

Fakultät Sprach- , Literatur- und Kulturwissenschaften

Institut für Information und Medien, Sprache und Kultur (I:IMSK)  
Lehrstuhl für Medieninformatik

Praxisseminar Master Medieninformatik

MEI-M 26.1 (M. Sc.)

SS 2018

Leitung: Florin Schwappach

Usability Testing Plattform – OBS Plugin

Dominik Deller

Matr.-Nr: 1679917

3. Semester M.Sc. Medieninformatik

Email: dominik.deller@stud.uni-regensburg.de

Julia Grötsch

Matr.-Nr: 1673540

3. Semester M.Sc. Medieninformatik

Email: julia.groetsch@stud.uni-regensburg.de

Khang Ho

Matr.-Nr: 1686639

3. Semester M.Sc. Medieninformatik

Email: khang.ho@stud.uni-regensburg.de

Philipp Weber

Matr.-Nr: 1699855

3. Semester M.Sc. Medieninformatik

Email: philipp1.weber@stud.uni-regensburg.de

Abgegeben am 07.10.2018

Inhalt

[1 Einleitung 4](#_Toc526587782)

[2 Angaben zu Ressourcen und Dokumenten 4](#_Toc526587783)

[3 Problemstellung und Projektbeschreibung 4](#_Toc526587784)

[4 Projektmanagement 5](#_Toc526587785)

[4.1 Agiles Projektmanagement 5](#_Toc526587786)

[4.2 Projektmanagement-Tool 6](#_Toc526587787)

[4.2.1 Taiga 6](#_Toc526587788)

[4.2.2 Asana und Instagantt 7](#_Toc526587789)

[4.2.3 Projektplan 7](#_Toc526587790)

[4.2.4 Codeverwaltung 9](#_Toc526587791)

[4.3 Meetings 9](#_Toc526587792)

[4.3.1 Meetings mit Kursleiter und Stakeholder 9](#_Toc526587793)

[4.3.2 Sprints 12](#_Toc526587794)

[5 Architektur und Implementierung 15](#_Toc526587795)

[5.1 Anforderungsanalyse 16](#_Toc526587796)

[5.1.1 Contextual Inquiry 16](#_Toc526587797)

[5.1.2 User Stories 16](#_Toc526587798)

[5.1.3 Mock-Ups 18](#_Toc526587799)

[5.2 Implementierung 21](#_Toc526587800)

[5.2.1 Funktionalitäten 22](#_Toc526587801)

[5.2.2 Systemarchitektur 26](#_Toc526587802)

[5.3 Spezifikation 27](#_Toc526587803)

[6 Evaluation 27](#_Toc526587804)

[6.1 Testing mit Mitkommilitonen 27](#_Toc526587805)

[6.2 Evaluation mit Stakeholder 29](#_Toc526587806)

[7 Setup-Anweisungen 30](#_Toc526587807)

[8 Ausblick 30](#_Toc526587808)

Abbildungen

[Abbildung 1 - Taiga Übersicht 6](#_Toc526587762)

[Abbildung 2 - Anforderungsanalyse 7](#_Toc526587763)

[Abbildung 3 – Projektplanung 7](#_Toc526587764)

[Abbildung 4 – Projektdurchführung 8](#_Toc526587765)

[Abbildung 5 – Design 8](#_Toc526587766)

[Abbildung 6 – Evaluation 8](#_Toc526587767)

[Abbildung 7 - Sonstige Miletones 8](#_Toc526587768)

[Abbildung 8 - Projektplanübersicht 9](#_Toc526587769)

[Abbildung 9: Hauptseite 18](#_Toc526587770)

[Abbildung 10 – Testleiter Ansicht im Mockup 19](#_Toc526587771)

[Abbildung 11 – Testpersonen Ansicht im Mockup 20](#_Toc526587772)

[Abbildung 12 – Export Seite im Mockup 21](#_Toc526587773)

[Abbildung 13 – Hauptseite der Anwendung 22](#_Toc526587774)

[Abbildung 14 – Tutorial Seite der Anwendung 22](#_Toc526587775)

[Abbildung 15 – Testleiter Ansicht der Anwendung 23](#_Toc526587776)

[Abbildung 16 Testmanager – Einen bestehenden Test wählen 24](#_Toc526587777)

[Abbildung 17 Testmanager – Einen neuen Test anlegen 24](#_Toc526587778)

[Abbildung 18 Testmanager – Tasks 25](#_Toc526587779)

[Abbildung 19 Testmanager – Einen freien Test starten 25](#_Toc526587780)

[Abbildung 20 – Taskansicht auf Testperson Seite 26](#_Toc526587781)

# Einleitung

Evaluierungen sowie Usability Testing sind ein zentraler Bestandteil des Softwareengineering Prozesses, weshalb Remote-Testing ein sehr wichtiger Aspekt ist, da nicht jeder Usability Test vor Ort absolviert werden kann. Für dieses Szenario existiert noch keine Open Source Software an der Universität Regensburg, weshalb eine Implementierung einer solchen Software erwünscht war. Die gestellte Aufgabe war es, eine Usability Testing Plattform zu gestalten, die es dem Nutzer ermöglicht, mit möglichst wenig Aufwand einen Usability Test zu absolvieren, welcher sowohl lokal als auch online stattfinden kann. Dies wurde mit einer Homepage realisiert, die vom Testleiter sowie Proband abgerufen werden muss. Dabei unterscheiden sich die beiden Ansichten. Der Testleiter hat mehr Funktionalitäten zur Verfügung und sieht zusätzlich den Bildschirm des Probanden, während der Proband zwei Monitore benötigt, um auf einer Seite seine Tasks durchzuführen und auf der anderen Seite mit dem Testleiter zu kommunizieren. Die einzigen konkreten Kosten, die sich hierbei bilden, wären die Kosten für einen Server, welcher aber auf einem Server des Medieninformatik Lehrstuhls realisiert werden könnte. Diese Software kann von jeder Person genutzt werden, hierbei werden keinerlei Lizenzen oder sonstige Berechtigungen benötigt. Durch zusätzliche Plugins soll es zudem möglich sein, weitere Funktionalitäten hinzuzufügen.

# Angaben zu Ressourcen und Dokumenten

Um diese Software nutzen zu können, werden einige zusätzliche Elemente benötigt. Da die jeweiligen Tests nicht immer lokal ablaufen, sondern zum Teil auch über größere Distanzen, wird ein Server benötigt, um über diesen den Stream zu hosten. Testweise wurde dies mit der „Digital Ocean“ Plattform realisiert, zur realen Inbetriebnahme dieser Software sollte der Stream auf einem eigenen Server gehostet werden. Auf dem Server müssen PHP, Apache2, NGINX und MySQL installiert werden. Eine genaue Erläuterung zur Inbetriebnahme der Software sowie alle benötigten Schritte entnehmen Sie bitte dem beigefügten Dokument. Theoretisch wäre es möglich über einen Raspberry Pi zu streamen, doch dieser würde nach längerer Zeit überlasten, weshalb dies keine sinnvolle Alternative ist.

# Problemstellung und Projektbeschreibung

Der Fokus bei der Entwicklung dieser Anwendung lag auf einem Weg eine möglichst kostenfreie Usability Plattform zu gestalten, damit verschiedene Nutzergruppen diese Software nutzen können. Hierbei sollen grundlegende Elemente der Usability im Vordergrund stehen: Es soll möglich sein, Tasks zu erstellen und eine Kommunikation zwischen Testleiter und Proband existieren. Dies ist einerseits durch einen Chat realisiert, aber wir empfehlen für einen realen Usability Test Voice Chat zu benutzen, damit die Kommunikation schneller und effizienter ablaufen kann. Unsere Zielgruppe sind zunächst Studenten und Dozenten der Universität Regensburg, die ihre Tests Remote absolvieren wollen. Aber auch andere Nutzergruppen sollten ohne Probleme die Anwendung nutzen können.

Es hat sich relativ schnell herausgefiltert, dass eine Implementierung der Anforderungen rein auf OBS Seite nicht möglich ist, daher wurde früh entschieden, die meisten Funktionalitäten auf einer Homepage auszulagern. Daher ist ein Server unbedingt nötig, um die Anwendungen zu nutzen. Das Projekt umfasst eine Implementierung der Anwendung auf Server Ebene. Hierbei existiert eine Webseite mit verschiedenen Ansichten, welche für die jeweiligen Teilnehmer (Testleiter sowie Proband) einige Funktionalitäten bietet.

# Projektmanagement

Das Projektmanagement ist ein wesentlicher Bestandteil des Softwareengineering Prozesses und ist vor allem für die organisatorische Ebene zuständig. Dabei ist es von besonderer Bedeutung, einen Zeitplan sowie einen Meilensteinplan zu erstellen, um den Fortschritt des Projektes zu gewährleisten und diesen auch einzuhalten, um weitere Kosten zu vermeiden. Dabei wurde eine leicht abgeänderte Form des agilen Projektmanagements benutzt.

## Agiles Projektmanagement

Als Projektmanagement Methode wurde die SCRUM Methode gewählt (vgl. Projektmanagement: Definitionen, Einführungen und Vorlagen), wobei hier nicht strikt alle Vorgaben des SCRUM Modells eingehalten wurde, sondern eine abweichende Form des SCRUMs angewendet wurde, die besser an die Projektlage angepasst war. Wöchentliche Sprint-Meetings wurden gehalten, wobei diese teils in Person und teils via Kommunikationsapplikationen (Teamspeak) absolviert worden sind.

