Woche 3
Features brainstormen, MVP und GUI Unterstützung für den / die Moderator:in

In dieser dritten Woche versuchen wir, etwas früher als sonst, zu klären was wir eigentlich fachlich bauen sollen. Wir sammeln die Anforderungen nun in einem Product Backlog, dieser kann ausgeprägt sein als Sammlung von Haftnotizen, als Tablelle in einem Word-Doument, als Excel-Sheet oder bereits als Tickets. Wir versuchen daraus eine schlüssige Auswahl zu finden, diese Auswahl heißt MVP (Minimal Viable Product) oder MMF (Minimal Marketable Featureset). Sie konkretisieren ihren Vorschlag für das MVP mithilfe eines ersten Entwurfs für das Interaktionsdesign.

Am Ende dieser Workshops sollten sie im Team ein klares Verständnis davon haben, was genau ihre Software leisten soll bzw. was die anderen Deliverables enthalten müssen. Wichtig wäre auch ein gutes „Bauchgefühl", dass das von ihnen vorgeschlagene MVP auch in der Zeit bis zum Semesterende für sie schaffbar ist.

Folgende Ergebnisse wären gut:

- Liste von Anforderungen, so dass sie diese auch ins Angebot übernehmen können (Tabelle)
- MVP (= Überschrift welche zusammenfasst, was das MVP leisten kann und eine Liste der entsprechenden Anforderungen
- Interaktionsentwurf / Taskflow (ggf. nur als einfache Kasten-und-Pfeile Grafik alternativ auch als Dialoglandkarte möglich)
- Wireframes, wenn sie eine Grafische Oberfläche habens

Ausblick: Sie schreiben Ihre Workshopergebnisse dann Mitte November in ihr Angebot / Grobkonzept in Form einer ***Leisungsbeschriebung bzw. als Beschreibung der Deliverables***. Was wird ihre Software bzw. was werden ihre anderen Ergebnisse für Eigenschaften bzw. Features haben? Die User Story Map hilft Ihnen dabei, für die Leisungsbeschreibung/Deliverables, also das MVP, eine schlüssige Story zu finden. Der in der User Story Map definierte „Narrative Flow" ist genau der rote Faden an dem Sie entlang Ihre Lösung beschreiben (mit wenigen Sätzen).

1. Brainstorming der Features (Backlog erstellen)

Wichtig für Anforderungen sind zwei Eigenschaften:

1. Die Anforderung erhält eine eindeutige Nummer (ist identifizierbar)

2. Die Anforderung kann gegen das gelieferte System geprüft werden, z.B. mithilfe von Akzeptanzkriterien

Sie haben ja vom Auftraggeber schon input erhalten. Eventuell müssen sie diesen noch veredeln bzw. übersetzen. Es bietet sich an, die Anforderungen als Excel-Liste zu erfassen. Denkbar sind aber auch (wenn es wenige Anforderungen sind) ein (online) Whiteboard mit Haftnotizen.

Im Workshop können sie eventuell noch fehlende Anforderungen erhänzen. Sie sollten aber in jedem Fall die bereits bekannten Anforderungen im Team überprüfen.

Die Anforderungsliste wird in der einen oder anderen Form Teil ihres Angebots.

2. Minimal Marketable Feature Set entscheiden

In keinem realen Projekt können *alle* Anforderungen in ein reales System umgesetzt werden, das wäre zu teuer und würde zu lange dauern. Ihr Ziel ist ein System, das Sie innerhalb von fünf Minuten Ihren Eltern oder ihren Nicht-IT-Freunden erklären können, daher ist Klarheit und Fokussierung besonders wichtig.

Gehen Sie Ihre Anforderungen durch, insbesondere die Pflichtanforderungen (Must). Entscheiden Sie, welche davon Sie bis zum Ende des Semesters umsetzen können. In den nächsten beiden Iterationen haben sie ***maximal noch 12-15 Arbeitstage pro Person*** zur Verfügung! Diese Anforderungen bilden das Minimal Marketable Feature Set. Entscheiden Sie nach folgenden Kriterien:

1. Welche Anforderungen haben den **größten Nutzen** für Ihren Kunden? Der Nutzen kann auch ein Erkenntnisgewinn sein und nicht notwendig laufende Software!

