

---

# SOFTWARE ENGINEERING 2

---

Endbericht



---

<https://informatik.hs-bremerhaven.de/docker-swe2-2023-team14-web/PrototypSWE2/TSKLogistik.html>

---



16. AUGUST 2023

HODEIFA DABOUR – 40016

JUNIOR EKANE – 40128

FLORIAN QUAAS – 39952

REZA AKBARI – 39620

STEVE AGUIWO II – 40088

HOSSEIN AKBARI – 39940

MAEVA LEKEUFACK – 39776

## Inhaltsverzeichnis:

### 0. Eigenständigkeitserklärung

#### 1. Überblick und Vorgehensbeschreibung des Gesamtprojektes

#### 2. Effektive Zusammenarbeit und Projektplanung mit OpenProject

#### 3. Welche Analyse- und Erhebungsmethoden habt ihr zur Erhebung der Anforderungen wie eingesetzt?

#### 4. Wie habt ihr die Anforderungen geordnet und strukturiert („FURPS“)? Usw.

#### 5. Erläuterung wichtiger Codeschnipsel

#### 6. Ergebnisse der Ist-Analyse

##### 6.1. Problembeschreibung

##### 6.2. Vorhandene Schwachstellen

#### 7. Zusammenfassung der Anforderungen bzw. Darstellung der eigenen Idee und des Anwendungskontextes (mit Verweis auf Software-Anforderungsspezifikation)

#### 8. Zusammenfassung der Anforderungen bzw. Darstellung der eigenen Idee, des Anwendungskontextes und der Zielgruppe (Personas).

#### 9. Erläuterung, wie die erhobenen Anforderungen modelliert und dokumentiert wurden

##### 9.1. Welche Methoden und UML-Diagramme wurden jeweils für welchen Zweck verwendet?

##### 9.2. Wie ergänzen sich die UML-Diagramme?

##### 9.3. Welchen Zweck hat die Software Anforderungs- spezifikation?

#### 10. Erläuterung, inwiefern der Prototyp die entsprechenden UML-Modelle realisiert, z.B.: Wo finden sich Klassen/Objekte im Prototypen? Welche Anwendungsfälle bzw. Aktivitätsdiagramme werden durch welche Funktionen und Interaktionen realisiert?

#### 11. Qualitätssicherung: Welche Methoden (z.B. Reviews, Testfälle, Walkthrough-Tests) wurden mit welchen Ergebnissen verwendet? Welche Verbesserungen wurden erreicht?

#### 12. Reflexion und Fazit

#### 13 Integration der Diskussionsergebnisse zu „Wissenschaftlichen Artikeln“ mit Bezug zum SWE2-Projekt

#### 14. Referenzen, Literatur- und Quellenverzeichnis

#### 15. Anhang

##### 15.1 Teamregeln

##### 15.2 Ideenfindung

##### 15.3 Interviewleitfaden

- **0.Eigenständigkeitserklärung**

Eigenständigkeitserklärung

Eigenständigkeitserklärung Hiermit bestätigen wir, dass wir die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Hilfsmittel benutzt habe. Die Stellen der Arbeit, die dem Wortlaut oder dem Sinn nach anderen Werken (dazu zählen auch Internetquellen) entnommen sind, wurden unter Angabe der Quelle kenntlich gemacht.

17.08.2023



Hossein  
Akbari



Florian Quast



- **1.Überblick und Vorgehensbeschreibung des Gesamtprojektes**  
**(Hodeifa Dabour)**

Im Rahmen des Moduls "Software Engineering II" haben wir die spannende Gelegenheit, eine realitätsnahe Projektplanung und Durchführung eines Software-Unternehmens zu simulieren. Diese Veranstaltung richtet sich sowohl an Studierende der Informatik als auch der Wirtschaftsinformatik und fördert das praktische Anwenden unserer erlernten Fähigkeiten. In Gruppen von drei bis fünf Studierenden arbeiten wir gemeinsam an einem Projekt mit dem Ziel, für einen echten Kunden ein Sollkonzept, ein Prototyp und eine webbasierte Softwareanwendung zu entwickeln.

Unser Glück war es, dass ein Mitglied unserer Gruppe eine Verbindung zu einem renommierten Umzugs- und Immobilienunternehmen in Bremerhaven hatte, welches sich bereit erklärte, als unser Kunde für das Projekt zu fungieren. Die Zusammenarbeit mit dem Unternehmen verlief größtenteils reibungslos, obwohl es einige Herausforderungen in der Kommunikation mit dem CEO gab, der als eher verplanter Typ galt. Nichtsdestotrotz legte diese Zusammenarbeit eine solide Grundlage für eine erfolgreiche Kooperation, während wir uns darauf konzentrierten, eine webbasierte Softwareanwendung zu entwickeln. Diese sollte sowohl eine informative Website als auch ein effizientes Terminplaner-System beinhalten. Als Gruppe entschieden wir uns für die Entwicklung eines Terminplaner-Systems, um einen besseren Überblick über unsere Termine im Unternehmen zu haben und effizienter zu arbeiten. Ziel ist eine optimierte Terminplanung für verbesserten Kundenservice und langfristigen Erfolg.

Um das Design für unseren Prototypen zu entwickeln, führten wir ein ausführliches Interview mit dem Kundenunternehmen durch. Dieses Interview lieferte uns wertvolle Einblicke und Informationen, die uns bei der Gestaltung des Prototyps entscheidend halfen. Es ermöglichte uns, die Anforderungen und Wünsche des Kunden besser zu verstehen und daraus abzuleiten, wie die Softwareanwendung idealerweise aussehen und funktionieren sollte.

Gestützt auf die gewonnenen Erkenntnisse aus dem Kundeninterview begannen wir mit der Erstellung des Designs für unseren Prototypen. Dabei legten wir besonderen Wert auf ein ansprechendes und benutzerfreundliches Interface, das den Bedürfnissen des Kunden und potenzieller Nutzer gerecht wurde. Durch regelmäßige Absprachen und Feedbackschleifen mit dem Kunden gewährleisteten wir, dass wir seine Erwartungen erfüllten und ein Prototyp entwickelten, der seinen Anforderungen in bestmöglicher Weise entsprach.

Die Entwicklung dieser webbasierten Softwareanwendung für das Umzugs- und Immobilienunternehmen in Bremerhaven war ein herausforderndes, aber äußerst lehrreiches Projekt. Es ermöglichte uns, unsere theoretischen Kenntnisse in der Praxis anzuwenden und unsere Teamarbeit und Projektmanagement-Fähigkeiten zu stärken. Das Ergebnis unserer Arbeit erfüllte die gestellten Anforderungen und präsentierte sich mit einem gelungenen Design, auf das wir stolz sind.

- **2.Effektive Zusammenarbeit und Projektplanung mit OpenProject:**

**(Hodeifa Dabour)**

Im Verlauf des zweiten Semesters setzten wir unsere Teamarbeit fort, wobei drei Teammitglieder das Projekt verließen und vier neue Mitglieder hinzukamen. Trotz dieser Veränderungen verlief die Teamarbeit reibungslos und wir trafen uns wöchentlich zweimal, um unsere Fortschritte zu besprechen. Dabei gab es keine Probleme oder Unstimmigkeiten.

Unsere persönlichen Treffen an der Hochschule Bremerhaven und an der Uni Bremen erwiesen sich als äußerst effektiv und wertvoll im Vergleich zu Online-Meetings.

Meistens haben wir uns in der Hochschule im Gebäude S 2 in der oberen Etage in den Lernräumen getroffen. In Bremen haben wir uns ebenfalls in Lernräumen getroffen, in denen wir große Bildschirme nutzen konnten. Wir haben uns bewusst auch in Bremen getroffen, um sicherzustellen, dass der Weg für jeden im Team gleich ist. Manchmal fielen die Züge häufig aus, weshalb wir uns entschieden haben, auch mal ab und an nach Bremen zu fahren und dort unsere Treffen durchzuführen.

Wir haben uns anfangs wöchentlich einmal getroffen, später wurden es sogar zwei Treffen pro Woche. Bei unseren Zusammenkünften haben wir die anstehenden Aufgaben ausführlich besprochen und die Aufgaben untereinander verteilt. Immer beim nächsten Treffen haben wir dann den Fortschritt und die Ergebnisse unserer Arbeit verglichen.

Die regelmäßigen Treffen dienten nicht nur dazu, uns über unsere individuellen Aufgaben im Klaren zu sein, sondern ermöglichten uns auch, als Team gut koordiniert zu arbeiten. Durch den stetigen Austausch konnten wir eventuelle Unklarheiten frühzeitig aus dem Weg räumen und sicherstellen, dass wir alle in die gleiche Richtung arbeiten.

Es war beeindruckend zu sehen, wie wir durch diese organisierten Treffen immer effizienter wurden und unser Teamgeist wuchs. Jeder konnte seine Stärken einbringen, und gemeinsam haben wir Herausforderungen bewältigt und erfolgreich Projekte abgeschlossen.

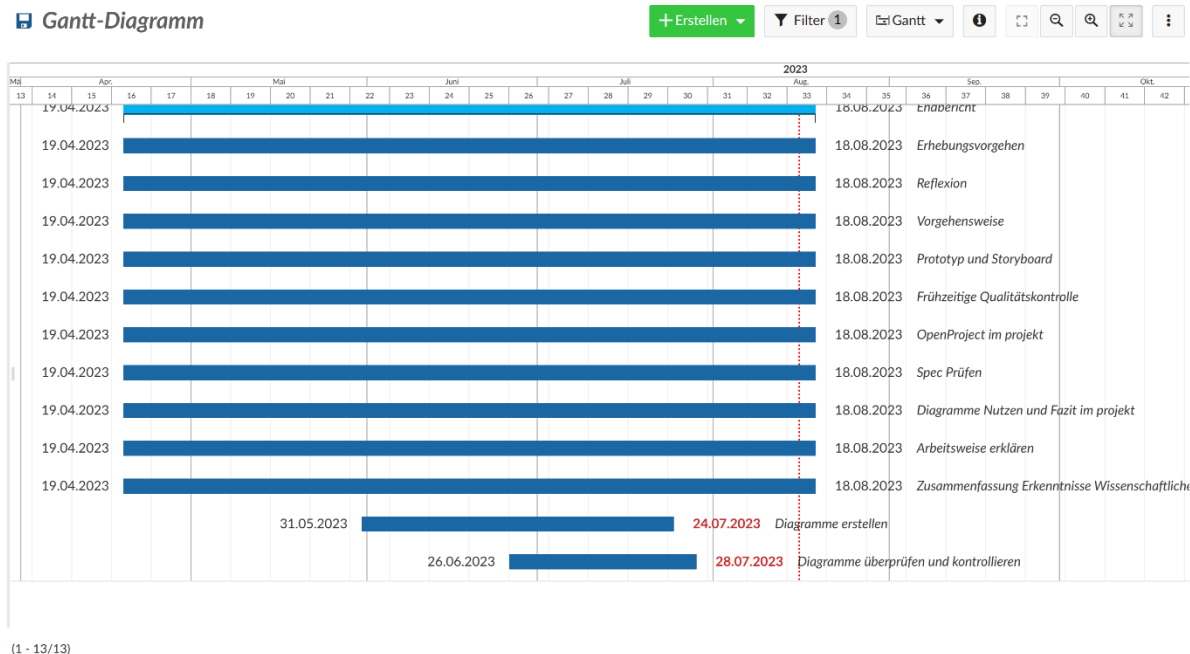
Die wöchentlichen Meetings wurden schnell zu einem festen Bestandteil unseres Arbeitsablaufs, der maßgeblich zu unserem Erfolg beitrug. Wir lernten voneinander, unterstützten uns gegenseitig und schafften es, unsere Projekte termingerecht und in hoher Qualität abzuschließen.

Ich bin dankbar für die Möglichkeit, in einem so gut organisierten und kooperativen Team zu arbeiten, und bin stolz auf das, was wir gemeinsam erreicht haben. Unsere regelmäßigen Treffen waren zweifellos ein Schlüsselfaktor für unseren Erfolg.

In der Vorlesung "Software Engineering" wurde uns OpenProject vorgestellt, eine Projektmanagement-Software, die uns bei der Planung und Durchführung unserer Projekte unterstützte. Mit OpenProject konnten wir unsere Projekte strukturiert planen und die Fortschritte hochladen. Es erlaubte uns, den Projektstatus wöchentlich zu aktualisieren und auf dem neuesten Stand zu halten.

OpenProject erwies sich als äußerst nützliches Werkzeug, um eine zuverlässige Projektzusammenarbeit und -kommunikation von der Projektidee bis zur abschließenden Dokumentation zu gewährleisten. Die Software half uns, Ideen zu sammeln und den Umfang des Projekts klar festzulegen. Zudem behielten wir mithilfe von OpenProject alle Termine und Meilensteine im Blick.