Im Laufe des Projektes wurde das Projektmanagement nach und nach an die Bedürfnisse angepasst. Zunächst haben wurden wöchentlichen Spring-Meetings abgehalten, um den Fortschritt des Projektes zu gewährleisten. Dies wurde aber gegen Mitte der Projektphase zunehmend schwieriger, aufgrund von persönlichen Gründen. In dieser Zeit fanden die Meetings sehr unregelmäßig statt und es wurde meist nur über Teamspeak kommuniziert. Aus Erfahrung lässt sich sagen, dass persönliche Treffen effektiver sind, da man nicht mit der Versuchung in Berührung kommt, unkonzentriert zu arbeiten. Teamspeak Meetings sind hingegen besser geeignet, wenn man nur einen kurzen Austausch anstrebt bzw. die Arbeitspakete untereinander austauschen wollte. Letztendlich ist eine Mischung aus beidem die beste Lösung, da die Arbeitspakete unterschiedlich viel Zeit beanspruchen. Da es sich bei der gewählten Methode um ein agiles Management handelt, war es dadurch möglich etwas flexibler zu handeln. Gegen Ende war eine intensivere Meeting-Dichte zu verzeichnen.

## Projektmanagement-Tool

Als Projektmanagement Tools werden meist externe Softwares bezeichnet, die zur Dokumentation des Projektes verwendet werden. Dabei werden Milestones sowie Arbeitspakete unterteilt, um eine bessere Übersicht zu bekommen und den Fortschritt des Projektes zu gewährleisten. Zunächst wurde mit dem Tool „Taiga“ (taiga.io) gearbeitet. Nach dem ersten Monat kam es zu der Entscheidung auf die Plattformen Asana (asana.com) und Instagantt (instagantt.com) umgestiegen, da diese besser an die Anforderungen angepasst sind.

### Taiga

Das Projektmanagement wurde zunächst auf Taiga durchgeführt, hier wurden die ersten Milestones und sämtliche Abläufe der bis zu diesem Zeitpunkt absolvierten Meetings dokumentiert. Im Laufe des Projektes wurde jedoch festgestellt, dass es besser angepasste Alternativen für das Projekt existieren, weshalb schlussendlich zu Asana bzw. Instagantt gewechselt wurde. Diese bieten die Möglichkeit aus dem Projektplan einen Milestone Plan bzw. ein Ganttdiagramm zu generieren, was den Managementaufwand deutlich reduzierte.

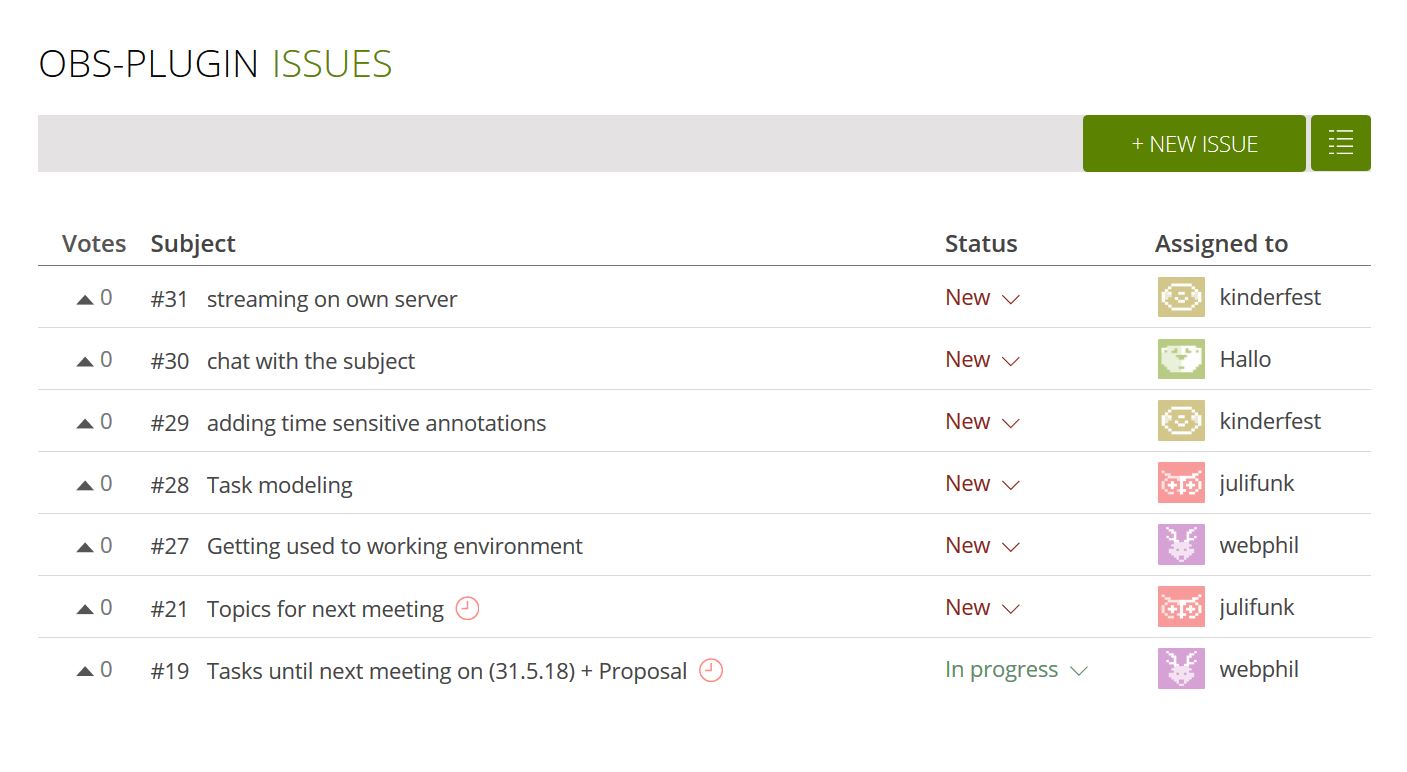


Abbildung 1 - Taiga Übersicht

### Asana und Instagantt

Auf Asana wurde ein detaillierter Projektplan angelegt, welcher zu Beginn der Projektphase sehr strikt eingehalten werden konnte. Gegen Mitte der Projektlaufzeit wurden hier Nachlässigkeiten verzeichnet, welche sich auf private Umstände der Projektmitglieder zurückführen lassen. Dies wurde aber im späteren Verlauf des Projektes zunehmend aufgefangen, um die Implementierung der Anwendung zu gewährleisten. Dies Geschah durch eine erhöhte Intensität sowie Dichte der Meetings innerhalb der finalen Wochen. Der Projektplan dient hier als Orientierungsanker für das Projekt, dieser wurde zeitlich so aufgeteilt, dass genügend Zeit für die einzelnen Tasks eingeplant wurde und diese ohne Probleme zu bewältigen waren. Ebenfalls diente der Plan zur Kontrolle des Fortschritts des Projektes, es wurde stets darauf geachtet, dass nicht zu viele Arbeitspakete nicht rechtzeitig abgeschlossen werden, da es sonst zum „Stau“ kommt und den weiteren Arbeitsablauf beeinträchtigt.

### Projektplan

Der erste grobe Projektplan wurde auf Asana erstellt und immer weiter spezifiziert, hier wurden im Wesentlichen in fünf größere Abschnitte und zahlreiche, kleinere Arbeitspakete unterteilt, die im Folgenden genauer erklärt werden:

****

Abbildung 2 - Anforderungsanalyse

Die Anforderungsanalyse beinhaltet das Erheben von Daten für das Projekt. Hierbei ging es primär darum, mit dem Stakeholder zu reden, um die Anforderungen zu bekommen und die Machbarkeit des Projektes zu verifizieren.

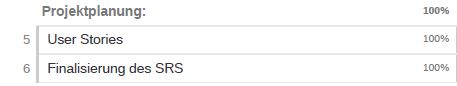


Abbildung 3 – Projektplanung

Hierbei ging es generell noch um die Anforderungen sowie die Abgabe des SRS. Dieser Schritt gehört noch zur Anforderungsanalyse.



Abbildung 4 – Projektdurchführung

Die Projektdurchführung war der längste Teil und zog sich von Mitte Juni bis einschließlich Mitte September durch. Hier wurden einige Änderungen an der Umsetzung vorgenommen, die im späteren Verlauf der Dokumentation erläutert werden.



Abbildung 5 – Design

Die Design Phase lief über den gesamten Ablauf des Projektes und wurde stetig angepasst. Aus den ersten MockUps entstand der erste Prototyp auf HTML-Basis, welcher dann erweitert wurde.



Abbildung 6 – Evaluation

Dies ist die letzte Phase des Projektes. Hier werden die letzten Feedbacks von den Endnutzern erhoben und im Endprodukt berücksichtigt.

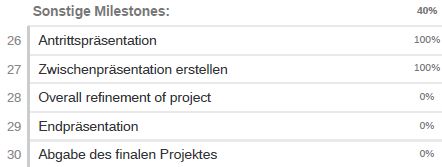


Abbildung 7 - Sonstige Miletones

Dies sind Arbeitspakete, die nicht zu den oben genannten Schritten zugeordnet werden können, weshalb diese extra kategorisiert wurden.

