FH AACHEN
UNIVERSITY OF APPLIED SCIENCES

FH Aachen

Fachbereich Elektrotechnik und Informationstechnik

Studiengang Media and Communications for Digital Business

Bachelorarbeit

Component Sprints: Entwicklung eines neuartigen Ansatzes zur Optimierung des Übergangs vom Prototypen zum MVP

vorgelegt von **Leo Bernard**

Matrikel-Nr. **3069756**

Referent: Prof. Dr. rer. pol. Marco Motullo

Datum: 2. Oktober 2019

Besonderer Dank gilt René Nauheimer, Daniel Wirtz, Moritz Gunz und Marcus Weiner

Eidesstattliche Erklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig angefertigt habe. Es wurden nur die in der Arbeit ausdrücklich benannten Quellen und Hilfsmittel benutzt. Wörtlich oder sinngemäß übernommenes Gedankengut habe ich als solches kenntlich gemacht.

Ort, Datum	Leo Bernard

Disclaimer

In dieser Arbeit wird aus Gründen der besseren Lesbarkeit teilweise das generische Maskulinum verwendet. Weibliche und anderweitige Geschlechteridentitäten sind dabei ausdrücklich mitgemeint, soweit es für die Aussage erforderlich ist.

Bei Zitaten ohne Angabe einer absoluten Seitenzahl handelt es sich um Webseiten oder E-Books, deren Seitenzahlen von der Bildschirmgröße abhängig sind. Bei diesen Quellen werden alternativ Kapitel- und Abschnittsangaben verwendet.

Bei Abbildungen ohne Quellenangaben handelt es sich um Eigenkreationen, die im Rahmen der Bachelorarbeit erstellt wurden.

Die Arbeit sowie alle im Rahmen der Arbeit entstandenen Ergebnisse sind open-source und können online[1] eingesehen werden.

[1] `https://github.com/leolabs/bachelor`

Inhalt

Abkürzungsverzeichnis

CEO: Chief Executive Officer.

CI: Corporate Identity.

HTML: HyperText Markup Language.

MVP: Minimum Viable Product.

OBS: Open Broadcaster Software.

RAD: Rapid Application Development.

UI: User Interface.

UX: User Experience.

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Der Markt für Apps und andere digitale Produkte wächst schnell. So stieg beispielsweise die Anzahl der verfügbaren Apps im App Store von ca. 61.000 im Jahr 2009 auf über 4 Mio. im Jahr 2019 (PocketGamer.biz 2019). Dazu nutzen im Jahr 2019 geschätzt 3,3 Mrd. Menschen weltweit ein Smartphone (Newzoo 2018). Auf ProductHunt, einer Plattform, auf der Entwickler ihre digitalen Produkte vorstellen können, wurden von 2017-2019 über 19.000 Produkte veröffentlicht. Das sind ca. 614 Produkte pro Monat oder ca. 20 Produkte pro Tag (siehe Abb. 11.1). Die zeitlichen Anforderungen an Entwicklerteams nehmen dadurch zu.

Methoden wie iterative und agile Softwareentwicklung, Scrum und Rapid Application Development (RAD) helfen Entwicklerteams dabei, effizienter zu arbeiten, auch wenn sich beispielsweise die Anforderungen an das Produkt häufig ändern (Larman 2004, S. 27). Weitere Methoden wie beispielsweise Minimum Viable Products (MVPs) werden dazu eingesetzt, das Potential eines Produktes zu testen, ohne zu viel Zeit in die Entwicklung dieses zu investieren (Ries 2009). So können Produkte schneller an den Markt gebracht werden.

Für die Ausarbeitung von Ideen werden Methoden wie Design Sprints und Design Thinking eingesetzt. Vor allem Design Sprints haben das Ziel, in einer möglichst kurzen Zeit Ergebnisse zu erzielen (Poguntke 2019, S. 37). Bei Design Sprints, die von Crisp Studio GmbH, einer Design Sprint Agentur aus Aachen, durchgeführt wurden, hat sich diese Methode bewährt, jedoch verlor die Entwicklung des

Produkts nach dem Sprint bei Kunden schnell an Momentum. Dies beeinflusst den Mehrwert des Design Sprints negativ.

Zwei große Aspekte der Effizienz von Design Sprints sind die genaue Struktur und das unabhängige Arbeiten an Aufgaben. Diese Arbeit beschäftigt sich mit der Frage, ob es möglich ist, diese Aspekte auf einen neuen Sprint zu übertragen, der auf den Design Sprint folgt und mit der selben Effizienz die Grundlage für die Entwicklung eines MVPs aufbauen kann.

2 Zielsetzung

Ziel dieser Bachelorarbeit ist es, ein Konzept für einen Sprint zu entwickeln, der den Übergang vom Prototyp aus einem Design Sprint zum fertigen Minimum Viable Product (MVP) optimal überbrückt.

Im Rahmen dieser Bachelorarbeit soll die folgende Frage beantwortet werden:

> Wie kann aus einem fertigen Prototyp in einem festen, kurzen Zeitraum eine Sammlung an Design Komponenten entwickelt werden, die Entwicklern des MVPs die Arbeit erleichtert?

3 Begründung

Design Sprints wurden 2012 von Jake Knapp entwickelt (Knapp u. a. 2016, S. 5) und werden dazu eingesetzt, innerhalb von einer Woche eine Produktidee zu einem Prototyp auszuarbeiten und an potentiellen Nutzern zu testen. Design Sprints können für jede Art von Produkten – physisch und digital – eingesetzt werden (Knapp u. a. 2016, S. 9). Diese Arbeit konzentriert sich jedoch auf den Einsatz von Design Sprints im Softwarebereich, also beispielsweise für Apps und Websites.

Nach einem erfolgreichen Design Sprint entwickeln Firmen häufig auf Basis des dort entwickelten Prototyps ein Minimum Viable Product, das sie auf den Markt bringen können (DockYard, Inc 2017).

Die in dieser Bachelorarbeit ausgearbeitete Methode, die im Folgenden Component Sprint genannt wird, soll Entwicklern die Implementierung des MVP vereinfachen, indem sie bereits alle nötigen UI-Komponenten in einer strukturierten Bibliothek zur Verfügung gestellt bekommen. So können sie sich bei der Entwicklung des MVP mehr auf die Implementierung der Logik konzentrieren.

4 Vorgehen

Um das Konzept für Component Sprints zu entwickeln, werden zuerst bestehende Grundlagen erörtert und relevante Konzepte und Frameworks vorgestellt.

Basierend auf diesen Frameworks wird die erste Version des Component Sprints entwickelt. Darauf folgend wird ein empirischer Test des entwickelten Konzepts in Form eines Design Sprints mit anschließendem Component Sprint durchgeführt und die Ergebnisse sowie Erkenntnisse zusammengefasst.

Um das Konzept weiter zu verfeinern, wird eine qualitative Studie in Form von vier Interviews mit Personen aus relevanten Bereichen durchgeführt, in denen das Konzept, sowie bestehende Ergebnisse, vorgestellt werden und Einschätzungen über das Potential und mögliche Ausarbeitungsmöglichkeiten des Sprints eingeholt werden.

Zuletzt werden Schritte zur Weiterentwicklung des Konzepts angerissen und dargelegt, wie Component Sprints in Zukunft ausgebaut werden können.

5 Theoretische Grundlagen / Einordnung

In den letzten 20 Jahren hat sich der Prozess der Softwareentwicklung und -veröffentlichung bzw. -verbreitung stark verändert. Vor 20 Jahren wurde Software hauptsächlich auf CDs verkauft. Ein normaler Veröffentlichungszyklus lag bei mehreren Monaten. Fehler in der Software konnten erst in der nächsten Version behoben werden, Feedback von Benutzern konnte erst Monate später umgesetzt werden (*Jake Knapp – The Design Sprint: One Small Change to Create a Culture of Innovation* 2019, 4:06).

Durch die Verbreitung des Internets wurde es einfacher, neue Versionen regelmäßiger zu veröffentlichen. Heute ist es normal, dass wöchentlich neue Updates für Software veröffentlicht werden. Analytics Tools wie Google Analytics[1] und Mixpanel[2] ermöglichen es Firmen, genaue Einblicke zum Benutzerverhalten in ihrer Software zu erlangen. So können potentielle Problemstellen in Programmen genauer identifiziert und behoben werden (*Jake Knapp – The Design Sprint: One Small Change to Create a Culture of Innovation* 2019, 14:23).

Die folgenden Kapitel konzentrieren sich auf die heutige Entwicklung von Apps und Websites. Um eine theoretische Grundlage zu schaffen, werden in diesem Kapitel

[1] `https://analytics.google.com`
[2] `https://mixpanel.com/`

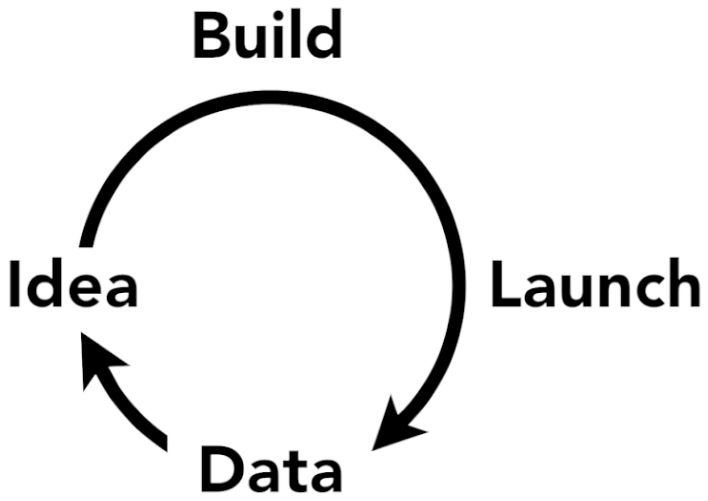

Abb. 5.1: Klassischer Entwicklungszyklus nach Jake Knapp [Knapp (2018), S. 65]

Konzepte und Frameworks vorgestellt, die für die Entwicklung des Component Sprints relevant sind.

5.1 Design Sprints

Ein üblicher Zyklus in der heutigen Entwicklung von Apps und Websites kann in vier Schritte unterteilt werden (siehe Abb. 5.1):

- **Ideate:** Generierung und Ausarbeitung von Ideen
- **Build:** Umsetzung der Ideen in Software
- **Launch:** Veröffentlichung der Software auf dem Markt
- **Data:** Sammeln von Insights und User Feedback

Bei der Entwicklung von Apps ist es normalerweise notwendig, diesen Zyklus mehrere Male zu durchlaufen, um eine optimale Version zu erhalten. Dabei ist

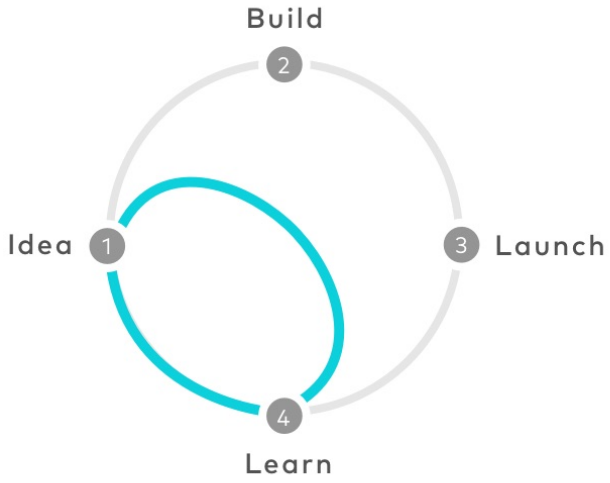

Abb. 5.2: Design Sprint Zyklus [Google Ventures (2019)]

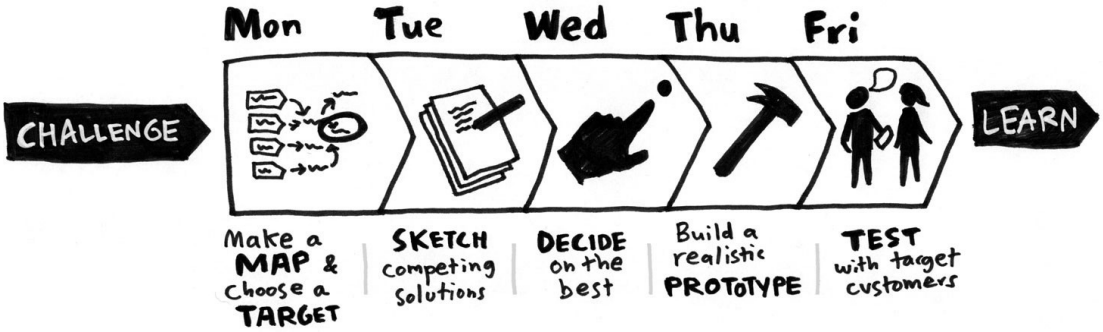

Abb. 5.3: Struktur des gesamten Design Sprints [Knapp u. a. (2016), S. 17]

vor allem der Build-Schritt zeitaufwändig. Wenn der Launch nicht erfolgreich ist, wurden die dafür benötigten Ressourcen verschwendet.

Der Design Sprint ist eine Methode, die versucht, diesen Zyklus zu optimieren. Hierbei werden die Build- und Launch-Schritte durch Prototyping und User Testing ersetzt, sodass User Feedback schneller gesammelt werden kann (siehe Abb. 5.2). Die Struktur eines Design Sprints wird von Jake Knapp in seinem Buch “Sprint: How to solve big problems and test new ideas in just five days” (Knapp u. a. 2016) erläutert.

Ein Design Sprint dauert in der Regel fünf Tage – Montag bis Freitag (siehe Abb. 5.3) – und wird vor Ort mit einem interdisziplinären Team von vier bis sieben Personen und einem Moderator durchgeführt. Die Tage sind zeitlich fest strukturiert und jegliche Form von Ablenkung – z.B. Handys oder Laptops – ist im Sprintraum nicht erlaubt. Ziel dieser Maßnahmen ist es, dass sich das Team fokussiert mit den Aufgaben auseinandersetzen kann (Knapp u. a. 2016, S. 41). Im Sprint gilt das Prinzip “Together Alone”: Längere Aufgaben werden in Einzelarbeit durchgeführt, über Ergebnisse stimmt das Team gemeinsam ab. Erarbeitete Informationen werden systematisch auf Post-its festgehalten, sodass jedes Teammitglied jederzeit darauf zugreifen kann (Knapp u. a. 2016, p. 20).