Besonders hilfreich war das Gantt-Diagramm in OpenProject. Dieses Diagramm zeigte die Arbeitspakete in einer übersichtlichen Zeitleiste an und ermöglichte es uns, den Projektplan gemeinschaftlich zu erstellen und zu verwalten. Wir konnten Start- und Endtermine festlegen und sie bei Bedarf im Diagramm durch einfaches Verschieben anpassen. Zudem ermöglichte es uns, Abhängigkeiten zwischen den Arbeitspaketen zu definieren, um die Beziehungen zwischen den einzelnen Aufgaben darzustellen.



In Openprojekt verfolgten wir stets das Ziel, uns selbst klare Aufgaben zu setzen, die bis zu einem bestimmten Datum erledigt werden sollten. Diese Funktion erwies sich als äußerst hilfreich, da sie uns ermöglichte, unsere Ziele schrittweise und mit klarem Fokus vor Augen zu erreichen. Es war eine wunderbare Möglichkeit, unsere Fortschritte zu verfolgen und uns kontinuierlich motiviert zu halten, während wir Schritt für Schritt auf unsere Ziele hinarbeiteten.

Insgesamt erwies sich OpenProject als wertvolles Werkzeug für unsere Teamarbeit und half uns, den Projektverlauf transparent und strukturiert zu gestalten. Durch die effektive Nutzung dieses Projektmanagement-Tools konnten wir unseren Projekterfolg sicherstellen und unsere Zusammenarbeit optimieren.

### • 3. Welche Analyse- und Erhebungsmethoden habt ihr zur Erhebung der Anforderungen wie eingesetzt? (Junior Ekane)

Im Rahmen unserer Produktentwicklung haben wir verschiedene Analyse- und Erhebungsmethoden angewendet, um die Anforderungen an die zu entwickelnde Software zu erfassen. Es standen aus der Vorlesung eine Vielzahl von Methoden zur Verfügung, die wir uns zunutze gemacht haben. Wir haben gezielt bestimmte Methoden gewählt, die wir für unser Projekt interessant und nützlich gefunden

haben. Diese Methoden stammen alle ursprünglich aus der Veranstaltung Softwareengineering II. Unsere Herangehensweise umfasste die Nutzung folgender Methoden: Recherchen, Brainstorming, Interviews, User Stories, Personas.

Recherchen: Wir haben intensiv im Internet recherchiert, um bestehende Lösungen, ähnliche Produkte oder relevante Industriestandards zu analysieren. Wie haben uns zahlreiche Webseiten von Umzugsunternehmen angeschaut, damit wir ungefähr wissen wie das generell aussieht, um ein ähnliches Produkt zu entwerfen, das sich allerdings von den schon existierenden abhebt. Im Bezug auf Terminbuchungssysteme haben wir uns solche Systeme wie das von der Ausländerbehörde Bremerhaven und das von dem Rathaus Wolfenbüttel angeschaut. Hilfreich war das sehr, weil wir bemerkt haben, dass die fast alle die gleichen Eigenschaften haben, aber unser Kunde wollte etwas Einzigartiges. Deshalb haben wir uns entschieden, eine Single-Page-Application kurz SPA zu bauen.

Brainstorming (Ideensammlung siehe Anhang, vergleiche [12]): In Gruppensitzungen beziehungsweise im Rahmen der SWE2-Übung haben wir Ideen frei ausgetauscht und kreative Gedanken gesammelt. Das Brainstorming hat uns dabei geholfen, potenzielle Funktionen, Nutzerinteraktionen und mögliche Anforderungen zu generieren, worauf sich der Kunde sehr gefreut hat. Die Ideen, die dabei gesammelt wurden, waren folgende: Urlaubsplanungssystem, Terminvergabesystem, Verbesserungsvorschlagssystem, Feedbacksystem. Das hier sind die Ideen, die wir am Anfang gehabt haben, als wir uns noch unsicher waren, was genau wir für unseren Kunden programmieren können und wie. Aber dann haben sich sehr viele Sachen im Laufe vom Projekt geändert und auch dazu gekommen.

Interviews (vergleiche [3]): Durch Gespräche mit Stakeholdern, einschließlich Kunden, möglichen Nutzern, haben wir direktes Feedback erhalten. Die Interviews halfen uns, Ihre Bedürfnisse, Wünsche und Schmerzpunkte zu verstehen und somit fundierte Anforderungen abzuleiten. Mit dem Kunden haben wir verschiedene Interviews beziehungsweise Gespräche durchgeführt, jedoch war eins am wichtigsten, wobei wir Kernfragen gestellt haben, die uns eine bessere Orientierung gegeben haben. Für das Interview haben wir die Vorlage aus den Übungen zum Protokollieren verwendet und dazu auch noch die Antworten notiert und aufgezeichnet, damit wir noch später das Interview hören und besser eine optimale Erhebung machen können. Dieses einen Interview haben wir protokolliert (siehe Anhänge) und dafür vorab zu stellenden Fragen aufgeschrieben. Fragen wie „Was wird gewünscht?“, „Was soll es können?“, „Wie lief der Prozess bisher?“, „Was sind die Kapazitäten?“, „Was sind die buchbaren Zeiten? Was sind spezielle Anforderungen?“, „Wie werden die Mitarbeiter informiert?“, „Was ist für die Benutzeroberfläche wichtig?“, „Was ist die Zielgruppe?“ und weitere wurden gestellt.

User Stories (siehe Anhänge, vergleiche [11]): Da wir im Rahmen der Produktentwicklung ein auf den Markt zu bringendes Endprodukt entwerfen sollten, mussten wir dafür sicherstellen, dass wir sowohl die Anforderungen der Kunden (Auftraggeber) und der künftigen Nutzer berücksichtigen, haben wir auch User Stories erstellt. Hierbei haben uns hauptsächlich auf die Meinung der Nutzer fokussiert und ihre Anforderungen aus deren Perspektive zu definieren. Für unsere User Stories haben wir keine Muster benutzt, sondern wir haben die Nutzer eine grobe Zusammenfassung von unserem Projekt gemacht und gefragt, was sie sich von so eine Anwendung wünschen.

Personas (vergleiche [10]): Für unser Projekt mussten wir Personas erstellen. Voraussetzung für deren Erstellung war aber die Identifizierung der Zielgruppe unserer Anwendung analysieren. Für diese Identifizierung haben wir uns überlegt, was für Leute die Anwendung benutzen würden. Erstmal haben wir Leute unter 16 Jahren ausgeschlossen und auch dann Studierende nicht betrachtet, aber auch nicht komplett ausgeschlossen, weil wir selber als Studenten selten solche Dienstleistungen brauchen. Dann haben wir uns eher auf berufstätige bzw. arbeitslose Leute fokussiert. Diese Art von Leuten sind für uns besonders interessant, denn sie ziehen entweder oft um oder brauchen ähnliche Dienstleistungen für

den Transport von schweren Möbeln. Durch die Erstellung von Personas haben wir fiktive, aber realitätsnahe Nutzerprofile erstellt und konnten Anforderungen aus Sicht einzelner Nutzergruppen als Rolle ermitteln. Dies half uns, uns in die verschiedenen Nutzergruppen hineinzusetzen und deren Anforderungen besser zu verstehen und unsere Vorstellung der Anwendung an diese anzupassen.

Insgesamt haben wir durch die Kombination dieser Analyse- und Erhebungsmethoden sicherstellen können, dass wir vielfältige Perspektiven und Anforderungen erfasst haben. Diese Ansätze ermöglichen es uns, eine umfassende Grundlage für die Entwicklung der Software zu schaffen, die den Anforderungen der Stakeholder gerecht wird und gleichzeitig innovative Lösungen bietet.

## • **4. Wie habt ihr die Anforderungen geordnet und strukturiert („FURPS“)? Usw.** (vergleiche [7]) (Junior Ekane)

Wir haben die erfassten Anforderungen an die zu entwickelte Software mithilfe des FURPS-Modells (Functionality, Usability, Reliability, Performance, Supportability) geordnet und strukturiert. Dieses Modell bietet einen bewährten Ansatz, um die verschiedenen Aspekte der Anforderungen zu kategorisieren und zu priorisieren. Die Strukturierung lief als Team ganz gut und wir haben uns geeinigt, wie wir das gezielt machen möchten. Hier ist, wie wir das FURPS-Modell angewendet haben:

-Functionality (Funktionalität): Wir haben die Hauptfunktionen der zu entwickelnde Anwendung identifiziert, die basierend auf den gesamten Informationen aus Brainstorming, Interviews, User Stories und Recherchen relevant waren. Diese von uns gewählten Hauptfunktionen (siehe Anwendungsfalldiagramm) bildeten den Kern der Funktionalitäten, die die Anwendung sowohl dem Nutzer als auch dem Geschäftsführer bieten sollte.

- Usability (Benutzerfreundlichkeit): Im Rahmen des Entwurfs der Software haben wir uns für das Thema User Centered Design entschieden. Daher war für uns von Belang, dass die Oberfläche benutzerfreundlich und sehr leicht zu verstehen ist. Die Anforderungen zur Benutzerfreundlichkeit wurden auf Grundlage der Nutzerinteraktionen, die wir durch User Stories, Brainstorming und Interviews definiert haben, strukturiert. Wir haben festgelegt, wie die Benutzeroberfläche gestaltet werden sollte, um eine intuitive und effiziente Nutzung der Software zu ermöglichen.

- Reliability (Zuverlässigkeit): Basierend auf den Erfahrungen aus Recherchen und den Gesprächen mit dem Kunden haben wir Anforderungen zur Stabilität und Fehlerfreiheit der Software festgelegt. Wir haben definiert, wie die Software mit unvorhergesehenen Situationen umgehen sollte, um eine zuverlässige Nutzung zu gewährleisten. Dennoch ist hier darauf hinzuweisen, dass nicht all das, was sich der Geschäftsführer gewünscht hat, mit unserem aktuellen Niveau und die zur Verfügung stehenden Werkzeuge umsetzbar waren. Doch war er am Ende zufrieden.

- Performance (Leistung): Die Leistungsanforderungen wurden durch eine Kombination von Informationen aus Recherchen und Brainstorming ermittelt. Hierbei haben wir definiert, wie schnell die Software reagieren sollte, welche Verarbeitungsgeschwindigkeiten erforderlich sind und wie die Skalierbarkeit gewährleistet wird. In Bezug auf die Schnelligkeit der Anwendung haben wir uns überlegt, wie wir die Interaktion mit der Seite beschleunigen könnten. So haben wir uns für den Aufbau einer Single-Page-Anwendung entschieden. Das Gute daran ist nicht nur, dass wir den Server mit einem Haufen von Anfragen belasten, sondern wir fragen auch nur nach Notwendigkeit an. Dadurch sind die Serverantworten schneller, weil die Seite nicht jedes Mal neu geladen werden soll.



- Supportability (Wartbarkeit): Die Anforderungen an die Wartbarkeit und Erweiterbarkeit der Software wurden unter Berücksichtigung von Recherchen festgelegt. Wir haben das eigentlich nicht anders machen können, als das so zu gestalten, dass Änderungen nur durch uns (Team Ahnungslos) möglich sind. Wir stehen dem Geschäftsführer jeder Zeit zur Verfügung, wenn es vorkommen sollte, dass er irgendeine Änderung vornehmen möchte. Und da die Anwendung in den Markt gebracht werden soll, haben wir den Quellcode so leicht, aber frei von Sicherheitslücken geschrieben, dass wir Erweiterungen leicht vornehmen können. Durch Erweiterungen werden das Hinzufügen und die Benutzung neuer Technologien verstanden.

Indem wir die erfassten Anforderungen in diese fünf Kategorien unterteilt und strukturiert haben, konnten wir eine klare Grundlage für die weitere Entwicklung schaffen. Das FURPS-Modell half uns dabei, die verschiedenen Aspekte der Anforderungen zu organisieren, Prioritäten zu setzen und sicherzustellen, dass sowohl die funktionalen als auch die nicht-funktionalen Anforderungen angemessen berücksichtigt wurden. Dies erleichterte die Kommunikation mit dem Kunden und anderen Stakeholdern sowie die gezielte Umsetzung der Softwareentwicklung.