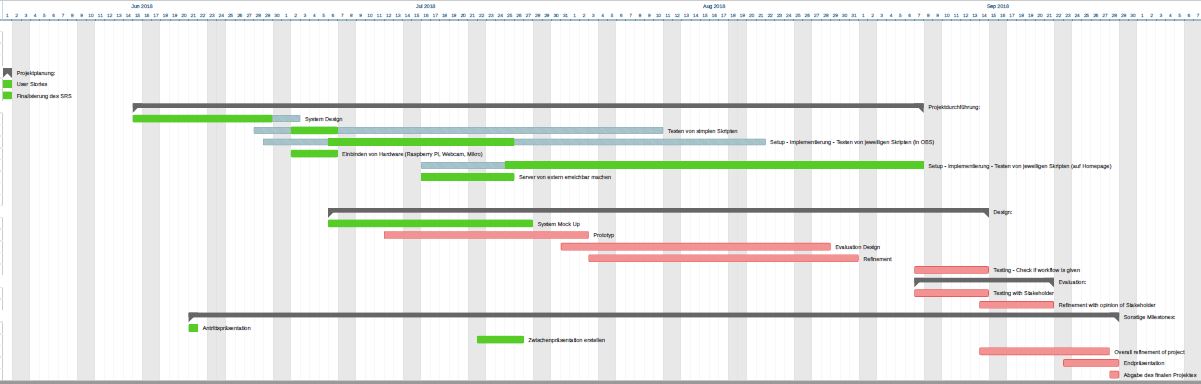


Abbildung 8 - Projektplanübersicht

### Codeverwaltung

Als Code Verwaltungsplattform wurde GitHub benutzt. In Git wurde das Projekt aktuell gehalten, damit jedes Teammitglied jederzeit an der Anwendung arbeiten konnte. Dafür wurden jeweils eigene Branches erstellt und am Ende wurden diese „gemerged“, um größere Probleme aus dem Weg zu gehen. Issues wurden hier intensiv gepflegt und es wurde explizit darauf geachtet, dass diese auch rechtzeitig erfüllt worden sind. Zusätzlich wurde die Anwendung Sourcetree verwendet.

## Meetings

Es wurden zahlreiche Meetings mit dem Stakeholder und Kursleiter ausgemacht, um den Stand der Dinge des Projektes zu klären.

### Meetings mit Kursleiter und Stakeholder

Die Meetings mit dem Stakeholder sowie Kursleiter wurde dazu genutzt, um die Machbarkeit der einzelnen Arbeitspakete sowie den Projektstand zu diskutieren. Es fanden zahlreiche Meetings mit dem Kursleiter sowie mit dem Stakeholder statt, welche in der TechBase stattgefunden haben.

#### Meeting 1 – 21.05.2018

Im ersten Meeting mit dem Kursleiter wurden allgemeine Informationen über das Projekt dargelegt, es wurde spezifisch über die Rahmenbedingungen und Architektur der Software diskutiert. Da wenig bis gar keine Erfahrung mit C/C++ vorherrschte, musste eine Alternative gefunden werden, da die meisten Skripte für OBS auf C/C++ Basis waren. Hier wurde als Alternativlösung einige Möglichkeiten gefunden, die Skripte via Python einzubinden. Zusätzlich wurde geklärt, was sich der Kursleiter vorstellt:

* Python Skripte lassen sich in OBS einbinden
* Grobe Anforderungen des Kursleiters erhalten
* Fragen sammeln für das Stakeholder-Meeting

#### Meeting 2 – 24.05.2018

In diesem Meeting wurde, durch den Stakeholder, die Anforderungen des Projekts dargelegt. Dieses Gespräch lief als eine sehr freie Form des Contextual Inquiries (vgl. Kapitel 5.1.1) ab, der Stakeholder erklärte, inwiefern er die Applikation während der Arbeit benutzen würde. Der Stakeholder nannte die verschiedenen Grundfunktionalitäten, die die finale Version des Projekts enthalten muss. Auf Basis dieser Informationen, entstanden die Anforderungsliste bzw. die User Stories, die als Basis für unsere Anwendung gelten. Die Applikation soll als Open Source Software für alle zur Verfügung stehen, da momentan nur Morae als Alternative für Studentische Zwecke existiert und die Lizenz nicht sehr kostengünstig ist. Daher sollte auch der Aufbau des Codes so gestaltet werden, dass dieser von anderen Personen ohne Probleme erweitert werden können.

* Task modellieren und stellen 🡪 wird dem User im Interface angezeigt
* Bildschirm + Keylogs + Webcam + Mouse Events; Areas/Hotspots loggen
* Als Testleiter Timeline loggen; Metadaten eingeben
* Vgl. Morae 🡪 Kein Manager
* Secondary: Chat
* Machbarkeit des Projektes allgemein testen

#### Meeting 3 – 07.06.2018

Bei diesem Meeting wurden weitere Informationen dargelegt und die Umsetzbarkeit des Themas diskutiert. Es wurde die momentane Situation beschrieben und das geplante weitere Vorgehen dem Stakeholder erläutert. Hierbei wurden folgende Probleme mit dem Stakeholder diskutiert:

* Delay des Stream (sehr wichtig für einen Usability Test)
* Nur Zuschauer sieht kompletten Screen 🡪 Zweiter Monitor wird benötigt
* Twitch als erster Streamingdienst zu Testzwecken, für die tatsächliche Testumgebung wäre ein eigener Server besser (zunächst Raspberry Pi als Alternative)
* Evtl. einen „Wrapper“, der Skripte zusammenführt anstatt einzelner Skripte (Soll einen flüssigen Workflow geben)
* Workflow muss gegeben sein (Sehr wichtig!)

#### Meeting 4 – 05.07.2018

Das zentrale Thema dieses Meeting war die Besprechung des Projektplans, welcher auf Asana bzw. Instagantt erstellt wurde und die Änderungen, die im Laufe des Projektes vorgenommen wurden. Dafür fand ein weiteres Meeting mit dem Kursleiter statt, um die Details des Vorgehens zu besprechen. Der Projektplan war im Allgemeinen in Ordnung, dennoch sollte er etwas genauer untergliedert sein, da die Arbeitspakete noch etwas zu groß waren. Der Plan an sich war schlüssig, es gab keine weiteren negativen Einwände des Kursleiters. Wichtig war es zudem, dass die einzelnen Arbeitspakete in dem Zeitraum abgeschlossen werden, die angegeben wurde. Zusätzlich wurde eine Zeiteinschätzung für die jeweiligen Pakete angegeben, um einen groben Überblick darüber zu geben, wie viel Zeit und Aufwand eingeschätzt werden muss, um ein Paket abzuschließen. Die tatsächlich gebrauchte Zeit variiert jedoch ein wenig, da es sich hierbei nur um Schätzungen handelt. Das zweite Thema der Besprechung war die Auslagerung mehrerer Funktionalitäten auf eine externe Webpage. Nach einigen Recherchen wurde der Entschluss gefasst, dass es nicht möglich sei, alle Funktionalitäten als „Plug-Ins“ in OBS zu implementieren, da der Testleiter lediglich den Stream sieht und daher es nicht möglich wäre, den Stream selbst zu manipulieren. Folglich wurde der Plan, die Funktionalitäten auf eine extern gehostete Webseite auszulagern gefasst. Dies sollte zunächst auf einen Raspberry Pi geschehen.

* Projektplan genauer untergliedern
* Zeitschätzungen anpassen
* Projektplan abgeben
* Aufteilung der Funktionalitäten:
  + Webpage erstellen mit Funktionalitäten (Chat, Tasks, Video)
  + Server soll auf einem Raspberry laufen
  + Auf OBS Ebene: Stream und Keylogger

#### Meeting 5 – 19.07.2018

Dieses Treffen fand nur mit dem Stakeholder statt und hatte als Ziel, den momentan Stand zu präsentieren sowie mögliche Änderungen zu diskutieren. Der Stakeholder wurde über die durchgeführten Änderungen am Konzept informiert. Unter anderem sollten einige Funktionen auslagert werden und verschiedene Ansichten der Webpage erstellt werden. Einerseits wird es eine Testleiter Ansicht und eine Probanden Ansicht geben. Diese unterscheiden sich von ein paar wenigen Funktionalitäten, da der Testleiter mehr Funktionen braucht. Diese beiden Ansichten sollen über einen Chat kommunizieren können. Dabei soll der Testleiter den Stream sehen, sowie Tasks erstellen und Annotationen hinzufügen können, während der Proband nur die Tasks einsehen und mit dem Testleiter über den Chat kommunizieren kann. Des Weiteren haben wurde darüber diskutiert, inwiefern der Raspberry Pi auf längerer Sicht genutzt werden kann, da dieser, Hardware technisch nur bedingt für eine längere Nutzung der Anwendung geeignet ist. Alternativ müsste ein Server gemietet werden, um das Streaming dort zu hosten.

#### Meeting 6 – 24.09.2018

Dieses Treffen war zur Evaluation der Anwendung im Alpha Stadium geplant. Dabei wurde dem Stakeholder die Anwendung präsentiert und alle Funktionalitäten getestet. Dies war das letzte Treffen vor der finalen Abgabe des Projekts, weshalb möglichst viel Feedback gesammelt werden sollte, um die Anwendung noch weiter an die Anforderungen anzupassen. Den detaillierten Verlauf sowie die einzelnen Erkenntnisse können Sie dem Kapitel 6.2 entnehmen.

### Sprints

Die wöchentlichen Sprints dienten der Gewährleistung des Projektfortschritts. Hier fanden persönliche Treffen statt, alternativ kam es zu Meetings via der Kommunikationsapplikation Teamspeak. Es wurden Arbeitspakete durchgeführt oder besprochen. Der allgemeine Aufbau der Sprints war wie folgt:

* Allgemeine Besprechung: Was wurde geschafft? Was wurde nicht geschafft?
* Momentan Stand des Projektes – Wurde der Projektplan eingehalten?
* Was sind die nächsten Schritte?
* Kurzes Sprint Review für alle Mitglieder
* Aufgaben Aufteilung bis zur nächsten Sitzung

Während des Projektes sind einige signifikante Änderungen vorgenommen worden, da manche Umsetzungen nicht so funktioniert haben, wie sie geplant waren. Eine komplette Auflistung der Sprints würde den Rahmen dieser Dokumentation sprengen, weshalb nur die signifikantesten Sprints aufgelistet werden, um die getroffenen Entscheidungen besser nachvollziehen zu können.

#### Besprechung des Themas

Die ersten Besprechungen wurden geführt, um in das Thema einzuführen und das Projektteam über OBS bzw. die Anforderungen zu informieren. Zu Beginn wurden zunächst einige Beispiel-Skripte analysiert. Die meisten Skripte basierten auf Lua bzw. C/C++. Dennoch gab es einige Anzeichen, dass man auch über Python Skripte einfügen kann, was ein Indiz war, dass dieses Thema eventuell doch umsetzbar ist.

