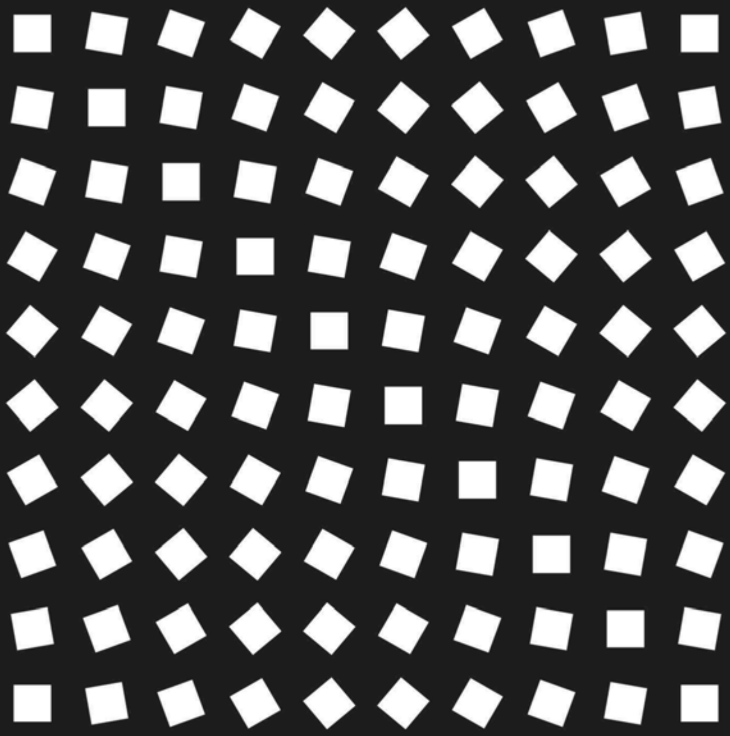


Managing UX Debt

Using a Design System

Jelle Van de Velde

PDEng. User System Interaction
Eindhoven University of Technology

15 September 2018
Eindverslagen Stan Ackermans Instituut; 2018/065

THECIRQLE

TU/e

Company Coach

Steven Lammertink

The Cirqle

University Coach

Jun Hu

Eindhoven University of Technology

Summary

Startups often operate in uncertainty with customers, product requirements and objectives susceptible to rapid change. To be able to be responsive to this uncertainty, startups often need to gain agility by intentionally occurring a limited amount of technical debt. This process allows companies to develop software faster, with the understanding that compromises made will negatively affect the product quality and thus the User Experience (UX).

The lack of an integrated user-centered design approach managing the impact of this debt poses a risk to both the product and the company as a whole; it leads to a fragmented UX, lack of consistency in the User Interface (UI), and a disconnect from changing user needs. Over time, the accumulation of this debt becomes a great weight that slows down growth and may cause users to seek out competitors.

The project outlined in this report describes the process of introducing a design system at The Cirqle to address uncovered UX gaps in the products while pro-actively reducing the amount of debt introduced into the UX design and UI development process. An initial research phase, followed by two case studies illustrate the implementation and use of the design system, together with the learnings and insights acquired through this process.

The design system had a considerable impact on the way products are designed and built at The Cirqle. Products showed an increase in design consistency and redundancies in the design and development process were reduced. Furthermore, this led to an increase of the team's efficiency; more resources could be dedicated to user-centered design activities and experiments, new features were released at a faster pace and the team was able to iterate faster on the design of features.

The current implementation of the design system can be considered to be in its early stages of maturity. Meticulous documentation of design decisions, UI components, and overarching design patterns is likely to further increase the product quality. Additionally, the introduction of both unit and visual regression testing of the building blocks of the design system would allow the team to release and update the design system with more confidence as the system scales.

1 Introduction

The Cirqle is an Amsterdam based marketing technology startup with a focus on influencer marketing. As of January 2018, I have started working at the company as a UX designer. During my first seven months at The Cirqle, I performed an innovation project in collaboration with the Professional Doctorate in Engineering (PDEng) program User System Interaction (USI) at Eindhoven University of Technology (TU/e). This report outlines the project's approach, research, and implementation based on two case studies.

1.1 About The Cirqle

The Cirqle offers brands and marketing agencies, further in this report referred to as clients, a platform to manage data-driven influencer marketing campaigns at scale. The platform enables clients to establish their online social presence through the world's most influential bloggers, creatives and social media influencers. At the time of writing, The Cirqle consists out of a multidisciplinary team of 15 people. The employees operate out of the headquarters in Amsterdam together with regional operations in both New York and Shanghai.

A typical marketing campaign run at The Cirqle starts by defining the campaign objectives and Key Performance Indicators (KPIs) in collaboration with the client. These goals are translated into a campaign brief, which is sent out to a curated network of 13 000 content creators and social media influencers. Comparable to a job description, a campaign brief highlights what the campaign is about; it includes requirements, deliverables, and incentives for influencers.

Influencers that are interested in collaborating with the client can opt-in to the campaign by applying to the brief. From all incoming applications, a limited set of influencers that fit the requirements set by the client is selected. These selected influencers create user-generated content (UGC) that is later distributed on their own social media channels to engage and activate the predetermined target audience.

The Product

The typical campaign described above is produced and managed by a campaign manager working at The Cirqle. The company developed a platform in-house to support the internal team to deliver on the above. The long-term strategy of The Cirqle, however, entails licensing this platform to third parties as a Software as a service (Saas) product. There are three distinct components to the platform:

APPS FOR INFLUENCERS AND CONTENT CREATORS

The apps for influencers and content creators connect users with clients; influencers can view and apply for campaigns, compose personal profiles and manage their account. The company currently maintains apps for three platforms: a responsive web application and a mobile app for both iOS and Android devices.

DASHBOARD FOR CLIENTS

Clients can draft, edit and approve campaign briefs, select influencers they would like to collaborate with and track the performance of their campaigns through a web dashboard.

ADMIN DASHBOARD FOR CAMPAIGN MANAGERS

The admin dashboard offers campaign managers and employees at The Cirqle the tools measure, analyze and report on the results of influencer campaigns and to manage influencers and clients.

The Team

The main departments at the Cirqle are as follows:

- Sales and Account Management: manages the relationships with clients
- Campaign Management: creates marketing strategies and coordinates campaigns
- Influencer Management: ensures collaboration with the right influencers
- Engineering: designs and develops the platform

The engineering team at The Cirqle that is responsible for developing the platform consists out of two back-end developers, two full-stack developers and myself as a UX designer. As an employee of The Cirqle, I am leading the User Experience (UX) and design efforts while contributing to the development of products described above.

1.2 Project Goal

Startups in search for product-market fit often operate in uncertainty with customers, product requirements and objectives susceptible to rapid change (Andreesen, 2007; Blank, 2006; Ries, 2011). To be able to be responsive to this uncertainty, startups often need to gain agility by intentionally occurring a limited amount of technical debt (Nilsson & Petersson, 2013). In short, technical debt is the result of shortcuts taken during the software development process in order to achieve rapid gains (Cunningham, 1992, Kruchten et al., 2012). The impact this technical debt has on the experience of the end user, referred to as UX debt, is often left unacknowledged. Leaving UX debt unaddressed does not only make the product harder to use over time but may also limit product adoption, cause users to seek out competitors or may cause potential bottlenecks in future product development (Dunwoody & Teague, 2015; Moffett, 2016; Wright, 2014).

This report focuses on the implementation of a design system in order to address UX debt systematically at startups, as they potentially offer a modular and flexible solution to the problem. The ultimate goal of this project is to enable the company to speed up the product development pace using a maintainable and robust design system that can swiftly address gaps in UX.

1.3 Project Approach

The project is divided into two phases: a research phase with an introduction to UX debt and an overview on current and best practices of implementing design systems, and a design phase with an initial technical implementation of a design system and two case studies conducted at The Cirqle.

2 UX Debt in startups

The metaphor technical debt was first coined about two decades ago by Ward Cunningham, one of the authors of the Agile Manifesto. The concept reflects the additional development work needed to maintain software due to technical compromises made during the development process (Cunningham, 1992, Kruchten et al., 2012). Kerievsky later extended the technical debt metaphor to UX design (Dunwoody & Teague, 2015). Similarly to technical debt, UX debt is an accumulation of mainly design – but also development – decisions that negatively impact the users of a product or service (Moffett, 2016).

Although technical debt itself is not part of the scope of this report, it's worth noting that the build-up of technical debt itself also negatively impacts UX. On the one hand, cutoffs may be made while implementing new designs. On the other hand, technical debt may possibly impact the performance or maintainability of the system, and thus cause a negative impact on the user experience (Knight, 2016).

2.1 Classifications of UX debt

Kruchten et al. (2012) define two types of technical debt: *visible debt*, debt that can be identified by people that aren't software developers, and *invisible debt*, debt that can only be identified by software developers. Although invisible debt such as bad architecture or code complexity may negatively impact the UX of the product in the long term, UX debt is mainly situated in the first category of visible debt.

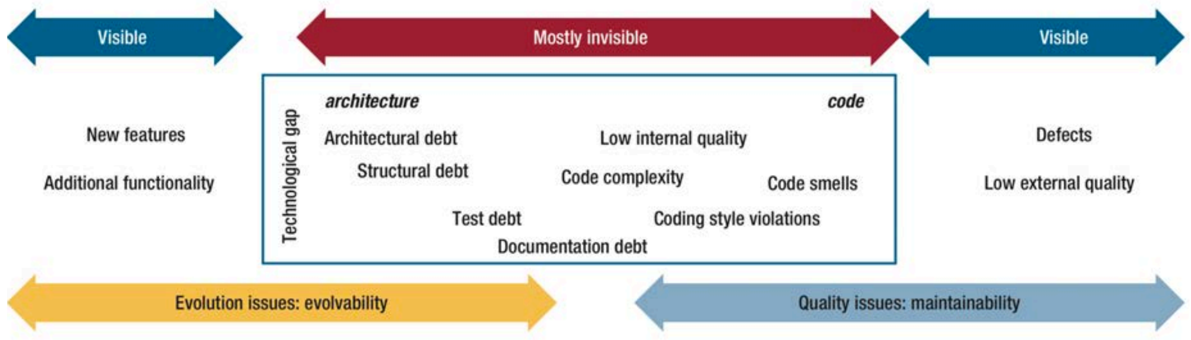


Figure 1. Visible and Invisible Technical Debt, from Kruchten et al. (2012).