## • 5.Erläuterung wichtiger Codeschnipsel (Junior Ekane)

```
15 //Vorbereitung der Verbindung zu Datenbank
16 $conn = mysqli_connect($servername, $username, $password, $db);
17 $abholort = mysqli_real_escape_string($conn, $_POST['abholort']);
18 //richtige Codierung, um Fehler zu vermeiden
19 mysqli_set_charset($conn, 'utf8');
20 mysqli_query($conn, "SET NAMES 'utf8'");
21
22 // Überprüfen, ob ein Termin für den gleichen Kunden mit den gleichen Daten bereits existiert
23 $query = "SELECT * FROM Termine WHERE Datum = ? AND Uhrzeit = ? AND Abholort = ? AND Zielort = ? AND Notiz = ?";
24 $stmt = mysqli_prepare($conn, $query);
25 mysqli_stmt_bind_param($stmt, 'sssss', $datum, $uhrzeit, $abholort, $ziel, $notizen);
26 mysqli_stmt_execute($stmt);
27 $result = mysqli_stmt_get_result($stmt);
28
29 if ($result && mysqli_num_rows($result) > 0) {
30     // Ein Termin mit denselben Daten existiert bereits
31     echo "Der Kunde hat bereits einen Termin mit den gleichen Daten vereinbart.";
32 } else {
33     // Ein Termin mit den angegebenen Daten existiert noch nicht, füge den neuen Termin hinzu
34     $sql = "INSERT INTO Termine (KundenID, Name, Zielort, Abholort, Notiz, Datum, Uhrzeit) VALUES (?, ?, ?, ?, ?, ?, ?)";
35     $stmt = mysqli_prepare($conn, $sql);
36     mysqli_stmt_bind_param($stmt, 'sssssss', $id, $name, $ziel, $abholort, $notizen, $datum, $uhrzeit);
37     mysqli_stmt_execute($stmt);
38
39     // Überprüfen, ob das Einfügen erfolgreich war
40     if (mysqli_stmt_affected_rows($stmt) > 0) {
41         echo "Termin erfolgreich angelegt";
42     } else {
43         echo "Fehler beim Einfügen des Termins";
44     }
45 }
46 mysqli_close($conn);
47 }
```

23,1

Bot

Bei unserer Anwendung handelt es sich um ein Terminverwaltungssystem. Zur Terminbuchung und damit sowohl der Geschäftsführer als auch künftige Nutzer Termine buchen und einsehen können, müssten wir die Termindaten in einer Datenbank Mariadb speichern und abrufen. Hier Oben ist der Code dafür.

Im ersten Schritt stellen wir mit PHP eine Verbindung zur Datenbank her. Dafür verwenden wir aus Sicherheitsgründen unsere Verbindungsdaten (Servername, Username, Password und Datenbank), die sich in einem separaten Ordner befinden. Bei den ersten Versuchen ist es uns aufgefallen, dass solche Buchstaben wie ä,ü,ö usw. vom Server nicht in „UTF-8“<sup>[1]</sup> codiert waren und deshalb falsch vom Server interpretiert wurden. Aus diesem Grund haben wir mit der PHP-Funktion „mysqli\_real\_escape\_string“ die richtige Codierung gesetzt. Und noch sicherheitshalber die gleiche Codierung für all die übergebenen Elemente.

Nachdem alle Daten empfangen und richtig codiert wurden und eine sichere Verbindung zur Datenbank erfolgreich hergestellt wurde, wird erst einmal geprüft ob, der Kunde bereits einen Termin mit identischen Daten gebucht hat. Dies passiert, damit die Datenbank nicht umsonst belastet wird und der Geschäftsführer keinen Termin verwechselt. Dabei haben wir auch an die Sicherheit der Anwendung gedacht und unsere Datenbank gegen SQL-Injection-Angriffe[2] geschützt. Dabei kodieren wir die zu prüfende Daten mit als Key bevor, sie an den Datenbankserver gesendet werden. „s“ bezeichnet den Typ der zu sendende Daten.

Bei unserem Projekt haben wir nur mit dem Datentypen String (Zeichenkette) gearbeitet. Je höher die Anzahl dieser Daten, desto öfter kommt dieses „s“ vor. Im Falle der Fälle, dass ein Termin vom gleichen Kunden mit den gleichen Daten vereinbart wurde, wird die Meldung „Der Kunde hat bereits einen Termin mit den gleichen Daten vereinbart“ angezeigt. Sonst wird dann versucht, die Termindaten in der Datenbank zu speichern und da haben wir die gleiche Vorgehensweise verwendet, nur dass wir jetzt gar keine Abfrage zur Überprüfung gemacht haben, sondern eine zum Hinzufügen. Anschließend prüfen wir, ob das Einfügen vom Termin erfolgreich war. Das machen wir, indem wir die Anzahl der hinzugefügten Zeilen in der Tabelle prüfen. Wenn sie größer null sind, dann bekommt der Nutzer die Meldung „Termin erfolgreich angelegt“ und wenn kleiner, dann erhält er eher die Meldung „Fehler beim Einfügen des Termins“. Zum Schluss wird die Verbindung zur Datenbank getrennt, damit keiner versuchen kann auf die zuzugreifen, wenn der gewünschte Vorgang schon abgeschlossen ist.

## • **6. Ergebnisse der Ist-Analyse** (Steve Aguiwo II)

Während der Analysephase unseres Projekts zur Produktentwicklung haben wir den aktuellen Zustand untersucht und dabei Probleme sowie Schwachstellen identifiziert. Unsere Untersuchungen ergaben Folgendes:

### • **6.1. Problembeschreibung** (Steve Aguiwo II)

Vor unserer zweiten Präsentation (Endpräsentation), haben wir eine Umfrage bei unseren Kameraden durchgeführt, damit sie uns helfen unser Projekt zu testen um uns positive oder negative Rückmeldungen geben zu können, um Fehler zu korrigieren und auch zu testen, ob es einwandfrei funktioniert. Nach dieser Umfrage ergeben sich verschiedene Ergebnisse:

Alles, was auf Englisch war, sollte ins Deutsche übersetzt werden, denn für die meisten, die kein Englisch verstehen und sprechen, wird dies nicht offensichtlich sein. Dies geschieht also, damit jeder unser Projekt verstehen und nutzen kann. Die Buchungsdauer von Terminen müssen weg gemacht werden, denn das war unnötig und konnte zur Verwirrung führen, so haben wir das weg gemacht und den Prozess bezüglich der Terminbuchung zu vereinfachen. Nachdem man sich registriert hat, sollte das Anmeldeformular weg und die Nachricht "ihre Registrierung war erfolgreich" sollte angezeigt werden. Diese verschiedenen Bemerkungen wurden dann ernst genommen und wir haben verbessert was es zu verbessern gab. Allerdings gibt es weitere Probleme und zwar, wenn sich ein neuer Benutzer auf der Plattform anmeldet, erhält er eine sichtbare Bestätigung, dass seine Registrierung erfolgreich war. Es wird eine Meldung angezeigt, die angibt, dass die Registrierung erfolgreich war und die vom Benutzer angegebene E-Mail-Adresse enthält. Das eigentliche Problem besteht jedoch darin, dass das System nicht überprüfen kann, ob diese vom Benutzer angegebene E-Mail-Adresse tatsächlich existiert

und gültig ist. Daher können wir als Systemadministratoren nicht sicher sein, dass die von den Benutzern angegebenen E-Mail-Adressen authentisch sind. Dies stellt eine große Herausforderung für die Zuverlässigkeit und Integrität der Daten in unserem System dar. Die Tatsache, dass es vorkommen kann, dass eine falsche E-Mail oder eine nicht existierende E-Mail eingegeben werden kann, kann dazu führen, dass Mitarbeiter und Kunden gezwungen sind, manuell zwischen verschiedenen Systemen und Kommunikationskanälen zu wechseln, um Termine festzulegen und abzustimmen. Dieser manuelle Ansatz birgt die Gefahr von Missverständnissen, Doppelbuchungen von Terminen den Kunden. Weiterhin ist der derzeitige Terminplanungsprozess aufgrund der Ineffizienz der Terminplanung und der Verschwendung von Ressourcen nicht optimal. Es ist ein chaotischer Prozess, der viel Zeit und Energie in Anspruch nimmt.

## • **6.2.Vorhandene Schwachstellen** **(Steve Aguiwo II)**

**Mangelnde Effizienz:** Der Terminplanungsprozess ist schlecht organisiert, was zu Verzögerungen und Ineffizienzen führt. Da es keine klare Struktur für die effiziente Organisation von Terminen gibt, werden Zeitressourcen unnötig verschwendet.

**Fehlende Überprüfung der Existenz von E-Mail:** unser System kann zur Zeit nicht überprüfen, ob die E-Mail von den Benutzern real oder existent ist.

**Die Unsicherheit der Datenqualität:** Aufgrund der Unfähigkeit, E-Mail-Adressen zu verifizieren, gibt es einen Mangel an Vertrauen in die Qualität der Datensätze

**die fehlende Authentizität der Benutzerdaten:** Aufgrund der fehlenden Verifizierung der E-Mail-Adressen kann der Benutzer eine beliebige E-Mail-Adresse angeben, was für den Administrator problematisch ist.

**Vertrauensverlust im System:** Unfähigkeit zu garantieren, dass E-Mail-Adressen ungültig sind, kann zu einem Vertrauensverlust in die Zuverlässigkeit des Systems führen.

**Unkoordiniertes verläuft:** Mangel an effektiver Koordination zwischen den verschiedenen Phasen des Terminplanungsprozesses kann zu Fehlern führen.

**Verzögerungen bei der Bearbeitung:** Aufgrund der Ineffizienz des Prozesses kann die Bearbeitung von Terminanfragen länger dauern.

**Unsachgemäße Nutzung von Ressourcen:** Ineffiziente Planung von Terminen kann zu einem Missbrauch der verfügbaren Ressourcen, einschließlich Personal und Zeit, führen.

**Hohes Fehlerrisiko:** Fehler können aufgrund mangelnder Koordination auftreten, was sich stark auf die Qualität der Dienstleistung auswirken kann.

**Ineffizienz:** Die Ineffizienz des Systems kann zu unnötigem Zeitverlust und ineffizienter Ressourcennutzung führen.

- **7.Zusammenfassung der Anforderungen bzw. Darstellung der eigenen Idee und des Anwendungskontextes (mit Verweis auf Software-Anforderungsspezifikation)**  
**(Steve Aguiwo II)**

Durch die Anwendung einer Vielzahl von Analysetechniken wurden die Anforderungen für die zu entwickelnde Software erfasst und strukturiert. Wir konnten viel über die Bedürfnisse und Anforderungen der Stakeholder erfahren, indem wir Recherchen, Brainstorming, Interviews, User Stories und Personas kombinierten. Um sicherzustellen, dass sowohl funktionale als auch nicht-funktionale Aspekte angemessen berücksichtigt wurden, konnten wir die erfassten Anforderungen durch die Verwendung des FURPS-Modells in klare Kategorien gliedern und priorisieren.

Wir konnten mit diesem umfassenden Ansatz eine solide Grundlage für die Entwicklung einer Single-Page-Application (SPA) schaffen, die sowohl den Anforderungen des Kunden als auch den Erwartungen der Nutzer entspricht. Wir konnten die Anforderungen durch die Verwendung des FURPS-Modells organisieren. Die gesamte Analyse stellt sicher, dass die Softwareentwicklung auf einer soliden Basis aufbaut und kreative Lösungen bietet.

- **8.Zusammenfassung der Anforderungen bzw. Darstellung der eigenen Idee, des Anwendungskontextes und der Zielgruppe (Personas).**  
**(Maeva Lekeufack)**

Gerade viele kleinere Unternehmen arbeiten noch ohne Terminplanungssystem, wodurch alle Termine persönlich abgemacht werden müssen. So war es auch der Fall bei unserem Kunden. Unser Terminplanungssystem trägt dazu bei, ineffektive Terminplanung und verschwendete Ressourcen, durch den unkoordinierten und zeitintensiven Prozess zu vermeiden. wenig Arbeit und mehr Transparenz für die Kunden und den Geschäftsführer zu ermöglichen. Eine erfolgreiche Lösung besteht darin, es möglich zu machen, dass verschiedene Personen mit verschiedenen Berechtigungen für sie relevante Daten schnell und einfach einsehen können, sowie dass mehrere Personen gleichzeitig freie Termine buchen können, schnell, zu jeder Zeit und bequem.

Von unserer Lösung profitieren:

-Geschäftsführer: Leiter des Unternehmens =>verwaltet die Termine und nimmt Termine an.  
-Kunde : Person, die das System benutzt => Der Kunde nutzt das System, um Termine zu buchen, seine Termine einzusehen und diese zu stornieren.

-spezifische Anforderungen(Funktionale Anforderungen):

Die Geschäftsführer kann Rollen und Berechtigungen einsehen und verwalten, Anfragen ablehnen, Termine hinzufügen und einsehen, Kundendaten und Anfragen einsehen, Anfragen annehmen. Kunden können ihre Termine einsehen, absagen und Terminanfrage erstellen und ein Unbekannt kann

sich registrieren und sich anmelden. Es geht hier um eine Person, die die Seite zum ersten Mal benutzt. Dafür haben wir ein UML-Anwendungsfalldiagramm und Anwendungsfall-Spezifikationen in Form von UML-Aktivitätsdiagrammen erstellt.

Zu nicht-Funktionalen Anforderungen gehören Benutzungsfreundlichkeit, Zuverlässigkeit, Leistung, Servicefreundlichkeit, Designeinschränkungen, Anforderungen an die Onlinedokumentation für Benutzer und Hilfesystem, gekaufte Komponenten, Schnittstellen (Benutzer-, Hardware-, Software-, und Übertragungsschnittstellen), Lizenzanforderungen, Rechtliche Hinweise, Copyright und Bemerkungen, geltende Standards.