Generell wird in Design Sprints auf offene Diskussionen verzichtet. Diese werden als problematisch angesehen, da sie Teilnehmern mit einer starken Meinung zu viel Gewichtung und Einfluss auf die Entscheidungen geben (Kahneman 2013, K. 7). Stattdessen werden Entscheidungen per Abstimmung, meist mit Hilfe von Klebepunkten, getroffen. Der CEO, Gründer, Produktmanager oder Head of Design hat als sogenannter Decider[3] in Abstimmungen eine größere Entscheidungskraft (Knapp u. a. 2016, S. 34).

Am Montag wird die Problemstellung definiert. Vormittags legt das Team nach einer kurzen Einführung das Langzeitziel des Projekts und Fragen, die für den Sprint interessant sein könnten, fest. Darauf folgend wird eine Abbildung der erwarteten Customer Journey[4] angelegt (Knapp u. a. 2016, S. 65).

Nachmittags werden sogenannte Experteninterviews durchgeführt. Ziel dieser Interviews ist das Sammeln und Festhalten von Wissen, das häufig auf einzelne Teammitglieder verteilt ist (Knapp u. a. 2016, S. 68). Während der Interviews legen alle Mitglieder in Stille Notizen auf Post-its in Form von “How Might We” (“Wie

[3] Übersetzt: Entscheidende Person

[4] Eine Customer Journey (dt. Kundenreise) beschreibt die Schritte, die ein potentieller Kunde im Zusammenhang mit einem Produkt durchläuft, bis er ein definiertes Ziel erreicht (Mattscheck 2019).

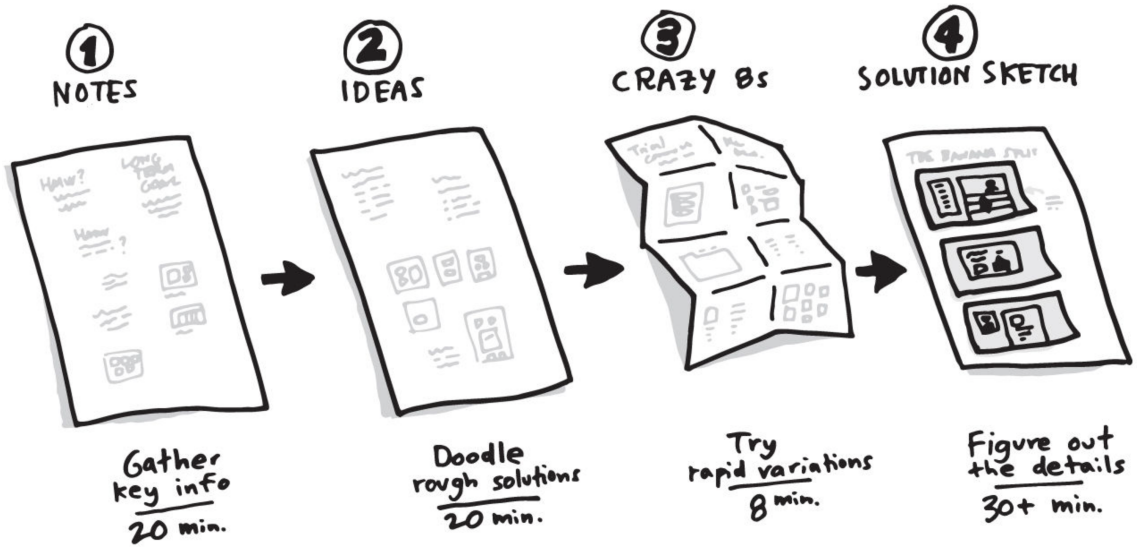

Abb. 5.4: Struktur des Dienstages [Knapp u. a. (2016), S. 109]

können wir") Fragen an. Eine Frage kann beispielsweise lauten: "Wie können wir das User-Onboarding vereinfachen?" oder "Wie können wir unser Produkt freundlich wirken lassen?". Diese Fragen werden daraufhin an einer Wand sortiert, per Abstimmung priorisiert und die wichtigsten in die Customer Journey eingeordnet (Knapp u. a. 2016, S. 80-88).

Am Dienstag sucht das Team online nach Konzepten, Komponenten und anderen Beispielen, die im Kontext der am Montag definierten Fragen und Ziele relevant sein könnten und präsentiert diese kurz. Für jede Komponente wird ein Post-it erstellt. Diese Post-its dienen als Inspiration für die spätere Erstellung von Lösungsansätzen (Knapp u. a. 2016, S. 98-100).

Basierend auf diesen Ideen werden nun von allen Teammitgliedern Skizzen und Notizen angelegt. Diese werden über weitere Übungen zu einem sogenannten Solution Sketch pro Person ausgearbeitet. Diese Sketches bilden in maximal drei Schritten detailliert die Interaktion der Nutzer mit dem Produkt ab und werden den anderen Teammitgliedern gegenüber bis zum nächsten Tag nicht offengelegt (Knapp u. a. 2016, S. 114-118).

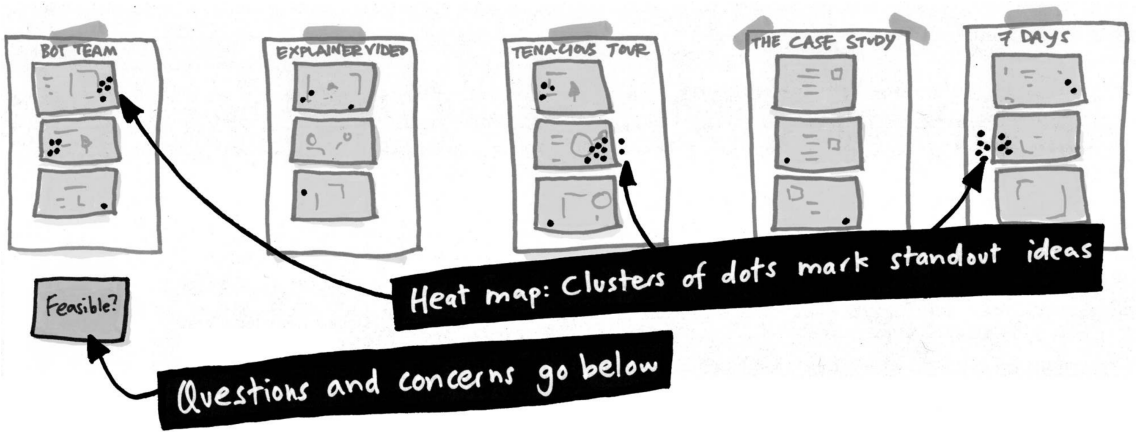

Abb. 5.5: Ergebnis der Heat Map Übung

Am Mittwoch entscheidet sich das Team für den besten Solution Sketch. Diese Entscheidung wird in fünf Schritten durchgeführt: “Art museum”, “Heat map”, “Speed critique”, “Straw poll” und “Supervote” (Knapp u. a. 2016, S. 131).

Im ersten Schritt werden die Solution Sketches, die am Vortag erstellt wurden, aufgedeckt und nebeneinander aufgehängt. Die Sketches sind anonym, um eventuellen persönlichen Einflüssen entgegenzuwirken (Knapp u. a. 2016, S. 132).

Darauf folgend markiert jedes Teammitglied auf jedem Solution Sketch mit Klebepunkten interessante Konzepte und Ideen. Fragen und Anmerkungen können per Post-it hinzugefügt werden. Dieser Schritt wird in Stille durchgeführt und gibt dem Team einen guten Überblick über interessante Bereiche in den Sketches (Knapp u. a. 2016, S. 132-135).

Im dritten Schritt wird jeder Solution Sketch drei Minuten lang besprochen. Die Besprechung folgt einer festen Struktur, damit sie konstruktiv bleibt und sich das Team nicht in Details verliert. Die Ergebnisse der Besprechungen werden auf Post-its über den Solution Sketches aufgehängt (Knapp u. a. 2016, S. 135-137).

Im vierten Schritt stimmt jedes Teammitglied für den Solution Sketch, den er am

besten findet, ab und erklärt seine Entscheidung in einem ein-minütigen Vortrag (Knapp u. a. 2016, S. 138-140).

Im letzten Schritt entscheidet sich der Decider für seine Favoriten, basierend auf den Abstimmungsergebnissen der anderen Teammitglieder, den Sprintfragen und dem festgelegten Langzeitziel. Diese Entscheidung ist absolut und die Grundlage für das folgende Prototyping (Knapp u. a. 2016, S. 140-142). Wenn mehr als ein Favorit ausgewählt wurde, wird gemeinsam entschieden, ob alle favorisierten Solution Sketches in einen Prototyp übernommen werden oder ob mehrere Prototypen erstellt werden sollen (Knapp u. a. 2016, S. 145).

Am Mittwoch Nachmittag wird basierend auf den gewählten Solution Sketches ein oder mehrere Storyboards für den Prototyp auf einem Whiteboard erstellt. Dies geschieht im Team, wobei der Decider auch hier das letzte Wort hat. Das Storyboard dient als Grundlage für den Prototyp und sollte so detailliert sein, dass der User Flow, also die Schritte, die der Benutzer im Prototyp durchlaufen wird, klar ist (Knapp u. a. 2016, S. 148-156). Zusammengefasst gilt in diesem Schritt das Mantra “When in doubt, take risks.” (Knapp u. a. 2016, S. 156): Der Design Sprint ist eine gute Möglichkeit, um riskante Entscheidungen zu testen.

Der Donnerstag ist dem Prototyping gewidmet. Hier wird auf Basis des erstellten Storyboards ein Klick-Dummy, bzw. Throwaway Prototype erstellt. Dieser Prototyp soll sich für Testnutzer am Freitag realistisch anfühlen, muss aber nur die Funktionalität abbilden, die am Freitag wirklich getestet wird. Je mehr sich der Prototyp wie ein echtes Produkt anfühlt, desto mehr werden die Testnutzer authentisch reagieren (Knapp u. a. 2016, S. 168-170).

Das Prototyping geschieht in Teamarbeit. So erstellen beispielsweise zwei Personen einzelne Komponenten oder Screens und eine Person kombiniert diese zu einem einheitlichen Prototyp. Zusätzlich ist eine Person für das Schreiben von

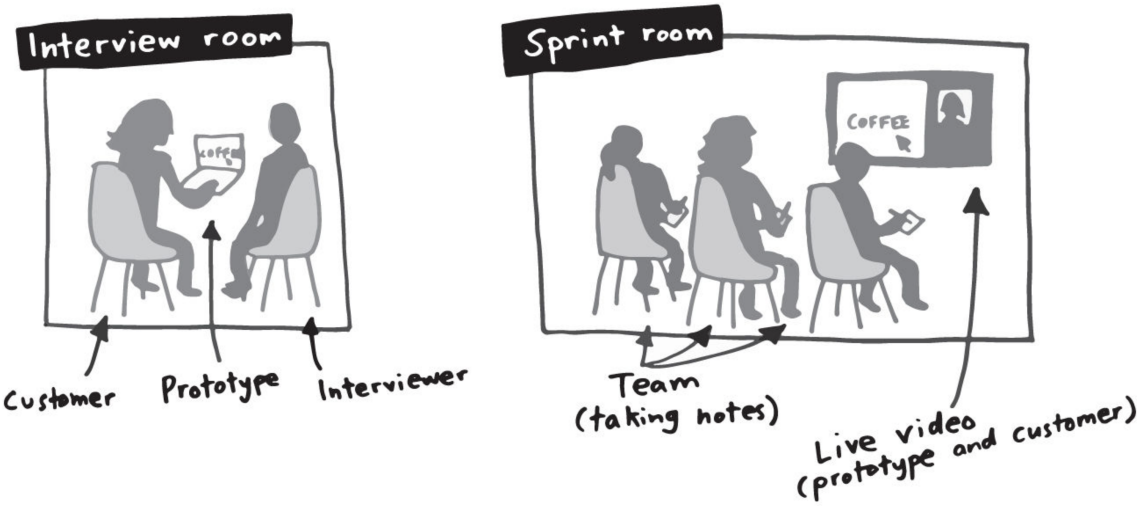

Abb. 5.6: Aufbau der Interviews [Knapp u. a. (2016), S. 203]

Texten und eine oder mehrere Personen für das Sammeln von Assets, wie Bilder und Icons, verantwortlich (Knapp u. a. 2016, S. 171).

Die Person, die am nächsten Tag die Interviews mit Testnutzern durchführen wird, konzipiert während des Prototypings ein Skript, mit dem die Nutzer durch den Prototypen geführt werden sollen (Knapp u. a. 2016, S. 171).

Am Freitag wird der Prototyp an fünf Nutzern getestet. Nach einer kurzen Einleitung und generellen Fragen zum Nutzer wird dieser durch den Prototyp geführt. Die interviewende Person versichert dem Nutzer dabei, dass nicht er, sondern das Produkt getestet wird. Er kann daher nichts falsch machen und es wird nicht vorausgesetzt, dass er alle Aufgaben verstehen oder ausführen kann (Knapp u. a. 2016, S. 207).

Interviewer und Testnutzer sitzen beim Interview zu zweit in einem Raum. Das Gespräch wird per Video an die anderen Mitglieder des Sprint-Teams übertragen, die in einem anderen Raum sitzen. Diese Personen machen sich während des Interviews in stiller Einzelarbeit auf Post-its Notizen zu dem Feedback des Testnutzers (Knapp u. a. 2016, S. 202-203).

Nach den Interviews werden die Notizen sortiert und gruppiert, sodass sich daraus Rückschlüsse über das weitere Vorgehen ergeben. Im Team wird nun mit Blick auf die am Montag definierten Sprintfragen und Langzeitziele über das weitere Vorgehen abgestimmt. Der Decider hat auch hier das letzte Wort (Knapp u. a. 2016, S. 216-222).

Design Sprints werden von vielen Unternehmen nicht zu 100% so durchgeführt, wie es in dem Buch *Sprint* des Erfinders Jake Knapp (Knapp u. a. 2016) vorgeschlagen wird. Die Design Sprint Agentur AJ&Smart[5] bietet beispielsweise einen Design Sprint in vier Tagen unter dem Namen “Design Sprint 2.0” an. Hier werden die ersten beiden Sprinttage in einen Tag zusammengeführt und die Übungen so angepasst, dass sie in der kürzeren Zeit durchgeführt werden können. So werden beispielsweise einzelne Experteninterviews in ein Gruppeninterview zusammengeführt, das auf 30 Minuten beschränkt ist (Smart u. a. 2018).