According to McConnel (2007) however, can technical debt also be regarded as either *intentional* or *unintentional*. Many authors have extended these categories to UX debt (Nilsson & Petersson, 2013). Intentional or strategic UX debt, on the one hand, is a result of decisions made based on project constraints while unintentional UX debt on the other is collected due to misconceptions or false assumptions made about users and due to changes in the user's needs and expectations over time.

2.2 Reasons for UX debt at startups

Moffett (2016) states that UX debt is particularly common in enterprise contexts due to their product and organizational complexity. However, there are some characteristics typically associated with startups that make these young and fast-paced companies susceptible to collecting UX debt over time.

2.2.1 Shifts in Requirements

Startups often know rapid changes in requirements that can be attributed to the search for product/market fit (Andreessen, 2017). Finding this fit is a step in between customer validation and customer creation, and is of the utmost importance for startups, as it's often a precondition for successfully growing the company (Blank, 2006; Ellis, 2009).

In order to achieve product/market fit, feedback from customers and prospects who go through the sales pipeline is extremely valuable input to steer the direction of the product roadmap (Snyder, 2018). Startups often need to adjust their roadmap to see what gains traction and iterate on existing features. When developing new features, speed and agility are important to keep a competitive edge; this way a company might release a product with technical or UX debt in order to meet strategic timelines (Moffet, 2016). However, when this debt is not repaid shortly after the release, or when the process doesn't account for timely refactoring, UX debt gets accumulated (Chen, 2011, 2015; Gothelf, 2016).

2.2.2 Early Stage of UX Maturity

Just as any other company, startups progress through a sequence of fairly universal stages in order to potentially become a fully user-centered organization (Nielsen, 2006a, 2006b). Each organization moves through these steps at their own pace, with some sources suggesting that progressing through each stage may take up to 2 to 3 years (Buttiglieri, 2012). Startups move, generally speaking, from a development centered or ad hoc approach to design in the early stages to a more rigid UX process once they achieve product-market fit (Giacoppo, 2013). The lack of an integrated user-centered design approach to manage accumulated debt leads to an incoherent UX and, possibly, a misunderstanding of the end users of the product.

2.3 Risks of UX debt

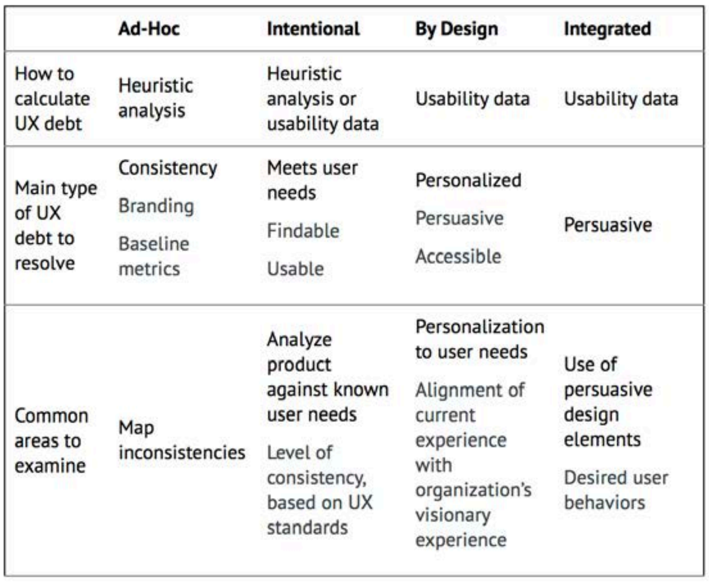
Both intentional technical and UX debt allow companies to develop software faster, with the understanding that compromises made will negatively affect the product quality and the user experience (Huffine, 2017).

The buildup of UX debt over time often poses a risk for both the product and the company as a whole. Unaddressed UX debt leads to a fragmented UX, lack of consistency in the User Interface (UI) and more importantly, a disconnect from changing user needs (Dunwoody & Teague Rector, 2015; Moffet, 2016). Over time, the accumulation of this debt becomes a great weight that slows down growth (Suarez et al., 2017).

Unintentional UX debt is arguably the worst type of UX debt to collect, as the disconnect with users naturally grows bigger over time (Wright, 2014). We should, therefore, aim to minimize the intentional technical debt while avoiding unintentional debt altogether (Moffet, 2016).

2.4 Uncovering UX Debt

Dunwoody and Teague Rector (2015) recommend different approaches to uncover UX debt, depending on the UX maturity of the organization and the product. Most features of the product didn't have any formative usability data available; the majority of these features were situated at the Ad-Hoc level, few at the Intentional level of UX maturity (Buttiglieri's, 2012). The first activity I performed upon joining the company was a heuristic evaluation of the current products to familiarize myself with the platform and discover potential usability problems.



	Ad-Hoc	Intentional	By Design	Integrated
How to calculate UX debt	Heuristic analysis	Heuristic analysis or usability data	Usability data	Usability data
Main type of UX debt to resolve	Consistency Branding Baseline metrics	Meets user needs Findable Usable	Personalized Persuasive Accessible	Persuasive
Common areas to examine	Map inconsistencies	Analyze product against known user needs Level of consistency, based on UX standards	Personalization to user needs Alignment of current experience with organization's visionary experience	Use of persuasive design elements Desired user behaviors

Figure 2. A summary of UX Debt by Maturity Level, from Dunwoody and Teague Rector (2015).

2.4.1 Heuristic Evaluation

Typically, this method involves having a small group of usability experts evaluate a user interface using a set of guidelines (Lauesen & Musgrove, 2005). In this scenario, I was the only expert reviewing the interface. The guidelines used for this evaluation were a combination of Shneiderman's 'Eight Golden Rules of Interface Design' (1986) and Nielsen and Molich's (1990) '10 Heuristics for User Interface Design' (1990).

Since all products were fully operational, a task-based evaluation could be carried out (Lauesen & Musgrove, 2005). These tasks replicated typical tasks a user might carry out using the platform. The users were assumed to have a basic understanding of (influencer) marketing and the relevant terminology.

ANALYSIS

The following phase of the evaluation involved the analysis of the listed problem descriptions. The discovered potential problems were rated using a five-point scale. All the potential problems were initially captured in a separate spreadsheet on a product basis but were later also combined to explore if there were recurring or overlapping problems.

According to Nielsen (1995), the severity of can best be judged by the following three factors: the *frequency of the problem* in the interface, the *impact of it and whether or not users are able to overcome* the issue, and lastly the *persistence* of the problem.

The severity ratings were used to allocate resources to immediately address the most serious problems. Furthermore, they also provided a rough estimate of the need for additional usability efforts (Nielsen, 1995).

RESULTS

Overall, the evaluation identified 32 total usability problems on average per product. Moffet's (2016) classifications of types of UX debt provided a structure to categorize the overarching issues. After consolidation, seven overarching issues emerged across all products:

- Features that are regularly used are tucked away layers deep in the UI. A confusing information architecture may fail to represent the user's mental model of the product. (Functional UX debt)
- Although the needs and priorities of the users may have changed over time, old features were still present in the platform. (Functional UX debt)
- Lack of brand and visual UI consistency in regards to color, spacing and, typography. Additionally, three different icon fonts were used throughout the platform. (Visual UX debt)
- As the platform grew, the number of influencers and the data campaign managers have to manage increased exponentially. Efficient tools to handle this data, e.g. performing bulk actions, searching, sorting and filtering, are lacking or used inconsistently across screens. (Functional UX debt)
- Lack of consistency and conventions in the navigation patterns between screens and pages. Interactions differ on a feature basis. (Behavioral UX debt)

- The language used in the copywriting may be confusing for new users. Additionally, notifications and error messages lack descriptive feedback. (Visual UX debt)
- Lack of error handling and descriptive error messages. (Behavioral UX debt)

3 Design Systems

A design system offers a library of interconnected visual guides, UI components and patterns documented and released by an organization as tools for designers and/or developers so that adopting products can be more efficient and cohesive (Curtis, 2017a; Kholmatova, 2017).

Design systems are mainly being developed at large organizations considering they require a considerate amount of organizational buy-in, investment, coordination and knowledge sharing (Curtis, 2017c; Ruissalo, 2018). However, the advantages offered by a design system provide a solution to the gaps in UX uncovered by the initial heuristic evaluation and potentially address the needs of startups that are searching for product/market fit or startups trying to scale and mature. Due to the increase in efficiency and consistency in the product development process, design systems could potentially offer a sustainable approach to managing UX debt at startups.

3.1 Modular Interface development

The usage and development of design systems in software design gained significant traction in recent years, but the usage of patterns in design disciplines is not novel.

The idea of a pattern library was first introduced by the architect Christopher Alexander. In his book 'A Pattern Language' (1977), Alexander attempted to answer why certain buildings feel great to be in while others may feel dull or lifeless. He argues that the feeling is based on a result of certain tangible and specific patterns, rather than on subjective emotions (Alexander, et al., 1977). Each one of the 253 architectural design patterns defined in the book, describes a problem and it's solution in such a way it can be used in a repeatedly.

The publications of the books Design Patterns: Elements of Reusable Object-Oriented Software (Gamma et al. 1994) and Common Ground (Tidwell, 2015) later introduced the idea of modular patterns to software engineering and Human-Computer Interaction. When applied to building interfaces, design patterns capture common solutions to design tensions or user needs (Kholmatova, 2017; van Welie & Traetteberg, 2000).

In 2006, Yahoo! released the Yahoo! Design Pattern Library, the first pattern library for the web. The library communicated design ideas, interaction design idioms that were new to the web (e.g., drag and drop) and discussed interface patterns commonly used by the company (Scott & Malone, 2006). The processes and considerations in this work are familiar to popular design systems created today such as Google's Material Design and the Salesforce Lightning Design System.

3.2 Rise in popularity

The wide adoption of design systems across large organizations can be attributed to the advantages design systems were offering in comparison to the current tools:

OFFERS CONSISTENCY

Providing a consistent UI, UX and branding across all platforms are vitally important to organizations; a cohesive experience is more easily understood by their users (Bailey, 2016; Saarinen, 2016).

In an industry research, 59% of the interviewed enterprises report that improving consistency is a relevant problem to them (UXPin, 2018). In many organizations, UX and UI design are still bespoke; teams offer tailor-made solutions to individual problems (Suarez et al., 2017). Although shipping one-off solutions in software development and design is not inherently bad, it often leads to inconsistencies and the accumulation of both technical and design debt when the solutions aren’t built upon a solid foundation (Saarinen, 2016).