Bedarf und Features :

-Funktionen mit hohen Prioritäten (Rechte-System => Zugriff auf verschiedene Daten je nach Berechtigung; Termin Verwaltung => Termine verwalten; Nachvollziehbarkeit => Änderungen sollen nachvollziehbar sein und müssen bestätigt werden; Zuverlässigkeit=>alle Termine müssen von beiden Seiten bestätigt werden). Bedarfe mit mittels Prioritäten (E-Mail Schnittstelle=>über die Schnittstelle sollen Bestätigungen und Erinnerungen versandt werden; Erleichterung der Terminbuchung => standardwerte sollen die Eingabe vereinfachen, Übersichtlichkeit soll die Eingabe vereinfachen). Täglicher Terminplan mit niedriger Priorität => es soll täglich ein Plan der Termine geschickt werden an Mitarbeiter/ Geschäftsführung.

Die Produktpositionierung ist für Familien, Alleinerziehende Eltern und Senioren, Firmen. Es ermöglicht einfaches und simples Terminbuchen, sowie einfaches Einsehen, Zuweisen und Verwalten von Terminen. Unser Terminverwaltungssystem zeichnet sich dadurch aus, dass es eine einfache übersichtliche Handhabung ermöglicht, sowie die Möglichkeit diese zentral zu bearbeiten im Gegensatz zu manuellen Terminvereinbarung mit folgenden Eigenschaften: sichere und einfache Terminbuchung, zentrale Bearbeitung und die Möglichkeit für seine Rolle wichtige Daten schnell einzusehen.

- **9.Erläuterung, wie die erhobenen Anforderungen modelliert und dokumentiert wurden**
- **9.1.Welche Methoden und UML-Diagramme wurden jeweils für welchen Zweck verwendet?**  
(Hossein Akbari)

Als Team wurde durch Brainstorming für unser Projekt Ideen gesammelt, um zu schauen, in welche Richtung das zu machende Projekt ungefähr gehen soll, und diese mit der Gruppe ausdiskutiert.

Hierfür hat das Team in Form von einem Mind-Map alle zum Softwareprojekt zu machenden Schritte und möglichen Aufgaben notiert, und sich dadurch visuell dargestellt, um mögliche Konzepte und sonstige Informationen für das Projekt, die in einer hierarchischen Struktur im Zusammenhang stehen, strukturell zu präsentieren, sowie um dadurch eine grobe Einschätzung für den Projektumfang zu bekommen.

Indem für das zu machende Softwareprojekt kreative Ideen gesammelt worden sind, wurden alle Beteiligten im Team dazu angeregt, sowie inspiriert, durch das gemeinsame Assoziieren im Team, ihre für das Softwareprojekt relevanten Gedanken das ganze Team zuteilwerden zu lassen, sowie alle Aspekte des Projekts in Betracht zu ziehen.

Bezüglich der Strukturierung wurde das Mindmap dazu benutzt, um die gesammelten Ideen, in einer Hierarchieform zu strukturieren. Hierfür wurden Hauptaufgaben/Hauptziele in der Mitte positioniert,

während Subthemen, die von Wertigkeit eine kleinere Rolle gespielt haben, sich der größeren Themen, in verschiedenen Zweigen, unterkategorisiert haben.

Durch das Mindmap wurde ebenfalls erreicht, dass man durch eine sinnvolle, miteinander im Zusammenhang stehenden Themen in einer Struktur, sich einen schnellen und übersichtlichen Überblick für das zu machende Projekt erschafft, wodurch jedes Teammitglied schnell sehen kann, wie die jeweiligen Schritte und Aufgaben im Projekt miteinander interagierbar sind, sowie wie die Themen im Gesamtbild passen.

Zudem diene die Mindmap dazu, verschiedene Aufgaben und Schritte nach ihrer Wertigkeit, sowie Dringlichkeit zu priorisieren, um dadurch eine bessere Strategische Vorgehensweise im Verlaufe des Entwicklungsprozesses zu erleichtern, und hierdurch im Voraus das wichtigere und/oder dringlichere vom unwichtigeren und/oder nicht dringlicherem zu unterscheiden (vgl. [2]).

Des Weiteren hat das Team sich im Internet ähnliche Systeme angeschaut, um sich davon bei ihrer Vorgehensweise orientiert. Wie sind ähnliche Systeme aufgebaut, wie könnte das Team bei sich im Projekt Ähnlich anwenden, was sollte das Team bei der Entwicklung besser nicht nachbauen, etc.

Durch das Anschauen von ähnlichen Systemen hat das Team es sich ermöglicht, sich von verschiedenen in anderen ähnlichen Systemen schon bereits implementierten Ansätzen, Designen und Funktionen inspirieren zu lassen, mit der es das Team wiederum ermöglicht wurde, sich zu neuen Ideen anzuregen bezüglich dessen, wie bestimmte andere Funktionen im System implementiert werden sollten, sowie wie ähnliche komplexe Programmierprobleme im System gelöst werden sollten.

Zudem hat das Team dadurch gewonnen, die von in anderen Systemen schon bereits implementierten sowie angewendeten Methoden und Verfahren als Beispiel anzusehen und dadurch mögliche in der Zukunft zu entstehenden Fehlern im Voraus zu vermeiden und im Allgemeinen von den negativen sowie positiven Erfahrungen anderer Entwickler zu lernen (vgl. [2]).

Das Team hat für die Dokumentation der Anforderung des Softwaresystems ebenfalls mit dem Kunden ein Online-Interview durchgeführt. Hierzu hat das Team sich im Voraus Fragen überlegt, die für das Erheben der Anforderungen des Softwaresystems relevant sind.

Die Fragen hierzu waren: Was für ein Softwaresystem wird gewünscht? Was soll das Softwaresystem können? Wie läuft der Geschäftsprozess bisher ab? Was sind die Kapazitäten? Wann sind die buchbaren Zeiten? Was sind spezielle Anforderungen des Kunden? Was für Funktionen soll das System haben? Was ist für die Benutzeroberfläche wichtig? Was stellen Sie sich vor, wie Sie die Anwendungen nutzen können? Was sind für ihre Kunden wichtig? Was ist ihre Zielgruppe? Was ist der Bedarf?

Dadurch, dass der Kunde diese Fragen beantwortet, werden zu den verschiedenen Aspekten des Softwaresystems, verschiedene Anforderungsarten bezüglich der Softwarefunktionalität, der Geschäftsprozesse, der Benutzererwartungen, sowie technische Aspekte modelliert, sowie dokumentiert.

Die Funktionsanforderungen des Softwaresystems werden durch die Fragen erhoben, bei denen danach geschaut wird, was konkret für ein Softwaresystem gewünscht wird, sowie was das System konkret können muss. Durch diese Fragen werden eine grobe Übersicht über die zu programmierenden Funktionalitäten des Systems notiert, die im weiteren Verlaufe der Entwicklung präzisiert werden können, wie zum Beispiel die tatsächliche Buchung des Termins, der Prüfung der Verfügbarkeit eines Termins, usw. Durch die Frage, was für Funktionen das System haben soll, können präzisere Details über manche zu entwickelnde Funktionen dokumentiert werden, die im Nachhinein im System implementiert werden sollen.

Die Geschäftsprozessanforderungen des Softwaresystems werden durch die Fragen erhoben, bei denen danach geschaut wird, wie konkret der Geschäftsprozess bisher läuft. Hierdurch kann der bis zu dem aktuellen Zeitpunkt ablaufenden Prozesse im Unternehmen dokumentiert werden, die im Nachhinein dazu führen könnten, manche Prozessverbesserungen oder Automatisierungen im Verlaufe der Entwicklung des Softwaresystems zu berücksichtigen.

Die Ressourcen- und Kapazitätsanforderungen des Softwaresystems werden durch die Fragen erhoben, bei denen danach gefragt wird, was konkret die Kapazitäten im Unternehmen sind, sowie was die buchbaren Zeiten für den Umzugsservice im Unternehmen sind. Hierdurch werden über die im Unternehmen übrigen Ressourcen und Kapazitäten im Unternehmen Informationen gesammelt, die man in der Implementierung der Funktionalitäten des Softwaresystems, sowie in der Entwicklung des Systems berücksichtigen kann.

Die Benutzeranforderungen des Softwaresystems werden durch die Fragen erhoben, bei denen danach gefragt wird, was die speziellen Anforderungen des Kunden sind, sowie was für den Kunden im Allgemeinen wichtig ist. Durch diese Fragen kann man die Erwartungen und Bedürfnisse, die ein Kunde an einem Softwaresystem hat, erheben, mit der man an der benutzerdefinierten Anpassung, sowie der Benutzerfreundlichkeit des zu entwickelnden Softwaresystem arbeiten kann.

Die Nutzererwartungen und Nutzungsszenarien des Softwaresystems werden durch die Fragen erhoben, bei denen danach gefragt wird, was der Kunde sich bezüglich dessen vorstelle, wie er die Anwendung nutzen könnte, mit der man Informationen über die zu erwartenden Szenarien der Nutzung des Softwaresystems sammeln kann, mit der man zur benutzerfreundlicheren Entwicklung des Softwaresystems ebenfalls beitragen kann.

Die Zielgruppen- und Marktanforderungen des Softwaresystems werden durch die Frage erhoben, in der danach fragt wird, was für die Zielgruppe des Kunden als Unternehmer und Geschäftsführer wichtig ist. Dies hilft dabei, bei der zu entwickelnden Softwaresystems die Zielgruppe der Software zu visualisieren und die Softwareanforderungen, an den Anforderungen des Kunden anzupassen (vgl. [3]).

Das Team hat außerdem zu repräsentativen Vertretern der Zielgruppe, Personas erstellt. Hierzu hat das Team die wesentlichen Eigenschaften der potenziellen Nutzer des Systems zusammengetragen, die wichtigsten Benutzergruppen zusammengetragen, und die repräsentativen Eigenschaften dieser Gruppen, in einer Persona zusammengefasst. Danach die Personas beschrieben: Name, Foto der Person, demographische Informationen, Beruf und Hauptaufgabe, Ziele, Wünsche, Erwartungen, und etc. ausgearbeitet.

Das Erstellen von Personas bietet eine gute Methodik dafür dar, um die zukünftigen Nutzer des Systems, sowie dessen Erwartungen und Wünschen besser nachvollziehen zu können. Von diesen Informationen aus kann man herausleiten, welche verschiedene Arten von Anforderungen an das zu entwickelte Softwaresystem zu identifizieren und zu dokumentieren sei. Dies versteht man als benutzerzentrierte Anforderung.

Für die funktionalen Anforderungen kann man Ziele und Aufgaben der Personas Als Basis der Identifikation von verschiedenen Funktionen und Interaktionen des Softwaresystems verstehen, die im Laufe der Entwicklung des Softwaresystem zu implementieren sind (vgl. [3.2]).

Das Team hat sich während der Entwicklung auch mit User Stories beschäftigt und danach die Ergebnisse der Anforderungserhebung sortiert. Durch das Erstellen von Personas als Ausgangspunkt hat das Team für das Softwaresystem User Stories entwickelt, mit der andere spezielle Anforderungspunkte aus Perspektive der Benutzer beschrieben werden können. Unter funktionale User Stories versteht man hierbei, dass man als Planer eines Umzugs in der Lage sein möchte,

individuelle Termine für verschiedene Kunden festzulegen, wodurch die Anforderungen eines Benutzers des Softwaresystems präziser und benutzerzentrierter dokumentiert werden können (vgl. [3.1]).

Das Team hat für das Softwaresystem ebenfalls ein Anwendungsfalldiagramm erstellt, mit der die funktionalen Anforderungen modelliert und dokumentiert wurden. Durch das Anwendungsfalldiagramm wurden zwischen den Akteuren des Systems die Interaktionen, die im System implementiert sind graphisch dargestellt, sowie die der verschiedenen Anwendungsfällen des Systems.

Anforderungen als Benutzerinteraktion werden repräsentiert dadurch, dass jeder dargestellte Anwendungsfall im modellierten Diagramm eine bestimmte Interaktion für zum Beispiel einen Benutzer oder ein externes, mit dem Softwaresystem verbundenen System repräsentiert.

Anforderungen als Benutzerziele und -funktionen werden repräsentiert dadurch, indem durch das modellieren eines Anwendungsfalles, immer auch ein Benutzerziel oder eine Funktion im System dargestellt wird, welches implementiert im Softwaresystem ausgeführt wird.

Anforderungen als Benutzerfreundlichkeit werden repräsentiert dadurch, indem durch die Modellierung der Funktionalitäten der Benutzer dargestellt wird, wie die verschiedenen Benutzer des zu entwickelnden Softwaresystems miteinander interagieren, sowie wie die zu entwickelnde Benutzeroberfläche des Systems gestaltet wird (vgl. [4.1]).

Für das Softwareprojekt wurden ebenfalls Qualitätssicherung, sowie das Skizzieren von Storyboards durchgeführt, mit der ebenfalls Anforderungen an das zu entwickelnde Softwaresystem dokumentiert wurden. Besonders durch das Skizzieren von Storyboards wurde es erreicht, Bezogen auf das Design, die Benutzerinteraktion, und die Benutzererfahrung des Systems zu modellieren und hierdurch einige Anforderungen zu dokumentieren.