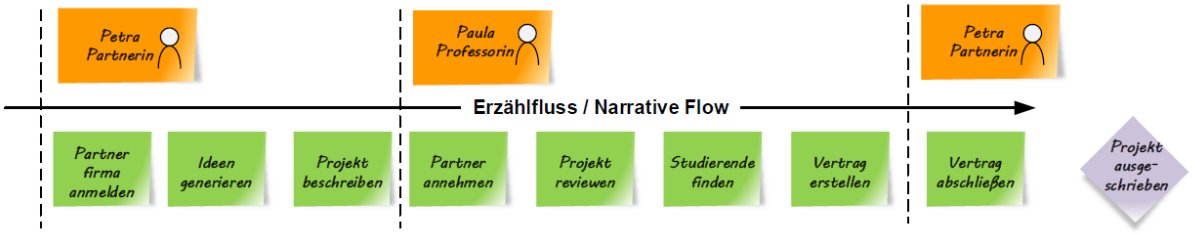
2. Welche Anforderungen bergen die **größten Risiken** für Sie und / oder Ihren Kunden?

3. Wenn diese Anforderung weggelassen wird, hat das System keinen Nutzen mehr. Bzw. welche Anforderungen können Sie weglassen, ohne dass Ihr Auftraggeber Nutzen verliert?

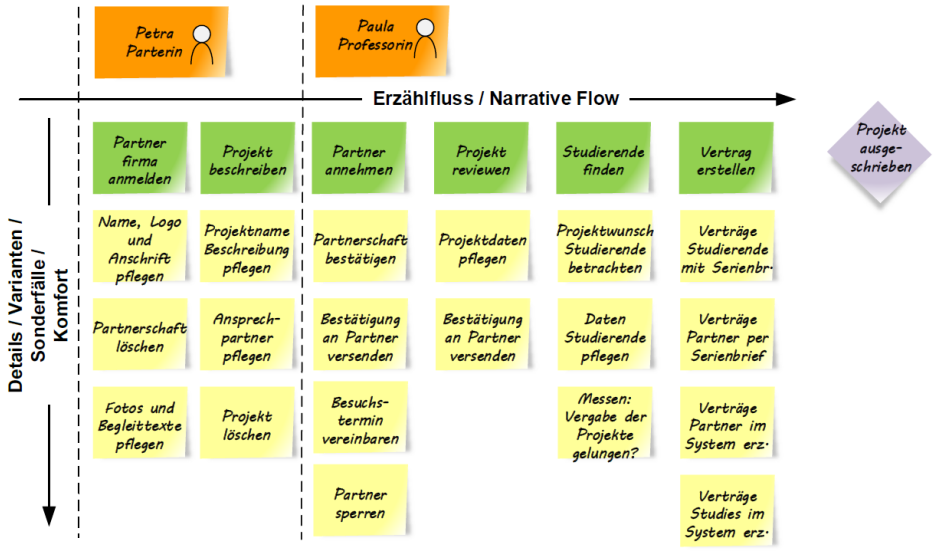
Bitte beachten Sie: Nicht alle Anforderungen müssen in Software umgesetzt werden. Sie können ja auch Sachbearbeiter oder Monteure schulen und diese dann Teile manuell erledigen lassen (vgl. Autobahn-Metapher). Eventuell ist die Automatisierung durch Software auch langfristig teurer als der manuelle Prozess. Versuchen Sie mit der Software die schlimmsten „Pain-Points“ ihrer Nutzer zu beheben. Worüber würden sich Ihre Benutzer langfristig (!) freuen? Schauen Sie sich dazu eventuell noch mal den Katzenfutter-Versand auf YouTube an (https://www.youtube.com/watch?v=jHyU54GhfGs).