ENABLES SCALABILITY ORGANIZATIONS

Scaling a consistent UI and UX design across a multitude of teams and different platforms has become a new challenge in the software industry (Rohde, 2014; Suarez et al., 2017). As an organization grows, it’s common for designers to focus on discrete areas of the product. This can lead to an inconsistent user experience across the platform (Suarez et al., 2017). Design systems enable teams to easily scale a product by using design standards and patterns (Curtis, 2010; Rohde, 2014). Having a set of standards in place also helps to mitigate impacts of reduced face-to-face collaboration in large, distributed teams.

INCREASES SPEED & EFFICIENCY

Using a design system helps teams to build a cohesive product at a faster pace as it helps reduce redundancies and inconsistencies in design and engineering implementations and gives teams a common language to work with (Kholmatova, 2017; Saarinen, 2016).

Many teams have reported achieving a more efficient implementation process, savings, and better design (Curtis, 2010). Common metrics to calculate this benefit from an organizational perspective include the time saved per feature, increased delivery, reduction in the number of bugs reported and improved the internal satisfaction of the design and development team (Martin, 2018). In a case study, Loomer (n.d.) highlighted a decrease of 50% of the time to market, while reducing software development costs and customer support tickets by 20% and 52% respectively.

TECHNOLOGY

In addition to these benefits, more component-driven web standards, development libraries, and frameworks made the technical implementation of end-to-end design systems more effective (Lambert, 2016; Priego, 2017). Popular JavaScript frameworks such as Angular and React encourage developers to create encapsulated UI components that manage their own state and can be combined to build up complex interfaces (React, 2018).

3.3 Assets of a Design System

To gain a better understanding of what a design system in practice consists out of, five existing design systems where analyzed: Salesforce Lightning, Shopify Polaris, WeWork Plasma, Atlassian's Design Language, and IBM Carbon (see Appendix 1). These examples were selected based on their current popularity in the UX design community (UXPin, 2018; Walrack, 2018).

3.3.1 Style Guide

Style guides and design systems are often wrongfully confused with each other. While style guides are static artifacts of the design process, design systems can be perceived as a living product serving an ecosystem of products with explicitly defined design patterns (Griffiths et al. 2000; Curtis, 2017).

All analyzed examples include a style guide that lays the foundation for the design system. Style guides often provide brand-defined visual attributes, such as colors, typography, spacing, icons and visual style elements, and guidelines on how to achieve visual and functional consistency (Gale, 1996). Traditionally, style guides and their graphical assets would be defined by designers and implemented by developers. This means a designer is a dependant on a member of the development team to implement even relatively minor design decisions (Chenais, 2018):

1. The designer updates the color in the design tool.
2. This design update is shared with the developer.
3. The developer updates the value in the code.
4. The designer can see the result in a development environment.

This workflow has several drawbacks:

- It slows down the development and design process (Chenais, 2018)
- The implemented visual design may differ from the style guidelines defined by designers (Gale, 1996)
- designers and developers would speak with different terminology (Ruissalo, 2018)

To address these potential pitfalls, there are industry recommendations to define visual UI styles as design tokens (Rohde, 2014). Tokens are code-level attributes that store a style value. The separation of the name and the content allows the token to be used independently of the value it represents throughout the entire design system; changes to the value of a design token will propagate through all products (Ruissalo, 2018). Using design tokens, or variables, to represent visual styles in a design system is a necessity for scalability, consistency, and maintainability as it creates a single source of truth for the entire system (Rohde, 2014). Additionally, naming conventions for design tokens offer context to all users involved, enabling designer and developers alike to use the same terminology (Ruissalo, 2018; Suarez et al., 2017).

3.3.2 User Interface Components

UI components are the technical implementation of user interface patterns. Components allow designers and developers to break up the UI into independent, self-contained, reusable parts and think about each piece in isolation (React, 2018). This abstraction enables teams to build large-scale applications in a modular way (Vue.js, 2018).

A library of UI components enables one of the key objectives of a design system; it discourages the use of one-off solutions and thus reduces inconsistencies in the UI and improves reusable of design and code solutions across products (Kholmatova, 2017; Suarez et al., 2017).

Most design systems rely on a variation of the Atomic Design methodology introduced by Frost (2013) to bring structure to their collection of components. Frost lays out a systematic approach for creating a component-based pattern library, and in turn design system. He argues that interfaces should be thought of as a hierarchy of elementary building blocks, rather than uniform or isolated pages.

The name atomic design refers to the metaphor he puts in place to refer to these building blocks. In this metaphor, the most basic building blocks, called atoms, are progressively used and grouped together to form increasingly larger concepts such as molecules and organisms, before being used to construct templates, and finally pages from those templates.

- Atoms: foundational building blocks that comprise all user interfaces
- Molecules: simple groups of UI elements functioning together as a unit
- Organisms: more complex UI components composed of groups of molecules, atoms or other organisms
- Templates: page-level objects that place components into a layout and articulate the design's underlying content structure.
- Pages: specific instances of templates that show what a UI looks like with real representative content in place

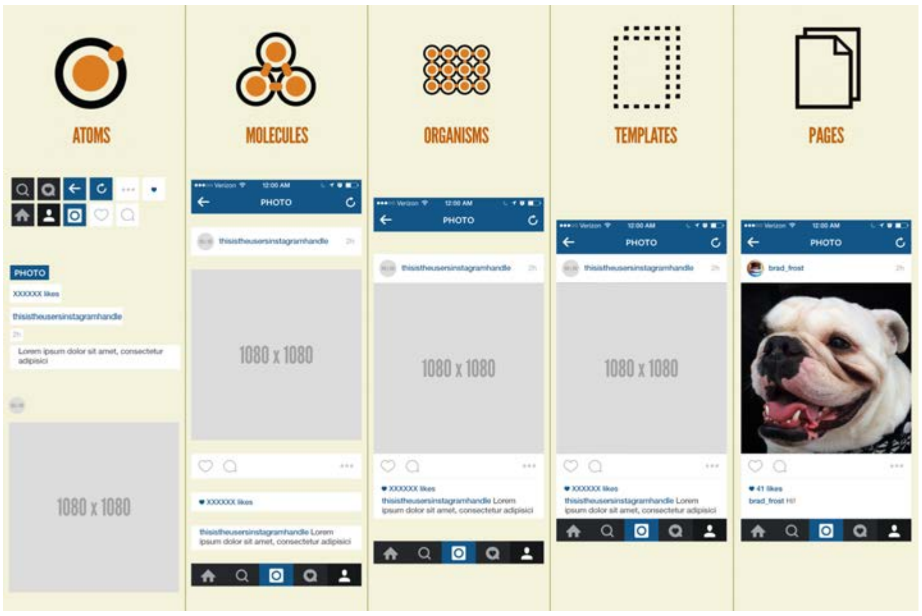


Figure 3. Atomic Design methodology as illustrated by Frost (2013).

Airbnb takes a different approach to design and develop their design system. Saarinen (2016), Principal Designer at Airbnb, argues that building a user interface from atoms only works well in theory. He explains that an atom based design system initially allows teams to design a coherent and flexible product, but that individual atoms also leave room for interpretation and that re-usable atoms often end up being used in many different ways. Moreover, he claims that this approach leads to disjointed experiences and a codebase that is hard to maintain.

Rather than various atoms, the Airbnb component library only consists of elements (comparable to molecules in the atomic design paradigm) that are iteratively developed to fit the product's needs. The contents of these components, however, are based on a static style guide.

3.3.3 Documentation & Design Principles

Documentation, design principles and guidelines offer team members across the organization more insight into how a particular design pattern, UI component or design solution should be used. While UI components ensure consistency across the smaller elements in the product, documentation provides guidelines and best practices on how they should be implemented (Connolly, 2018; Suarez et al., 2017).

4 Design

This chapter goes into more depth on how a design system was implemented and used at The Cirqle. The chapter consists of the initial implementation of a design system tailored to the company's needs and based on the knowledge gathered during the initial research phase.

Furthermore, two case studies offer insight as to how the system was later used in the product development process and matured over time. These case studies include a comprehensive description of the design process and reflect on the use of the design system during the design and development phase.

4.1 Initial Implementation

The objective of the initial implementation was to lay the groundwork for the iterative development of the design system. As the first step in building a design system, most resources recommend starting with composing a UI inventory first (Treder, 2017). Frost (2013) defines a UI inventory as "a comprehensive collection of the bits and pieces that make up your interface."

Rather than starting by creating such an inventory, however, the team was aiming to iteratively develop the design system based on the core workflows, tasks, and feedback collected from users. For a design system to be purposeful, the contents need to solve specific problems and be inspired by user-centered design activities (Ruissalo, 2018; Wroblewski, 2018).

The main objective of a design system should not be achieving agility, efficiency, and consistency, but rather leveraging these advantages to create meaningful experiences (Kholmatova, 2017; Madsen, 2017).

4.1.1 Approach

The team considered two different approaches when implementing the design system: the *Big Bang* and the *Slow drip* approach (Gothelf & Seiden, 2016).

Within the first approach, a team would take a limited time away from product development to design, develop and implement all the necessary building blocks for the design system. Gothelf and Seiden (2016) argue that this approach works best for products that are relatively young and simple. The main drawback, however, is that this approach either halts product development or requires a dedicated multidisciplinary task force (Curtis, 2017a).

An alternative is the Slow drip approach, in which a team keeps working on the product while continuously adding components as they go. This approach comes with one big caveat; it could initially slow down iterations and the development of new features since the system would be incomplete.

The Big Bang approach seems to be the most common strategy in the industry, with organizations creating standardized and modular alternatives to their current UI and design elements (Curtis, 2017a; Saarinen, 2006; Treder, 2017). Since none of the team members had experience building a design system and we didn't want to halt ongoing product developments, we opted for the slow drip approach.

This approach would allow us to iteratively design and build the design system inspired by user-centered design activities (Madsen, 2017; Ruissalo, 2018; Wroblewski, 2018). Furthermore, Loranger (2015) also points out that incremental design changes and continuous improvements are known to lead to more high-quality products.

4.1.2 Implementation

During the first sprint, we focused on establishing a structure that would allow the team to expand the design system over time. Since we were planning to design the system iteratively, the design system would scale both horizontally and vertically; new building blocks will be introduced, while existing elements will gain more options, versions, and refinements.