Anforderungen als Benutzerinteraktion werden dokumentiert dadurch, indem durch die Skizzierung klargemacht wird, dass die Benutzer in der Lage sein müssen die verschiedenen Funktionalitäten des Systems benutzerfreundlich nutzen zu können, indem sie beispielsweise in der Lage sein müssen, die Termine auf eine intuitive Weise auszuwählen, sowie buchen zu können.

Durch die Skizzierung wird zum Beispiel außerdem klar, dass die Benutzer, zwecks der Benutzerfreundlichkeit in der Lage sein müssen, eine gute Navigation und Menüführung zu genießen. So sollte ein Benutzer zum Beispiel in der Lage sein, zwischen den verschiedenen Abschnitten des Systems leicht hin und her wechseln zu können (vgl. [5]).

## **(Steve Aguiwo II)**

Das Team hat für jede Funktion des Anwendungsfalles ebenfalls ein Aktivitätsdiagramm erstellt. Hierdurch sollten innerhalb einer bestimmten Funktion des Softwaresystem, die internen Abläufe, sowie Interaktionen zu modellieren. Hierdurch können mehrere Formen von Anforderungen bezogen auf bestimmte Prozesse, sowie Interaktionen modelliert werden.

Hierzu gibt es beispielsweise einmal die Ablaufanforderungen, in der es heißt, dass während des Buchens eines Termins für den Benutzer, der abzulaufende Prozess klar definiert sein muss. Durch den



Aktivitätsdiagramm sieht man Schritt-für-Schritt den gesamten abzulaufenden Prozess oder Terminbuchung, einschließlich aller Eingaben, Entscheidungen und sonstiges, die erforderlich sind.

Bezüglich der Bedingungen und Entscheidungen muss das System zum Beispiel gut in der Lage sein, die Verfügbarkeit der Termine zu prüfen. Diese Anforderungen werden dadurch gewährleistet, dass das Aktivitätsdiagramm den gesamten Entscheidungsprozess bezüglich dessen, wie das System die benötigte Verfügbarkeit der Termine beispielsweise überprüft, sowie darauf basierend wiederum dem Benutzer die verfügbaren Termine anzeigt.

Als Anforderung zur Parallelität und Nebenläufigkeit ist es zum Beispiel wichtig, dass das System dazu in der Lage ist, mehrere Buchungen gleichzeitig bearbeiten zu können. Diese wird dadurch erreicht, indem durch das Aktivitätsdiagramm mehrere zu betätigende Buchungsanfragen gleichzeitig bearbeitet werden und parallel hierzu auch Verfügbarkeitsprüfungen durchführt (vgl. [6]).

Das Team hat sich danach ebenfalls mit Testfällen beschäftigt. Hierbei wurden für manche Funktionalitäten des Systems Testfälle erstellt. Durch das Erstellen von Testfällen zu manchen Funktionalitäten des Systems, kann man überprüfen, ob die Funktionen des Systems, sowie im Gesamtbild, das gesamte Softwaresystem einwandfrei funktioniert (vgl. [4]).

Für manche Funktionalitäten des Softwaresystems wurden ebenfalls Klassendiagramme erstellt.

Dadurch, dass man für das Softwaresystem Klassendiagramme erstellt, modelliert man, in welchen Beziehungen und in welcher Struktur die verschiedenen Klassen und Objekte im Softwaresystem stehen. Hierdurch könnte man die Anforderung überprüfen, ob das Softwaresystem dazu in der Lage ist, Informationen zu Benutzern und Terminen zu speichern. Mit dem Klassendiagramm kann man die Gültigkeit dieser Anforderung überprüfen, indem man die Klassen der „Benutzer“ und „Termine“, mit ihren jeweiligen Attributen und Beziehungen modelliert.

Im Bereich Vererbung und Polymorphismus kann man die Anforderung ableiten, ob das Softwaresystem verschiedene Arten von Benutzern mit unterschiedlichen Rechten gewährleistet. Mit dem Klassendiagramm kann man hierfür die Vererbungsbeziehung zwischen der sogenannten Basisklasse „Benutzer“, sowie der Klassen „Administrator“ und „Kunde“, die sich davon ableiten, prüfen (vgl. [14]).

## **(Reza Akbari)**

Danach hat das Team sich mit Sequenzdiagrammen beschäftigt, mit der versucht wurde, die Zusammenarbeit bzw. den Nachrichtenaustausch zwischen Objekten oder Komponenten des Systems darzustellen.

Durch das Sequenzdiagramm wird sichergestellt, zu modellieren, wie der zeitliche Ablauf von verschiedenen Interaktionen zwischen mehreren Objekten und Klassen im Softwaresystem ist. Bezüglich der Interaktionen zwischen Benutzern und System, wird durch das Sequenzdiagramm und das Zeigen von Abläufen überprüft, wie ein bestimmter Benutzer mit einem System interagiert, bei dem Versuch ein Termin zu buchen oder zu ändern.

Möchte man die Anforderungen überprüfen, ob der Datenfluss und Datenverarbeitung im Softwaresystem einwandfrei funktionieren und die Daten dabei richtig verarbeitet und gespeichert werden, so kann man durch den Sequenzdiagramm den Datenfluss zwischen den im Softwaresystem zuteilwerdenden verschiedenen Klassen anschauen, sowie betrachten, wie die Daten verarbeitet und letztendlich in der Datenbank gespeichert werden (vgl. [15]).

Zuletzt hat das Team sich ebenfalls mit der Qualitätssicherung vorhandener Artefakte beschäftigt, bei dem wir für die Aktivitätsdiagramme eine Checkliste durchgeführt haben, dasselbe für Testfälle und Klassendiagrammen. Zudem haben wir Prototyp-Testing durchgeführt. Hierfür haben wir anhand von vorgegebenen Beispieldaten, Prototyp-Tests durchgeführt.

Durch den Prototyp-Test ermöglicht man, die zum Beispiel für das zu verwendende Softwaresystem, die Benutzerfreundlichkeit und die damit gemachte Erfahrung, mit einem früheren Prototyp zu überprüfen. Für die Benutzerfreundlichkeit und der Interaktion mit dem Softwaresystem ist es wichtig, dass zum Beispiel die Benutzeroberfläche des Softwaresystems intuitiv und leicht verständlich ist. Durch den Usability-Walkthrough ermöglicht man es den Benutzern, zu testen, wie simpel und einfach es für sie ist, sich durch die Benutzeroberfläche zu bewegen und dort manche Aufgaben und Funktionen zu machen und zu betätigen.

Für die Einhaltung der Anforderung bezüglich der Effizienz der Interaktion ist es wichtig, der der Benutzer, die zu machende Aufgaben, wie zum Beispiel das Buchen von Terminen schnell und effizient buchen könnte. Durch das Prototyp-Testing kann der Benutzer schauen, ob er die zu machende Aktionen in gewünschter Zeit ausführen könnte.

Für die Einhaltung von Benutzererwartungen ist es für das System wichtig, dass die zu erwartende Anforderung in Bezug auf zum Beispiel Terminbuchung und Terminverwaltung erfüllt werden sollten. Diese erreicht man dadurch, indem der Benutzer, während der Benutzung des Systems feststellt, ob der Prototyp die Erwartungen und Wünschen des Kunden erfüllt oder nicht (vgl. [16]).

## • 9.2. Wie ergänzen sich die UML-Diagramme? (Maeva Lekeufack)

Anwendungsfalldiagramm: Durch den Anwendungsfalldiagramm wurden die verschiedenen Benutzerinteraktionen und die funktionalen Anforderungen des Systems kenntlich gemacht. Pro Akteure (Unbekannt, Geschäftsführer und Kunden) des Systems wurden hierbei die jeweiligen zur Möglichkeit gestellten Funktionen des zu betreibenden Systems verdeutlicht zu jedem Akteur bereitgestellt (vgl. [4.1]).

Aktivitätsdiagramm: Als Ergänzung zum Anwendungsfalldiagramm wurden durch den Aktivitätsdiagramm zu jeder funktionalen Anforderung des Systems, der interne Ablauf der einzelnen Anwendungsfälle modelliert, einschließlich der einzelnen Schritte, Entscheidungen, sowie Bedingungen, die bei Ausführung eines Anwendungsfalls auftreten. Als Ergänzung des Anwendungsfalls, helfen Aktivitätsdiagramme, die Arbeitsabläufe, sowie die internen Prozesse innerhalb der jeweiligen zu machenden Anwendungsfalls zu visualisieren (vgl. [6]).

Klassendiagramm: Durch den Klassendiagramm werden die statischen Strukturen des zu programmierenden Softwaresystems, zudem die Klassen, Attribute und die im Zusammenhang stehenden Beziehungen, die als Schnittstelle der Klassen repräsentiert werden, verdeutlicht. Hierbei werden die Daten, die zur Benutzung des Softwaresystems benötigt und implementiert werden, sowie die Methoden und ebenfalls die dazugehörigen Verhaltensweisen der jeweiligen Klassen modelliert (vgl. [14]).

Sequenzdiagramm: Durch den Sequenzdiagramm werden zwischen den verschiedenen Objekten und Klassen des im Voraus gemachten Klassendiagramms, zu möglichen Interaktionen und ablaufenden Kommunikationen, für einen gezielt ausgewählten Szenario und/oder Anwendungsfall, verdeutlicht. Grob gesagt werden durch diese in einem zeitlichen Rahmen beobachtet, wie zwischen den zuteilwerdenden Elementen, Nachrichten und Methodenaufrufe verlaufen. Das Sequenzdiagramm zeigt seinen Stellenwert in der Softwareentwicklung am besten dadurch, indem es zu verstehen gibt, wie die Elemente, die Teil des Systems sind, während das Ausführen des Systems miteinander integrierbar ist, wodurch man mögliche Probleme zwischen Klassen und Objekten erkennen und beheben kann (vgl. [15]).

## • 9.3. Welchen Zweck hat die Software Anforderungsspezifikation? (Maeva Lekeufack)

Unter einer Software-Anforderungsspezifikation versteht man in der Softwareentwicklung ein Dokument, das dazu dient, einen Leitfaden dafür festzulegen, wie und mit welchen Methoden man die Anforderungen des Softwaresystems erhebt, versteht, sowie generell festhält. Im Verlaufe des Entwicklungsprozesses eines Softwaresystems, spielt die Software-Anforderungsspezifikation eine zentrale Rolle, da hierdurch für das Konzipieren, Programmieren, Implementieren, Testen, sowie Managen des Projekts eine Verlaufsgrundlage dargestellt wird, die wiederum das erfolgreiche Gelingen der Anforderungen des Softwaresystems massiv beeinflusst.

Die Software-Anforderungsspezifikation dient dem Hauptzweck, durch das Klarstellen der Anforderungen zwischen den im System beteiligten verschiedenen Akteuren, in einer klaren Struktur, eine verbindliche Darstellung zu schaffen.

Durch das Organisieren aller wertvollen Informationen, wird für ein im Entwicklungsteam, also aller Softwareentwickler, Softwaretester, sowie anderen Beteiligten des Systems, ein gemeinsames Verständnis gesorgt. Die Software-Anforderungsspezifikation kann als ein Mittel benutzt werden, um zwischen allen Beteiligten im System, für eine gemeinsame Kommunikation, einen Leitfaden zu herstellen und somit auf der einen Seite die Missverständnisse zu reduzieren, sowie auf der anderen Seite die zu erreichenden Anforderungen des Systems besser zu definieren.

Des Weiteren wird erreicht, dass indem die Anforderungen in ihre Details dargestellt werden, für alle Beteiligten eine auf eine gemeinsame Stufe bringende Grundlage der Interpretation der Softwareanforderungen dargestellt wird, um bei erhöhter Komplexität der Anforderungen, Fehlinterpretationen zu verhindern, wodurch im Verlaufe der Entwicklung des Softwaresystems das Risiko von Missverständnissen zu reduzieren, die sonst zu Fehlern, Verzögerungen, unnötige Kostenüberschreitungen, und etc. verursachen könnten.

Somit könnte man zusammenfassend wiedergeben, dass das Führen der Software-Anforderungsspezifikation im Verlaufe des Entwickelns eines Softwaresystems zwischen mehreren Teilnehmern des Softwaresystems, inklusive aller Softwareentwickler, sowie Softwaretester und etc., eine Schlüsselmethodik für das Koordinieren, Organisieren, sowie Managen des Softwaresystems darstellt.

- **10.Erläuterung, inwiefern der Prototyp die entsprechenden UML-Modelle realisiert, z.B.: Wo finden sich Klassen/Objekte im Prototypen? Welche Anwendungsfälle bzw. Aktivitätsdiagramme werden durch welche Funktionen und Interaktionen realisiert?**  
**(Florian Quaas)**

Zuerst hatten wir uns stark an die UML-Diagramme gehalten, beim Programmieren des Prototyps, aber wie es nun mal so ist haben wir während des Programmierens festgestellt, dass manches so zum einen vielleicht nicht umsetzbar ist oder auf eine andere Weise besser umsetzbar wäre. Zwar haben uns die Diagramme eine klare Struktur und einen roten Faden für den Prototyp gegeben aber beim Programmieren stellte man den schnell fest, dass man einige Sachen vernachlässigt hat oder noch einmal überarbeiten musste, sowie die Redundanz von einigen Features oder Aspekten der Diagramme.