Erstellen Sie eine gemeinsame User Story Map (z.B. mit Miro, Workshop)

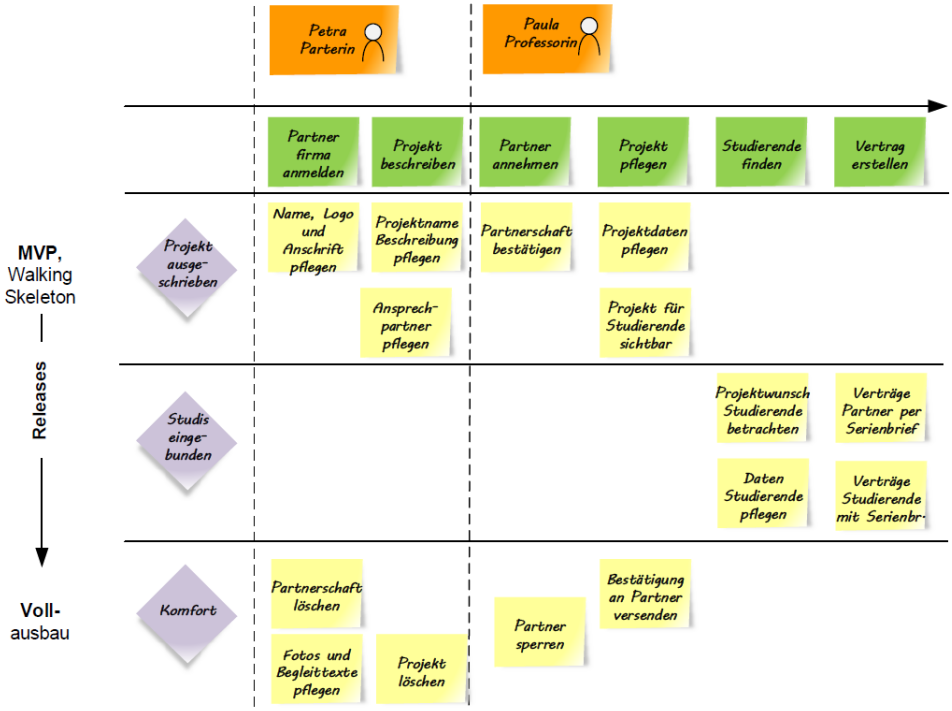
Sehr hilfreich zur Definition des MVP/MMF hat sich die Technik der User Story Maps erwiesen. Sie überlegen, was will ihre Kernpersona mit ihrem System erreichen und leiten daraus einen / den „Narrative Flow“ ab.



Sie Brainstormen dann die Teile, die Sie in Software tun könnten, also die Features ihrer möglichen Software:



Im dritten Schritt sortieren sie die Features nach Releases.



Dokumentieren sie ihr MVP in Form von Tickets.

Erstellen Sie dann zu jeder Anforderung aus dem MVP ein Ticket in GitLab und vergeben Sie auch ein neues Label mit dem Titel „User Story“ und für sehr große Anforderungen „Epic“ sowie „Anforderung“ für Anforderungen, die Sie nicht als User Story formulieren können.