DESIGN TOKENS

A collection of design tokens was defined to bring the design attributes used in brand guidelines, marketing materials, and existing interfaces together. Furthermore, meaningful combinations of different design tokens were grouped together and defined as utility classes; these are layout specifications that can are often widely used in the development process. Using development classes creates consistency in the naming, makes the code more readable for developers and creates an additional level of modularity.

Since there was no need at The Cirqle to create technology-agnostic design tokens, the values of the style guide were defined as Sass variables. Sass is a CSS preprocessor that enables designers and developers to use features that don't exist yet in CSS such as variables, nesting, inheritance, etc. To make the design tokens more accessible, a consistent and descriptive naming pattern was used. Curtis (2016) promotes the idea to organize tokens in two categories: an option-based naming describes the value of the design token, while a decision-based naming implies the use case of a token in the final product.

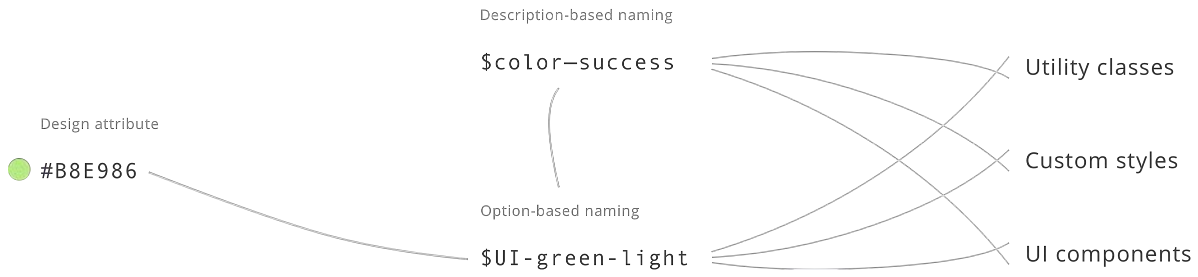


Figure 4. Example of design tokens at The Cirqle

UI COMPONENTS

The initial implementation of the Design System was structured based on the Atomic Design methodology introduced in the previous chapter, albeit with a more industry-related naming scheme that fit the code structure and naming conventions used at The Cirqle prior to the design system (Kholmatova, 2017; Meier 2017).

At the beginning of 2018, the development team at The Cirqle decided to develop all new products moving forward using Vue. Vue is an open-source JavaScript framework for building a component-driven UI (Vue.js, 2018). An interface in Vue is composed based loosely-coupled components.

A Vue component typically consists out of three parts: a template defining the HTML markup, CSS style definitions and a script tag with JavaScript to define the components' logic. Collocating and coupling these three layers in a single component file makes the component more cohesive and maintainable (Vue.js, 2018).

UI screens in Vue are build up using a tree structure of multiple components. This tree structure can either be achieved by nesting components or by using slots. Slots serve as distribution outlets for content and are modeled after the current Web Components spec draft. This allows you to effectively pass DOM elements into a child component.

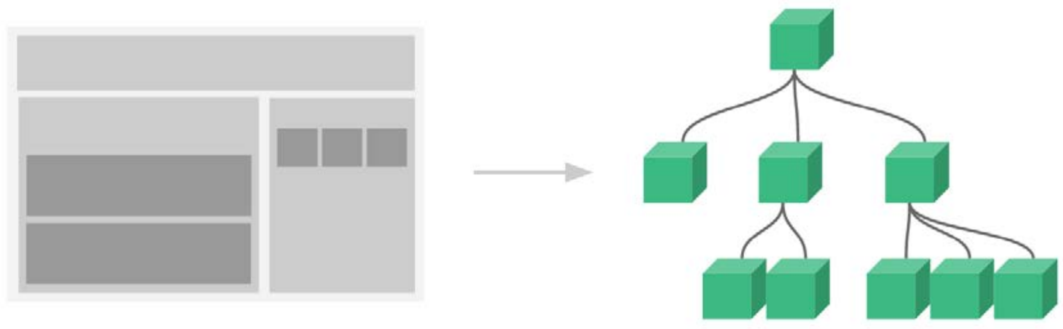


Figure 5. "Composing with Components" as shown on the Vue.js documentation website.

Suarez recommends using a unique namespace to prefix UI components and CSS classes, e.g. 'cq-[component-name]'. This will prevent attribute and class collisions on a page, especially when working with legacy code (Suarez et al., 2017).

SELF CONTAINED REPOSITORY

The design system was set up in a source control repository, independent from the main codebases used at The Cirqle. This allows for both short and long-term benefits:

Versioned releases

Importing the package through a package manager allows the team to use Semantic Versioning. Versioning a design system prevents updates from breaking code or functionalities in the respective products where it is being used (Preston-Werner, n.d.; Suarez et al., 2017).

Shared across multiple products and projects

The design system's repository is imported as a standalone dependency across all products. A single change made to the design system automatically propagates through all products and platforms.

Single source of truth

The design system repository serves as a single source of truth for interface design, containing all the necessary building blocks to construct a UI. The final products' codebases only define the state of the UI and give context to the design system's assets.

DOCUMENTATION

The initial implementation of the design system was documented in two places: a Sketch file is maintained with all the design tokens and UI components and shared documents were used to communicate the use cases and different variations of UI components.

4.1.3 Reflection

The introduction of the design system to the product development process positively impacted the product's visual consistency from the beginning. The design tokens enforced a consistent usage of design attributes, suspended the use of hard-coded values, and offered a shared vocabulary to the team around all the visual properties.

Keeping the documentation up-to-date with the system's code instantly proved to become a challenge. Multiple team members were contributing to the design system in parallel at a rapid pace, causing the documentation to become outdated quickly. This, in turn, led to an inconsistent use of certain elements, essentially defeating the purpose of creating a design system.

To address this challenge, the next steps were taken during the following sprints:

- Meaningful presets were added to low-level components that are used frequently, e.g. the button UI component has a ‘save’ preset that automatically defines all the component’s options to visually style the save button.
- During the design handoff and daily standups, the team discussed what parts of the design system to use for the implementation of the feature we were working on, keeping all members in sync.
- If needed, documentation was added co-located to the components code rather than in separate documents.
- Promoting the use of templates over elements reduced the need for detailed documentation for developers. Additionally, this also reduces repetition of code, improves visual consistency, and decreases the amount of time spent on the implementation of the design.

5 Case Study: User Sign Up Flow

Influencers and content creators are required to create an account on the platform in order to be able to view and apply for campaigns. During the signup process, new users complete their personal profiles and authorize themselves to connect their social accounts and websites. The data collected from those channels, together with a personal profile highlighting their work and content, is used to determine if someone would be a great fit for a client or a campaign.

The redesign of sign up flow was part of a complete overhaul of the influencer dashboard. The process for this redesign was selected as a case study in this report because of the following reasons:

- The way the sign up workflow is designed is critical, it's essentially a user's first impression of the product (Vijayashanker, 2016)
- A task analysis showed that this is one of the most important tasks on the platform
- A change in user and data requirements leading up to the redesign; the requirements of this feature differed heavily from the previous implementation

Up until the redesign of this feature, the team's efforts designing and developing the design system were mainly focussed on the client dashboard and tools for internal use. A redesign of the influencer dashboard, together with the design of a new sign up flow, gives us the possibility to evaluate the already existing UI components with external users.

The main objective of this redesign effort is to iteratively design a new signup flow that increases the amount and quality of the data that gets collected, increases the number of successful signups and reduces the amount of time spent by the team to answer recurring questions.

5.1 Conceptual Design

5.1.1 Research

Over the course of 3 months leading up to the redesign effort, we were actively collecting customer support records and (remote, unmoderated) session recordings from users signing up to the platform. While the user feedback was immediately captured in the product backlog, the recorded visitor sessions were later analyzed.

Screen recording allows for an unobtrusive collecting of a rich record of a user interacting with a system in the natural context (Tang et al., 2006). It is a cost-effective way of collecting a rich record of data, but it's worth noting that this approach lacks the qualitative characteristics a typical Contextual Task Analysis offers; there is no means to interview the user and only the interactions that happen within the platform itself are recorded.

55 recordings in total were evaluated. During the reviews, special attention was paid to errors users encountered, screens of the product they would spend a longer than average time on and the quality of the data they would enter. Two different types of codes were added to every recording; the area of the product being used in the recording and the possible issues the part of interface element the users were having problems with.

The results of this analysis can be summarised as follows:

- High churn rate
  38% of users are abandoning the signup process when they land on the profile screen, leading us to think that the current form might be too lengthy.
- Confusing data entry
  The screen recordings showed some users were struggling entering the necessary data. Possible causes for these problems include: lack of descriptive labels and input validation on some of the input fields
- Lack of clear call to actions
  Lack of distinct call to actions (CTA) in the signup flow, causing users to leave the flow or encountering errors they couldn't recover from. Some users would revert to the customer support, leading to a very time-consuming process for both the development team, the

support team, and the end users.

5.1.2 Stakeholder Requirements

I conducted several informal interviews with the product owner of the influencer dashboard and other stakeholders within the company. These interviews were set up to understand better what challenges they were facing from an operational perspective with the current design, what data they expected from influencers and what the long-term goals are for the product. Defining a long-term vision for software products is essential to not create any bottlenecks early on in the project that may hinder product development down the road (Bricklin, 2004).