Ein Beispiel hierfür ist das wir zuerst vorhatten Termine, Anfragen und Absagen alle in eine Datenbank zu packen und diese Anhand einer Zeile Status zu unterscheiden aber während der Entwicklung des Prototyps ist uns aufgefallen, dass dies eventuell nicht so schlau wäre, da sie auch unterschiedliche Informationen benötigen und klarer getrennt werden sollten. Aber wir haben uns versucht sehr stark an den Diagrammen entlang zu arbeiten. So stellt unser Klassendiagramm zum Beispiel unsere Datenbankstruktur da, wir haben eine Tabelle für Accounts und 3 Tabellen für die Termine, Absagen und Anfragen deren Spalten jeweils den Inhalten aus dem Klassendiagramm entsprechen.

Das Anwendungsfalldiagramm haben wir aufgrund der Überarbeitungen während der Arbeiten am Prototyp noch einmal komplett überarbeitet. Die Anwendungsfälle haben wir alle eingebaut, so kann der Geschäftsführer, solange er als Admin eingeloggt ist Termine direkt erstellen, falls er mit einem Kunden telefonisch etwas abklärt, indem er einfach auf Termin erstellen geht und die Eingaben alle einträgt.

Er kann sich auch alle Anfragen Termine/Anfragen einsehen indem er auf den richtigen Knopf in der navbar drückt und dann kann er dort bei Anfragen nach einem Termin auf bearbeiten/annehmen oder ablehnen klicken um die jeweiligen Anwendungsfälle auszuführen. Er kann auch Kunden einsehen und ihre Berechtigungen ändern also einen Account zum Admin machen oder auch entfernen. Der Kunde hingegen kann Anfragen erstellen, indem er sich anmeldet falls er registriert oder sich erst registriert und dann anmeldet und dann auf Anfrage erstellen klickt, er kann auch alle seine bisherigen Termine einsehen sowie seine Absagen, Anfragen und auch die Anfragen die der Geschäftsführer erstellen kann, indem er bei einer Anfrage auf Gegenangebot erstellen klickt. Und wenn man als Unbekannt auf die Seite kommt kann man sich erstmal nur registrieren oder anmelden und nicht mehr.

Da wir die Aktivitätsdiagramme zuerst erstellt haben mussten wir sie im Prozess der Programmierung öfters anpassen, da wir zum Beispiel erst nach relativ spät festgelegt haben, wie die Klassen und Objekte aussehen sollen, was an sich nicht so schlimm war, da damit nur sehr minimale Änderungen verbunden waren also so etwas wie die Datenbanken anpassen und das Klassendiagramm sowie die Objekte bzw. Startangaben in den Aktivitätsdiagrammen dies konnten wir aber an einem Tag am Anfang der Prototyperstellung beheben, da wir vor dem Programmieren noch einmal alles kurz theoretisch durchgegangen waren, eine weiterer Aspekt, der hier herein gespielt hatte war, das wir zu der Zeit, zu der wir die Aktivitätsdiagramme gemacht hatten noch nicht alle Techniken gelernt hatten und noch nicht alles gefestigt hatten.

Die meisten Anwendungsfälle werden als Interaktion des Users mit dem System und dem System mit der Datenbank realisiert, zum Beispiel werden Termine einfügen, ändern, bearbeiten, annehmen, ablehnen, Anfrage erstellen, Registrieren und Admin hinzufügen sowie Berechtigungen verwalten alle als so eine Interaktion realisiert werden, der Benutzer des Systems macht eingaben die dann von dem System überprüft und verarbeitet werden um dann in eine Datenbank hinzugefügt zu werden oder zu Änderungen eines bestimmten Eintrages anhand der KundenID oder der AuftragsID zu führen.

Ansonsten haben wir uns stark an den Diagrammen orientiert seien es Objekte aus dem Klassendiagramm wie Kunden oder Termine die als Einträge in eine Datenbank realisiert wurden oder die Aktivitätsdiagramme mit den Eingaben sowie den Ausgaben des Systems zu bestimmten Situationen oder auch die Resultate der Testfälle also wie sich zum Beispiel das Terminbuchen oder anmelden verhält je nachdem was für Bedingungen gegeben sind, zum Beispiel kann ein Termin nur angelegt werden, wenn alle Eingaben getätigt wurden und noch kein Termin mit denselben Daten gegeben ist während wir für die Bedingung weder Sonn- noch Feiertag keine Funktion programmiert haben, dies ist nur Teil der Modellierung und nicht Teil des Prototypen sowie das E-Mail schreiben, da dies auf hopper nicht möglich ist.

- **11.Qualitätssicherung: Welche Methoden (z.B. Reviews, Testfälle, Walkthrough-Tests) wurden mit welchen Ergebnissen verwendet? Welche Verbesserungen wurden erreicht?**

**(Reza Akbari)**

Qualitätssicherung: Welche Methoden (z.B. Reviews, Testfälle, Walkthrough-Tests) wurden mit welchen Ergebnissen verwendet? Welche Verbesserungen wurden erreicht? „Vorher – Nachher“-Darstellung mit Screenshots.

Im Rahmen unseres Softwareprojektes haben wir eine Webanwendung für ein

Umzugsunternehmen entwickelt. Um sicherzustellen, dass wir eine stabile, funktionale und benutzerfreundliche Anwendung erstellen, die den Nutzerbedürfnissen gerecht wird und das Vertrauen der Kunden stärkt, haben wir verschiedene Methoden der Qualitätssicherung eingesetzt. Diese Methoden umfassten Reviews, Testfälle und Walkthrough-Tests.

Verwendete Methoden: 1. Reviews: Nach der Erstellung unseres Prototyps haben wir im Team eine kritische Überprüfung durchgeführt. Wir haben individuell Notizen gemacht, in denen potenzielle Verbesserungen und Änderungen festgehalten wurden. Zudem führten wir ein Gespräch mit dem Kunden bzw. Geschäftsführer, um sicherzustellen, dass die Anforderungen korrekt verstanden wurden und unsere Lösungen ihren Bedürfnissen entsprechen.

2. Testfälle: Um Fehler zu vermeiden und die Funktionalität sicherzustellen, haben wir für jedes Szenario Testfälle definiert. Ein Beispiel für einen Testfall ist die Registrierung und Terminbuchung. Für die Registrierung haben wir fünf Testfälle erstellt, von denen einer positiv und die anderen vier negativ ausfallen. Diese negativen Testfälle demonstrieren, dass die Anforderungen nur unter bestimmten Bedingungen erfüllt werden können. Jeder

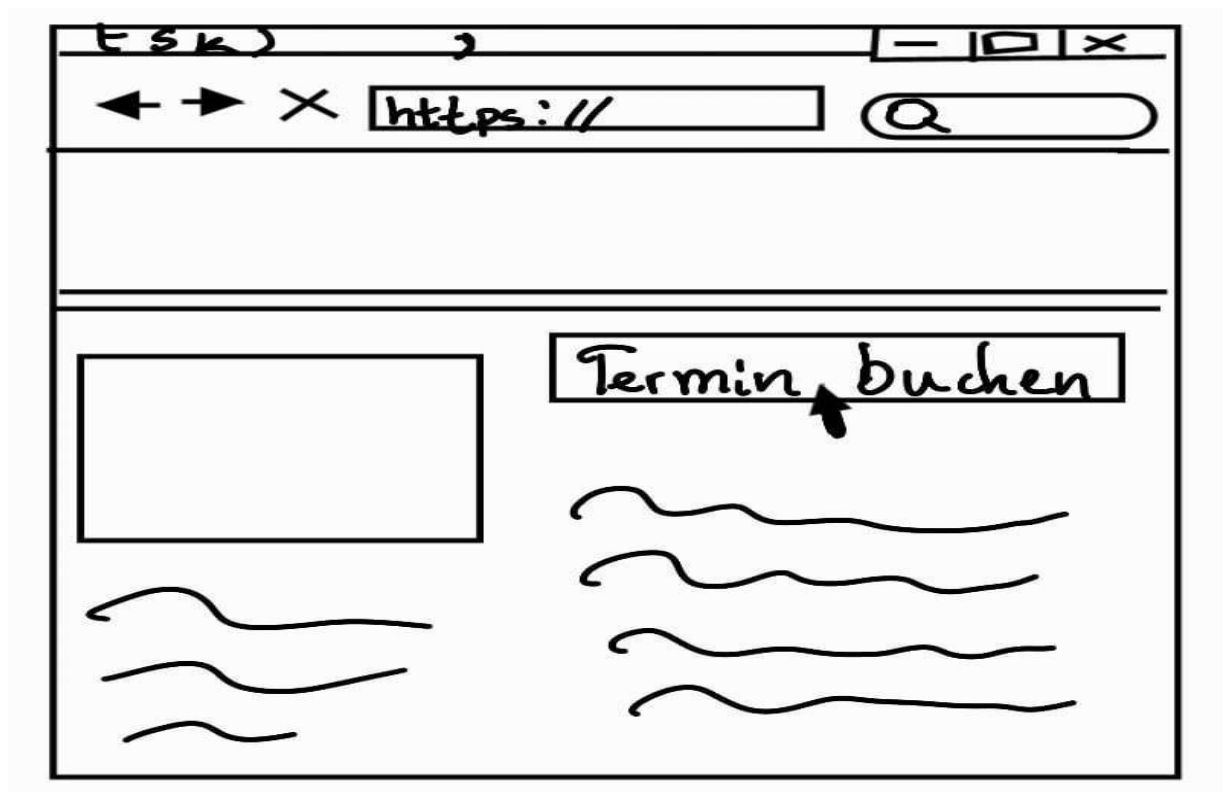
Testfall umfasst Vorbedingungen, erforderliche Daten wie Passwort, E-Mail-Bestätigung und Name, sowie die erwarteten Ergebnisse, ob die Registrierung Erfolgreich war oder Fehlgeschlagen ist.

ID	Vorbedingung	Passwort	E-Mail bestätigt	Name	Ergebnis/Nachbedingung
01	Noch nicht registriert	gültig	Ja	Neu	Erfolgreiche Registrierung
02	-	ungültig	-	Neu	Fehlgeschlagen
03	-	gültig	-	Vorhanden	Fehlgeschlagen
04	-	ungültig	-	Neu	Fehlgeschlagen
05	-	gültig	Nein	Neu	Fehlgeschlagen

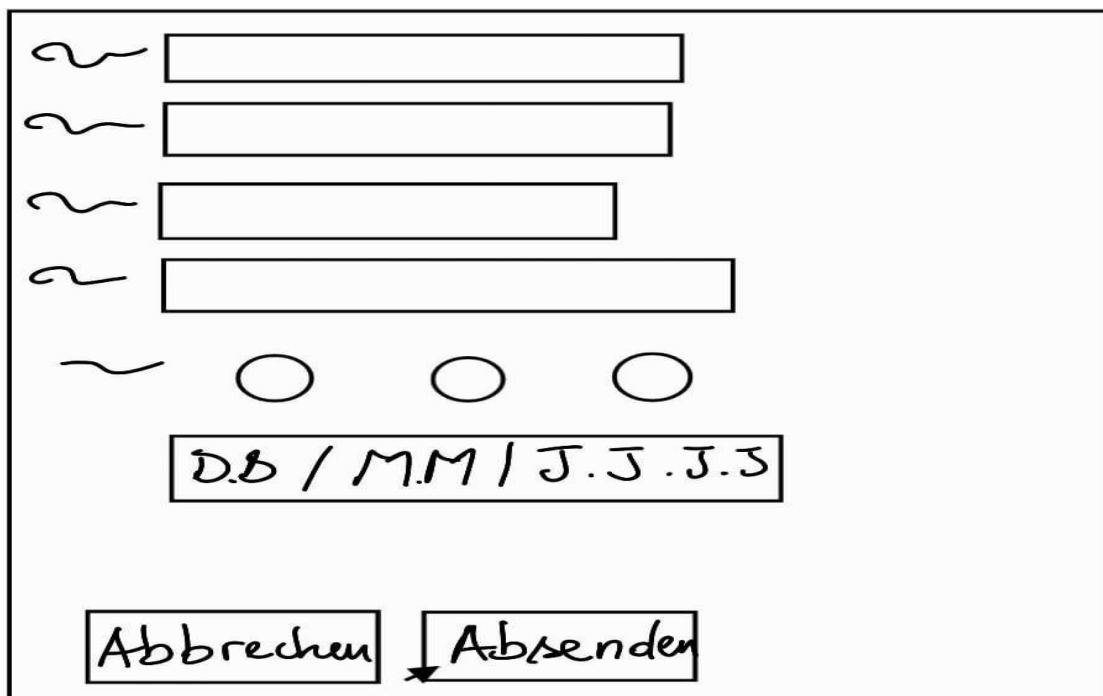
Für die Terminbuchung haben wir acht Testfälle erstellt. Dabei gab es vier Testfälle mit Fehlermeldungen, zwei Testfälle für die erfolgreiche Erstellung einer Anfrage und je einen Testfall für das Nicht-Anzeigen des Formulars sowie für den Fall, dass nichts passiert. Diese Testfälle beinhalteten zwei Vorbedingungen, ob die Person „Angemeldet“ oder „Nicht Angemeldet“ ist sowie wie das Fehlen eines vorhandenen Termins mit denselben Daten und notwendige Eingabedaten wie Datum ob etwas eingetragen ist oder Leer, Das gleiche für Zielort, Abholort und Notizen.