#### Machbarkeit des Projektes

Um sicherzustellen, dass das Projekt auch abschließbar ist, musste die Machbarkeit des Projektes überprüft werden. Dafür mussten intensive Recherchen betreiben und einige Setups bzw. Skripte getestet werden. Da die Anforderungen zuvor vom Kursleiter gestellt und nähere Spezifikationen durch den Stakeholder angeführt wurden, konnte grob die Architektur der Anwendung erstellet und somit die gebrauchten Frameworks sowie Anwendungen beschaffet werden, um die ersten Tests durchzuführen. Diese Tests waren ausschlaggebend, ob dieses Thema tatsächlich absolviert oder eine Alternative gefunden werden muss.

#### Nacharbeit der Anforderungsanalyse

Aus den zuvor erhobenen Daten der Meetings konnten die ersten groben Anforderungen generiert und die ersten User Stories sowie Epics (Funktionen) in Taiga erstellet werden. Hier wurden die Funktionen in reale Szenarien verpackt und als User Stories aufgestellt, um einen besseren Bezug zu der tatsächlichen Nutzung zu bekommen. Hierbei haben sich zwei Grundfunktionen herauskristallisiert, die auf ihre Machbarkeit getestet werden müssen. Dies war einerseits das Einbinden von Skripten auf Python Basis und andererseits das Hosten eines Streams. Diese beiden grundlegenden Funktionen müssen umsetzbar sein, damit dieses Projekt abgeschlossen werden kann.

#### Skripteinbindung mit Python

Der erste Test war die Einbindung eines simplen Skripts (Countdown Zähler) in OBS. Das Skript selbst war in Python geschrieben, damit sichergestellt werden kann, dass auch ohne Kenntnisse der Sprache Lua bzw. C Anwendungen geschrieben werden können. Anfangs gab es kleinere Probleme, da das Skript nicht erkannt wurde, nach ein paar Testläufen wurde das Skript jedoch korrekt ausgeführt. Dies war der Startschuss, da nun davon ausgegangen werden konnten, dass das Projekt durchführbar ist. Dennoch musste noch ein zweiter Test gemacht werden, da der Stream noch auf einem Server gehostet werden musste.

#### Stream hosten via Raspberry Pi

Das Hosten des Streams war das nächste große Arbeitspaket, dass durchgeführt wurde. Der erste Gedanke war, den Stream auf einem privaten Raspberry Pi zu konfigurieren und zu hosten. Hierzu wurde der Raspberry Pi in einem privaten Netzwerk angeschlossen und konfiguriert. Nach anfänglichen Schwierigkeiten konnten der Raspberry Pi so konfigurieren werden, dass der Stream gehostet werden konnte. Der Stream konnte als Medienquelle in OBS eingebunden werden und auch von außerhalb des privaten Netzwerkes erreicht werden. Nach einiger Zeit traten jedoch Performance Probleme auf, da Streaming an sich relativ viel Rechenleistung beansprucht und ein Raspberry Pi auf längerer Sicht nicht dafür geeignet ist. Auch die Konfiguration der Ports über den Routern war sehr komplex, weshalb eine andere Lösung angestrebt wurde.

#### Grober Projektplan erstellt

Nachdem die Grundvoraussetzungen des Projektes erfüllt waren, wurde der Projektplan erstellt, welcher auf Asana bzw. Instagantt einzusehen war. Die Details zu diesem Plan sind im Kapitel 4.2.3 näher erläutert.

#### Erster Grundbauplan der Anwendung

Bei diesem Sprint wurde der Grundbauplan der Anwendung ausgearbeitet. Erste Unterteilungen der Anwendung wurde schon bei der Erstellung des Projektplans gemacht und hier kam die Erkenntnis, dass es nicht möglich war, alle Funktionalitäten auf OBS Seite zu implementieren, da der Testleiter keinen Zugriff auf den Stream hat, sondern nur als Zuschauer fungiert. Deshalb wurde sich darauf geeinigt, eine externe Webpage zu gestalten, um dort die Kommunikation zu ermöglichen. Die Kommunikation ist ein Schlüsselelement des Usability Testing, da Fragen gestellt werden können und Tasks übermittelt werden müssen. Folglich waren es keine Plugins an sich selbst mehr, sondern eine Auslagerung der Funktionen auf einer Webpage. Dies ermöglicht es flexibler zu sein und durch eine externe Webpage können einfacher weitere Elemente hinzugefügt werden. Lediglich der Keylogger musste noch separat bearbeitet werden. Dies wurde vom Kursleiter sowie Stakeholder akzeptiert. Die nächsten Schritte wurden geplant und die jeweiligen Arbeitspakete auf die Mitglieder verteilt.

#### Design der Webpage

Das Design der Webpage wurde anfangs als MockUps erstellt (vgl. Kapitel 5.1.3) und später genauer spezifiziert. Der erste Prototyp der Webpage wurde als Grundlage genutzt und im weiteren Verlauf des Projektes stetig angepasst und modifiziert. Die Funktionalitäten wurden hierbei noch nicht hinzugefügt, lediglich das Gerüst der Webpage wurde erstellt, um einen Groben Plan zu bekommen, welche Funktionalitäten wie funktionieren sollen.

#### Streaming hosten via RTMP

Der Raspberry Pi bzw. das Streamen selbst funktioniert nur auf RTMP Basis, da über den JW-Player gestreamt wird. Die weitere Methode wäre das Streaming via HLS (HTTP Live Streaming), welches für zukünftige Erweiterungen oder Verbesserungen in Betracht gezogen werden sollte.

#### Server Hosting anstatt Raspberry Pi

Nach längerer Diskussion fiel der Entschluss, den Raspberry Pi aufgrund von Performance Problemen durch einen Server zu ersetzen. Hierbei fiel die Entscheidung auf die Server Plattform „Digital Ocean“. Nachdem das Streamen erfolgreich verknüpft wurde, wurden die ersten Funktionalitäten implementiert.

#### Taskmodellierung

Die Taskübermittlung war ebenfalls ein zentraler

#### Chat

Bei dem verwendeten Chat handelt es um eine sehr simple Methode, welche jedoch einen entscheidenden Vorteil im Bezug auf das Projekt beinhaltet. So schreiben beide Chatpartner lediglich in ein .txt File. Diese Methode ermöglicht es am Ende des Tests, dieses File zum Download bereitzustellen, um somit dem Testleitet im Nachhinein die Möglichkeit zu bieten dieses einfach und schnell auszuwerten.

#### Server

Als Grundlage für unseren Server wurde ein einfaches Droplet der Plattform „Digital Ocean“ verwendet. Bei dem verwendeten Betriebssystem handelt es sich um die Ubuntu Version 18.04. Das genau Setup des entnehmen Sie bitte dem Dokument „Server Setup“.

#### Datenbankanbindung

Die Anbindung der MySQL Datenbank erfolgt über phpMyAdmin und ist Bestandteil der Severkonfiguration. Im Code selbst fungiert hier PHP als Schnittstelle zwischen der Webpage (Frontend) und der Datenbank (Backend)

#### Finalisierung der Anwendung

Nachdem wir unsere Anwendung dem Stakeholder präsentiert haben, haben wir die abschließenden Modifizierungen hinzugefügt. Es wurden sämtliche Anforderungen des Kursleiters sowie dem Stakeholder implementiert und auf die jeweiligen Wünsche des Stakeholders eingegangen. Da der Stakeholder keine größeren Mängel an unserer Anwendung bemerkt hatte und zufrieden mit dem Ergebnis war, konnten wir die Finalisierung der Anwendung relativ zügig abschließen und für die Abgabe vorbereiten. Hierzu wurden kleinere Bugs gefixt sowie Quellenangaben hinzugefügt und der Code aufgeräumt, damit dieser auch für andere Leute schlüssig ist.

# Architektur und Implementierung

Das Projekt untergliedert sich in wesentliche Teilblöcke, die sukzessiv abgearbeitet worden sind. Zunächst wurde die grundlegende Anforderungsanalyse durchgeführt, um einen Überblick über das Thema zu bekommen sowie weitere Informationen über Funktionalitäten sowie Anforderungen zu erheben. Daraus resultierten die ersten Mock-Ups sowie erste Annäherungen des Designs.

## Anforderungsanalyse

Die Anforderungsanalyse wurde relativ früh mit unserem Stakeholder durchgeführt. Dies geschah in insgesamt drei Meetings, da die jeweiligen Anforderungen einer Machbarkeitsstudie unterlaufen mussten und wir die Anforderungen näher spezifiziert haben. Die Funktionalitäten wurden dann durch User-Stories dargestellt, um die Funktionalitäten in reellen Situationen widerzuspiegeln.

### Contextual Inquiry

Die grundlegenden Anforderungen wurde vor den ersten Meetings schon von dem Kursleiter bereitgestellt, weshalb die ersten groben Spezifikationen der Systemarchitektur vorgenommen und vorgestellt werden konnten. In den ersten Meetings wurde uns das Vorhaben der Anwendung erläutert. Dies geschah durch eine relativ freie Form des Contextual Inquiries (Beyer & Holtzblatt, 1997, S.37f), bei welcher der Stakeholder kurz erklärte, wie die Anwendung im Alltag genutzt werde und uns so einen Überblick gegeben hat, welche Funktionalitäten wichtig sind. Auf Basis dessen entstanden so genannte User Stories, die im folgendem zusammengefasst werden.