The list of data collection requirements was categorised into logical groupings:

PERSONAL INFORMATION	PROFESSIONAL INFORMATION	CHANNELS	LEGAL
Name	Type of content they produce	Connect social accounts	Accept Privacy Policy
Location	Brief introduction	Website	Agreement with Terms and Conditions
Gender			
Age			
	only required for users registered as a company:		
Phone number	Invoicing details		
Email address	VAT number		

5.1.3 Best Practices

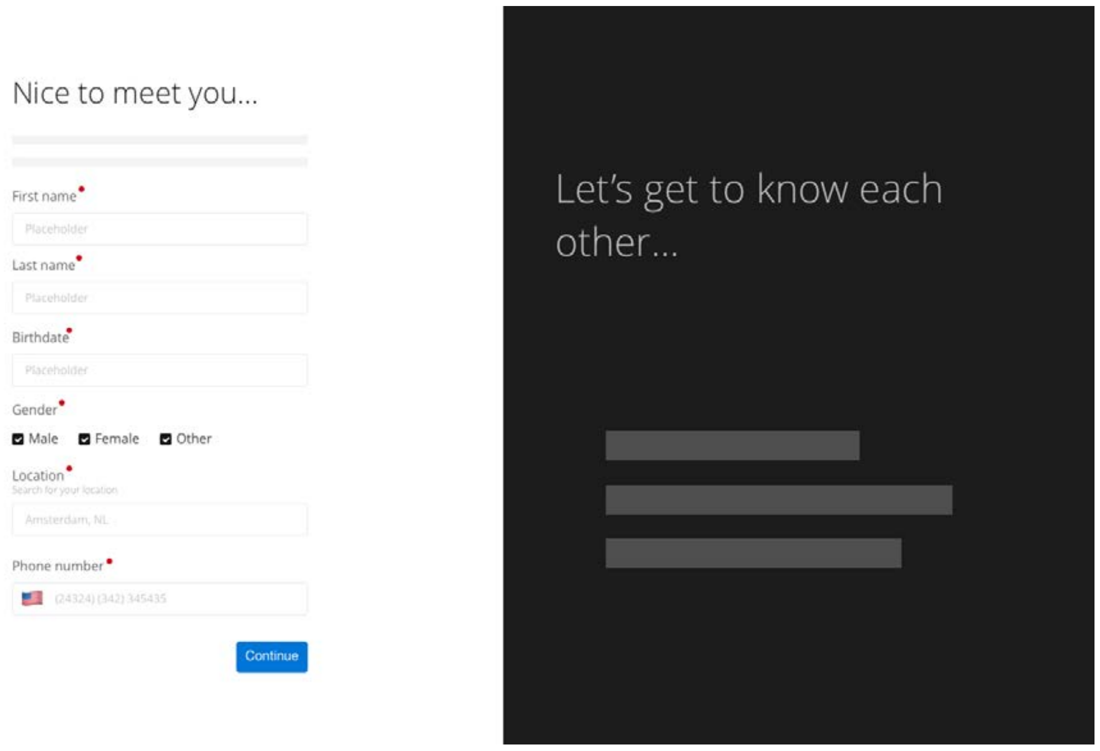
An analysis of different signup flows showed the following set of shared characteristics and best practices across competitors and online services that was not present in the current design:

- Lengthy forms are chunked
  Signup flows are often chunked into different steps focussing on only one type of data input at the same time. Dividing a form up into multiple steps helps to keep the number of input fields shown to a minimum, reduces the workload for users and increases the completion rate (Tidwell, 2005; Formstack, 2015).

- **Only necessary fields are included**
  Only the necessary steps are included to keep the form as short as possible. Reducing the number of fields increases the conversion of a form (Kirkpatrick, 2011; Gardner, 2013). Optional fields are should be left out as much as possible, reducing the length of the form. Checkboxes to accept Terms and Conditions and Privacy Policies would often be omitted. However, with GDPR regulations becoming active in 2018 this was no longer possible.
- **Definition of the value proposition**
  Explaining the user what is expected from them and communicating why they need to fill in the questions asked helps to build trust. (Rao, 2018)

5.1.4 Conceptual Design

A digital low fidelity prototype of a potential solution was created based on the problems identified with the current design, opportunities for improvement, and the updated data requirements.



Nice to meet you...

First name*
Placeholder

Last name*
Placeholder

Birthdate*
Placeholder

Gender*
☑ Male ☑ Female ☑ Other

Location*
Search for your location
Amsterdam, NL

Phone number*
(24324) (342) 345435

Continue

Let's get to know each other...

Figure 6. A screen (step 3/7) of the low fidelity prototype for the redesigned signup flow.

The signup form was chunked into meaningful steps based on the logical groupings defined above to set the flow of the form up as a conversation (Barry, 2012; Wroblewski, 2008). Each step is build up using a conversational tone and offers the user a clear heading and information about why the type of data is required (Barry, 2012). The steps were sorted based on the perceived difficulty of the data entry and the importance of the entered data to the stakeholders. For some users signing up, specific steps could be left out of the signup process, showing only relevant input fields.

A pilot test was conducted with five employees of The Cirqle. The results can be summarized as follows:

- Users understood how the interaction of the flow worked, but were lacking the ability to move backward and forward between steps
- Users were unsure if they could also connect multiple channels
- Users were confused that, after clicking the 'finish' button, they would immediately be redirected to the homepage of the dashboard
- The panels on the right explaining the value to the user were not perceived as useful for every step
- Some form sections were still too lengthy (see figure 4), especially on mobile screens where the fields at the bottom of the form and the primary CTA would appear below the fold

5.2 Design

The steps that were perceived as too lengthy were chunked down even further based on the feedback from the pilot test. The copywriting on the additional panels was made more concise and replaced with relevant images where possible. Additional help text was added were users showed uncertainty.

As a next step, I implemented the design and the structure for the signup process in code. This would allow us to carry out an interactive user test; we first conducted another pilot test with members of the team and later evaluated the flow with external users.

The majority of the necessary visual UI components to build up this flow was already defined in code, allowing us to skip the sprint to define the high fidelity design. The time saved on this step was later used to improve the interaction design, form validation, and error handling.

5.2.1 Implementation Using the DS

The following section gives a detailed overview of how the design was implemented using the design system.

A form pattern component lays out a form header, an array of input fields, and form actions on a screen. This assures consistent layout of all input forms across the platform and handles form validation, form submission and error handling.

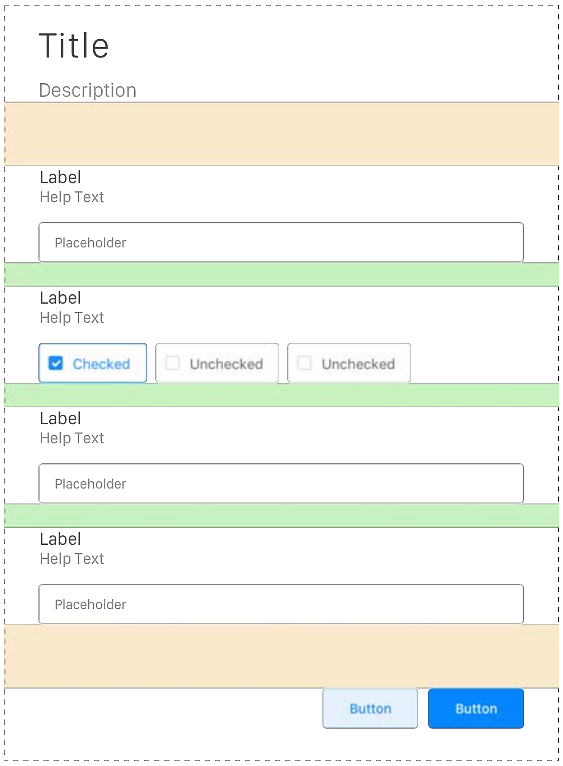


Figure 7. Wireframe showing how a UI component (pattern) distributes content on a page.

The different forms that make up the steps of the signup flow are displayed on the screen using a wizard component. A wizard design pattern is best suited for tasks that are either unfamiliar or are very complex. It enables untrained users to achieve a goal by splitting up the task into a sequence of steps and by only responsively disclosing to the user what is expected (Tidwell, 2015). Both the client and the influencer dashboard contain some steps that are necessary to complete a marketing campaign, but aren't used frequently throughout the customer journey. Iteratively designing such an interface pattern as part of the design system showed great potential to improve the usability and UX for multiple features across different products, while improving the maintainability and consistency of the product.

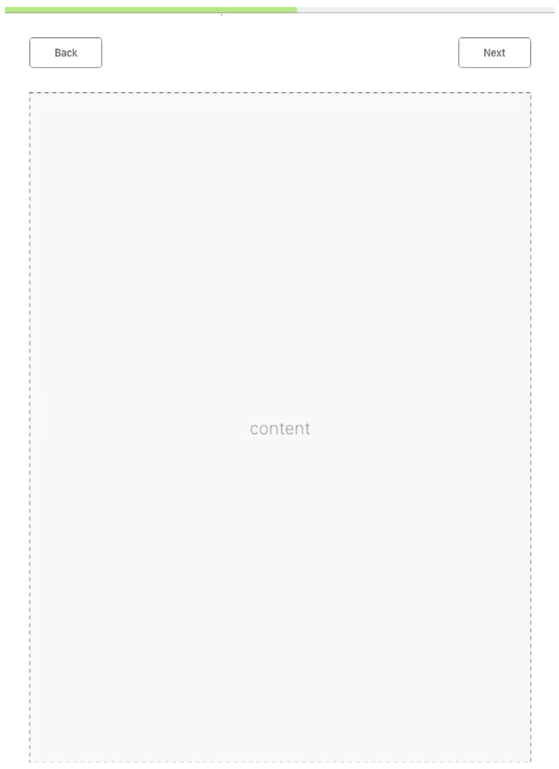


Figure 8. Wireframe of the wizard component. Any other UI element can be slotted into the content section.

5.2.2 User Evaluation

An on-site user test was conducted with 5 users of the platform. Three of the users that were invited were recent signups, 2 of them were long-term users. All of the participants were also using other platforms from competitors. This type of users is interesting to include in tests since end-user tend to compare the experience with similar applications they have used in the past (Sakhardande & Thanawala, 2014). As the study was conducted with five participants, we aimed to gather qualitative results by asking the participants to think out loud and explain the reasons behind their actions. We ended the user tests with a semi-structured interview. The results from the user test can be summarized as follows:

- The buttons in the navigation at the top of the wizard were not immediately noticed by desktop users; on a mobile device the buttons were clear, but desktop users had trouble locating the 'next' button. To accommodate this, an additional button was added at the bottom of the screen after the form has been successfully validated.

- The length of the signup process was not clear to users when they landed on the page; a progress bar was present, but went unnoticed. The visual design of the progress bar was changed to make the progress more clear.
- The flow seemed to match the mental model of the users. All of the users were able to successfully complete the steps.
- Users preferred having images on the right side of the screen over text.
- The signup process explains why certain data is collected, but does not show to users how their data is displayed to clients. Users have access to their profile however, but only after they complete the signup flow. Images and screenshots of these profiles were included in the signup flow later.
- Users actively compared the wording and industry terminology the platform used in the signup flow with the terms used by competitors. The copywriting was found confusing when it did not match with the wordings they were used to.

5.3 Outcome

The first month after launching the updated sign up flow, the platform saw an increase of 82.3% of successful signups. The second month after the redesign signups were 66.31% higher than prior to the redesign.

Reflection on the Use of the Design System

The redesign of the signup process was part of a larger project, in which we refactored the entire influencer web application using existing assets from the design system. It was the first time that UI components were included across different products.