ID	Vorbedingung	Noch kein Termin mit denselben Daten	Datum	Zielort	Abholort	Notiz	„Termin buchen“ angeklickt	Ergebnis
01	Nicht Angemeldet	-	-	-	-	-	-	Formular wird nicht angezeigt
02	Angemeldet	Ja	eingetragen	eingetragen	eingetragen	eingetragen	Ja	Anfrage wird erstellt
03	Angemeldet	-	-	-	-	-	Nein	Nichts passiert
04	Angemeldet	Ja	eingetragen	eingetragen	eingetragen	leer	Ja	Anfrage wird erstellt
05	Angemeldet	Ja	-	-	leer	-	ja	Fehlermeldung
06	Angemeldet	Ja	-	leer	-	-	Ja	Fehlermeldung
07	Angemeldet	Ja	leer	-	-	-	Ja	Fehlermeldung
08	Angemeldet	Nein	eingetragen	eingetragen	eingetragen	eingetragen	Ja	Fehlermeldung

Terminbuchung und -verwaltung:

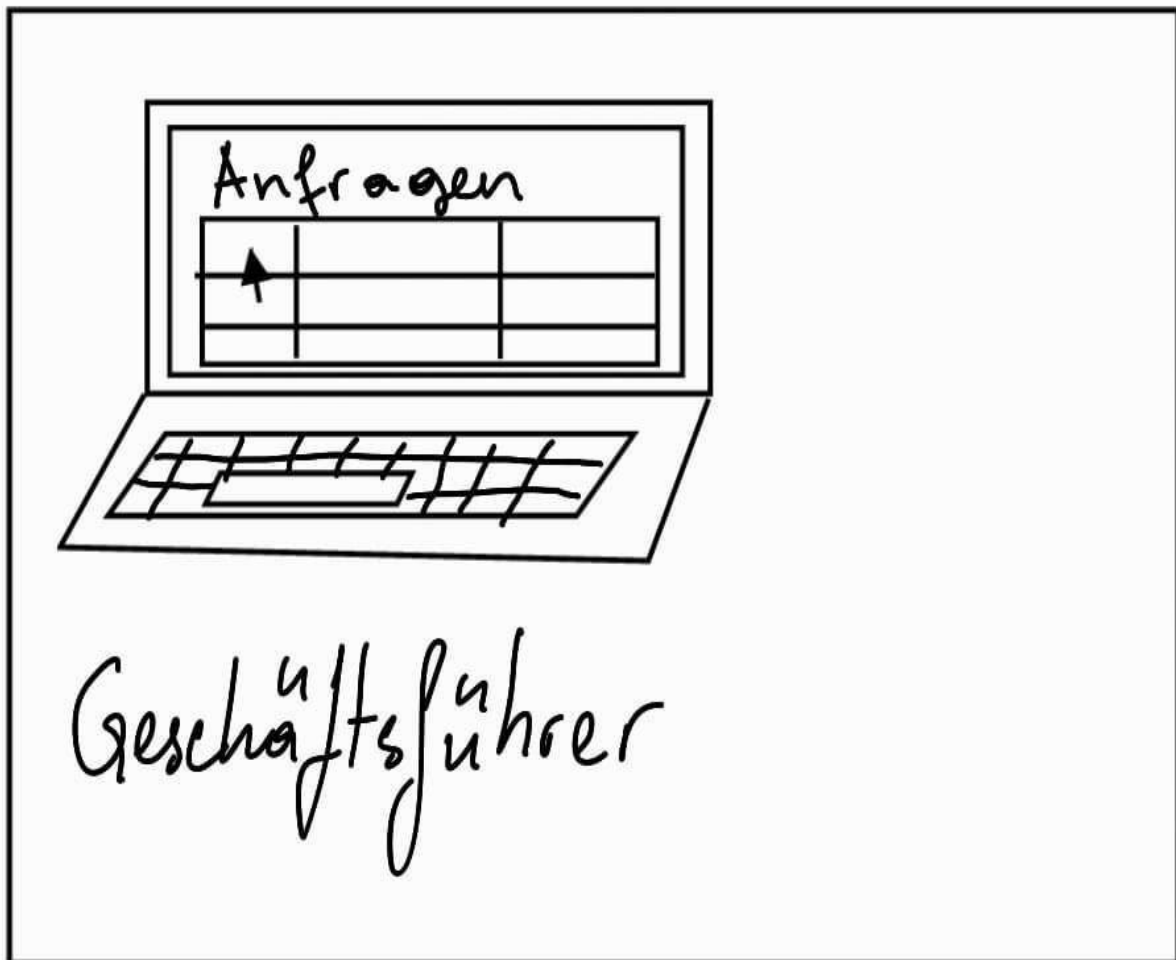


Wir haben Storyboards erstellt, die den Prozess der Terminbuchung und die darauffolgende Verwaltung illustrieren.



Dies beinhaltet Schritte wie das Buchen eines Termins, das Abbrechen eines Termins und das Absenden dieser Termine. Diese Storyboards helfen uns, den Ablauf und die

Benutzerinteraktion klar zu verstehen und mögliche Schwachstellen frühzeitig zu erkennen Anfragen für Geschäftsführer:



Ein weiteres Storyboard zeigt den Prozess der Anfragen an den Geschäftsführer. Hierbei wurden Schritte wie die Übersicht über Anfragen, die Zuweisung von Zeiten, das Bestätigen und Ablehnen von Anfragen sowie das Zuweisen alternativer Zeiten visualisiert.



Name: \_\_\_\_\_  
 Tage: DD.MM.YYYY  
 vorgeschlagene Uhrzeit: \_\_\_\_\_  
 Adresse: \_\_\_\_\_  
 E-Mail: \_\_\_\_\_  
 Art des Termins: \_\_\_\_\_  
 Notizen: \_\_\_\_\_  
 \_\_\_\_\_

Dieser Prozess ermöglicht es dem Geschäftsführer, flexibel auf die Anfragen der Kunden zu reagieren.

Kalender mit Tagen und Terminen:

Termine Tage X-Y

K.L	~	~	~	~	~
~	~	~	~	~	~
~	~	~	~	~	~
~	~	~	~	~	~
~	~	~	~	~	~
~	~	~	~	~	~

Wir haben eine Skizze für den Kalender erstellt, um die Darstellung von Datum und

Termin details zu veranschaulichen. Dies hätte uns geholfen, die Benutzeroberfläche des Kalenders zu gestalten und sicherzustellen, dass wichtige Informationen klar und verständlich präsentiert werden.

Verwaltung von angefragten Terminen für Kunden:

K.L.O

## Angefragte Termine

Name : \_\_\_\_\_

Tag : DD.MM.YYYY

Uhrzeit : — — —

Adresse : \_\_\_\_\_

E-Mail : \_\_\_\_\_


bestätigenbearbeitenAblehnen  
↑

Ein weiteres Storyboard behandelt die Verwaltung von angefragten Terminen aus Sicht der Kunden. Hierbei haben wir die Schritte skizziert, um Termine zu bestätigen, zu bearbeiten oder abzulehnen. Zudem haben wir eine Möglichkeit zur Eingabe von Ablehnungsgründen hinzugefügt, um den Prozess transparenter zu gestalten.

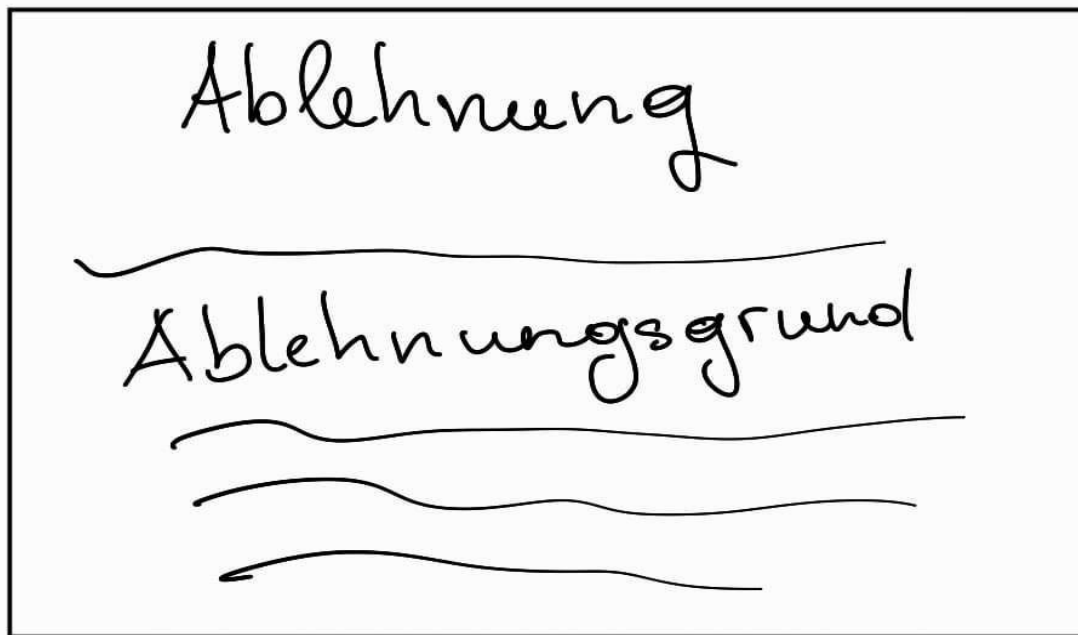
Stornierung und Ablehnung von Terminen:

K.L

<input checked="" type="checkbox"/>			
<input type="checkbox"/>			
<input type="checkbox"/>			
<input type="checkbox"/>			

stornieren 

Wir haben auch zwei Skizzen erstellt, die den Prozess der Stornierung oder Ablehnung von Terminen darstellen.



Hierbei haben wir besonderes Augenmerk auf die Möglichkeit gelegt, einen Grund für die Ablehnung anzugeben. Im Laufe der Entwicklung stellten wir fest, dass einige dieser Funktionen nach Kunden- und Teamgesprächen sowie unserem Walkthrough-Test überflüssig waren und haben entsprechende Anpassungen vorgenommen.

#### **Walkthrough-Tests:**

Nach Treffen mit verschiedenen Teammitgliedern haben wir Walkthrough-Tests durchgeführt, um die Benutzererfahrung zu verbessern. Dabei haben wir folgende Verbesserungen vorgenommen:

Aktive Seiten wurden farblich hervorgehoben, um die Navigation zu erleichtern.

Alle englischen Texte wurden ins Deutsche übersetzt, um die Benutzerfreundlichkeit für deutschsprachige Nutzer zu erhöhen.

Begriffe wie "Datum" wurden angepasst, um klarer zu kommunizieren, worum es geht (z.B. "Gewünschter Termin" oder "Datum des Umzugs").

Überflüssige Informationen wie die Buchungsdauer wurden entfernt, um die Benutzeroberfläche übersichtlicher zu gestalten.

Des Weiteren haben wir die Anmeldungs- und Terminbuchungsprozesse optimiert. Nach einer erfolgreichen Registrierung oder Anmeldung wird nun eine

Erfolgsmeldung angezeigt, um dem Nutzer Rückmeldung zu geben. Ebenso wurde darauf geachtet, Eingabefelder wie "Zielort" und "Abholort" klarer zu kennzeichnen.

## Termin Buchen Vorher:

[Home](#) [Kontakt](#) [About](#) [Services](#) [Willkommen reza.akbaril12@gmail.com](#) [Termin buchen](#) [Meine Termine](#) [Logout](#)

Name:

Datum:

tt.mm.jjjj

Uhrzeit:

\_\_:\_\_

Art:

Umzug

Abholort:

Zielort:

Buchungsdauer:

Sonstige Notizen:

Abschicken

Mithilfe von vorher-nachher Screenshots unserer Prototypen haben wir die Seite für das Terminbuchungserlebnis erheblich verbessert, indem wir überflüssige und unnötige Informationen entfernt haben.

Im Vergleich zu unseren früheren Entwürfen ist die überarbeitete Seite nun auf das Wesentliche fokussiert.

Diese Informationen sind bewusst reduziert worden, um den Geschäftsführer in die Lage zu versetzen, diese Details direkt mit dem Kunden abzustimmen und sie gemäß seinen individuellen Anforderungen zu gestalten. Dies ermöglicht eine maßgeschneiderte Buchungserfahrung, ohne die Komplexität für den Kunden zu erhöhen.