Crisp Studio orientiert sich am viertägigen Sprint von AJ&Smart. Dazu wird die Reihenfolge einiger Übungen getauscht, da sie so aus Sicht des Unternehmens mehr Sinn ergeben. Die Experteninterviews werden beispielsweise an den Anfang des ersten Sprinttages gelegt, sodass alle Teammitglieder bei den Sprintfragen und dem Langzeitziel bereits auf dem gleichen Stand sind. Zusätzlich enthalten die Sprints neue Übungen, wie z.B. Note-n-Map (siehe Cruchon 2019), die den Prozess weiter vereinfachen. In den Design Sprints von Crisp Studio sind nicht alle Teammitglieder an allen vier Tagen anwesend – die Teammitglieder des Kunden müssen hier nur an den ersten beiden Tagen teilnehmen. Dies vereinfacht die Organisation des Sprints erheblich, da die Kunden sich nicht eine ganze Woche freihalten müssen. Das Prototyping und Testing wird von der Agentur übernommen (Nauheimer 2019).

[5] https://ajsmart.com

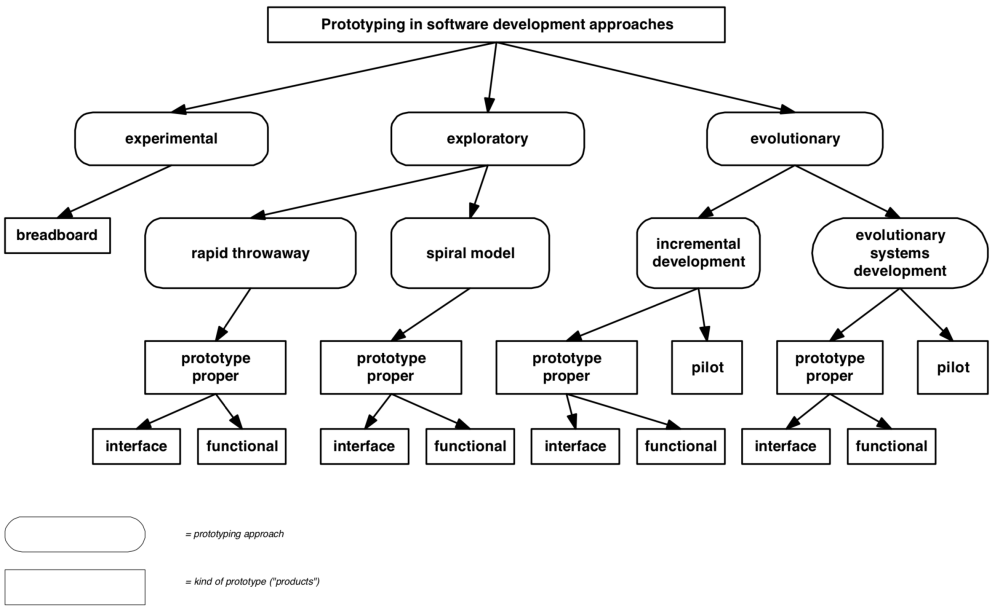

Abb. 5.7: Kategorisierung der Arten von Prototyping [Verner u. a. (o.D.), S. 7]

5.2 Prototyping

Prototypen sind in ihrer Grundform im Duden als “[. . .] zur Erprobung und Weiterentwicklung bestimmte erste Ausführung [. . .]“ definiert. Das Wort stammt vom Griechischen “prōtótypos”, übersetzt “ursprünglich”, ab (Dudenredaktion (o. J.) 2019). Prototypen werden dazu eingesetzt, Ideen mit minimalem Aufwand zu testen. Prototyping bezeichnet die Erstellung eines solchen Prototypen.

Im Kontext der Softwareentwicklung lässt sich Prototyping grob in drei Kategorien einteilen (siehe Abb. 5.7): Experimentelles, erforschendes und evolutionäres Prototyping (Verner u. a. o.D., S. 7). Diese Einteilung ist nicht absolut und die Begriffe variieren je nach hinzugezogener Quelle.

Der Prototyping-Schritt in Design Sprints ist ein Rapid Throwaway Prototyping oder kurz Throwaway Prototyping, das sich dem erforschenden Prototyping unterordnen lässt. Bei diesem Ansatz werden Prototypen mit Fokus auf Geschwindigkeit entwickelt. Die Qualität des Programmcodes und die Sorgfältigkeit der Entwicklung sind

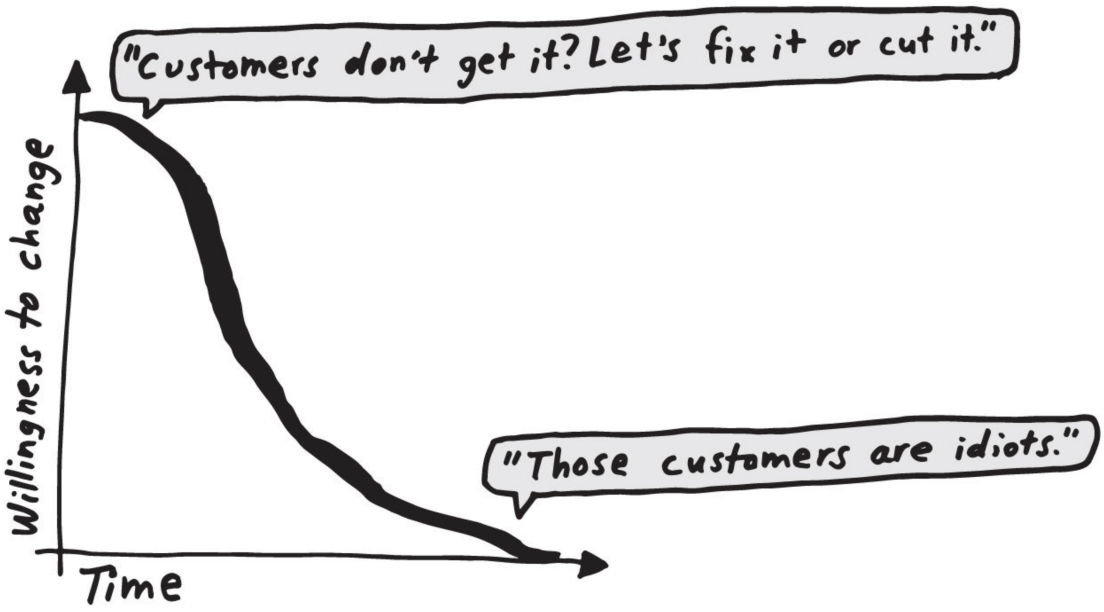

Abb. 5.8: [Knapp u. a. (2016), S. 168]

dabei nicht vorrangig, da der Prototyp nach der Evaluation nicht weiterentwickelt, sondern verworfen wird (Crinnion 1992, S. 18).

Durch diese Grundlage fällt es dem Team meist leichter, Ideen zu verwerfen, da es so noch nicht zu viel Zeit in deren Entwicklung investiert hat (Knapp u. a. 2016, S. 168). Abb. 5.8 stellt diesen Sachverhalt vereinfacht dar.

5.3 Minimum Viable Products

Ein Minimum Viable Product, kurz MVP, auf Deutsch “minimal überlebensfähiges Produkt“ (*Minimum Viable Product* 2019), ist laut Eric Ries ein Produkt, das dem Entwicklerteam mit dem geringsten Aufwand die maximale Menge an validiertem Wissen über seine Kunden bietet (Ries 2009).

MVPs werden eingesetzt, um zu erforschen, wie der Zielmarkt auf ein potentielles Produkt oder ein neues Feature bei bestehenden Produkten reagieren würde. So

wird das Risiko minimiert, Zeit in die Entwicklung eines Produktes oder Features zu investieren, das letztendlich nicht genutzt wird und/oder sich nicht verkaufen lässt.

Die Definition von “Minimum” im Kontext des MVP ist laut Ries nicht festgelegt. So kann der Entwicklungsaufwand je nach Produkt zwischen wenigen Tagen und mehreren Monaten liegen (Ries 2009).

Ähnlich wie im Fall des Begriffs “Prototyping” gibt es auch für das MVP verschiedene Definitionen. Der Begriff wurde erstmals 2001 von Frank Robinson als “[. . .] unique product that maximizes return on risk for both the vendor and the customer”[6] (Robinson 2001) definiert. Steve Blank definierte MVPs 2010 als “[. . .] Customer Development tactic to reduce engineering waste and to get product[s] in the hands of Earlyvangelists soonest”[7] (Blank 2010).

Ries’ Definition hat aktuell jedoch die größte Reichweite. Die meisten Definitionen bauen entweder auf seiner Definition oder auf der Definition von Steve Blank auf (Lenarduzzi u. a. 2016, S. 119).

5.4 Atomic Design

Atomic Design ist ein System zur hierarchischen Organisation und Strukturierung von UI Komponenten, das erstmals 2013 von Brad Frost auf seinem Blog vorgestellt und 2016 in dem Buch “Atomic Design” veröffentlicht wurde (Frost 2016, Foreword).

[6] Übersetzt: “einzigartiges Produkt, das den Return on Risk sowohl für den Lieferanten als auch für den Kunden maximiert”

[7] Übersetzt: “Kundenentwicklungs-Taktik, um technische Verschwendung zu reduzieren und das Produkt schnellstmöglich in die Hände von Earlyvangelists zu bekommen”

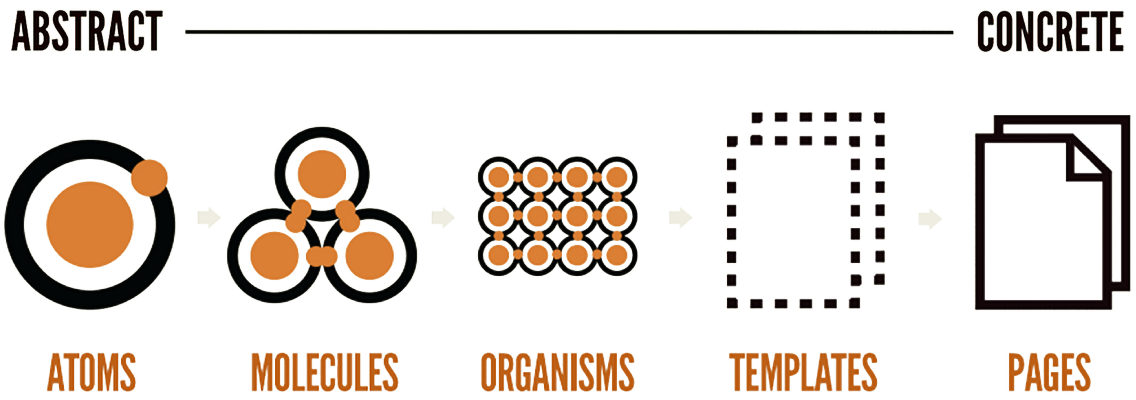

Abb. 5.9: Struktur des Atomic Designs [Frost (2016), K. 2]

Das in seinem Buch beschriebene System bezieht sich hauptsächlich auf die Entwicklung von Websites, lässt sich jedoch auch für jede andere Form von Benutzerinterfaces einsetzen. Atomic Design unterteilt die eingesetzten UI Elemente in Atome, Moleküle, Organismen, Templates und Seiten (siehe Abb. 5.9). Komponenten aus diesen Kategorien bauen jeweils auf Komponenten aus der vorherigen Kategorie auf (Frost 2016, K. 2).

Atome sind im Atomic Design die kleinste Einheit der Komponenten und können nicht weiter in kleinere Teile zerlegt werden. Gültige Atome sind beispielsweise Buttons, Eingabefelder, Labels, Überschriften und alle Elemente, die sich durch einen HTML Tag darstellen lassen (Frost 2016, K. 2).

Moleküle bauen auf diesen Atomen auf und kombinieren diese zu Einheiten, die einen bestimmten Zweck erfüllen. So könnte ein Molekül “Suchfeld” beispielsweise aus den Atomen “Button” und “Eingabefeld” bestehen. Dieses Molekül kann nun überall verwendet werden, wo ein Suchfeld benötigt wird (Frost 2016, K. 2).

Organismen sind komplexe Komponenten, die aus mehreren Atomen, Molekülen und anderen Organismen bestehen. Ein gültiges Molekül ist beispielsweise ein Header, so wie er in Abb. 5.10 abgebildet ist. Dieser könnte aus einem Logo-Atom, einem Menü-Molekül und einem Suchfeld-Molekül bestehen (Frost 2016, K. 2).

Abb. 5.10: Header-Organismus [Frost (2016), K. 2]

Templates bestehen aus mehreren Organismen und definieren das Layout in dem diese auf der Seite angeordnet werden sollen. Templates sind unabhängig von Inhalten und verwenden Platzhalter, um die Inhaltsstruktur zu definieren (Frost 2016, K. 2).

Seiten sind mit Inhalten gefüllte Templates. Sie repräsentieren den fertigen Zustand der Website und lassen sich gut dazu verwenden um sicherzustellen, dass die Inhalte in den definierten Templates gut dargestellt sind. So können eventuelle Probleme, wie zu lange Überschriften oder unzureichende Bildformate, behoben werden, indem entweder das Template oder die Inhalte angepasst werden (Frost 2016, K. 2).

Die von Frost vorgestellte Hierarchie ist nicht linear. So müssen beispielsweise nicht erst alle Atome gestaltet werden, bevor die Gestaltung der Moleküle beginnt (Frost 2016, K. 2).

Durch den Einsatz von Atomic Design ist es einfacher, ein Benutzerinterface sowohl als Ganzes zu betrachten, als auch sich auf einzelne Komponenten zu konzentrieren. Die hierarchische Struktur trägt dazu bei, dass Komponenten nicht versehentlich in verschiedenen Kontexten mehrmals gestaltet werden. Dadurch werden spätere Anpassungen an einzelnen Komponenten ermöglicht, die sich konsequent auf das gesamte Design auswirken (Frost 2016, K. 2).

Benutzerinterfaces, die auf dieses System setzen, sehen laut Frost meist einheitlicher aus und sind dadurch verständlicher für Benutzer. Sie lassen sich dazu in Zukunft schneller um neue Komponenten erweitern, und bleiben dabei für das Entwicklungs- und Designteam übersichtlich (Frost 2016, K. 4).