Changes made to UI components propagate through all products. This is often beneficial as improvements to the UI and UX are immediately present in other products. In this case, e.g. the additions to the form validation and error handling are immediately present in the client dashboard, without the need for additional design or development work. However, it is easy to lose oversight of the content of the design system when multiple designers and developers contribute to it simultaneously.

This led to the following challenges:

- Bugs, either visual or functional, in assets of the UI components are present in all products when they occur in the design system.
- Visual changes are hard to track; it's hard to predict or evaluate how the changes to the styling of a component will impact other screens.
- When a component requires a fundamental change, all other screens containing that component need to be refactored.

6 Case study: Content Approval

The online platform was set up to automate influencer marketing campaigns from start to finish. One of the steps in the campaign process is making sure that the quality of the published content is up to par with the client's expectation while guaranteeing the creative freedom of influencers. On the one hand, clients often require strict control over the content that is published, while influencers want to retain their creative freedom and want the content to appear organically within their feeds.

Up until now, this process was managed manually by campaign managers, which led to a substantial amount of emails back and forth between them, the influencers and clients. Campaign managers had to log every request and every update in a lengthy spreadsheet while going back and forth communicating over different channels with all parties involved. This proved to be a cumbersome, time consuming and repetitive task.

6.1 Conceptual Design

Together with the campaign management team we brainstormed on how we might improve this process so that users are more efficient and have to spend less time managing the process rather than focussing on the quality of the content produced. During interview sessions with influencers, we discussed how this approval process fits into their workflow and how they keep track of all the content that needs to be submitted and reviewed on their end.

We defined high-level objectives and desired outcomes, that were captured as user stories (Appendix 2). Based on this list, we prioritised a set of functionalities that would be fundamental to create a minimal viable product (MVP). Ries defines an MVP as the version of a new product which allows a team to collect the maximum amount of validated learning about customers with the least effort (Ries, 2009).

- upload content
- View the content that was submitted
- Approve content

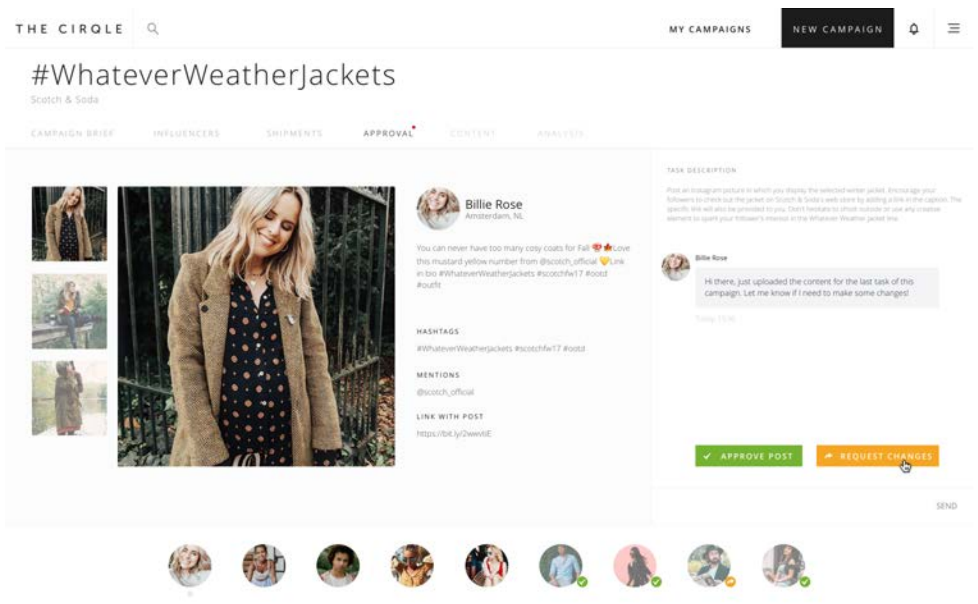
- Communicate feedback
- Edit content

A clickable prototype was designed to evaluate the feasibility of the product with the campaign management team. Although some minor improvements had to be made to the UI, the internal stakeholders acknowledged the current design's potential.

6.2 First design

Based on the conceptual design, we built a coded version of the feature using existing components. This allowed us to not only test the workflow, interaction and functionality of the feature, but also evaluate the visual design and brand elements present in the product.

Influencers are able to submit their deliverables using a form that consisted out of an upload button and input fields to add a description and related information to the content. Following this, campaign managers and clients are able to view submitted deliverables, communicate feedback, approve the deliverables or request additional changes.



Billie Rose
Amsterdam, NL

You can never have too many cosy coats for Fall 🍁 Love this mustard yellow number from @scotch_official 💛 Link in bio #WhateverWeatherJackets #scotchfw17 #ootd #outfit

#WhateverWeatherJackets #scotchfw17 #ootd

@scotch_official

https://bit.ly/2wwvtiE

Post an Instagram picture in which you display the selected winter jacket. Encourage your followers to check out the jacket on Scotch & Soda's web store by adding a link in the caption. The specific link will also be provided to you. Don't hesitate to shoot outside or use any creative element to spark your follower's interest in the Whatever Weather jacket line.

Billie Rose

Hi there, just uploaded the content for the last task of this campaign. Let me know if I need to make some changes!

Figure 9. First design of the content approval feature in the client dashboard.

The first design was evaluated by a client and two internal campaign managers running a marketing campaign with 40 influencers that had to deliver two items of content each. At the end of the campaign, we set up a feedback session with the client and sent out questionnaires to influencers. The results can be summarised as follows:

- While most influencers successfully managed to submit their content, they argued there was insufficient guidance when uploading their deliverables.
- Both the client and the campaign managers also made similar remarks; it was not always clear to them what actions they had to take next. Additionally, they mentioned it was challenging for them to keep track of changes; by only displaying the most recent version of the content it was challenging to keep track of changes.
- Campaign managers found the feature helpful but were often bumping up against the limits of the scope of the MVP. Functionalities to handle edge cases were not included yet.
- When we initially evaluated the wireframes internally, the design did not anticipate for the amount of content that would be submitted. The client explained the need to filter or sort the submitted content.

6.3 Second design

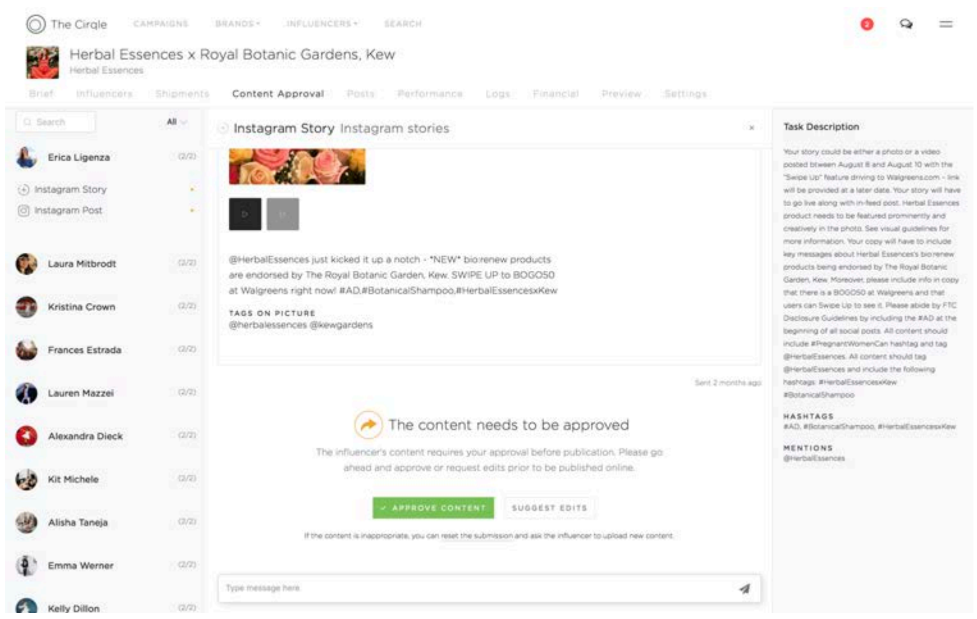
The feedback we collected from the MVP was prioritized using a prioritization matrix. In this decision-making exercise, possible actions are weighted against each other based on two factors: the effort required to implement and the potential impact (Gibbons, 2018).

Based on the problems we identified in the first iteration, the prioritisation resulted in the following steps to be taken:

- Using the wizard style interface we discussed above, we were aiming to offer influencers better guidance while submitting content. Since this process is clearly more sequential than the already existing signup flow, a stepper indicator was added to the UI pattern.
  The functionality of the upload component was expanded; users are now able to upload multiple files of any filetype, disregard previously uploaded files and view and playback any uploaded content in a lightbox.
- The submitted content and chat messages are being displayed in the same chronological overview. By capturing the revision history of the content together with the feedback, users are

able to see the progress of a deliverable over time. Changes made to the content are being highlighted.

- Status messages are added to the bottom of the conversation. These messages explain the current state of the approval process and show the relevant call to actions.
- Real-time notifications have been added, notifying users about feedback and status changes to the deliverables. Additionally, users would receive email notifications about updates.
- The overview of the submitted content is now organized in a vertical list. The user is able to sort, search and filter the list in order to find the deliverables that are relevant to them.



Herbal Essences x Royal Botanic Gardens, Kew
Herbal Essences

Erica Ligenza (2/2)
Laura Mitbrodt (2/2)
Kristina Crown (2/2)
Frances Estrada (2/2)
Lauren Mazzei (2/2)
Alexandra Dieck (2/2)
Kit Michele (2/2)
Alisha Taneja (2/2)
Emma Werner (2/2)
Kelly Dillon (2/2)

Instagram Story Instagram stories

@HerbalEssences just kicked it up a notch - *NEW* bio:renew products are endorsed by The Royal Botanic Garden, Kew. SWIPE UP to BOGO50 at Walgreens right now! #AD,#BotanicalShampoo,#HerbalEssencesxKew

TAGS ON PICTURE
@herbalessences @kewgardens

Sent 2 months ago

The content needs to be approved

The influencer's content requires your approval before publication. Please go ahead and approve or request edits prior to be published online.

If the content is inappropriate, you can reset the submission and ask the influencer to upload new content.