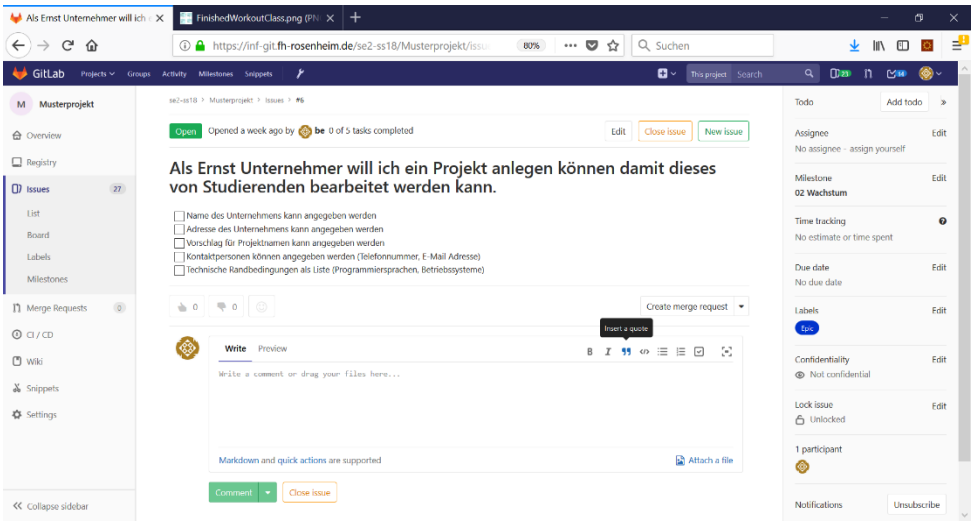


Abbildung 1: User Story mit Akzeptanzkriterien in GitLab

3. Planung der Interaktion mit dem Benutzer

Nutzer „mögen“ GUI-Entwürfe normalerweise sehr gerne. Damit sind die Prototypen / Wireframes auch gut als Bebilderung für ihr Angebot. Damit können sie gut illustrieren, was sie dem Auftraggeber fachlich vorschlagen. Achten sie beim Übernehmen der Entwürfe darauf, dass sie zu den Bildern etwas Text ergänzen, der jeweils erklärt, was sie sich zu dem jeweiligen Dialog überlegt haben.

Einige unserer Systeme verfügen nicht über eine grafische Oberfläche, sondern über ein Sprachinterface, wenige Buttons in Hardware oder eine komplexe VR/AR Interkation. Daher sind die im folgenden dargestellten Techniken nicht universell einsetzbar. Außerdem ist es leicht sich in den ersten grafischen Entwürfen mit Kleinigkeiten zu verzetteln Daher starten wir hier abstrakter.

1. Erstellen Sie eine **Liste mit einem oder mehreren Zielen**, die ihr Anwender mithilfe des Systems erreichen will. Für diese Ziele überlegen wir nun, wie die Interaktion stattfinden soll. Als Grundlage dient der Narrative Flow aus dem User Story Mapping.
2. Überlegen Sie sich einen Dialog zwischen Ihrem System und dem Benutzer. Was gibt der Benutzer ein, bzw. was sagt er oder welche Geste macht er, wie reagiert darauf das System? Wie reagiert wiederum der Benutzer. Beschreiben Sie diesen Ablauf zunächst in Form einer Liste (wie sie es von den Use Cases [Happypath] aus Software-Engineering 1 gewohnt sind).

   Beispielsweise:

Schritt	**Benutzer**	**System**
Ziel:	Blutdruck erfassen (= Intent)	
1.	„Hallo Alexa“	Alexa leuchtet blau in Richtung Sprecherin.
2.	„Starte Gedundheitstagebuch“	„Gesundheitstagebuch gestartet
3.	„Mein Blutdruck ist 180 zu 120“	„Ich habe verstanden, dass dein Blutdruck 180 zu 120 ist“, „Stimmt das.
4.	„Korrekt“	Alexa schaltet sich wieder aus.

   Ähnlich können Sie auch die Interaktion mit einer grafischen Oberfläche beschreiben oder mit Objekten iner virtuellen Welt, sowie mit Buttons oder Sensoren auf ihrem Embedded Device. Achten Sie beim Aufschreiben darauf, dass sie noch keine Implementierungsdetails preisgeben. Also statt „drückt Bestätigen Button“ eher „Benutzer bestätigt die Eingabe“ (eventuell wollen Sie ja später keinen Button haben, sondern vielleicht eine Swipe-Geste).