Frost schlägt in seinem Buch vor, die entwickelten Komponenten in einer Komponentenbibliothek, oder auch Pattern Library, zu sammeln. Diese ist direkt mit dem fertigen Produkt verbunden, sodass die dort enthaltenen Komponenten immer den aktuellen Zustand des Produkts abbilden. So werden in Zukunft Änderungen an den Komponenten direkt auf das Produkt angewendet und es kommt nicht zu unerwarteten Unterschieden zwischen Design und Code (Frost 2016, K. 3).

5.5 Design Systems

Design Systems ist ein von Alla Kholmatova entwickeltes Konzept und gleichnamiges Buch, das 2017 veröffentlicht wurde. Kholmatova beschreibt Design Systems als "[. . .] a set of connected patterns and shared practices, coherently organized to serve the purposes of a digital product"[8] (Kholmatova 2017, S. XII).

Ähnlich wie beim Atomic Design empfiehlt Kholmatova Unternehmen, eine Bibliothek für Muster, hier Pattern Library, anzulegen und zu pflegen. Sie unterscheidet bei diesen Patterns zwischen funktionalen Patterns, die das Verhalten und das Layout bestimmter Komponenten definieren und Wahrnehmungsmustern, die beispielsweise die Ästhetik und das Branding bestimmen (Kholmatova 2017, S. XI). Jedes Pattern stellt in dieser Bibliothek eine wiederverwendbare Lösung zu einem bestimmten Problem dar (Kholmatova 2017, S. 22).

Anders als Frost in seinem Atomic Design Ansatz legt Kholmatova in ihrem Konzept keine feste Hierarchie für diese Patterns fest und unterscheidet explizit zwischen der Pattern Library und dem Design System selbst (Kholmatova 2017, S. 18).

Eine gute Pattern Library hat keinen Wert, wenn sie im Team nicht richtig eingesetzt

[8] Übersetzt: Eine Reihe von miteinander verbundenen Mustern und gemeinsamen Praktiken, die einheitlich organisiert sind, um den Zwecken eines digitalen Produkts zu dienen

wird. Um diesem Problem vorzubeugen, ist es wichtig, dass alle Teammitglieder den gleichen Prinzipien folgen und eine geteilte Sprache sprechen. Diese sind neben der Pattern Library beide Teil des Design Systems (Kholmatova 2017, S. 28-30).

Einem guten Design System liegen stabile Design Prinzipien zugrunde. Diese Prinzipien definieren, was im Kontext des Unternehmens und des entwickelten Produkts gutes Design ist und bieten eine Grundlage für die Entwicklung der Design Patterns (Kholmatova 2017, S. 46-48).

Die Prinzipien des Salesforce Lightning Design Systems sind beispielsweise "Clarity", "Efficiency", "Consistency" und "Beauty"[9] (Salesforce 2019). Sipgates Prinzip lautet "Purpose first"[10] (sipgate GmbH 2019). Beide Unternehmen haben ihre Pattern Library auf diesen Prinzipien aufgebaut und öffentlich zur Verfügung gestellt.[11]

Im Falle von sipgate ersetzt das aktuelle Design System die vorherige Pattern Library, welche mit Frosts Atomic Design Prinzipien entwickelt wurde. Corinna Baldauf beschreibt in einem Artikel auf sipgates Blog, dass die Pattern Library mit Atomic Design durch seine hierarchische Struktur auf Dauer unübersichtlich wurde, was dazu führte, dass Patterns nicht wiederverwendet wurden – "im Zweifelsfall baute man einfach ein neues Pattern" (sipgate GmbH 2018) – und Restrukturierungen aufgrund der Abhängigkeiten zwischen Komponenten nicht mehr möglich war. Zusätzlich fehlte der Pattern Library ein übergreifender Style Guide, um die Einheitlichkeit des Aussehens und Verhaltens der Komponenten zu gewährleisten (sipgate GmbH 2018).

[9] Übersetzt: "Klarheit", "Effizienz", "Beständigkeit" und "Schönheit"
[10] Übersetzt: "Zuerst die Absicht"
[11] `lightningdesignsystem.com` und `sipgatedesign.com`

5.6 Rolle des Component Sprints

Component Sprints können in Unternehmen und für Produkte eingesetzt werden, die noch kein Design System nutzen. Ziel des Component Sprints ist die Optimierung des Übergangs vom Prototyp zum fertigen MVP eines Produkts. Die in einem Component Sprint entwickelte Pattern Library verwendet eine abgewandelte Variante von Atomic Design. Sie soll den Entwicklern des MVPs die Arbeit erleichtern, indem sie auf definierte Designelemente zurückgreifen und sich dadurch auf die Implementierung der Geschäftslogik des Produktes konzentrieren können.

Ein Component Sprint sollte zwischen einem Design Sprint und der Entwicklung des MVPs stattfinden, da so der validierte Prototyp aus dem Design Sprint als Grundlage für die Patterns genutzt werden kann.

6 Ausarbeitung des Konzepts für Component Sprints

Das Ziel eines Component Sprints ist die Erstellung einer Pattern Library für ein Produkt. Damit ein Component Sprint in einem festen Zeitraum zuverlässig wertvolle Ergebnisse liefert, sollte er einer festen Struktur folgen, ähnlich wie es bei Design Sprints der Fall ist.

Hypothesen, die in diesem Kapitel aufgestellt werden, werden im folgenden Kapitel in Form eines empirischen Tests überprüft.

Voraussetzung für einen Component Sprint ist ein validierter Prototyp, der bereits das finale Design des Produkts abbildet. Dieser wird als Grundlage bzw. Input des Component Sprints verwendet. Wenn das Unternehmen bereits ein definiertes Corporate Design bzw. Style Guide besitzt, sollte dieses zur Orientierung mit in den Component Sprint eingebracht werden.

Angelehnt an den Zeitrahmen von Design Sprints sollte ein Component Sprint in einem Zeitraum von vier bis fünf Tagen erfolgen. Aus persönlicher Erfahrung als Entwickler dürfte dieser Zeitraum für die Implementierung der Pattern Library ausreichen. Diese Hypothese wird im empirischen Test des Component Sprints überprüft.

Strukturell orientiert sich die Pattern Library des Component Sprints an einem Hybrid aus Atomic Design und Design Systems. Die Komponenten der Pattern Library sind in Atome, Moleküle und Templates unterteilt, wobei ein Molekül ausschließlich aus Atomen und nicht aus anderen Molekülen bestehen darf um die Komplexität der Struktur zu limitieren. Organismen und Seiten werden aus dem selben Grund aus der Struktur der Pattern Library ausgeschlossen. Die Prinzipien von Design Systems schließen die Unterteilung der Komponenten nicht explizit aus und die Hierarchie soll Entwicklern während des Sprints helfen, die einzelnen Komponenten zu priorisieren, sodass keine Abhängigkeitsprobleme entstehen.

Um einem Unternehmen einen nachhaltigen Wert zu liefern, sollte das Ergebnis eines Component Sprints gut dokumentiert sein. Dazu sollte es seitens des Unternehmens einfach genutzt und um neue Komponenten erweitert werden können. Um dies zu erreichen, ist ein gutes Handoff[1] des Sprint Teams an das Unternehmen wichtig.

Genau wie ein Design Sprint, kann ein Component Sprint sowohl innerhalb eines Unternehmens oder durch eine externe Agentur durchgeführt werden.

Das Sprint Team sollte für optimale Produktivität insgesamt aus maximal zehn Personen bestehen (Brooks 1995, p. 31; Coplien 1994, p. 7). Je nach Komplexität des Produkts und damit der Anzahl an Komponenten dürfte ein Team von mindestens drei Personen jedoch auch ausreichen. Auch diese Hypothese wird im empirischen Test überprüft.

Als Basis für die Pattern Library kann Storybook[2] verwendet werden. Storybook ist ein kostenloses, offenes Framework, das mit beliebten Web Libraries wie React[3],

[1] Übersetzt: Übergabe

[2] `https://storybook.js.org/`

[3] `https://reactjs.org/`

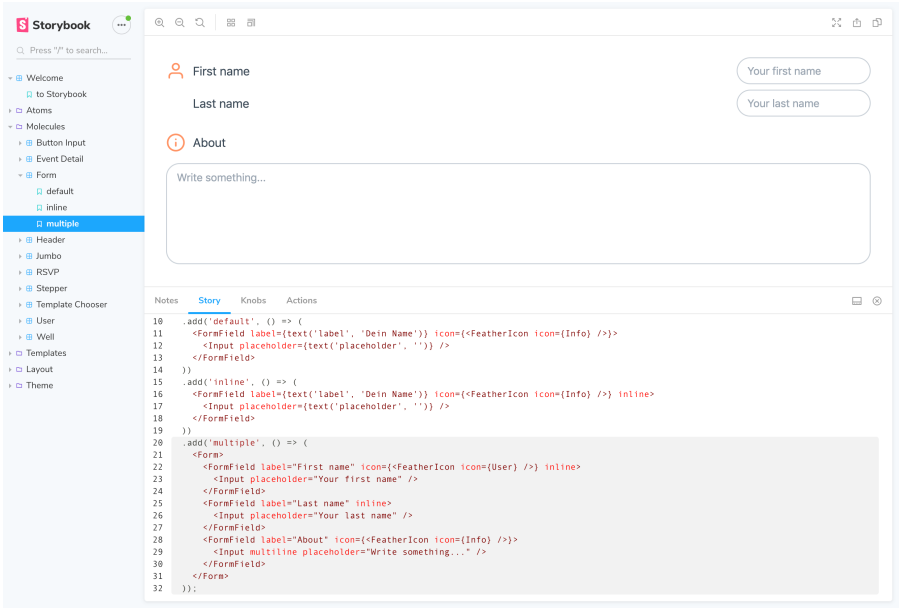

Abb. 6.1: Die Benutzeroberfläche von Storybook

Vue[4] und Angular[5] zusammenarbeitet. Storybook ermöglicht Entwicklern, Komponenten zu implementieren, zu dokumentieren und zu testen. Die Komponenten werden den Nutzern automatisch in einer übersichtlichen Benutzeroberfläche (siehe Abb. 6.1) in allen möglichen Variationen zusammen mit Codebeispielen angezeigt (*Storybook* 2019).

Um die Zusammenarbeit zwischen den Teammitgliedern zu vereinfachen, bietet sich eine Versionsverwaltungssoftware wie beispielsweise Git[6] an. Diese Software erlaubt es mehreren Entwicklern, parallel an dem selben Projekt zu arbeiten. Änderungen werden in Form von sogenannten Commits gespeichert und mit anderen Entwicklern oder einem zentralen Server synchronisiert (*Git - Book* 2019, K. 1.1). Als zentrale Platform für Git bieten sich GitHub[7], GitLab[8] oder BitBucket[9] an.

[4] https://vuejs.org/
[5] https://angular.io/
[6] https://git-scm.com/
[7] https://github.com
[8] https://about.gitlab.com/
[9] https://bitbucket.org

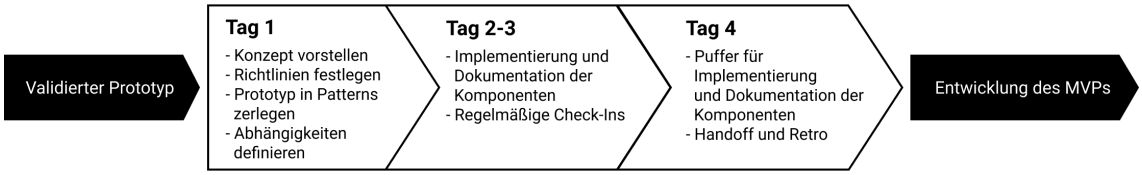

Abb. 6.2: Zusammenfassung des Component Sprints [eigene Darstellung]

Das Aufsetzen von Storybook und der Entwicklungsumgebung lässt sich nicht parallelisieren bzw. auf mehrere Teammitglieder aufteilen und sollte daher bereits vor dem Component Sprint geschehen, da alle anderen Aufgaben davon abhängen.

Die Struktur des Component Sprints lässt sich essentiell in drei Sektionen unterteilen: Vorbereitung, Implementierung und Handoff (siehe Abb. 6.2). Angelehnt an den Design Sprint 2.0 von AJ&Smart (AJ&Smart 2018a) beginnt jeder Tag um 10:00 Uhr und endet um 17:00 Uhr. So können alle Teilnehmer des Sprints vor Beginn des Tages noch geschäftliche Angelegenheiten erledigen.

Ebenfalls orientiert am Design Sprint gibt es einen Moderator, der das Team durch den Component Sprint leitet und dem Team bei möglichen Fragen zur Verfügung steht.

Der erste Tag ist der Vorbereitung gewidmet. Hier erklärt der Moderator vorerst allen Teilnehmern das Konzept von Component Sprints und die Entwicklungsumgebung, sowie Richtlinien zur Entwicklung. Es ist wichtig, dass alle Entwickler den selben Richtlinien folgen, damit das Ergebnis einheitlich und übersichtlich bleibt. Diese Richtlinien können beispielsweise die Ordnerstruktur, Namenskonventionen, Codestil und Art der Dokumentation festlegen. Eventuelle Fragen können hier geklärt werden, sodass alle Teilnehmer sich auf dem selben Wissensstand befinden.

Nach diesem Schritt wird der Prototyp bzw. das Ausgangsdesign Screen für Screen vom Team in einzelne Atome, Moleküle und Templates unterteilt. Jede

Komponente wird als Post-it mit Name, Skizze und den möglichen Eigenschaften festgehalten und auf einer Wand gesammelt. Bei der Anordnung der Komponenten sollte bereits darauf geachtet werden, welche Komponenten voneinander abhängig sind. So entsteht schnell eine Übersicht über alle Komponenten, die implementiert werden müssen.

Am zweiten und dritten Tag werden die Komponenten implementiert. Dafür nimmt sich jeder Entwickler eine Komponente als Post-it von der Wand und implementiert und dokumentiert diese. Dieser Prozess lässt sich gut parallelisieren, da die meisten Komponenten aufgrund der vorgegebenen Struktur unabhängig voneinander entwickelt werden können.

Alle zwei Stunden führt das Team in Anleitung durch den Moderator ein Check-in durch, in dem der Status der Komponenten abgeglichen wird und mögliche Fragen und Probleme geklärt werden können.