Task Description

Your story could be either a photo or a video posted btween August 8 and August 10 with the "Swipe Up" feature driving to Walgreens.com - link will be provided at a later date. Your story will have to go live along with in-feed post. Herbal Essences product needs to be featured prominently and creatively in the photo. See visual guidelines for more information. Your copy will have to include key messages about Herbal Essences's bio:renew products being endorsed by The Royal Botanic Garden, Kew. Moreover, please include info in copy that there is a BOGO50 at Walgreens and that users can Swipe Up to see it. Please abide by FTC Disclosure Guidelines by including the #AD at the beginning of all social posts. All content should include #PregnantWomenCan hashtag and tag @HerbalEssences. All content should tag @HerbalEssences and include the following hashtags: #HerbalEssencesxKew #BotanicalShampoo

HASHTAGS
#AD, #BotanicalShampoo, #HerbalEssencesxKew

MENTIONS
@HerbalEssences

Figure 10. Second design of the content approval feature in the client dashboard.

6.4 Outcome

The usability of the second design was assessed using the System Usability Scale (SUS). SUS is a 10-item scale and an effective and reliable tool for measuring the usability; scores have a range of 0 to 100 (Brooke, 1996). In total, 22 participants evaluated the content approval feature after using it for the first time. In this evaluation, the mean SUS score was 78.5, with a standard deviation of 14,7.

Reflection on the Use of the Design System

Having already a set of components in place allowed us to create a working MVP in under two weeks. This way, we could create a fully functional working prototype that replicates the design and behavior of other features in the platform and connects to live data. This helped us to evaluate the feasibility and user flow of the feature but did not necessarily warrant a great user experience. Ruissalo (2018) also touches on this matter; having a component library may help a team achieve efficiency or consistency, but designers should not lose track of the different contexts and the user's job to be done they're designing for.

The design system also had a clear influence on the prioritization exercise described in the chapter above; some costly ideas became less expensive to implement thanks to readily available UI components in the design system. The design and development work needed to create a submission flow that offers influencers more guidance could clearly be reduced due to already existing components.

Alternately, ideas with a seemingly smaller impact on this particular feature were rated with a higher priority. Improving the way notifications are handled on the platform and visualizing status messages would know a bigger long-term payoff because of their reusability across the product.

Although this approach did not impact the prioritization for critical UX or usability issues, it would affect prioritization for uncovered issues with a lower potential impact on the feature.

7 Discussion & Reflection

The design system had a considerable impact on the way products are designed and built at The Cirqle. Products showed an increase in design consistency, and redundancies in the design and development process were reduced. Furthermore, this led to an increase of the team's efficiency; more resources could be dedicated to user-centered design activities, new features were released at a faster pace and the team was able to iterate faster on the design of features.

As the current implementation of the design system can be considered to be in its early stages of maturity (Beck, 2017; Somers, 2015), there are still some challenges left to be addressed. Most elements of the design system are left insufficiently documented. While we made efforts to document the design system using written documents, this turned out to be very time consuming due to the rapid pace of iterations. Therefore, limited documentation of design decisions during the design and development process of the design system's assets is currently included in the code files. This keeps documentation, design, and code closely connected, but fails to offer a holistic overview of the design systems and makes the documentation of overarching design patterns that include multiple assets challenging. Although this didn't cause any bottleneck in the development process so far, this approach might fail to provide sufficient clarity to new members that are being introduced to the team and might slow down future development as the system grows larger. The design system could benefit from a documentation tool that automatically generates use cases, sample code, and annotations.

Better documentation of overarching design patterns will further increase the consistency of the usage of UI components and interaction design patterns across products. The initial structure of the design system was based on the Atomic Design principles, leading to an abundance of modular UI components and a project structure and naming that was considered to be both limiting and confusing for developers. This would lead to an incoherent usage of UI components, opening the door for inconsistent UX and essentially defeating the purpose of having a design system.

Another challenge that was not fully addressed is the need for governance of the design system. Due to the fast development pace of the organization, the system is being designed and developed in parallel. While it's very positive that multiple team members contribute to the system, it also makes it hard to track changes and predict the impact of changes being made to UI components. Since UI components are being used so interconnectedly, updates to design components propagate to all screens that make use of that component. The majority of large organizations have a dedicated product team that governs the (design) quality of the design system, with the team being the only one responsible for developing and maintaining the design system. Additionally, Suarez et al. (2018) recommend the introduction of both unit and visual regression testing in order to release components with more confidence.

8 Conclusion

This report demonstrates the usage of a design system to reduce UX debt at startups. This design system was iteratively designed and developed based on user-centered activities. The design and development of the design system described in this report paves the way for a more systematic and scalable approach to design at The Cirqle. The usage of a single source of truth for all design elements (design tokens, UI elements, and UI patterns) promotes consistency throughout the interface and helps to reduce redundancies in design and engineering implementations. This increase in efficiency resulted in a more effective design process that allowed the team to reduce the time to market, design features more iteratively through user-centered design efforts and pay back UX debt over time.

Unintentional UX debt, as described earlier in this report, was addressed with additional user research and a more lean design process (see Gothelf, 2016). The resources saved through a more efficient design process freed up valuable resources to conduct user research, leading to a better understanding of the end users. The findings of these research efforts could be put into practice more easily by utilizing a readily available component library; functional prototypes and assumption could be evaluated by end users through the creation of MVPs. Furthermore, when MVPs were received positively, the design and development required little additional effort due to the reuse of production ready UI components. The amount of intentional debt, however, is often inherently reduced through the reuse of the assets of the design system. Still, when UX debt occurs the team has a more systematic way to resolve it. Through the reuse of UI components, we described earlier that components are constantly undergoing revisions. For these revisions to be successful, however, they need to be based on user-centered design activities.

9 References

Alexander, C., Ishikawa, S., Silverstein, M., & Jacobson, M. (1977). A Pattern Language. New York, NY: Oxford Univ. Press.

Andreesen, M. (2007). Product/Market Fit. Retrieved August 29, 2018, from https://web.stanford.edu/class/ee204/ProductMarketFit.html

Bailey, D. (2016). What is GEL? Retrieved September 10, 2018, from https://www.bbc.co.uk/gel/articles/what-is-gel

Barry, N. (2012). Forms are a Conversation. Retrieved September 1, 2018, from https://uxmag.com/articles/forms-are-a-conversation

Beck, C. (2017). The Path to Design System Maturity. Retrieved September 12, 2018, from https://medium.com/ux-power-tools/the-path-to-design-system-maturity-d403daba692a

Blank, S. G. (2006). The Four Steps to the Epiphany.

Bricklin, D. (2004). Software That Lasts 200 Years. Retrieved from http://www.bricklin.com/200yearsoftware.htm

Brooke, J. (1996). SUS: A Quick and Dirty Usability Scale. In: P.W. Jordan, B. Thomas, B.A. Weerdmeester & I.L. McClelland (Eds.), Usability Evaluation in Industry.

Buttiglieri, R. (2012). How UX Evolves At Companies: A New Look at Maturity Models. Retrieved July 10, from https://www.slideshare.net/UPABoston/ux-maturity-modelbuttiglieri.

Chen, A. (2011). When has a consumer startup hit product/market fit? Retrieved September 4, 2018, from https://andrewchen.co/when-has-a-consumer-startup-hit-productmarket-fit/

Chen, A. (2015). Product design debt versus Technical debt. Retrieved September 4, 2018, from https://andrewchen.co/product-design-debt-versus-technical-debt/

Chenais, L. (2018). Design tokens for dummies – UX Collective. Retrieved September 1, 2018, from https://uxdesign.cc/design-tokens-for-dummies-8acebf010d71

Connolly, E. (2018). The full stack design system - Inside Intercom. Retrieved September 10, 2018, from https://www.intercom.com/blog/the-full-stack-design-system/

Cunningham, W. (1992). The WyCash portfolio management system. In Addendum to the proceedings on Object-oriented programming systems, languages, and applications (Addendum) - OOPSLA '92. ACM Press. https://doi.org/10.1145/157709.157715

Curtis, B., Krasner, H., & Iscoe, N. (1988). A field study of the software design process for large systems. Communications of the ACM, 31(11), 1268–1287. https://doi.org/10.1145/50087.50089

Curtis, N. (2010). So You Wanna Build a Library, Eh? Retrieved September 1, 2018, from http://boxesandarrows.com/so-you-wanna-build-a-library-eh/

Curtis, N. (2016). Tokens in Design Systems. Retrieved September 15, 2018, from https://medium.com/eightshapes-llc/tokens-in-design-systems-25dd82d58421

Curtis, N. (2017). Defining Design Systems. Retrieved September 10, 2018, from https://medium.com/eightshapes-llc/defining-design-systems-6dd4b03e0ff6

Curtis, N. (2017a). Designing a Systems Team. Retrieved September 1, 2018, from https://medium.com/eightshapes-llc/designing-a-systems-team-d22f27a2d81d

Curtis, N. (2017b). Defining Design Systems. Retrieved September 11, 2018, from https://medium.com/eightshapes-llc/defining-design-systems-6dd4b03e0ff6

Curtis, N. (2017c). Starting a Design System. Retrieved September 11, 2018, from https://medium.com/eightshapes-llc/starting-a-design-system-6b909a578325

Dunwoody, K., Teague Rector, S. (2015). UX Debt in the Enterprise: A Practical Approach. User Experience Magazine, 15(1). Retrieved July/August, 2018, from http://uxpamagazine.org/ux-debt-in-the-enterprise/

Ellis, S. (2009). The Startup Pyramid. Retrieved September 4, 2018, from http://www.startup-marketing.com/the-startup-pyramid/

Fromstack. (2015). Form Conversion Report. Retrieved from https://www.formstack.com/report/form-conversion-2015.

Frost, B. (2013). Atomic design.

Gale, S. (1996). A collaborative approach to developing style guides. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems Common Ground - CHI 96. doi:10.1145/238386.238572

Gardner, O. (2013). How To Optimize Contact Forms For Conversions. Retrieved from https://unbounce.com/conversion-rate-optimization/how-to-optimize-contact-forms/

Giacoppo, S. (2013). User Experience in Startups, Part I: Challenges and Realities. Retrieved September 1, 2018, from https://www.uxmatters.com/mt/archives/2013/02/user-experience-in-startups-part-i-challenges-and-realities.php

Gibbons, S. (2018). Using Prioritization Matrices to Inform UX Decisions. Retrieved September 4, 2018, from https://www.nngroup.com/articles/prioritization-matrices/

Gothelf, J. (2016). Lean ux. Place of publication not identified: OReilly Media, Inc, Usa.

Griffiths, R., Pemberton, L., Borchers, J., & Stork, A. (2000). Pattern languages for interaction design. In CHI '00 extended abstracts on Human factors in computing systems - CHI '00. ACM Press.