### User Stories

Aus den vorher erhobenen Daten konnten User Stories erstellt werden, welche reale Szenarien darstellen. Diese dienen als Grundlage für die Anforderungsliste:

* As User I want to be able to see the subject's screen during the test in realtime
* As User I want to be able to save a recording of the subject's screen during the test
* As user I want to be able to model tasks to show them on the tester’s screen
* As user I want to be able to edit tasks
* As User I want to get keylogs of all activity of the subject during the test
* As User I want to record all mouse events during the test
* As User I want to be able to record the subject’s facial expressions during the test
* As User I want to be able to get an overview of hotspots/main areas of interest of the subject
* As User I want to be able to log a timeline for the test.
* As User I want to be able to add time-sensitive metadata to the recording of the test.
* As user I want to be able to have a live-chat with the subject during the test.

Aus den oben genannten User Stories sind folgende Anforderungen und dessen Prioritäten resultiert:

**Stream the subject’s screen**

Das Streamen des Bildschirmes der Testperson ist die Grundfunktion dieser Anwendung und hat somit die höchste Priorität, da ohne diese Funktion ein Remote-Test nicht stattfinden kann.

**Record mouse and key events**

Diese Funktion ist ein “nice to have“, da es den Workflow eines Tests nicht beeinflusst, sondern vor allem für die Nacharbeit eine wichtige Rolle spielt. Dadurch, dass ein Test auch ohne einen Keylogger funktionieren kann, hat diese Funktion nur eine mittlere Priorität.

**Chat with subject**

Kommunikation zwischen Testleiter sowie Testperson ist ein Schlüsselelement bei einem Usability Test, weshalb diese Anforderung eine sehr hohe Priorität hat.

**Task modelling**

Das Gestalten von Tasks sowie deren Bearbeitung hat ebenfalls eine sehr hohe Priorität, da es den Workflow beeinflusst. Wenn diese Funktion nicht einwandfrei funktioniert, ist es nicht möglich, einen effizienten Test durchzuführen, weshalb es sehr wichtig ist, dass diese Anforderung erfüllt wird.

**Add time-sensitive annotations**

Diese Anforderung hat eine mittlere Priorität, da es den Prozess des Tests nicht beeinflusst. Der Testleiter könnte theoretisch auch externe Notizen machen und diese für die Nacharbeit mit einbeziehen.

**Screen recording**

Diese Anforderung musste von uns nicht übernommen werden, da OBS eine Funktion hat, um den Stream zu recorden. Der Testleiter kann den Screen selbst recorden.

**Record the subject’s face**

Auch diese Anforderung wurde von OBS übernommen, die es ermöglicht eine Webcam als Medienquelle hinzuzufügen.

Diese sind die Hauptfunktionalitäten der Anwendung und dienen als Grundlage für die Anwendung.

### Mock-Ups

Aus den ersten Gesprächen mit dem Stakeholder haben wir Informationen über die Funktionalitäten der Anwendung sammeln können und die ersten Mock-Ups generiert. Diese dienten als Grundlage für die weitere Designspezifizierung und wurde nach und nach designtechnisch angepasst. Auch die verschiedenen Core-Funktionalitäten wurden abgebildet.

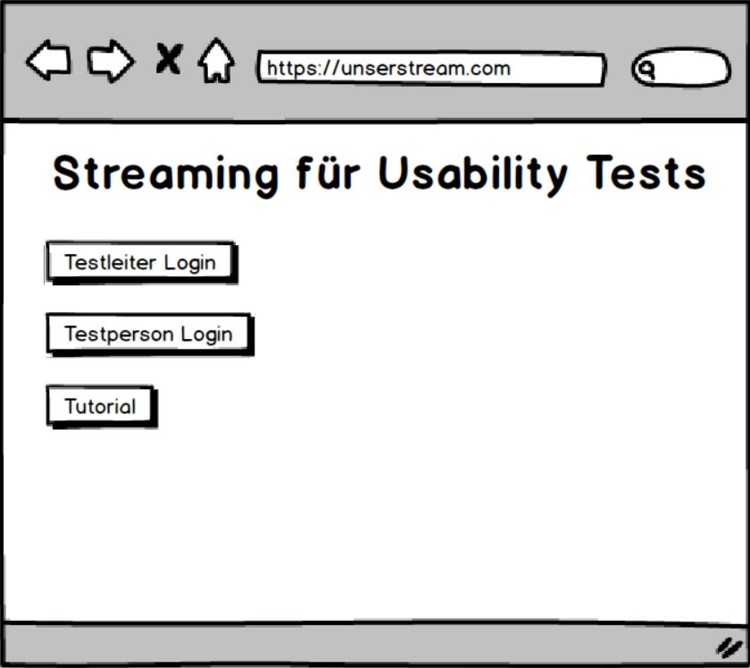


Abbildung 9: Hauptseite

In der obigen Abbildung sieht man den Hauptscreen der Webpage. Die Webpage untergliedert sich in 2 wesentlichen Screens, die im folgendem noch näher erläutert werden. Auf dem Hauptscreen kann man zwischen der Testleiterseite sowie Testpersonenseite auswählen. Zusätzlich soll noch ein kurzes Tutorial dem Nutzer die Applikation erklären.

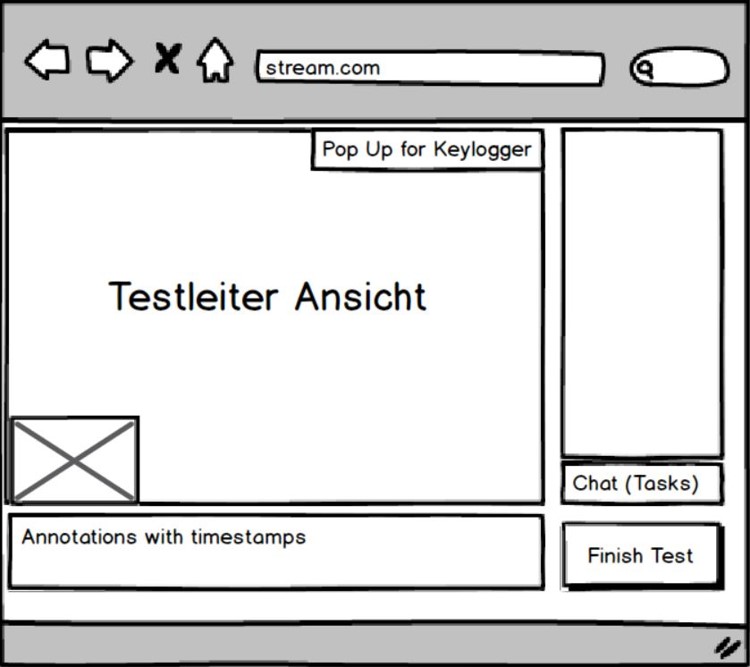


Abbildung 10 – Testleiter Ansicht im Mockup

Hier sieht man die Testleiter Ansicht der Webseite. Drei grundlegende Funktionalitäten sind hierbei auf der Seite abgebildet:

* Der Bildschirm des Probanden soll angezeigt werden, dabei nimmt dieser Teil den größten Abschnitt der Seite ein
* Es soll möglich sein, zeitsensitive Daten zu erfassen, dies kann der Leiter durch den „Annotationsabschnitt“ tätigen
* Der Testleiter braucht eine Möglichkeit, mit dem Probanden zu kommunizieren, hierbei ist der Chat zur Verfügung gestellt worden
* Der Tester soll einen Test abschließen können



Abbildung 11 – Testpersonen Ansicht im Mockup

Die Testpersonen Seite ist ähnlich der Seite des Testleiters aufgebaut. Einerseits soll die Testperson den Bildschirm streamen, auf welcher er die zu testende Software am laufen hat und zusätzlich einen Bildschirm haben, um Fragen bzw. die Tasks anzeigen zu lassen und ggf. mit dem Testleiter kommunizieren. Diese Ansicht hat deutlich weniger Funktionalitäten und hat hauptsächlich die Kommunikation im Vordergrund.

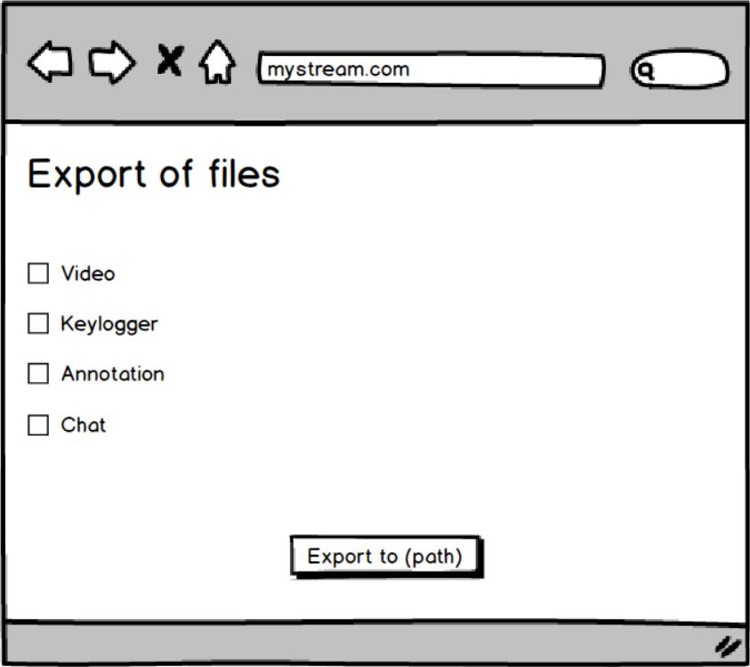


Abbildung 12 – Export Seite im Mockup

Diese Ansicht sollte dazu dienen, die jeweiligen Logs herunterzuladen, wurde jedoch im Laufe des Projektes verworfen und mit einem Button ersetzt.