Am Ende der beiden Tage sollte die Pattern Library fertig implementiert und zum Großteil dokumentiert sein.

Am vierten Tag überarbeitet das Entwicklerteam mögliche undokumentierte oder unvollständig dokumentierte Komponenten und korrigiert eventuelle Abweichungen von den Richtlinien. Falls am Vortag nicht alle Komponenten implementiert werden konnten, kann der Vormittag des Tages dazu genutzt werden, die Implementierung so weit wie möglich fertigzustellen.

Am Nachmittag des vierten Tages wird die Pattern Library an das Entwicklerteam abgegeben, welches diese in das MVP integriert. Im Rahmen des Handoffs stellt der Moderator die Pattern Library vor und zeigt, wie sie in ein Projekt integriert werden kann. Die Entwickler des MVP-Entwicklerteams können hier mögliche Fragen mit den Entwicklern des Component Sprint Teams klären und dieses Wissen an ihr Team weitergeben.

Nach dem Handoff ist die Zusammenarbeit mit dem MVP-Entwicklerteam vorerst beendet. Zum Ende des Component Sprints wird innerhalb des Component Sprint Teams eine Retrospektive durchgeführt. Hier hält der Moderator fest, was während des Sprints gut funktioniert hat und an welchen Stellen noch Optimierungsbedarf besteht. Dieses Wissen sollte dazu genutzt werden, den nächsten Component Sprint zu optimieren (Dybå u. a. 2014, S. 290).

7 Empirischer Test des entwickelten Konzepts

Um das im letzten Kapitel ausgearbeitete Konzept zu testen und die dort aufgestellten Hypothesen zu überprüfen, bietet sich die Durchführung eines Component Sprints im Anschluss an einen Design Sprint an, in dem eine neue Produktidee zu einem validierten Prototyp ausgearbeitet wird.

Um die zeitlichen und personenbezogenen Grenzen des Konzepts zu testen, wird der Component Sprint in vier Tagen und mit drei Teilnehmern durchgeführt.

Die Produktidee ist für diesen Test eine App, mit der Nutzer Events wie beispielsweise Geburtstage, Partys oder Spielabende im kleinen Rahmen planen können. Das Einladen von Gästen soll plattformunabhängig per Link funktionieren und eingeladene Personen sollen direkt über die App die Möglichkeit haben, zu- oder abzusagen. Die App trägt den Namen Wevent und soll so einfach wie möglich zu bedienen sein, sodass Nutzer keine großen Hindernisse überwinden müssen, um die App zu verwenden.

Der Design Sprint für diese Produktidee wird in Kooperation mit Crisp Studio durchgeführt. Das Design Sprint Team besteht aus Daniel Wirtz[1], René Nauheimer[2]

[1] https://danielwirtz.com/

[2] https://twitter.com/renaui

und Leo Bernard[3], dem Autor dieser Arbeit. Nauheimer übernimmt die Rolle des Moderators, beteiligt sich jedoch auch an den Aufgaben. Bernard übernimmt die Decider-Rolle. Für den Design Sprint wird auf die Struktur des Design Sprints 2.0 von AJ&Smart zurückgegriffen. So dauert der Design Sprint nur vier statt fünf Tage.

Der Component Sprint wird in der darauffolgenden Woche zusammen mit zwei Entwicklern, die im Bereich Web Development tätig sind, durchgeführt. Hier besteht das Sprint Team aus Marcus Weiner[4], Moritz Gunz[5] und Leo Bernard. Bernard übernimmt in diesem Sprint die Rolle des Moderators, beteiligt sich aber auch mit an der Entwicklung.

Da alle Teammitglieder bereits Erfahrung in der Entwicklung mit React haben, wird dieses Framework in Kombination mit Storybook für den Component Sprint eingesetzt. Für die Kollaboration wird Git auf GitHub eingesetzt.

Die Zeiten der Implementierung der Komponenten werden anhand der Commit Logs auf GitHub festgehalten und können nach dem Test ausgewertet werden.

7.1 Ablauf

Der Design Sprint begann am Montag der ersten Projektwoche nach Aufbau des Sprintraums und kurzer Einleitung mit einem Experteninterview. Hier beschrieb der Decider die Problemstellung und die Idee der App Wevent. Wichtige Punkte aus dem Gespräch wurden in Form von “How might we”-Fragen auf Post-its festgehalten und im Anschluss an das Gespräch in die Kategorien *Onboarding*, *Information*,

[3] https://leolabs.org
[4] https://marcusweiner.de/
[5] https://moritzgunz.de/

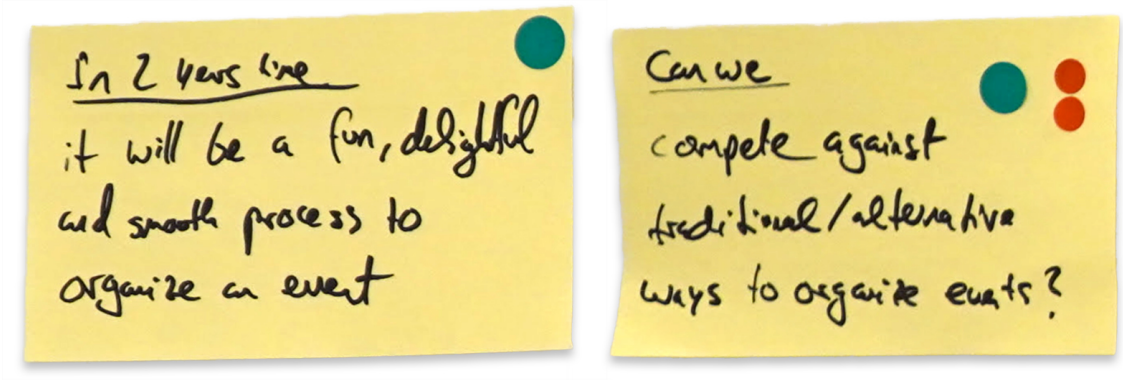

Abb. 7.1: Sprintziel und Sprintfrage

Abb. 7.2: Einzelne Schritte im User-Flow (ein Flow pro Zeile)

Ausrichtung und *Misc* unterteilt. Über diese Fragen wurde nun abgestimmt. Jeder Teilnehmer bekam dafür zwei Stimmen in Form von Klebepunkten, der Decider vier.

Im Anschluss wurden Langzeitziele der App sowie mehrere Sprintfragen vorgeschlagen. Mithilfe von Klebepunkten wurde jeweils über den passendsten Vorschlag abgestimmt. Das Langzeitziel des Sprints wurde auf “In two years’ time, it will be a fun, delightful, and smooth process to organize an event” festgelegt. Die Sprintfrage lautete: “Can we compete against traditional/alternative ways to organize events?” (siehe Abb. 7.1).

Um herauszufinden, welcher Bereich der App-Idee im Fokus des Sprints stehen sollte, legte jedes Teammitglied einen User-Flow in maximal acht Schritten an,

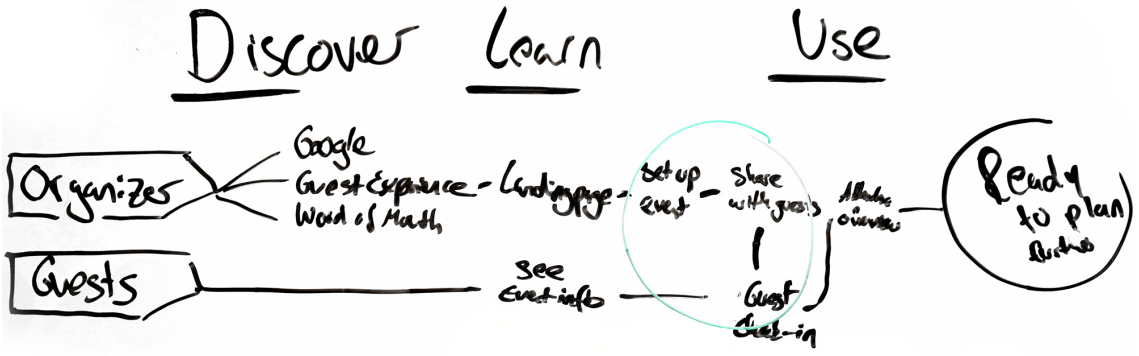

Abb. 7.3: Sprint-Map mit markiertem Fokusbereich

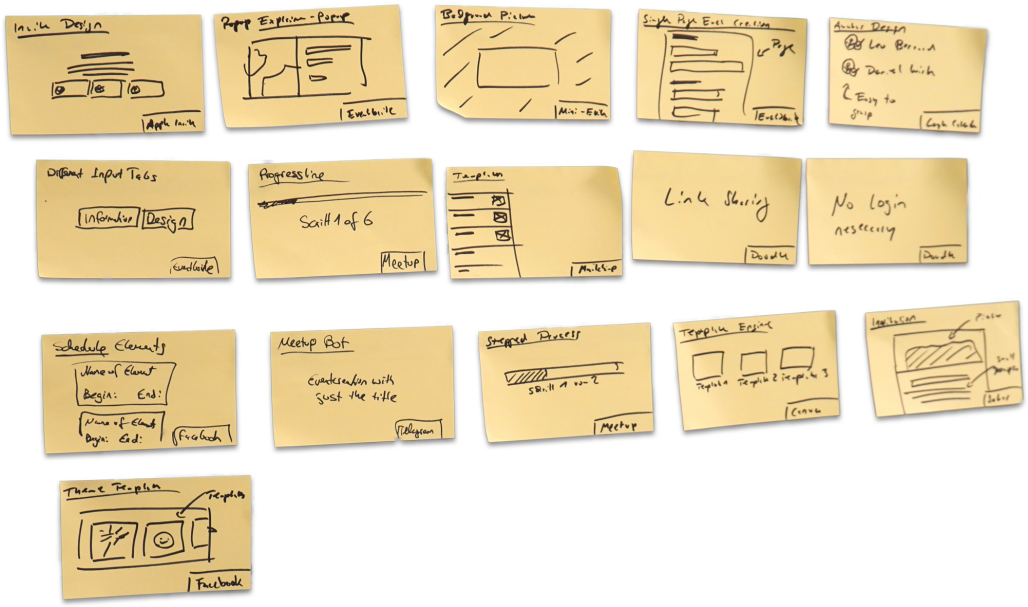

Abb. 7.4: Ergebnis der Lightning Demos

den ein potentieller Benutzer durchlaufen könnte (siehe Abb. 7.2). Diese Schritte wurden als Grundlage genutzt, um eine Map zu erstellen, die den Flow in drei Phasen – *Discover*, *Learn* und *Use* – unterteilt. Nach gemeinsamer Absprache wurde der Fokusbereich auf das Erstellen und Teilen eines Events und auf den Check-In für Gäste festgelegt (siehe Abb. 7.3). Dem Fokusbereich wurden relevante “How might we”-Fragen zugeordnet.

Am Nachmittag wurden zur UI-Inspiration Ideen aus bestehenden Websites gesammelt und in Lightning Demos vorgestellt. Die Ergebnisse wurden in Form von Post-its festgehalten (siehe Abb. 7.4).

Abb. 7.5: 3-Part-Sketches

Auf Grundlage dieser Ergebnisse und den am Vormittag erarbeiteten Fragen und Zielen zeichnete nun jedes Teammitglied Skizzen eines möglichen UI-Layouts. Diese wurden in Form von Crazy 8s optimiert und als 3-Part-Sketch (Abb. 7.5) zu einer vollständigen Skizze des Produktes ausgearbeitet.

Am Dienstag wurde der beste 3-Part-Sketch aus den drei erstellten Sketches durch mehrere Abstimmungen ausgewählt. Der Sketch mit dem Titel “Linky” erhielt am meisten Stimmen. Der in dem Sketch abgebildete Flow wurde – ähnlich wie im User-Flow-Schritt am Vortag – in einem User-Test-Flow abgebildet. Aus den gesammelten Ressourcen wurde nun in Teamarbeit ein acht-teiliges Storyboard erstellt (siehe Abb. 7.6).

Dieses Storyboard wurde am Mittwoch in Framer[6] als testbarer Prototyp gebaut (siehe Abb. 7.7). Framer ist ein Tool, mit dem sich Prototypen mit wenig Aufwand erstellen lassen. Aufwändigere Teile des Prototypen können direkt als Code auf Basis von React implementiert werden. Das Prototyping dauerte hier länger als geplant, sodass die Endzeit des Tages von 17:00 Uhr auf 23:00 Uhr verlegt werden

[6] https://www.framer.com/

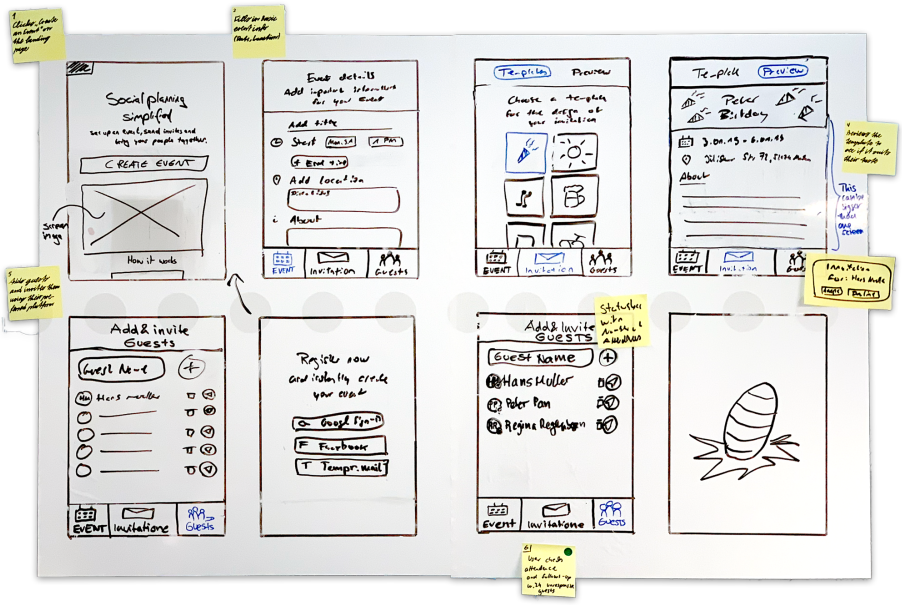

Abb. 7.6: Ausgearbeitetes Storyboard des Sketches “Linky”

musste. Da die Benutzerinterviews am nächsten Tag von dem fertigen Prototypen abhängig sind, musste zumindest der User-Flow noch vollständig abgebildet werden.

