than evaluating the guidelines and checklist at hand. These remarks were very useful and would have been more encouraged and applicable in the phase Discover and Ideate rather Evaluate.

It is uncertain whether the methodology succeeds at its purpose: to make applications accessible for end users. To measure the outcome of the methodology, it would need to be applied at a company for a longer period of time, for instance a year. The level of accessibility of the applications would need to be quantified at the start and end of the period to be able to measure the expected improvement. This project only evaluated the potential for the methodology to be complied with.

The thesis' validity is hence motivated by the method being founded on the needs of the target group and it succeeding in answering the research question. The thesis reliability is motivated by the large number of different techniques used, and the number of participants contributing, both experts and members of the target group. I am confident that if the project was to be repeated at a different company with the same method, it would encounter the same topics that have been discussed in Sections 6.1–6.7. The resulting methodology would be slightly different, but build on the same core concepts.
7 Conclusion

Working with digital accessibility is all about processes. To be successful in the long run, accessibility measures need to fit into the existing processes of a software company, not be made into a new, stand-alone process. For accessibility work to be sustainable, it needs to go beyond engaged individuals. It is thus important that the company management is involved and communicates a clear standpoint that the company’s applications should be accessible and that the work with accessibility is prioritized. Taking ownership of accessibility in this manner is proactive given that accessibility-related legal requirements can be expected to spread to more types of digital applications.

There is a strong belief among employees within software development, in automated testing to solve accessibility measures, but it has limitations. Because of the nature of the accessibility requirements, only some of them can be tested programmatically. Manual testing done by development teams and user testing with disabled users is hence needed to test the remaining requirements. Support material for manual testing can also foster an understanding of the requirements among designers and developers in a way automatic evaluation tools do not. When, for example, going through a checklist to test an application, a designer or developer learns things to keep in mind for their next design or coding task. Another benefit is that the employees can shape a checklist or other support material to fit their process and company. Automatic evaluation tools do not have this flexibility.

This thesis concluded that a design system can support designers and developers by making the work of producing accessible applications systematic and avoiding them getting overwhelmed. A design system has the potential to be more than a storage for designs and code snippets that have been made with accessibility in mind. Such a storage could inherently only undertake a fraction of WCAG, because many requirements need to be fulfilled in a context and cannot be addressed within general building blocks. Luckily, a design system can support on more levels. Through documentation, it can provide guidance on how to apply instances of components and patterns correctly in a context. At the end of the chain of accessibility work, testing procedures can reference the design system by telling employees which requirements have already been fulfilled if the design system has been followed. In this way, a design system can help designers and developers focus on one part of the requirements at a time.
8 Future Work

This thesis provided insights on setting a frame for the design system at a company, and how to benefit from it once it is in place. More research is needed to investigate what tools and methods are useful when making accessible components and patterns for a design system. When the guidelines have been extended with advice on that, the methodology should be tested by being applied at a company for a long period of time.

This thesis focused on designers and developers as the target group. It uncovered, however, that members of company management and product owners are also tightly tied to the process of producing and testing accessible applications. Researching their perspective would be valuable.

The perhaps most significant finding of the design project of the thesis was the high potential of implementing condensed checklists for testing accessibility requirements inside an issue tracking system. The designers and developers who tested such a checklist were convinced it would improve their work and expressed excitement about learning new things while going through the list items. One of the users even made an effort to improve accessibility in the application they were working on shortly after the test. Due to the weak connection to design systems, developing the checklist further was considered outside the project scope. A future project about accessibility testing procedures for whole applications could adopt these results as a starting point, and preferably include product owners as well.
Acknowledgments

Many admirable people contributed to this thesis. First and foremost, I want to thank Jill Aittamaa at Metria for taking me under your wings these past four months. Thank you for guiding me and cheering me on. You have been my inspiration and my rock. I also want to thank all the other Metria employees who have participated in my interviews, workshops, focus group and user tests. It has been a delight working with you all, sharing your individual personalities and positivity.

My friend and fellow student Anja Tjärnhage deserves a special thanks for supporting me and always being ready to exchange ideas. With you I have learned so much more than I would have alone. I want to thank my supervisor Ole Norberg and my peer reviewers Frida Aringskog, William Hägström, Adam Hörnemalm and Sandra Rannanpää for steering my writing in the right direction. I would also like to express my gratitude to the three experts who readily gave me their time and welcomed me to the community of accessibility ambassadors. Lastly, I want to thank Mía Petersson for helping me find an external partner for the thesis.
References


A Final version of guidelines

The design project resulted in 10 guidelines for Metria and other companies in the process of introducing a design system. The guidelines describe how to create and maintain a design system that promotes accessibility.

Designsystem som främjar digital tillgänglighet

Guidelines riktade till ansvarig för utveckling på ett företag och teamet som skapar företagets designsystem

Framtagna av Ingrid Berglund våren 2023 inom ramen för examensarbete i interaktion och design vid Umeå universitet


Guideline 1: Sätt upp egna tillgänglighetsmål på företaget som designsystemet kan utgå från.

Det är helt avgörande för att lyckas i längden att företaget äger frågan om tillgänglighet och att det går bortom enstaka individers engagemang. Det skapar välbehövd trygghet att ledningen står bakom en ram som sätts, exempelvis att följa standarden WCAG 2.1, nivå AA.

Stanna dock inte vid lagtexter och standarder, för de kan vara svårtolkade och dessutom garanterar de inte att en produkt blir användbar för alla. En produkt som följer lagen kan ändå vara användbar för de användare ni försöker inkludera.