## Implementierung

Zunächst war der Plan alle Funktionalitäten innerhalb OBS zu implementieren, doch nach ausgiebiger Recherche haben wir uns dazu entschlossen, die Funktionalitäten auszulagern und auf einer externen Homepage zu gestalten, da eine direkte Implementierung in OBS selber nicht möglich ist für unsere Zwecke. Problematisch hierbei war es, dass der Proband selber seinen Bildschirm streamt und es daher nicht möglich ist, von extern (also als Testleiter) darauf zu zugreifen. Daher wurden einige Funktionalitäten auf der externen Homepage transferiert. Lediglich der Key Logger muss beim Probanden eingebunden werden, welcher dann nach Abschluss des Tests manuell an den Testleiter gesendet werden muss. \*

### Funktionalitäten

Auf Basis der Anforderungsanalyse sowie zahlreichen Meetings wurden grundlegende Funktionalitäten herausgefiltert, die in unserer Anwendung implementiert wurden. Folgend wird kurz ein Überblick über alle Funktionalitäten auf der Webpage gegeben, die auf der Basis der User Stories beruhen:

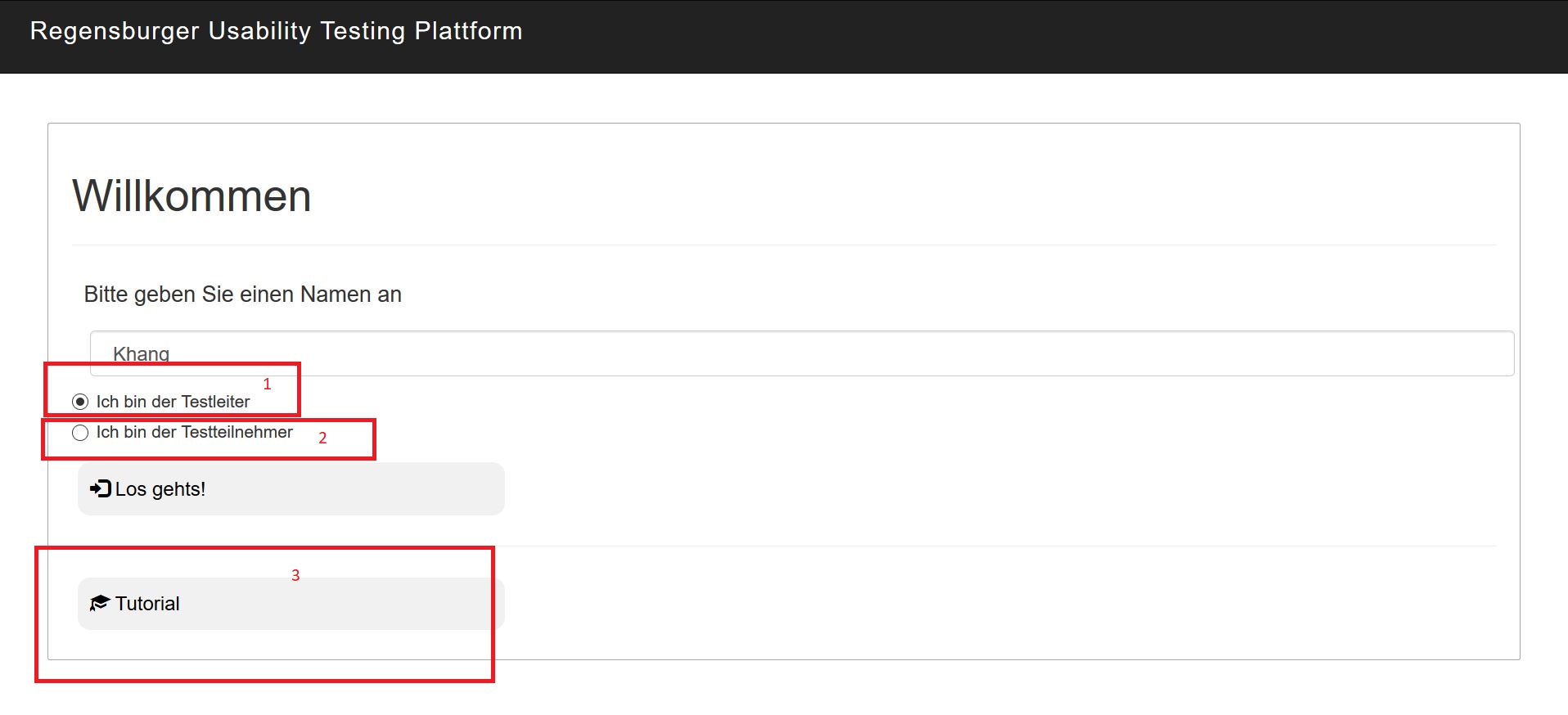


Abbildung 13 – Hauptseite der Anwendung

Zunächst muss ein Name angegeben werden, damit sich die jeweiligen Personen identifizieren können. Auf der Hauptseite der Webpage gibt es dann folgende Möglichkeiten:

1. Zur Testleiter Ansicht zu kommen
2. Zur Testperson Ansicht zu gelangen
3. Auf die Tutorial Seite zu gelangen

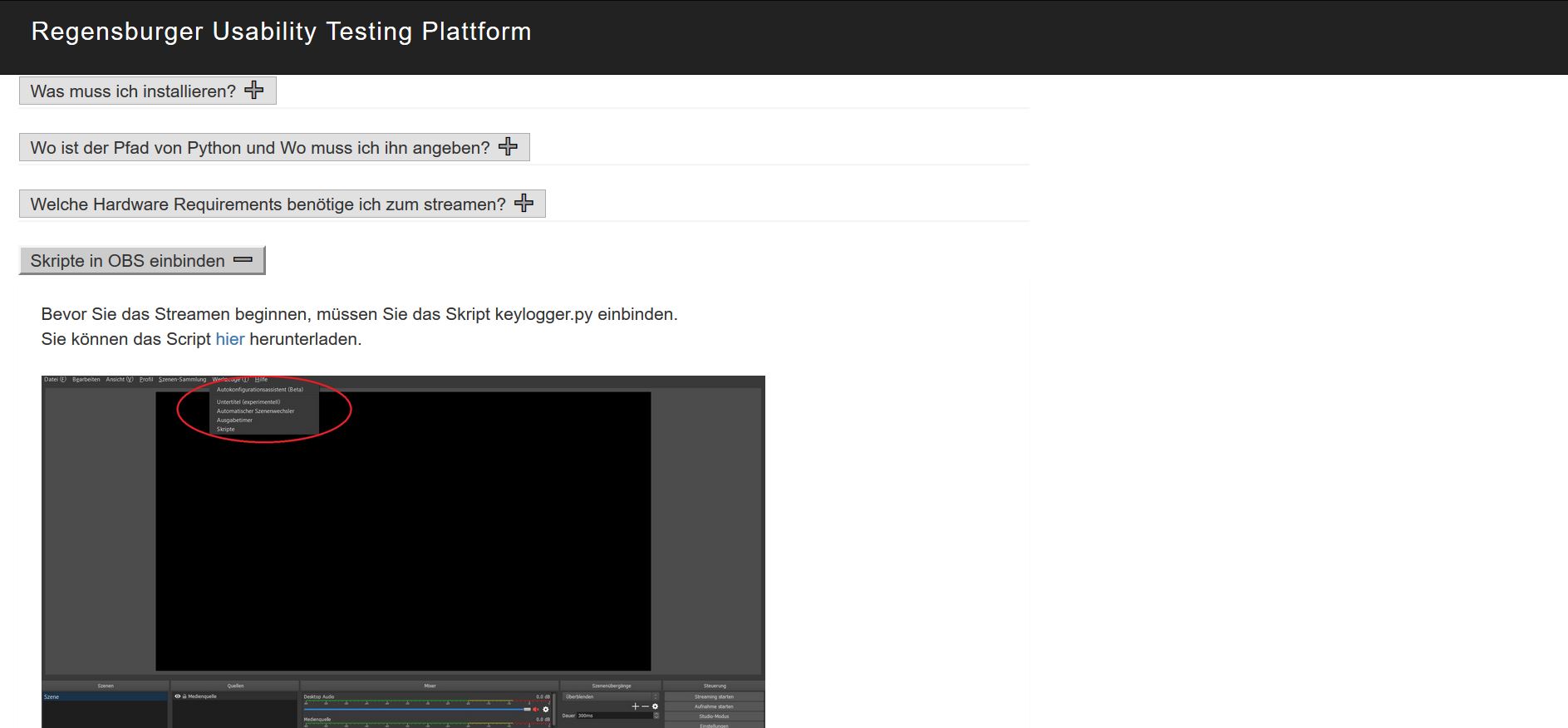


Abbildung 14 – Tutorial Seite der Anwendung

Auf der Tutorial Seite sind grundlegende Fragen beantwortet, wie die Anwendung funktioniert. Ebenfalls kann man sich hier die Keylogger Datei herunterladen und es gibt verschiedene kurze Anleitungen wie man OBS konfiguriert, um den Stream ordnungsgemäß zu starten. Ebenfalls wird kurz erläutert, welche Elemente die Webpage besitzt.