   Auch die Reaktionen des Systems können variieren, von einem Satz, den das System sagt, bis hin zu 3D-Objekten oder einfachen Leuchtdioden ist alles denkbar, was eine Rückmeldung an den Benutzer gibt.
3. Sie überlegen Sich nun Sonderfälle und Ausnahmesituationen, im Beispiel könnte Alexa ja den Blutdruck nicht verstanden haben, oder das Tagebuch ist nicht gestartet. Möglicherweise gibt es alternative Abläufe. Für wenig komplexe Sonderfälle kommen Sie mit der tabellarischen Darstellung aus.

   Für komplexe, umfangreiche Abläufe benötigen Sie eine umfangreichere Notation. Hierfür bieten sich Aktivitätsdiagramme oder ggf. Zustandsdiagramme der UML 2.x an. Die Aktionen des Aktivitätsdiagramms können dann die Benutzerinteraktionen darstellen, übergebene Daten können dort explizit als solche modelliert werden. Komplexere Kontrollflüsse sind möglich.

   Alternativ zur obigen Tabelle können Sie auch eine Dialoglandkarte zeichnen (siehe unten) oder einen Taskflow spezifizieren, als einfaches Box&Arrow Diagramm oder alternativ als Flow-Chart oder UML-Aktivitätsdiagramm. Als Werkzeuge stehen ihnen: PlantUML oder draw.io zur Verfügung.

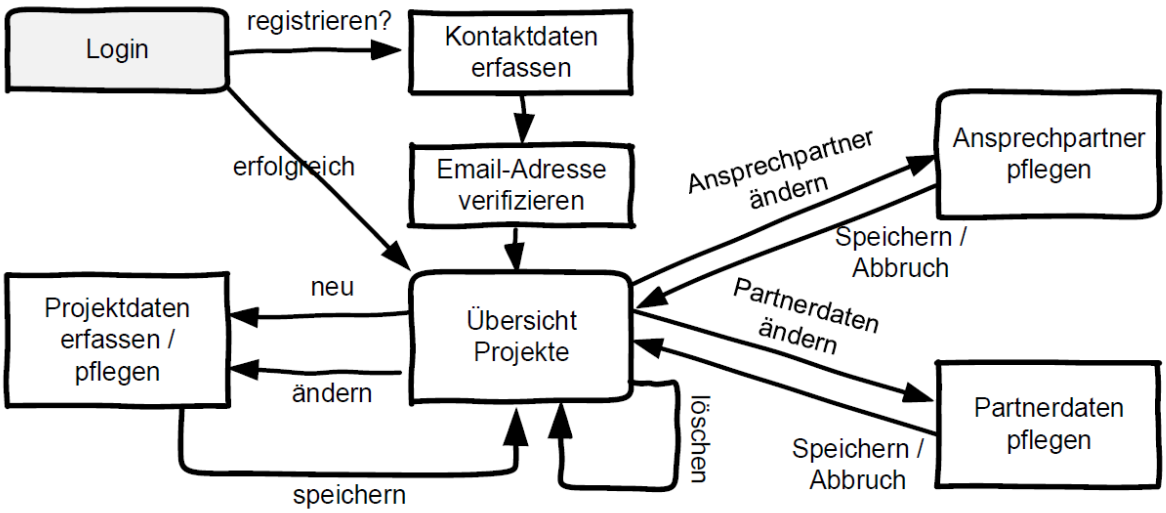


Abbildung 2: Beispiel für einen Taskflow

4. Maskenlayouts erstellen (Wenn sie eine GUI haben)

Überlegen Sie nun, wie sie die Interaktionen aus Aufgabe 3 zu Masken / Szenen / Dialogen zusammenfassen können. Erstellen Sie mit möglichst mit Stift und Papier einige Handskizzen, wie ihre Oberfläche aussehen könnte. Eventuell können sich alle Teammitglieder hier beteiligen. Versuchen sie sich danach auf einen gemeinsamen Entwurf zu einien und stimmen sie diesen mit ihrem Auftraggeber ab.