Harker, S. D. P., Eason, K. D., & Dobson, J. E. (1993). The change and evolution of requirements as a challenge to the practice of software engineering. In [1993] Proceedings of the IEEE International Symposium on Requirements Engineering. IEEE Comput. Soc. Press. https://doi.org/10.1109/isre.1993.324847

Huffine, T. (2017). What is technical debt? And why does almost every startup have it? Retrieved August 24, 2018, from https://medium.freecodecamp.org/what-is-technical-debt-and-why-do-most-startups-have-it-9a54458daabf

Kirkpatrick, D. (2011). Testing Form Field Length Reduces Cost-Per-Lead By $10.66. Retrieved from https://marketingexperiments.com/lead-generation/lead-generation-testing-form-field-length-reduces-cost-per-lead-by-10-66

Knight, A. (2016). Design debt. Retrieved August 23, 2018, from https://www.invisionapp.com/blog/design-debt/

Kholmatova, A. (2017), Design Systems: A practical guide to creating design languages for digital products. Smashing Media.

Kruchten, P., Nord, R. L., & Ozkaya, I. (2012). Technical Debt: From Metaphor to Theory and Practice. IEEE Software, 29(6), 18–21. https://doi.org/10.1109/ms.2012.167

Lambert, S. (2016). Styling Web Components Using A Shared Style Sheet. Retrieved from https://www.smashingmagazine.com/2016/12/styling-web-components-using-a-shared-style-sheet/

Lauesen, S., & Musgrove, M. P. (2005). User Interface Design - A Software Engineering Perspective

Loomer, D. (n. d.). Design Systems. Retrieved from https://projekt202.com/design-systems/.

Loranger, H. (2015). Radical Redesign or Incremental Change? Retrieved September 1, 2018, from https://www.nngroup.com/articles/radical-incremental-redesign/

Madsen, R. (2017). The User Experience Of Design Systems. Retrieved September 11, 2018, from https://runemadsen.com/talks/uxcampcph/

Martin, D. (2018). Getting Executive Buy-In For Your Design System. Retrieved from https://www.invisionapp.com/blog/getting-executive-ok-design-system/

Meier, D. (2017). Atomic Design Coding Sanity: Naming Conventions for Easy Recognition. Retrieved September 11, 2018, from https://medium.com/@DaveMeier/simple-rules-for-atomic-design-coding-sanity-cfc694474dda

Moffet, J. (2016). Eliminate UX Debt. Improving Consistency and Usability.

Monus, A. (2016). How to Recognize & Manage UX Debt. Retrieved July 12, 2018, from https://www.hongkiat.com/blog/managing-ux-debt/

Nielsen, J., & Molich, R. (1990). Heuristic evaluation of user interfaces. In Proceedings of the SIGCHI conference on Human factors in computing systems Empowering people - CHI '90. ACM Press.

Nielsen, J. (1995). Severity Ratings for Usability Problems. Retrieved July 12, 2018, https://www.nngroup.com/articles/how-to-rate-the-severity-of-usability-problems/

Nielsen, J. (2006). Corporate UX Maturity: Stages 1-4. Retrieved July 10, 2018, from https://www.nngroup.com/articles/ux-maturity-stages-1-4/

Nielsen, J. (2006). Corporate UX Maturity: Stages 5-8. Retrieved July 10, 2018, from https://www.nngroup.com/articles/ux-maturity-stages-5-8/

Nilsson, H., & Petersson, L. (2013). How to Manage Technical Debt in a Lean Startup.

Priego, P. (2017). CSS Evolution: From CSS, SASS, BEM, CSS Modules to Styled Components. Retrieved from https://medium.com/@perezpriego7/css-evolution-from-css-sass-bem-css-modules-to-styled-components-d4c1da3a659b

Preston-Werner, T. (n.d.). Semantic Versioning 2.0.0. Retrieved September 15, 2018, from https://semver.org.

Rao, R. (2018). 18 UX Design Tips for Registration and Login Forms – UX Planet. Retrieved September 11, 2018, from https://uxplanet.org/18-ux-design-tips-for-registration-and-login-forms-f897557358ba

React. (2018). A JavaScript library for building user interfaces. Retrieved September 1, 2018, https://reactjs.org.

Ries, E. (2009, July 23). Retrieved September 1, 2018, from https://www.youtube.com/watch?v=E4ex0fejo8w

Rohde, S. (2014). Living Design System. Retrieved September 11, 2018, from https://medium.com/salesforce-ux/living-design-system-3ab1f2280ef7

Ruissalo, M. (2018). Operating a design system in a large software company.

Saarinen, K. (2016). Creating the Airbnb Design System. Retrieved from https://karrisaarinen.com/posts/building-airbnb-design-system/

Sakhardande, P., Thanawala, R. (2014). UX Maturity Model: From Usable to Delightful. User Experience Magazine, 14(3). Retrieved from http://uxpamagazine.org/ux-maturity-model/

Seckler, M., Heinz, S., Bargas-Avila, J. A., Opwis, K., & Tuch, A. N. (2014). Designing usable web forms. In Proceedings of the 32nd annual ACM conference on Human factors in computing systems - CHI '14. ACM Press. https://doi.org/10.1145/2556288.2557265

Scott, B., & Malone, E. (2006). Yahoo! Design Pattern Library Released. Retrieved September 1, 2018, from https://yuiblog.com/blog/2006/02/13/yahoo_patterns_released/

Shneiderman, B. (1986). Seven plus or minus two central issues in human-computer interaction. ACM SIGCHI Bulletin, 17(4), 343–349.

Somers, M. (2015). A Maturity Model for Design Systems. Retrieved September 12, 2018, from https://medium.com/slalom-engineering/a-maturity-model-for-design-systems-93fff522c3ba

Snyder, B. (2018). Product Design in a Sales-driven organization – UX Collective. Retrieved September 1, 2018, from https://uxdesign.cc/product-design-in-a-sales-driven-organization-43fcdd6f14eb

Suarez, M., Anne, J., Sylor-Miller, K., Mounter, D. and Stanfield, R. (2017). Design Systems handbook. Retrieved September 1, 2018, from https://www.designbetter.co/design-systems-handbook

Tang, J. C., Liu, S. B., Muller, M., Lin, J., & Drews, C. (2006). Unobtrusive but invasive. In Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work - CSCW '06. ACM Press. https://doi.org/10.1145/1180875.1180948

Tidwell, J. (2005). Designing Interfaces: Patterns for Effective Interaction Design.

Treder, M. (2017). Design Systems Sprint 1: The Interface Inventory. Retrieved from https://medium.com/@marcintreder/design-systems-sprint-1-the-interface-inventory-1f78d376e49a

UXPin. (2018). Adele, The repository of publicly available design systems and pattern libraries. Retrieved April 6, 2018, from https://adele.uxpin.com.

van Welie, M., & Traetteberg, H. (2000). Interaction Patterns in User Interfaces. In 7th Pattern Languages of Programs Conference.

Vue.js (2018). Composing with Components. Retrieved September 1, 2018, https://vuejs.org/v2/guide/.

Vijayashanker, P. (2016). Differentiating with Design. Retrieved September 1, 2018, from https://articles.uie.com/differentiating-with-design/

Walrack, J. (2018). 8 Top Design Systems in 2018. Retrieved from https://blog.brightscout.com/8-top-design-systems-2018/

Whitenton, K. (2016). Website Forms Usability: Top 10 Recommendations. Retrieved September 11, 2018, from https://www.nngroup.com/articles/web-form-design/

Wright, A. (2014). User Experience Debt: Creating awareness and acting on missed opportunities. Retrieved August 28, 2018, from https://www.slideshare.net/andr3wjwright/ias14-uxdebt

Wroblewski, L. (2008) Web Form Design, Filling in the blanks.

Wroblewski, L. (2018). An Event Apart: Scenario-Driven Design Systems.

Appendix 1: Comparison of Design Systems

Style Guide

Organisation	Name Design System	Brand Guidelins	Colors	Typography	Icons	Spacing	Illustrations	Animations	Voice and Tone	Design Tokens
Salesforce	Lightning	no	yes	yes	yes	yes	yes	yes	yes	yes
Shopify	Polaris	no	yes	yes	yes	yes	yes	no	yes	yes
WeWork	Plasma	no	no	no	yes	no	no	no	no	no
Atlassian	Atlassian Design Language	yes	yes	yes	yes	no	yes	no	yes	no
IBM	Carbon	no	yes	yes	yes	yes	no	yes	yes	no

UI Components

Organisation	Name Design System	UI Components	Code Depth	Framework	CSS	Automated testing	UI Kit
Salesforce	Lightning	yes	HTML/CSS/JS	React	Sass	yes	yes, Sketch
Shopify	Polaris	yes	HTML/CSS/TS	React	Sass PostCSS	yes	yes, Sketch
WeWork	Plasma	yes	HTML/CSS/JS	React	Sass	no	no
Atlassian	Atlassian Design Language	yes	HTML/CSS/TS	React	Less postCSS	yes	yes, Sketch
IBM	Carbon	yes	HTML/CSS/JS	React, Vanilla JS	Sass	yes	yes, Sketch

Documentation

Organisation	Name Design System	Documentation website	Design Principles	Code Playground
Salesforce	Lightning	yes	yes	no
Shopify	Polaris	yes	yes	no
WeWork	Plasma	yes	no	yes
Atlassian	Atlassian Design Language	yes	no	yes
IBM	Carbon	yes	no	yes

Appendix 2: User Stories Content Approval

- As an influencer, I want to know what is expected from me
- As an influencer, I want to upload my content for approval through the platform
- As an influencer, I want to receive updates when I need to take action
- As an influencer, I want to view the feedback on the content I created
- As an influencer, I want to adjust my content if needed
- As an influencer, I want to be able to see what the content will look like when it gets published
- As a client, I want to view the content that is produced
- As a client, I want to approve content that fits the criteria specified up front
- As a client, I want to be able to request revisions if needed
- As a client, I want to get notified when content changes
- As a client, I want to get notified when updates to the content are made
- As a client, I want to be able to download the content that was created
- As a campaign manager, I want to view the content that is produced
- As a campaign manager, I want to keep an overview of what the various stages of the content approval process per item
- As a campaign manager, I want to be able to communicate with the client
- As a campaign manager, I want to be able to communicate with the influencer
- As a campaign manager, I want to leave feedback on the content
- As a campaign manager, I want to be able to request a revision from the influencer when the content clearly is inadequate
- As a campaign manager, I want to be able to download the content that was produced