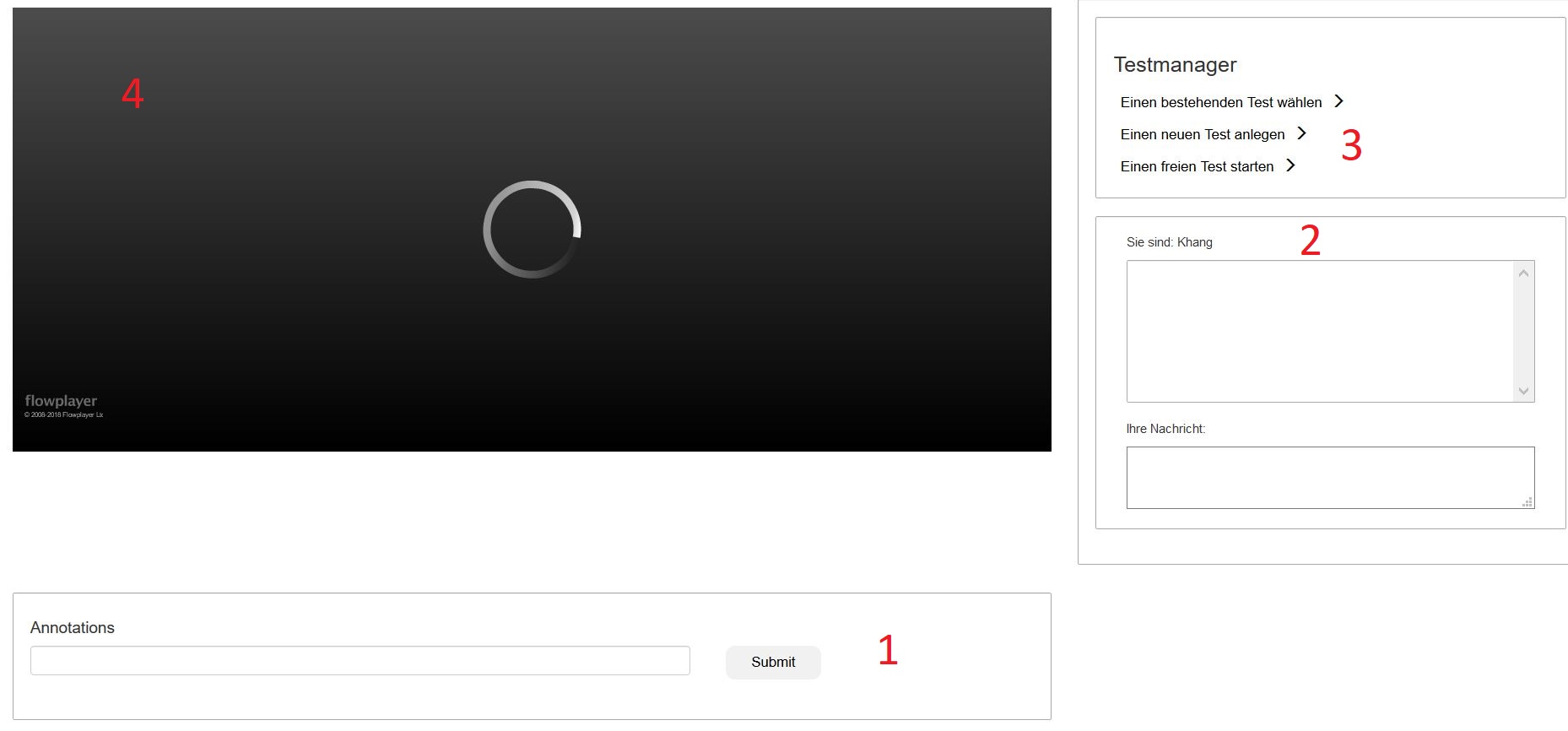


Abbildung 15 – Testleiter Ansicht der Anwendung

Abbildung 8 zeigt die Ansicht des Testleiters. Dieser benötigt lediglich einen Bildschirm, jedoch würde ein zweiter Bildschirm ebenfalls sinnvoll sein, um etwaige Informationen zu notieren bzw. den Stream zu maximieren und den Chat auf der rechten Seite zu öffnen. Folgende Aktionen können hier durchgeführt werden:

1. Zeitsensitive Annotationen können notiert werden, welche in einer Text Datei lokal abgelegt werden. Diese können zur Nacharbeit benutzt werden
2. Der Chat dient zur Kommunikation zwischen Testleiter und Testperson, da bei Remote-Testing sich Teilnehmer nicht gegenüber sitzen 🡪 Voice Talk evtl. auch sinnvoll
3. Der Taskmanager ist nur in der Testleiter Ansicht verfügbar und wird später näher erläutert
4. Der Stream ist ein Hauptelement dieser Ansicht und zeigt den Screen der Testperson an. Der Stream ist im Regelfall um einige Sekunden verzögert, sollte aber für ein Testing irrelevant sein

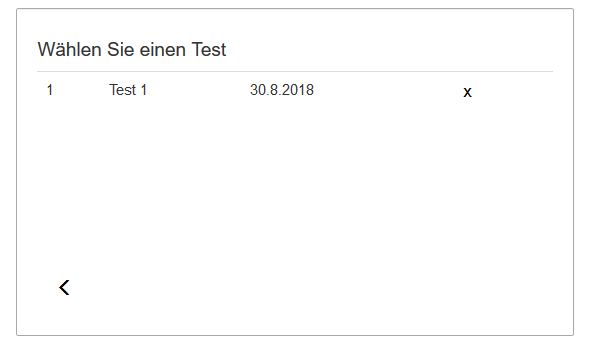


Abbildung 16 Testmanager – Einen bestehenden Test wählen

Hier kann ein bestehender Test gewählt werden mit den jeweiligen Tasks. Dies spart Zeit, wenn man mehrere Probanden mit demselben Testdurchlauf absolvieren will.

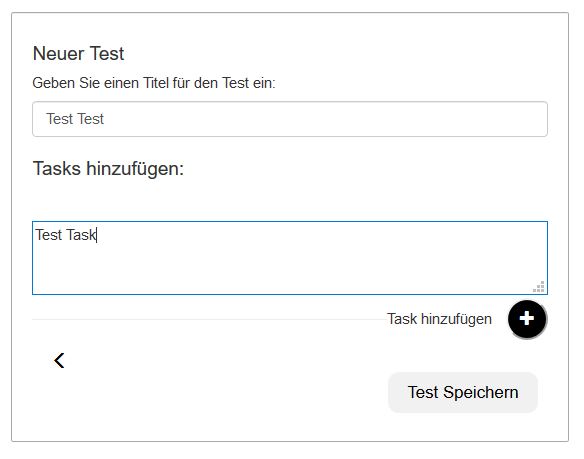


Abbildung 17 Testmanager – Einen neuen Test anlegen

Hier werden neue Tests angelegt, die dann in der Datenbank abgelegt werden. Es können beliebig viele Tasks hinzugefügt werden durch den „Task hinzufügen – Button“. Durch „Test Speichern“ wird der Test gespeichert.

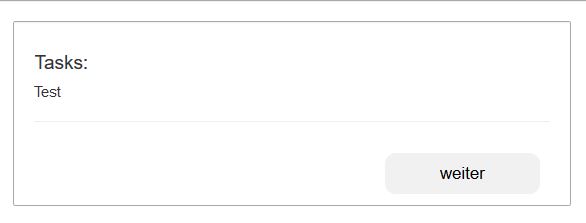


Abbildung 18 Testmanager – Tasks

Nachdem ein existierender Test gewählt bzw. ein neuer Test angelegt wurde, können diese gestartet werden. Die eingegebenen Tasks werden nacheinander der Testperson übermittelt, indem der „weiter-Button“ geklickt wird. Man kann erst wieder auf den Button klicken, sofern die Testperson den Task als „completed“ markiert hat.

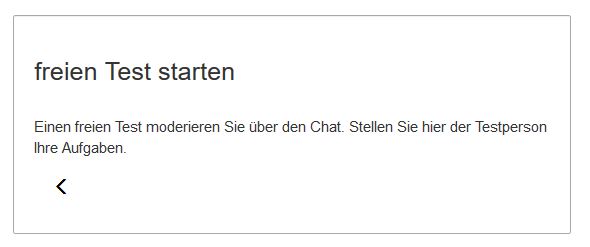


Abbildung 19 Testmanager – Einen freien Test starten

Ein Freies Test wird über den Chat durchgeführt, dies findet vor allem bei unstrukturierten Interviews Verwendung und wird meist von der Testperson geleitet.

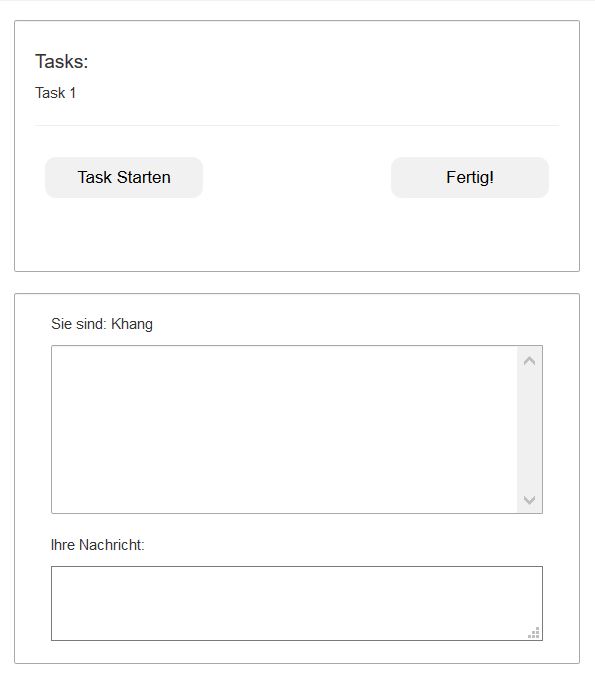


Abbildung 20 – Taskansicht auf Testperson Seite

Die Testpersonen Ansicht ist relativ simpel gestaltet. Der untere Bereich gestaltet sich aus dem Chat, der obere Bereich beinhaltet die Taskübermittlung. Sobald ein Task gestartet wird, läuft im Hintergrund ein Counter. Sobald der Task abgeschlossen ist, soll der „Fertig“ Button betätigt werden, sodass die Task Completion Time notiert werden kann. Diese Datei wird lokal abgelegt und muss noch hochgeladen werden. Dabei kann der Testleiter erst einen neuen Task übermitteln, wenn der vorherige abgeschlossen ist.