Definiera istället vad tillgänglighet innebär rent konkret för er. Här kan bra steg vara att ha en dialog med kunden om deras förväntningar på tillgänglighet och vilka deras användare är. I slutändan är det viktigt att ni på ert företag har en målbild som kommer från er själva. Ta reda på behoven men äg beslutet utifrån era förutsättningar och kunskaper. Gör målet till något mätbart och genomförbart.
Guideline 2: Prioritera och sortera bort irrelevanta krav.


Delar av tillgänglighetskraven är kanske inte relevanta för er verksamhet. Sysslar ni inte med video och ljud så uteslut de kraven. Har ni inga mobilanvändare så uteslut de kraven. Använder ni ett komponentbibliotek (Angular Material, Material UI etc.) så har de löst flera av kraven redan.

Guideline 3: Gör designsystemet till sitt eget projekt och definiera dess ansvar.

Tid som spenderas på att skapa och förfina generella komponenter och ta fram gemen-samma resurser slår igenom på flera projekt, så den är väl spenderad. Tillgänglighet är en del av den förfiningen och det är viktigt att inser vad ni vinner på att göra den.

Andra projekt som tar fram nya produkter kan bidra med komponenter, men för att designsystemet ska bli systematiskt behövs också ett helhetsperspektiv. Överskådligheten är viktig, och att de mest grundläggande komponenterna finns. Designsystemet behöver också tid för att underhållas.

Bestäm vad ni menar ni när ni pratar om ”designsystemet”. Ett designsystem kan finnas i olika skalor, från litet med tio komponenter och en grafisk profil till stort med hundratals komponenter i olika states, ikoner, designmönster och styleguides. Bestäm också vilken support teamet bakom designsystemet ska ge och huruvida tillgänglighetsinsatserna ska gälla både nya och gamla produkter.

Guideline 4: Använd språk som är lätt att ta till sig och förklara.


Guideline 5 : Börja litet och bygg på.

Guideline 6: Skaffa ambassadörer för tillgänglighet som ger stöd.

Ta vara på de individer som finns i organisationen som har ett driv för tillgänglighet och involvera dem i framtagandet av designsystemet. Låt dem utbilda sig och få en djupare förståelse för detaljerna i de krav som finns i WCAG eller den standard ni valt. Med sin kunskap kan de sedan skapa stödmaterial för andra i organisationen och svara på frågor. Det bästa är en kombination av att lärar medarbetarna tänka själva kring tillgänglighet och att ge vägledning med avseende på krav en i detalj.

Tyvärr tar ni er inte hela vägen på att bygga in tillgänglighet i generella komponenter och mönster. Många krav kan bara tillämpas i kontext. Genom att erbjuda stöd för hur komponenter och mönster ska användas, samt hur sidor ska testas, utnyttjar ni designsystemets potential till fullo.

Guideline 7: Bryt ner stödet på arbetets olika steg.

Skapa stödmaterial kopplat till designsystemet som hjälper till med tillgänglighet i hela processen från att skapa innehåll för designsystemet till att leverera en produkt. Stödmaterial kan ses som olika tillgänglighetsglasögon att sätta på sig, beroende på var i processen ni befinner er. På så vis behöver alla inte ha koll på allt.

Olika glasögon för olika tillfällen:

- när ni skapar generella komponenter och mönster
- när ett team använder en generell komponent eller mönster i ett sammanhang
- när en implementerad sida ska testas

Olika glasögon för design och utveckling


Kravförteckning och mall för att skapa innehåll i designsystemet

En utförlig guide av de tillgänglighetskrav ni har beslutat att uppfylla är ett bra stöd för de som skapar komponenter, mönster och annat innehåll i designsystemet. Det är smidigt om guiden delar upp kraven på olika funktionalitet som flöde, layout, formulär, brödtext, meny och knapp. En template eller mall som visar hur en komponentsida ska se ut, med de delar som ni bestämt, är också en stor hjälp.

Handledning för användning av komponenter

En komponent är inte självförklarande när det gäller hur och var den får användas. Det finns tillgänglighetskrav som inte går att bygga in i en generell komponent, som måste tas hänsyn till när en medarbetare plockar in komponenten i sitt projekt. Den som skapar komponenten skriver därför lämpligen en liten handledning till komponenten över saker.
som är viktiga att tänka på. Den ska vara invävd i dokumentationen i designsystemet, inte i en egen flik. Vissa komponenter kan behöva mycket handledning, andra lite.

### Checklistor för att testa sidor


### Guideline 8: Väv in stödet i resurser och processer som redan används.


### Guideline 9: Utnyttja automatisk testning för det den är kapabel till.


### Guideline 10: Skjut upp tillgänglighet till senare när det är motiverat.

Tillgänglighetsarbetet måste vara förankrat i verkligheten. Det går inte att göra 100% tillgängligt hela tiden. Uppmuntra medarbetare att uppnå en viss grad av tillgänglighet och öka graden för varje ny release. Det viktiga är att ni vet vilka krav ni rundar och varför, och att de tas om hand i ett senare steg så de inte bara fortsätter skjutas upp.

Tillfället när tillgänglighetsglasögonen inte behövs:

- för ta fram prototyp till användartest (om man testar med funktionshindrade kan man ofta förklara de delar de inte kan ta del av och på så vis ändå få nyttig feedback)
- för funktioner som används av få användare och går att lösa på annan väg genom att exempelvis kontakta kundtjänst

Tillgänglighetsbuggar ska behandlas jämte andra buggar, varken före eller senare. Det viktigaste är att ni prioriterar buggar som gör att vissa användare stängs ute från primära flöden.

Kom ihåg att alla tillgänglighetsinsatser är positiva, oavsett hur små de är.