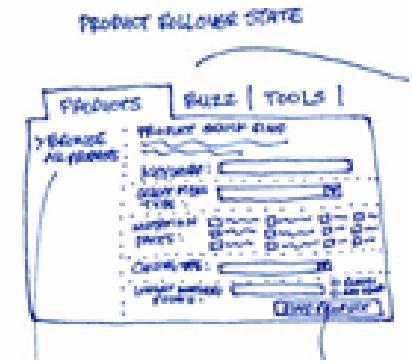


Abbildung 3: Handskizze von GUI Masken (https://www.flickr.com/photos/typehunter/4595959073)

Eine Umsetzung in einem Programm Ihrer Wahl (z.B. Balsamiq, Figma, Adobe XD oder ggf. GUI-Builder) ***kann*** später erfolgen**, wir kümmern uns hier erstmal um die wesentlichen Eckpunkte der Oberfläche**, da sind wir mit Handskizzen schneller. Beachten Sie dabei die Grundregeln der Ergonomie! Erstellen Sie für jede Maske (Fotos ihrer Zeichnungen) eine eigene Wiki-Seite. Verlinken Sie die Wiki-Seite mit den entsprechenden User Storys als Ticket.

5. Dialogablauf darstellen, Dialoglandkarte

Erstellen Sie ein Übersichtsdiagramm, das das Zusammenspiel Ihrer Masken zur Laufzeit darstellt. Dieses Diagramm wird auch als Screenflow oder als Dialoglandkarte bezeichnet. Also mit welchen Aktionen zwischen den Masken navigiert wird. Die nachfolgende Abbildung zeigt eine an die Pinnwand gezeichnete Dialoglandkarte. Ihre Karte sollte zusätzlich die Buttons/Funktionen darstellen, mit deren Hilfe Sie zwischen den Masken navigieren. Machen Sie ein Foto von Ihrem Entwurf und erstellen Sie eine entsprechende Wiki-Seite. Die Dialoglandkarte ist eine Alternative zum Taskflow aus Aufgabe 3.

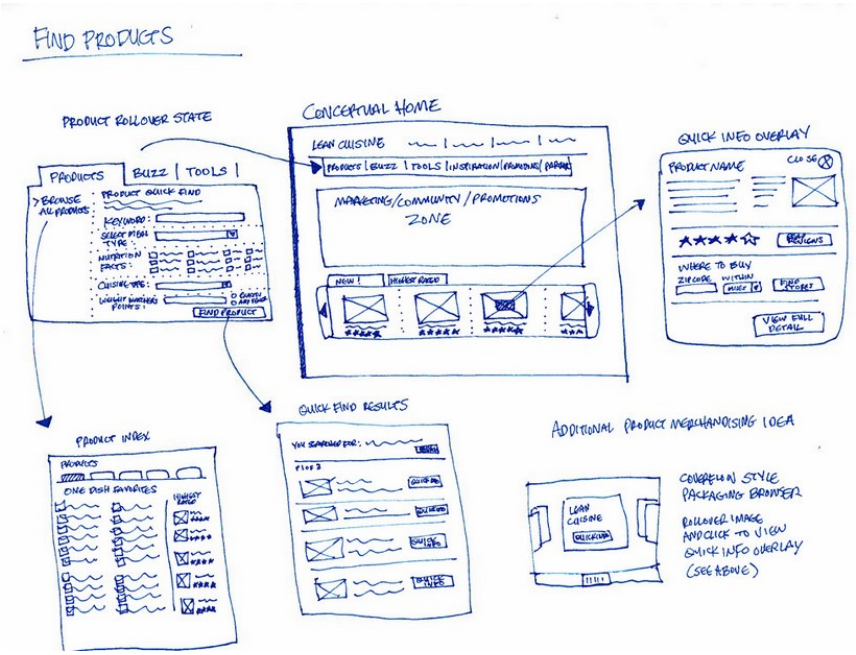


Abbildung 4: Dialoglandkarte als Handskizze (https://www.flickr.com/photos/typehunter/4595959073)