### Systemarchitektur

Beschreiben Sie die Systemarchitektur der Anwendung. Wie sind die verschiedenen Komponenten aufgebaut? Welche Frameworks oder Bibliotheken haben Sie eingesetzt und wie wurden diese in das System integriert. Sie sollten an dieser Stelle einen Überblick über die technische Umsetzung des Projekts und den implementierten Quellcode liefern. Fassen Sie größere logische Module Ihrer Software zusammen und beschreiben Sie die interne Kommunikation innerhalb Ihrer Anwendung. Der Abschnitt sollte dabei als Einführung und Ergänzung zur Ihrer Code-Base dienen. Unabhängig davon muss Ihr Code selbst auch sinnvoll dokumentiert und kommentiert sein. Verweisen Sie an dieser Stelle auf mögliche externe Dokumentationen, die Sie aus den Quellcode-Kommentaren generiert haben.

Das Ziel dieses Kapitel ist es, Dritte (z.B. Auftraggeber(!)) in die technische Implementierung Ihrer Anwendung einzuführen. Zusammen mit der Code-Dokumentation und der Setup-Anweisung sollte ein fähiger Programmierer in der Lage sein, Ihren Code nachzuvollziehen und diesen zu warten bzw. zu erweitern.

## Spezifikation

Beschreibung der auf der Basis der Anforderungen entwickelten Spezifikation für Ihr Vorhaben. Die Spezifikation hängt natürlich stark von der Art des Projektes ab (welche Art Software wird entwickelt, welche Plattformen werden genutzt, welche Rolle spielt das Interaktions- oder UX-Design etc.).

# Evaluation

Die Evaluation ist ein iterativer Prozess, welcher von Beginn an durchgeführt wurde, um den Anforderungen gerecht zu werden. Die ersten spezifischen Evaluationen wurden mit Mitkommilitonen durchgeführt, die im Laufe ihres Studiums schon einige Usability Test absolviert haben, daher ist das Feedback dieser Probanden signifikant und relevant, da diese schon Erfahrungen mit Usability Tests haben. Eine ausführliche Evaluation wurde dann mit dem Stakeholder durchgeführt, da dieser der Endnutzer ist und somit sein Feedback von hoher Bedeutung ist. Die erhobenen Daten wurden im Endprodukt berücksichtigt.

## Testing mit Mitkommilitonen

Im Rahmen des Projektes wurden sieben Medieninformatik Studenten getestet. Aus einer Studie von Virzi (1992) kann man entnehmen, dass auch schon bei einer geringen Anzahl an Probanden qualitative Daten entnommen werden können da die signifikantesten Probleme schon bei den ersten 4-5 Probanden erkannt werden. Hauptsächlich ging es hierbei um die Bedienbarkeit der Seite sowie Designtechnisches Feedback der Probanden.

Aus der Anforderungsanalyse sowie die daraus entstehenden User Stories haben wir Use Cases generiert, welche die Hauptfunktionalitäten abprüfen. Hierbei wurden folgende Funktionalitäten getestet:

* Auf die Testleiter bzw. Testteilnehmer Seite gelangen
* Einen Stream starten (mit einer kurzen Anleitung)
* Chatten
* Skript hinzufügen zu OBS
* Annotationen hinzufügen

Es wurden zwei Laptops für die Testpersonen Seite sowie einen Laptop für die Testleiterseite zur Verfügung gestellt. Es wurde eine Anwendung in einem prototypischen Zustand vorgestellt, welcher Kernfunktionalitäten beinhaltete. Das Feedback von der Zwischenpräsentation wurde berücksichtigt.

**Aufgabe 1:**

Die jeweiligen Probanden sollten von der Hauptseite in die jeweiligen Ansichten (Testleiter und Testteilnehmer) gehen. Hierbei konnten alle Probanden die Aufgabe ohne weitere Probleme absolvieren.

**Aufgabe 2:**

Die zweite Aufgabe war es, einen Stream zu starten. Dabei wurden den Probanden alle nötigen Informationen sowie ein kurzes Tutorial zur Verfügung gestellt. Hierbei konnten fünf Probanden ohne Probleme den Stream starten, zwei Probanden hatten anfangs Probleme, da Sie nicht wussten, was Sie in den jeweiligen Felder angeben müssen. Diese konnten aber auf Hinweis der Testleiter auf das Tutorial, die Aufgabe ohne weitere Probleme beenden. Hierbei konnte man erkennen, dass einige Probanden schon Erfahrungen mit dem Streaming haben und keinerlei Hilfe von dem Tutorial benötigten.

**Aufgabe 3:**

Eine weitere Kernfunktion war das Chatten zwischen Testleiter sowie Testperson. Dies wurde von unseren Probanden schnell gefunden und ohne Probleme erkannt. Die Chatfunktion wurde überwiegend positiv empfunden, jedoch wurden von einigen Probanden das persönliche Gespräch (oder auch Voicechat) bevorzugt, da man sich dann besser auf die Tasks konzentrieren kann.

**Aufgabe 4:**

Bei diesem Task wurde primär darauf geachtet, ob der Proband ein externes Skript in OBS einbinden kann. Hierbei wurde dem Probanden zunächst keine Hilfestellung gegeben, nur bei Bedarf wurde dem Probanden eine kurze Anleitung zur Verfügung gestellt. Vier Probanden konnte nach einer kurzen Zeit das Skript (dies war auf dem Desktop vorgegeben) erfolgreich einbinden. Die restlichen Probanden waren sich unsicher und haben nach der Hilfestellung die Aufgabe erfolgreich absolvieren können. Hierbei ist besonders zu vermerken, dass zwei der vier Probanden, die es geschafft haben da Skript einzubinden, nie zuvor OBS benutzt haben.

**Aufgabe 5:**

Diese Aufgabe wurde von allen Probanden ohne weitere Probleme abgeschlossen.

Zusammenfassend kann man sagen, dass alle Probanden die Tasks abschließen konnten. Die Tutorials sind sehr hilfreich und wurden von allen Leuten positiv empfunden. Da die Anwendung von verschiedenen Nutzergruppen benutzt werden soll, ist die Anwendung selber sowie das Tutorial relativ simpel gestaltet, damit jeder – auch medial weniger bewandte Personen - diese Anwendung nutzen kann.

## Evaluation mit Stakeholder

Diese Evaluation wurde gegen Ende durchgeführt, um auf letzte Wünsche des Stakeholders einzugehen. Die Anwendung war in einem Alpha Stadium, welches alle gewünschten Funktionalitäten beinhaltete. Dies war eine sehr freie Evaluation, der Stakeholder durfte sich eigenhändig die Anwendung anschauen und sollte bei Bedarf an den User Stories sich orientieren bzw. die Leiter fragen. Der Stakeholder explorierte alle Funktionalitäten eigenhändig, konnte die Kernfunktionen eigenständig ausführen und ohne Probleme erkennen. Zunächst hat sich der Proband die Testleiter Seite angeschaut und hat einige Tasks erstellt, welches kein Problem darstellte. Daraufhin hat er den Chat sowie die Annotationen getestet, die in einem separaten Log-File gespeichert werden. Auch dies verlief reibungslos ab. Ebenfalls wurde die Ansicht der Testperson exploriert, spezifisch wurde das Setup des Streams selber auf OBS Ebene erkundigt. Hier wurde dem Probanden kurz erläutert, wie er vorgehen müsste, dieser verstand den Prozess ohne Probleme. Es wurde darauf geachtet, dass alle Funktionen auf Basis der User Stories (vgl. Kapitel x) vom Probanden getestet werden, dabei konnten alle Funktionalitäten gefunden sowie getestet werden. Im groben und ganzen war der Stakeholder positiv überrascht und konnte keine signifikanten Mängel feststellen. Dennoch wurden einige Punkt erwähnt, die für das Endprodukt relevant sind und noch geändert werden mussten:

* Login Testperson bzw. Testleiter sollte umbenannt werden 🡪 irritiert, da man sich per se nicht „einloggt“ 🡪 Test durchführen bzw. an Test teilnehmen
* Nachdem die Tasks beendet werden, sollte ein Download Button erscheinen für die Log Files
* Das Tutorial musste erweitert werden
* Nachdem ein Test gespeichert wurde, sollte es die Möglichkeit geben, diese zu Clearen oder zu einem existierenden Test zu wechseln
* Chat noch fehlerhaft 🡪 wurde mehrmals geschrieben
* Completion time der Task als CSV ausgeben zur einfacheren Auswertung
* Der gleiche Test mit mehreren Probanden sollte entweder gleich nummeriert werden oder in einem Ordner gepackt werden

Abschließend lässt sich sagen, dass der Stakeholder sehr zufrieden mit unserer Anwendung ist, da alle Anforderungen sowie Funktionalitäten erfüllt sind. Die oben genannten Punkte wurden beim Endprodukt berücksichtigt. Das Aufsetzen des Servers zur Nutzung dieser Software wird im nächsten Kapitel genauer erläutert.

# Setup-Anweisungen

Je nach Art Ihres Projekts beschreiben Sie hier welche Schritte notwendig sind, um Ihr Projekt auszuführen bzw. verweisen auf eine entsprechende Web-Adresse, Datei o.ä. Ihre Anwendung sollte ggf. als ausführbare Datei oder über eine entsprechende Setup-Anwendung zugänglich sein.

Die hier adressierten Dateien sollen es dem Gutachter bzw. dem Auftraggeber ermöglichen Ihre Anwendung problemlos auszuführen und zu testen.

# Ausblick

Die Anwendung ist einem nutzbaren Zustand, kann dennoch weiter verfeinert werden und neue Funktionalitäten umgesetzt werden.

**Quellen:**

Beyer, H., & Holtzblatt, K. (1997). *Contextual design: defining customer-centered systems*. Elsevier.

Virzi, R. A. (1992). *Refining the test phase of usability evaluation: How many subjects is enough? Human factors*, *34*(4), 457-468.

Projektmanagement: Definitionen, Einführungen und Vorlagen. Retrieved from

<http://projektmanagement-definitionen.de/glossar/scrum/> [03.10.2018]