Am Donnerstag wurden die Benutzerinterviews durchgeführt. Dazu wurden im Vorhinein fünf Testpersonen organisiert, die zu dem Interview vor Ort erscheinen konnten. Diese Personen wurden einzeln von dem Moderator durch den Prototypen geleitet. Nach der Vorstellung und dem Test des Prototypen fragte er jede Testperson, welche drei Wünsche sie an die App hätte.

Die Interviews wurden mithilfe von OBS Studio[7] per YouTube live an den Rest des Teams übertragen, der sich in einem anderen Raum gleichzeitig Notizen dazu machen konnte. Diese Notizen waren in die fünf Schritte, die der Nutzer im Prototyp durchläuft, die drei Wünsche und weitere Anmerkungen unterteilt (siehe Abb. 7.8). Jede Notiz wurde entweder als positiv (gelb), negativ (blau), Wunsch (rot) oder als Anmerkung zum Interview selbst markiert.

[7] `https://obsproject.com/`

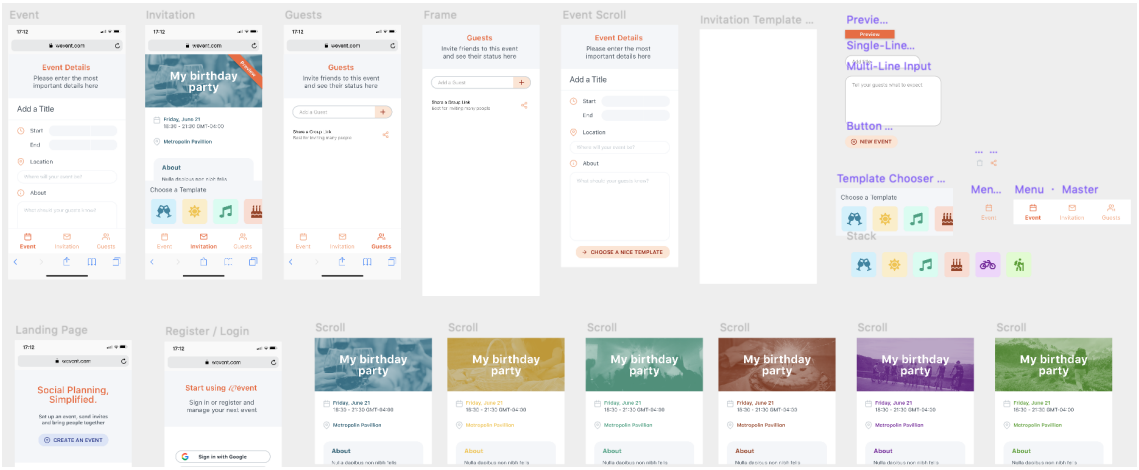

Abb. 7.7: Screenshot des fertigen Prototyps in Framer

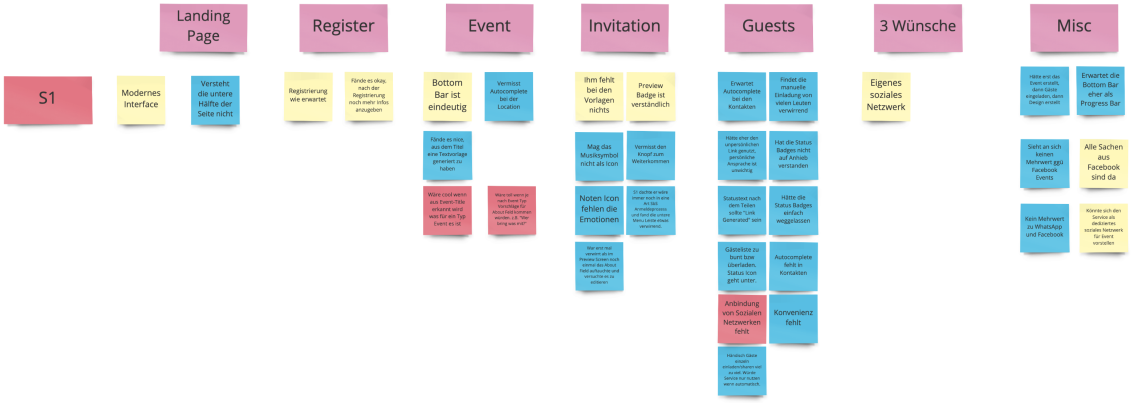

Abb. 7.8: Notizen zu einem der Nutzerinterviews

Am Freitag wurden die Notizen aus den Nutzerinterviews in Form eines verkürzten Iteration Sprints zusammengefasst und festgehalten.[8] Am Nachmittag wurde die Entwicklungsumgebung für den folgenden Component Sprint eingerichtet.

Der Component Sprint begann am Dienstag der folgenden Woche mit einer kurzen Einführung in das Konzept, einer Vorstellung des Prototyps und einem Austausch über den besten Codestil, sowie über die Anwendung der Prinzipien des Atomic Design im Kontext von Web Apps.

Nachdem alle offenen Fragen geklärt werden konnten, wurde auf den Computern der Teammitglieder die Entwicklungsumgebung eingerichtet, sodass alle Entwickler an der Pattern Library arbeiten konnten.

Daraufhin wurde der Prototyp unter Anleitung des Moderators in einzelne Komponenten zerlegt. Neben den normalen Komponenten wie Atomen und Molekülen entstanden hierbei auch Komponenten, die in keine der vorgegebenen Kategorien passten. Da die meisten dieser Komponenten das Layout beeinflussten, wurde eine neue Kategorie “Layout” erstellt. Insgesamt entstanden bei der Zerlegung 29 Komponenten, unterteilt in neun Atome, elf Moleküle, vier Layout-Komponenten und fünf Templates (siehe Abb. 7.9). Ein Atom und zwei Layout-Komponenten wurden erst im späteren Verlauf des Sprints hinzugefügt und sind daher in Abb. 7.9 noch nicht abgebildet.

Das Zerlegen des Prototyps dauerte nicht so lange wie geplant, sodass der Tag bereits um 15 Uhr beendet werden konnte.

Am Mittwoch begann das Team mit der Implementierung und Dokumentation der

[8] Iteration Sprints dauern in der Regel vier Tage und folgen ähnlichen Prinzipien wie Design Sprints, verwenden jedoch andere Methoden, um den Prototyp zu überarbeiten. Essentiell wird Nutzerfeedback in umsetzbare Änderungen zusammengefasst, diese werden in den Prototyp eingepflegt und erneut mit fünf Nutzern getestet (AJ&Smart 2018b).

Abb. 7.9: Sortierte Komponenten des Prototyps

Komponenten. Hierzu nahm sich jedes Teammitglied eine Komponente von der Post-it-Wand. Komponenten, an denen gerade gearbeitet wurde, wurden in eine “Doing”-Spalte verschoben, fertige Komponenten in eine “Done”-Spalte. So hatte jedes Teammitglied jederzeit einen Überblick über den Fortschritt.

Reguläre Check-Ins wurden alle zwei Stunden vom Moderator durchgeführt, jedoch konnten viele der Fragen auch schon während der Entwicklung geklärt werden.

Insgesamt konnte das Team am Mittwoch zehn der 29 Komponenten implementieren. Fünf Komponenten waren noch in Arbeit. Der Tag dauerte länger als geplant, sodass der Sprintraum erst um 20:00 Uhr verlassen wurde – drei Stunden nach dem ursprünglich geplanten Ende.

Am Donnerstag konnten alle Komponenten bis auf zwei Templates erfolgreich implementiert werden, jedoch dauerte der Tag auch hier länger als geplant und der Sprintraum wurde erst um 21:00 Uhr verlassen.

Die letzten beiden Templates wurden am Freitag Morgen implementiert. Daraufhin wurden alle Komponenten auf Korrektheit und Vollständigkeit hin kontrolliert, eventuelle Fehler behoben und die Dokumentation vervollständigt.

Da es in diesem Component Sprint keinen Kunden im klassischen Sinne gab, wurde das Handoff übersprungen und die Retrospektive vorverlegt. Diese dauerte insgesamt ca. anderthalb Stunden.

Der Sprintraum wurde am Freitag um 15:00 Uhr verlassen.

7.2 Ergebnisse

Zusammengefasst wurde in dem empirischen Test in einem Zeitraum von zwei Wochen eine grobe Idee zu einem validierten App-Prototypen und einer darauf abgestimmten Pattern Library ausgebaut. Mit Hilfe dieser Pattern Library kann im nächsten Schritt das MVP der App entwickelt werden.

Die Ergebnisse des empirischen Tests sind in Form des fertigen Prototyps[9] und der entwickelten Pattern Library[10] online verfügbar.

Da im Component Sprint zwei Tage um insgesamt sieben Stunden überzogen wurden, kann die Hypothese, dass drei Entwickler in vier Tagen für einen Component Sprint mit einer Größenordnung von 29 Komponenten ausreichen, als widerlegt gelten.

[9] `https://wevent-prototype.netlify.com/`
[10] `https://wevent-components.netlify.com/`

Tabelle 7.1: Durchschnittliche Entwicklungszeiten der Komponenten

Type	Count	ø Time	Median Time	ø Commits
Atom	9	1:30:23	0:48:00	13,2
Molecule	10	1:20:12	1:12:45	8,9
Layout	5	1:21:48	0:57:00	3,0
Template	5	2:30:30	2:27:00	4,0

Die durchschnittlichen Entwicklungszeiten der einzelnen Komponenten unterscheiden sich mit der Ausnahme von Templates nicht groß voneinander. Um zukünftige Component Sprints besser planen zu können, bietet es sich an, für Atome, Moleküle und Layout-Komponenten jeweils ca. 2 Stunden Entwicklungszeit einzuberechnen. Für Templates sollten 3 Stunden einberechnet werden. Diese Zeiten enthalten jeweils 30 Minuten Pufferzeit, um eine vollständige Entwicklung zu gewährleisten, auch wenn die Entwicklung der Komponenten länger dauern sollte als geplant. Die genauen Zeiten der einzelnen Komponenten sind in Tabelle 11.1 aufgeschlüsselt.

$$1 \text{ Tag} + \frac{24 \text{ Komponenten} * 2 \text{ h} + 5 \text{ Layouts} * 3 \text{ h}}{6 \text{ h/Tag} * 3 \text{ Entwickler}} = 4,5 \text{ Tage} \tag{7.1}$$

$$1 \text{ Tag} + \frac{24 \text{ Komponenten} * 2\text{h} + 5 \text{ Layouts} * 3\text{h}}{6 \text{ h/Tag} * 4 \text{ Entwickler}} = 3,625 \text{ Tage} \tag{7.2}$$

Basierend auf diesen Daten und mit der Annahme, dass die Arbeitsleistung proportional mit der Anzahl der Entwickler steigt, hätte der Component Sprint mit drei Entwicklern in fünf Tagen (Gleichung (7.1)) oder mit vier Entwicklern in vier Tagen (Gleichung (7.2)) ohne Überstunden gelingen können. Diese Annahme wurde jedoch bisher noch nicht bestätigt und könnte Gegenstand einer weiteren Überprüfung werden.

7.3 Erkenntnisse

Die folgenden Erkenntnisse wurden in der Retrospektive am Ende des Component Sprints gesammelt.

Das Tooling des Component Sprints mit Storybook, React und Linaria wurde positiv wahrgenommen. In Zukunft wäre es einfach, die Pattern Library des Component Sprints um Visual Testing[11] zu erweitern, um sicherzustellen, dass alle Komponenten auch nach möglichen Änderungen gegebene Grundvoraussetzungen erfüllen und sich das Aussehen nur kontrolliert verändert. Hierzu eignen sich Dienste wie Percy[12] oder Chromatic[13], die sich in Storybook integrieren lassen und dieses Testing automatisieren können.

Durch Konzentration auf die Implementierung von UI-Komponenten kam es selten vor, dass ein Teammitglied in der Umsetzung nicht weiterkam. Da es innerhalb einer Hierarchieebene, wie z.B. bei Atomen, keine Abhängigkeiten zwischen Komponenten gab, konnte zudem das “Together Alone”-Prinzip aus Design Sprints gut angewendet werden. So konnten die Teammitglieder unabhängig voneinander arbeiten, was zur Annahme führte, dass die Anzahl der Teammitglieder ohne große Einbußen der Produktivität variiert werden könnte.

Um den Component Sprint in Zukunft noch effizienter zu gestalten, wäre es sinnvoll, bereits im Vorhinein feste Vorgaben für Farben, Formen und die generelle Corporate Design des Produktes festzulegen um Unklarheiten bei der Implementie-

[11] Visual Testing beschreibt eine Methode, in der bei jeder Änderung im Code automatisch das Aussehen aller Komponenten auf Änderungen hin überprüft wird. Wenn Änderungen festgestellt wurden, wird der Entwickler darüber benachrichtigt. Dies verhindert ungewollte visuelle Änderungen (Storybook 2019).

[12] `https://percy.io/`

[13] `https://www.chromaticqa.com`

rung vorzubeugen. Im Falle des Test-Sprints war das größte Problem die Definition der Farben, die in den Komponenten verwendet werden sollten.

Zusätzlich zu strikteren Designvorgaben wünschte sich das Sprint Team eine Design Role als Teil des Teams, die sich um die visuelle Kohärenz der Pattern Library kümmert und eventuelle Fragen der Entwickler beantworten kann. Im Optimalfall sollte diese Design Role aus dem Unternehmen kommen, für das die Pattern Library entwickelt wird. So kann die Designsprache des Unternehmens von vornherein berücksichtigt werden.

Der Idee, Entwickler des Kundenteams mit in den Component Sprint einzubeziehen, stand das Sprint Team skeptisch gegenüber. Der größte Kritikpunkt war hier, dass diese Developer Role eventuell den Fortschritt zurückhalten könnte, da sie durch bisherige Verfahren im Unternehmen bereits geprägt ist. Auf der anderen Seite besitzt diese Person auch nützliches Wissen, das bei der Entwicklung der Pattern Library helfen könnte.

Dem Moderator gegenüber wünschte sich das Team mehr Moderation und eine ausführlichere Einführung in das Thema für Mitglieder, die noch nicht mit den Konzepten von Design Systems, Atomic Design und Component Sprints gearbeitet haben.

Zusammengefasst war das gesamte Sprint Team mit dem Component Sprint zufrieden, sah jedoch, wie in diesem Abschnitt beschrieben, noch Potential zur Optimierung des Konzepts für zukünftige Sprints.

8 Qualitative Studie zur Verfeinerung des Konzepts

Um das Konzept der Component Sprints weiter auszuarbeiten, bietet sich eine qualitative Studie in Form von Experteninterviews mit Personen aus digitalen Firmen verschiedener Größen an.

Diese Experteninterviews bieten qualitative Einblicke in das Wissen der Experten und deren Sicht auf bestimmte Situationen (Gläser u. a. 2010, S. 13), in diesem konkreten Fall auf die Arbeitsumgebung, die Arbeitsweisen und die Anforderungen in ihrem Unternehmen.

Der Begriff "'Experte' beschreibt [hierbei] die spezifische Rolle des Interviewpartners als Quelle von Spezialwissen über die zu erforschenden [. . .] Sachverhalte." (Gläser u. a. 2010, S. 12).

8.1 Studiendesign

Insgesamt werden vier Interviews durchgeführt – zwei davon mit Personen aus Agenturen, die digitale Projekte für andere Firmen konzipieren und umsetzen und zwei mit Personen, die aktiv an eigenen Produkten arbeiten. Die Firmengröße

reicht hierbei von einem Startup mit fünf Mitarbeitern (Bogdoll u. a. 2019) bis hin zu einem Großunternehmen mit über 50000 Mitarbeitern (anonym).

Das Ziel dieser diversen Auswahl ist die Gewinnung von vielen verschiedenen Perspektiven auf das Konzept der Component Sprints und dessen Potential als Dienstleistung in verschiedenen Situationen.

Als Interviewte haben sich folgende Personen bereiterklärt:

- Daniel Bogdoll, CEO bei SAYM[1] (23.8.2019)
- Niels Anhalt, Director bei nexum AG[2] (27.8.2019)
- Eine anonyme Person aus der Automobilbranche (28.8.2019)
- Markus Mazur, Geschäftsführer bei Duplexmedia[3] (2.9.2019)

Die anonyme Person wird im Folgenden als (anonym) bezeichnet.

8.1.1 Entwicklung des Interviewleitfadens

Um den interviewten Personen genug Freiraum zu lassen, eigene Ideen und Gedanken mit in das Interview einbringen zu können, wird das Interview semistrukturiert aufgebaut. Hierdurch soll ein offenes Gespräch ermöglicht werden, sollte die interviewte Person sich dies wünschen.

Zur Einordnung der interviewten Personen in ihr Arbeitsumfeld beginnen die Interviews vorerst mit Fragen zur Person. Hierzu werden folgende Fragen gestellt:

- Wo arbeitest du?

[1] https://saym.io/
[2] https://www.nexum.de
[3] https://www.duplexmedia.com/

- Was bietet dein Produkt / deine Firma?

Darauf folgend wird die interviewte Person bezüglich der theoretischen Grundlagen befragt, um zu erfahren, inwiefern sie bereits mit den in Kapitel 5 vorgestellten Konzepten – insbesondere mit Design Sprints und Component Systems bzw. Pattern Libraries – vertraut ist. Zusätzlich ist die Erfahrung der Person mit dem Einsatz dieser Konzepte interessant:

- Habt ihr schon mit Design Sprints gearbeitet?
- Welche Vor- und Nachteile siehst du bei Design Sprints?
- Mit welchen Frameworks arbeitet ihr?
- Nutzt ihr eine Pattern Library für das Design eures Produktes?
- Welche Vor- und Nachteile siehst du bei Pattern Libraries?

Im Anschluss daran wird der interviewten Person das Konzept der Component Sprints, sowie ein Bericht des empirischen Tests anhand des Beispielprojekts We-vent inklusive der Ergebnisse vorgestellt. Diese beinhalten sowohl den Prototypen als auch die auf Grundlage dessen entwickelte Pattern Library. Basierend auf diesen Informationen kann die interviewte Person nun nach ihrer Einstellung zu dem Konzept befragt werden.

- Hast du generell Fragen zu dem Konzept?
- Wo siehst du in Component Sprints Potential?
- Wo könnte es zu Problemen kommen?
- Könntest du dir für dein Unternehmen so einen Sprint vorstellen?

Daraufhin wird die Person nach ihrer Einschätzung zu potentiellen Zielgruppen für Component Sprints befragt. Diese Frage ist für die Weiterentwicklung des Konzepts interessant:

- Für welche Zielgruppe könnte der Component Sprint interessant sein?

Letztlich erhält die interviewte Person die Möglichkeit, eigene Gedanken oder Ideen zu dem Konzept zu formulieren:

- Hast du noch Gedanken oder Ideen zu dem Konzept?

Es werden nicht alle Fragen auf alle Experten zutreffen, sodass manche Fragen je nach Interview leicht angepasst oder ausgelassen werden.

Die Experteninterviews sind für eine Dauer von maximal einer Stunde angelegt. Um die Ergebnisse des empirischen Tests sehen zu können, ist es während der Interviews nötig, dass die interviewte Person den Bildschirm des Interviewers sehen kann. Daher werden Drei der Interviews per Videokonferenz und ein Interview in Person vor Ort durchgeführt.

Letztendlich nahmen die Interviews mit Markus Mazur und Niels Anhalt die gesamte Stunde in Anspruch. Die anderen beiden Interviews dauerten 25 (anonym) bzw. 40 (Daniel Bogdoll) Minuten.

8.2 Ergebnisse

Im Folgenden werden die Ergebnisse der vier Interviews zusammengefasst. Die Transkriptionen der Interviews befinden sich sowohl im Anhang als auch online.[4]

Das Konzept der Component Sprints wurde in allen Interviews grundsätzlich positiv aufgenommen, jedoch wurden einige Punkte angesprochen, die noch optimiert werden können. Bogdoll sieht in dem Konzept Potential (Bogdoll, Z. 263pp) und würde in Zukunft gerne einen Component Sprint für SAYM durchführen (Bogdoll, Z. 442pp) und Mazur erwähnte, dass der Sprint für die Entwicklung der eigenen

[4] https://github.com/leolabs/bachelor/tree/master/interviews

Firmenwebsite interessant gewesen wäre (Mazur, Z. 388pp). (anonym) sieht zwar persönlich keinen Mehrwert, da sie für das Zeichnen und Entwickeln von Prototypen und nicht für die Umsetzung der Produkte zuständig ist, kann sich die Sprints jedoch für ihr Unternehmen aus Perspektive der Entwicklungsteams gut vorstellen (anonym, Z. 232pp).

Sowohl (anonym) als auch Anhalt hatten bereits mit Design Sprints gearbeitet und gute Erfahrungen damit gemacht (anonym, Z. 35p; Anhalt, Z. 40pp). Mazur hat Design Sprints noch nicht verwendet, ist aber interessiert daran und hat sich mit der Methodik bereits auseinandergesetzt (Mazur, Z. 164pp). Anhalt gefällt an Design Sprints besonders der kurze Zeitraum, in dem ein Sprint durchgeführt wird (Anhalt, Z. 46p) und dass er "[. . .] so kompakt ist und wirklich alle Phasen [der Ideation] durchläuft" (Anhalt, Z. 43p). (anonym) sieht Design Sprints als sinnvoll an, um schnell zu verstehen, wie Kunden eine Idee sehen könnten (anonym, Z. 39p), hinterfragte im Interview jedoch, wie repräsentativ die Testergebnisse mit nur fünf Nutzern sind (anonym, Z. 42pp).

Keiner der vier Experten hatte bisher mit einer Pattern Library gearbeitet.

Alle interviewten Personen konnten sich als Zielgruppe der Component Sprints sowohl Startups als auch größere Unternehmen vorstellen (Bogdoll, Z. 364pp; anonym, Z. 235p; Anhalt, Z. 619pp; Mazur, Z. 469pp). Mazur fügte der Einschätzung hinzu, dass die Art der Produkte wichtiger sei als die Größe des Unternehmens. So sei der Component Sprint seiner Ansicht nach beispielsweise für Apps und Web-Apps besser geeignet als für Websites, da sich bei Websites nicht so viele Elemente wiederholen wie in Apps (Mazur, Z. 466pp).

Ein wichtiger Punkt, der sowohl von Bogdoll als auch von (anonym) und Anhalt angesprochen wurde, ist das Einbeziehen von Entwicklern und anderen Mitarbeitern wie z.B. UX-Designern seitens des Kunden in den Component Sprint. Diese Mitarbeiter können ihr Knowhow gut in die Entwicklung der Komponenten einbringen

und neues, im Component Sprint erworbenes Wissen im Gegenzug wieder an ihr Team weitergeben (Bogdoll, Z. 379pp). Dazu könnte die Akzeptanz der Pattern Library gegenüber diesen Entwicklern steigen, da sie im Sprint die Möglichkeit haben, die Entwicklung mitzugestalten (anonym, Z. 170pp). UX-Designer könnten in der Entwicklung helfen, wenn beispielsweise bestimmte Zustände von Komponenten noch nicht visuell definiert sind, da sie bereits Wissen über das Branding des Kunden und einen besseren Blick für Design haben (Anhalt, Z. 568pp).

Bei der Annahme, dass ein Component Sprint zwischen dem Design Sprint und der Entwicklung des MVPs stattfinden sollte, gingen die Meinungen der Interviewten Personen auseinander. So schlug Bogdoll vor, den Sprint nach der Entwicklung des MVPs und vor der Entwicklung des finalen Produktes anzusetzen, da MVPs meist wenige wiederverwendbare Elemente enthalten und nach der Testphase im Optimalfall verworfen würden, sodass die Entwicklung der Pattern Library so keinen nachhaltigen Wert liefere (Bogdoll, Z. 266pp).

Anhalt würde den Component Sprint vor der Entwicklung des MVPs durchführen, sieht aber einen zwischengeschobenen Iteration Sprint als wichtig an, um den Prototyp mithilfe des im Design Sprint gesammelten Feedbacks zu überarbeiten (Anhalt, Z. 241pp). Als Startup, das “relativ schnell mit einem MVP [. . .] an den Markt gehen muss” biete es sich jedoch an, den Component Sprint hinter die Entwicklung des MVPs zu schieben (Anhalt, Z. 384p).

(anonym) sah aus der Perspektive ihres Unternehmens mehr Wert darin, den Component Sprint einmalig für das Branding des gesamten Unternehmens anstatt nur für ein Produkt durchzuführen, sodass alle Teams bei der Entwicklung von neuen Projekten auf die selbe Pattern Library zugreifen können (anonym, Z. 226pp).

Aus Sicht der beiden Agenturen Duplexmedia und Nexus AG wurde die Abhängigkeit von Kunden und anderen Stakeholdern besonders betont. “Das Hauptproblem,

das ich immer sehe, auch bei den Design Sprints, ist, dass man es schafft, eine bestimmte Anzahl von Kunden wirklich für einen gewissen Zeitraum fokussiert an etwas arbeiten zu lassen, ohne Unterbrechung" (Mazur, Z. 280pp), beschrieb Mazur im Zusammenhang mit der Teamzusammenstellung der Component Sprints. Anhalt betonte: "[. . .] sobald du individueller wirst im Interface und auch viel näher Markenthemen berührst, dann hast du da einfach nochmal einen anderen Abstimmungsaufwand [. . .]" (Anhalt, Z. 289pp). Diese beiden Punkte sollten beim weiteren Ausbau des Konzepts beachtet werden.

9 Weiteres Vorgehen

Im Rahmen dieser Bachelorarbeit konnte nur eine Variante des Component Sprints empirisch getestet werden. Um das Konzept weiter zu optimieren, sollten weitere Component Sprints in verschiedenen Varianten durchgeführt und beurteilt werden.

Damit könnten beispielsweise folgende Fragen beantwortet werden, die sich aus den Ergebnissen des empirischen Tests und der Experteninterviews ableiten lassen:

- Wie verhält sich die Produktivität in Form von benötigten Personenstunden pro Komponente bei Sprints mit mehr Komponenten und/oder mehr Teilnehmern in einem kürzeren oder längeren Zeitraum?
- Welche Anpassungen müssen vorgenommen werden, damit das Konzept auch zwischen einem fertigen MVP und der Entwicklung des finalen Produkts oder projektunabhängig eingesetzt werden kann?
- Inwiefern können Entwickler und Designer auf Kundenseite mit in die Entwicklung einbezogen werden und welche Auswirkungen hat dies auf das Ergebnis?
- Wie kann die Abhängigkeit von mehreren Stakeholdern im Sprint gut bewältigt werden und wie kann sichergestellt werden, dass alle benötigten Informationen zum Beginn des Component Sprints verfügbar sind?

Zusätzlich dazu sollte die Nachhaltigkeit des Ergebnisses geprüft werden, indem die im Sprint entwickelte Pattern Library über einen längeren Zeitraum, beispielsweise über ein Jahr, im Entwicklerteam eingesetzt wird. Hier ist es wichtig, dass die Pattern Library nicht nur genutzt, sondern vom Entwicklerteam gepflegt wird. Ist dies nicht der Fall, könnte es vorkommen, dass sie nicht mehr ordentlich genutzt wird. Dadurch verliert sie ihren Wert, so wie es bei sipgates Pattern Library der Fall war.

Bei Crisp Studio wurde im Zeitraum dieser Arbeit bereits ein weiterer Component Sprint für nextAudit[1] durchgeführt, in dem der Inhalt des ersten Sprinttages auf eine Woche vor dem Sprint verlegt wurde. So stand mehr Zeit für die Rücksprache mit Stakeholdern und mehr Zeit zur Entwicklung während des Sprints zur Verfügung. Das Feedback seitens nextAudit war positiv, sodass es sich anbieten würde, diese Strukturänderung weiter zu verfolgen und zu testen.

Diese Tests überschreiten den Rahmen dieser Bachelorarbeit, könnten jedoch das Thema einer weiteren Bachelorarbeit sein oder unternehmensintern durchgeführt werden.

[1] `https://www.next-audit.de/`

10 Fazit / Zukünftige Ausbaumöglichkeiten

In dieser Arbeit wurde eine potentielle Lösung zur Frage erarbeitet, wie aus einem fertigen Prototyp in einem festen, kurzen Zeitraum eine Sammlung an Design Komponenten entwickelt werden kann, die Entwicklern des MVPs die Arbeit erleichtert.

Die Konzepte von Design Sprints, Atomic Design und Design Systems wurden verwendet, um das Konzept für einen eigenen Sprint unter dem Namen “Component Sprint” auszuarbeiten. Dieses Konzept wurde durch einen empirischen Test und darauf folgende Experteninterviews auf sein Potential und mögliche Probleme hin überprüft.

Für die Entwicklung des Konzepts war primär die Orientierung an den Prinzipien von Design Sprints hilfreich, die eine genaue, feste Sprintstruktur vorgeben. Durch die Orientierung an Atomic Design kann die im Component Sprint anstehende Arbeit strukturiert in kleine, meist voneinander unabhängige Aufgaben unterteilt werden.

Auch wenn noch einige Aspekte des Sprint-Konzepts überarbeitet und überprüft werden müssen, hat sich in den Tests gezeigt, dass dieses Konzept bereits in einer Woche ein für Unternehmen wertvolles Ergebnis liefern kann.

Um das Konzept zukünftig auszubauen, können weitere Variationen, die in Kapitel 9 beschrieben wurden, getestet werden. Hier ist es wichtig mehrere Component Sprint durchzuführen, um dabei reale Erfahrungen mit Kunden zu sammeln und auszuwerten. So kann das Konzept iterativ optimiert werden.

Dazu sollte das Konzept interessierten Firmen offen zur Verfügung gestellt werden, sodass sie es testen können und ein Diskurs über mögliche Problemstellen und Optimierungen entstehen kann.

Das ausgearbeitete Konzept ist nur eine von wahrscheinlich mehreren Lösungen zur Fragestellung dieser Arbeit. So könnten in Zukunft auch andere Konzepte entwickelt und getestet werden, die diese Fragestellung beantworten.

11 Anhang

Tabelle 11.1: Aufgeschlüsselte Entwicklungszeiten der einzelnen Komponenten

Komponente	Typ	Editor	Zeit	Commits
Avatar	Atom	Gunz	0:42:00	11
Button	Atom	Weiner	2:22:30	31
Datetime Picker	Atom	Weiner	4:57:00	13
Icon	Atom	Weiner	1:19:00	11
Input	Atom	Weiner	0:48:00	10
Primary Input	Atom	Bernard	0:18:00	4
Ribbon	Atom	Gunz	0:31:00	10
Template Icon	Atom	Bernard	0:30:00	4
Typography	Atom	Gunz	2:06:00	25
Button Input	Molecule	Gunz	0:57:00	8
Event Detail	Molecule	Gunz	1:33:00	10
Form	Molecule	Bernard	3:01:30	9
Header	Molecule	Bernard	0:34:30	6
Jumbo	Molecule	Gunz	1:13:30	15
RSVP	Molecule	Gunz	1:12:00	11
Stepper	Molecule	Bernard	2:27:00	11
Template Chooser	Molecule	Bernard	1:25:00	4
User	Molecule	Weiner	0:42:00	9
Well	Molecule	Gunz	0:16:30	6
Article	Layout	Weiner	0:24:00	1

Komponente	Typ	Editor	Zeit	Commits
Content Wrapper	Layout	Bernard	0:36:00	3
List	Layout	Bernard	0:57:00	2
Overlay	Layout	Bernard	1:27:00	4
Colors	Layout	Bernard	3:25:00	7
Landing Page	Template	Gunz	2:34:00	7
Register / Login	Template	Gunz	1:13:00	4
Create Event	Template	Weiner	2:22:30	4
Event Preview	Template	Weiner	2:27:00	2
Share	Template	Bernard	3:06:00	3

Tabelle 11.2: Anzahl der Produkte auf ProductHunt pro Monat von 2017-2019 für Abb. 11.1

Month	Number of submitted products
2017-02-01	596
2017-03-01	658
2017-04-01	538
2017-05-01	583
2017-06-01	565
2017-07-01	573
2017-08-01	641
2017-09-01	638
2017-10-01	674
2017-11-01	665
2017-12-01	588
2018-01-01	580
2018-02-01	660
2018-03-01	604
2018-04-01	566
2018-05-01	701

Month	Number of submitted products
2018-06-01	612
2018-07-01	611
2018-08-01	656
2018-09-01	590
2018-10-01	702
2018-11-01	624
2018-12-01	512
2019-01-01	616
2019-02-01	642
2019-03-01	631
2019-04-01	679
2019-05-01	642
2019-06-01	540
2019-07-01	570
2019-08-01	591

Die rohen Daten zu Abb. 11.1 wurden mit einem Web Scraper von ProductHunt bezogen und mit Exploratory[1] zu einer Grafik verarbeitet. Sie befinden sich zusammen mit dem Code des Scrapers auf dem beigelegten Datenspeicher und online.[2]

Die Transkripte der Interviews befinden sich ebenfalls auf dem beigelegten Datenspeicher und online.[3]

[1] https://exploratory.io/

[2] https://github.com/leolabs/bachelor/blob/master/crawlers/producthunt-crawler/results.json

[3] https://github.com/leolabs/bachelor/tree/master/interviews

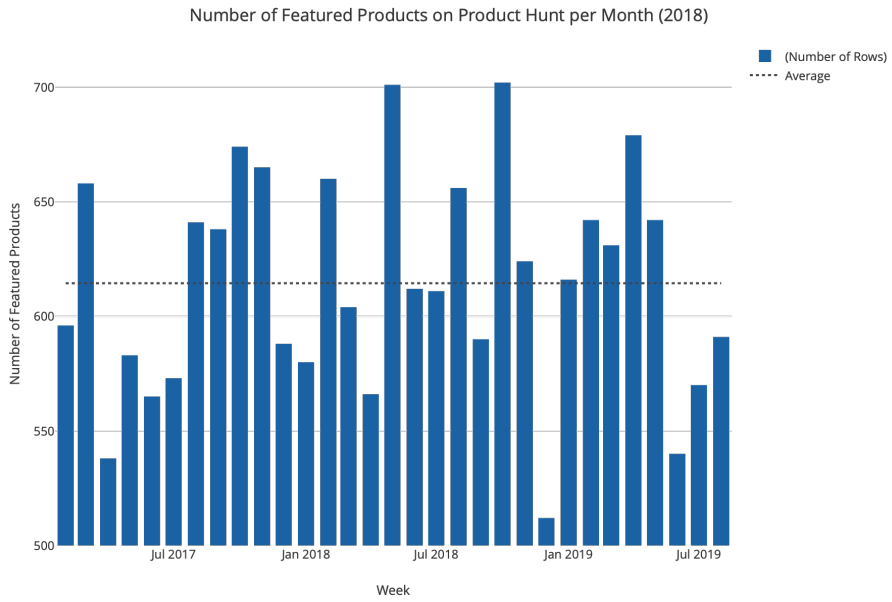

Abb. 11.1: Anzahl der Produkte auf ProductHunt pro Monat (2017-2019) [eigene Darstellung]

Literatur

AJ&Smart (2018a). *AJ&Smart Design Sprint Checklist*. URL: https://aj-smart.teachable.com/courses/307155/lectures/4764967.

– (2018b). *How the Iteration Sprint Works*. URL: https://aj-smart.teachable.com/courses/307155/lectures/4729041 (besucht am 21. 09. 2019).

Blank, Steve (4. März 2010). *Perfection By Subtraction – The Minimum Feature Set*. URL: https://steveblank.com/2010/03/04/perfection-by-subtraction-the-minimum-feature-set/ (besucht am 13. 08. 2019).

Bogdoll, Daniel und Benjamin Dörries (2019). *SAYM | Everyday Carpooling Simplified*. URL: https://saym.io (besucht am 12. 09. 2019).

Brooks, Frederick P. (1995). *The Mythical Man-Month: Essays on Software Engineering*. Anniversary ed. Reading, Mass: Addison-Wesley Pub. Co. 322 S. ISBN: 978-0-201-83595-3.

Coplien, James O (Aug. 1994). „A Development Process Generative Pattern Language“. In: S. 34.

Crinnion, John (1992). *Evolutionary Systems Development: A Practical Guide to the Use of Prototyping Within a Structured Systems Methodology*. Perseus Publishing. ISBN: 0-306-44139-X.

Cruchon, Stéphane (2. Mai 2019). *The Design Sprint Note-n-Map*. URL: https://sprintstories.com/the-design-sprint-note-n-map-a9bf0ca88f51 (besucht am 24. 07. 2019).

DockYard, Inc (2017). *Design Sprints | What's next*. URL: https://dockyard.com/design-sprints/whats-next/ (besucht am 23. 09. 2019).

Dudenredaktion (o. J.) (2019). *Prototyp*. In: *Duden online*. URL: https://www.duden.de/node/115811/revision/115847 (besucht am 30. 07. 2019).

Dybå, Tore, Torgeir Dingsøyr und Nils Brede Moe (2014). „Agile Project Management“. In: *Software Project Management in a Changing World*. Hrsg. von Günther Ruhe und Claes Wohlin. Berlin, Heidelberg: Springer Berlin Heidelberg, S. 277–300. ISBN: 978-3-642-55034-8. DOI: 10.1007/978-3-642-55035-5_11.

Frost, Brad (2016). *Atomic Design*. OCLC: 971562254. ISBN: 978-0-9982966-0-9.

Git - Book (2019). URL: https://git-scm.com/book/en/v2 (besucht am 28. 08. 2019).

Gläser, Jochen und Grit Laudel (15. Juli 2010). *Experteninterviews und Qualitative Inhaltsanalyse*. VS Verlag. 352 S. ISBN: 978-3-531-17238-5.

Google Ventures (2019). *The Design Sprint — GV*. URL: http://www.gv.com/sprint (besucht am 16. 07. 2019).

Jake Knapp – The Design Sprint: One Small Change to Create a Culture of Innovation (13. Mai 2019). URL: https://www.youtube.com/watch?v=KX7tc4UPnI&feature=youtu.be (besucht am 23. 09. 2019).

Kahneman, Daniel (2013). *Thinking, Fast and Slow*. Penguin.

Kholmatova, Alla (2017). *Design Systems: A Practical Guide to Creating Design Languages for Digital Products*. OCLC: 1010241383. Freiburg: Smashing Media AG. 288 S. ISBN: 978-3-945749-58-6.

Knapp, Jake (30. Okt. 2018). „How to Stop Wasting Time—Jake Knapp at Amplify“. Self Improvement. URL: https://www.slideshare.net/amplitudemobile/how-to-stop-wasting-timejake-knapp-at-amplify (besucht am 19. 08. 2019).

Knapp, Jake, John Zeratsky und Braden Kowitz (2016). *Sprint: How to Solve Big Problems and Test New Ideas in Just Five Days*. First Simon & Schuster hardcover edition. OCLC: ocn913304205. New York: Simon & Schuster. 274 S. ISBN: 978-1-5011-2174-6.

Larman, C. (2004). *Agile and Iterative Development: A Manager's Guide*. Agile Software Development Series. Addison-Wesley. ISBN: 978-0-13-111155-4. URL: https://books.google.de/books?id=76rnV5Exs50C.

Lenarduzzi, V. und D. Taibi (Aug. 2016). „MVP Explained: A Systematic Mapping Study on the Definitions of Minimal Viable Product“. In: *2016 42th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. 2016 42th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), S. 112–119. DOI: 10.1109/SEAA.2016.56.

Mattscheck, Markus (2019). *Customer Journey - Definition | Onlinemarketing-Praxis*. URL: https://www.onlinemarketing-praxis.de/glossar/customer-journey (besucht am 26. 09. 2019).

Minimum Viable Product (6. Aug. 2019). In: *Wikipedia*. Page Version ID: 191100310. URL: https://de.wikipedia.org/w/index.php?title=Minimum_Viable_Product&oldid=191100310 (besucht am 15. 08. 2019).

Nauheimer, René (23. Mai 2019). *Mister Spex – Rethinking The Way We Buy Eyewear Today*. URL: https://blog.crisp.studio/mister-spex-%E2%80%93-rethinking-the-way-we-buy-eyewear-today/ (besucht am 24. 07. 2019).

Newzoo (11. Sep. 2018). *Number of Smartphone Users Worldwide from 2016 to 2021 (in Billions)*. URL: https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/ (besucht am 24. 09. 2019).

PocketGamer.biz (2019). *Number of Available Apps in the Apple App Store from 2008 to 2019 (in 1,000s)*. URL: https://www.statista.com/statistics/268251/number-of-apps-in-the-itunes-app-store-since-2008/ (besucht am 24. 09. 2019).

Poguntke, Sven (2019). *Corporate Think Tanks Elektronische Ressource: Zukunftsforen, Innovation Center, Design Sprints, Kreativsessions & Co.* 3. Aufl. 2019. Wiesbaden: Springer Fachmedien Wiesbaden, Imprint: Springer Gabler. ISBN: 978-3-658-25663-0. DOI: 10.1007/978-3-658-25663-0.

Ries, Eric (3. Aug. 2009). *Minimum Viable Product: A Guide*. URL: http://www.startuplessonslearned.com/2009/08/minimum-viable-product-guide.html (besucht am 13. 08. 2019).

Robinson, Frank (2001). *Minimum Viable Product | SyncDev*. URL: http://www.syncdev.com/minimum-viable-product/ (besucht am 13. 08. 2019).

Salesforce (23. Juli 2019). *Design Guidelines - Lightning Design System*. URL: https://www.lightningdesignsystem.com/guidelines/overview/ (besucht am 25. 08. 2019).

sipgate GmbH (27. März 2018). *Warum unsere Pattern Library kein Atomic Design mehr benutzt*. URL: https://www.sipgate.de/blog/warum-unsere-pattern-library-kein-atomic-design-mehr-benutzt (besucht am 21. 08. 2019).

– (2019). *Purpose first*. URL: https://www.sipgatedesign.com/pattern-library/purpose-first (besucht am 25. 08. 2019).

Smart, Michael und Jonathan Courtney (14. Feb. 2018). *Design Sprint 2.0 | What Is Google Design Sprint Process and Framework*. URL: https://ajsmart.com/design-sprint-2-0/ (besucht am 24. 07. 2019).

Storybook (2019). *Automated Visual Testing*. URL: https://storybook.js.org/docs/testing/automated-visual-testing/ (besucht am 26. 09. 2019).

Storybook (2019). *Storybook: UI Component Explorer for Frontend Developers*. URL: https://storybook.js.org (besucht am 26. 08. 2019).

Verner, June und Mahil Carr (o.D.). *Prototyping and Software Development Approaches*.