## Termin buchen Nachher:

[Home](#) [Kontakt](#) [About](#) [Willkommen reza.akbaril12@gmail.com](#) [Termin buchen](#) [Meine Termine](#) [Logout](#)

Buchen sie jetzt einen unverbindlichen Termin kostenlos, der Preis und die Uhrzeit des Termines werden von dem Geschäftsführer kalkuliert und ihnen dann mitgeteilt, dann können sie das Angebot annehmen oder ablehnen.

Datum:

Abholort:

Zielort:

Sonstige Notizen:

Diese Spalte „Services“ wurde in unserem vorherigen Prototyp als Möglichkeit für den Geschäftsführer vorgesehen, verschiedene Dienstleistungen oder Leistungspakete darzustellen. Jedoch stellten wir fest, dass diese Informationen in Bezug auf die Benutzererfahrung und den spezifischen Bedarf des Geschäftsführers nicht essentiell waren. Sie führten eher zu visueller Überlastung und könnten das Nutzererlebnis komplizierter gestalten, ohne einen klaren Mehrwert zu bieten.

Die Qualitätssicherung durch Reviews, Testfälle und Walkthrough-Tests hat erheblich zur

Verbesserung unserer Webanwendung für das Umzugsunternehmen beigetragen. Die Identifikation von Fehlern, die Anpassung an die Bedürfnisse der Nutzer und die Schaffung

einer besseren Benutzererfahrung sind zentrale Ergebnisse dieser Qualitätssicherungsmaßnahmen. Die Anwendung ist stabiler, funktionaler und benutzerfreundlicher geworden, was letztendlich das Vertrauen der Kunden stärkt und dem Umzugsunternehmen einen Wettbewerbsvorteil verschafft.

Die umfassende Qualitätssicherung hat sich als unverzichtbarer Bestandteil unseres Projektes erwiesen und hat maßgeblich zur erfolgreichen Entwicklung einer hochwertigen Webanwendung beigetragen.

## • 12. Reflexion und Fazit (Florian Quaas)

Unsere Aufgabe während des Projektes war es ein Software Projekt in kleinem Maßstab durchzuführen und uns mit den Techniken und Methoden, die in einem solchen Projekt genutzt werden bekannt zu machen. So war es zum Beispiel eine unserer Aufgabe das Projekt über OpenProjekt zu managen und unsere Erkenntnisse daraus zu reflektieren.

Bei unserer Reflektion kamen wir zu dem Entschluss, dass zwar Projektmanagement an sich sehr wichtig ist, gerade zum Festhalten von Fortschritt, Aufgaben und Zielen, man erhält dadurch erst einen roten Faden und Struktur, wir sind zu dem Entschluss gekommen, dass Projekte ohne solche Planung und Projektmanagement komplett unmöglich sind, man hätte keine Struktur und könnte nicht effektiv arbeiten und Ziele erreichen. Gleichzeitig haben wir aber auch festgestellt, das OpenProjekt und seine vielzähligen Funktionen und Möglichkeiten für ein erstes Software Projekt in so einem kleinen Umfang etwas viel war und nicht ganz ausgeschöpft und wirklich gewinnbringend genutzt werden konnte, gerade da wir versucht haben uns stark an die zeitliche Vorgabe des Moduls für die unterschiedlichen Phasen zu halten.

Gleichzeitig jedoch haben wir in Hinblick auf die Zukunft und in Reflexion auf die Funktionen an sich sowie die benötigten Funktionen bei einem Software Projekt von Grund auf festgestellt, dass OpenProjekt alles zu bieten hatte, was wir uns in einem Projekt als wichtig vorstellen konnten, die Möglichkeit Meetings festzuhalten und zu dokumentieren, Aufgaben zu erstellen, zuzuweisen und den Zeitaufwand einzuschätzen, gerade wenn man mit mehreren Abteilungen oder Ebenen an einem Projekt arbeitet, die nicht alle an einem Ort sind ist es sehr wichtig dieses Zentral organisieren zu können.

Auch wenn es in diesem Fall von uns eher nicht vollständig zu den Grenzen seiner Möglichkeiten genutzt werden konnte, da wir meistens Protokolle vor Ort verfasst haben und diese dann in unserer Whatsapp Gruppe festgehalten haben. Für uns war OpenProjekt aber sehr nützlich in dem Projekt und zwar Zeitmanagement, so konnte man Personen Aufgaben zuteilen vor Ort und zu Hause oder danach wusste man genau was man tun sollte und bis wann. In dem Sinne war OpenProjekt vor allem in der späteren Projektphase für uns sehr wichtig und wurde deutlich mehr genutzt als zuvor.

Ein weiterer Punkt den wir in der Auseinandersetzung mit Projektmanagement als unerlässlich erwiesen hat war die Modellierung mithilfe der verschiedenen Diagramme sowie den Storyboards, Personas und User Stories, da man erst dadurch ein klares Bild von dem Ziel schafft und so erst in der Lage ist strukturiert das Projekt zu planen und die noch zu erledigenden Aufgaben zu dokumentieren. Wenn wir einfach am Anfang angefangen hätten zu programmieren, da sind wir uns einig hätte das Projekt zum einen deutlich länger gedauert und man hätte viel mehr Arbeit gehabt, da man sehr viele Teile dann wegwerfen müsste, da sie entweder nicht zu dem System passen würden oder einfach nicht den Anforderungen entsprächen, des Weiteren wäre das System auch deutlich schlechter geworden, da man kein klares Bild oder Ziel vor Augen hätte, jeder hätte irgendetwas gemacht aber man hätte als Team nicht effektiv zusammen gearbeitet, da jeder eine andere Vision oder Idee von dem System hätte.

Auch wenn selbst beim modellieren oft Diagramme oder Artefakte uns erstmal nur eine ungefähre Richtung geben haben war das genug um eine gemeinsame Idee von einem System zu schaffen, das aufgrund der vorher entstandenen Artefakte mit jedem weiteren weiter verfeinert wurde und somit auch die Artefakte angepasst wurden, so haben wir zum Beispiel anhand der des Storyboards die Möglichkeit gehabt und genutzt noch einmal unser Anwendungsfalldiagramm zu überdenken und anhand des Storyboards noch einmal zu überprüfen und zu überarbeiten die Diagramme helfen die anderen Artefakte noch einmal gegen zu checken und weiterzuentwickeln.

Zum Beispiel fiel einem bei dem Storyboard auf, dass manche Anwendungsfälle keinen Sinn machen und andere relevante zum Beispiel fehlen. Die von uns verwendeten Erhebungsmethoden in unserem Fall Interviews, User Stories, Personas Brainstorming, Mindmapping und Recherche erachten wir als sehr sinnvoll, um die Spezifikationen eines Systems und die Anforderungen die ein System erfüllen muss zu erheben. Während User Stories, Interviews, Mindmapping, Brainstorming und Recherche generell einem helfen Anwendungsfälle sowie Funktionen und Akteure in dem System herauszuarbeiten erlauben Personas eine Zielgruppe für sein System zu analysieren und anhand dessen dem System erst wirklichen Wert zu geben, ein System hat nur dann einen Wert, wenn es mit Fokus auf den Nutzer programmiert und erhoben wurde, das System muss intuitiv sein und es darf keine Qual sein das System zu nutzen.

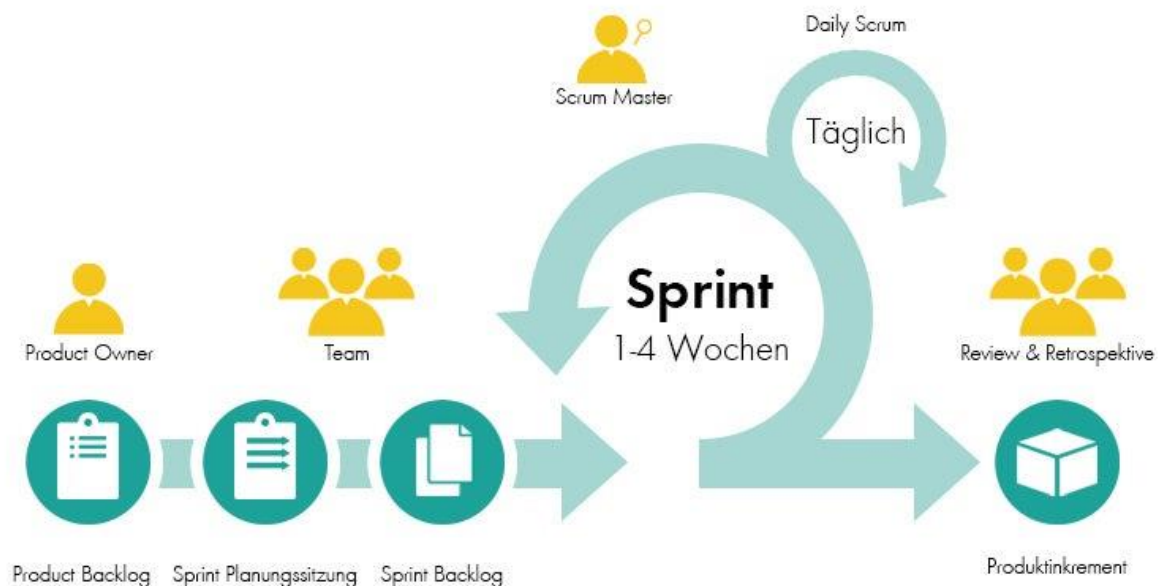
Systeme sind allgemein dazu da einen Prozess zu vereinfachen und wenn es unmöglich ist für die Zielgruppe des Systems ist dieses effektiv zu nutzen ist das Problem das System und es verliert jeden Wert, egal wie gut oder kompliziert es programmiert wurde. In diesem Kontext war die Spec sehr wichtig, sie sammelt alle Anforderungen des Systems seien sie funktional oder nicht funktional gerade um sie einem Kunden früh vorzuzeigen und als Grundbaustein der Zusammenarbeit ist diese sehr wichtig, sie erlaubt es einem früh auf einen gemeinsamen Nenner zu kommen in der Zusammenarbeit sowie das Verständnis beider Parteien abzugleichen, ob sie dasselbe Ziel haben oder nicht, es sei erwähnt das ein System, dass komplett an den Wünschen und Anforderung des Kunden vorbei entwickelt wurde ohne Absprache oder Abgleich in Form der Spec wertlos ist.

Man sollte möglichst früh anfangen Artefakt zu Verifizieren und zu Validieren so kann man viel Geld und Zeit sparen, Fehler oder Missverständnisse in der Planungsphase können relativ einfach behoben werden während diese später zu beheben mit viel Aufwand und vielen Kosten einhergehen, daher sollte man möglichst früh Qualitätssicherung betreiben sowohl mit Artefakten als auch mit Prototypen. Auch wenn der Prototyp sehr wichtig ist gibt er nur Auskunft über das Aussehen und die Vorgänge in dem System, während das Storyboard früh eine Auskunft über die Abläufe und Hauptfunktionen des Systems gewährt, während der Prototyp schon viel Arbeit und Geld gekostet hat in Software Projekten, kann man über Storyboards frühzeitig mit dem Kunden die gewünschten Aufgaben und Abläufe des Systems sowie das grobe Aussehen dieser festigen ohne viel Arbeit in Anspruch zu nehmen.

Für das Projekt an sich fanden wir dies Vorgegebenen Meilensteine und Phasen als sehr hilfreich und sinnvoll gerade am Anfang, am Ende des Projektes haben wir aber noch einmal über den Ablauf reflektiert und wir hätten deutlich früher mit der Qualitätssicherung angefangen, so hätte man anhand der Diagramme schon einmal die Qualität der eigenen Idee und des Verständnisses des Zieles des Systems überprüfen können, zum Beispiel hätte man die Diagramme schon früher dem Kunden zeigen können und vorstellen können, wodurch man frühzeitig eventuelle Fehler und Missverständnisse aus dem Weg räumen könnte des Weiteren würde man so auch dem Kunden die Möglichkeit geben sich von dem Projekt zu vergewissern und eventuell eigne Ideen außerhalb der Anforderungserhebung noch einzubringen. In diesem Fall war es relativ egal, aufgrund der niedrigen Komplexität des Systems und des Umfangs des Projektes gerade da es nur eine Simulation eines Software Projektes ging, jedoch bei größeren Projekten bei denen wirklich viele wirtschaftliche Mittel auf dem Spiel stehen sowie der Ruf des Betriebes könnten wir uns vorstellen, dass man versucht möglichst früh mit der Qualitätssicherung anzufangen.

Unser Vorgehen in dem Projekt haben wir an Scrum angelehnt, wir hatten zuerst die Anforderungserhebung durch Interviews, Recherche, Mindmapping etc.(vgl. [8]) erledigt und haben uns dann den Artefakten Sprintmäßig gewidmet, wir haben Aufgabe nach Aufgabe so gemacht und haben die Aufgaben die durch die Wochenaufgabe uns vorgegeben wurde für den jeweiligen 1-Woche Sprint erledigt und haben bis zum Ende des Projektes vorgearbeitet und für diesen Bericht und diesen Abschnitt über den Sprint reflektiert, also haben wir uns erst am Ende dem Auswerten des Prozesses

gewidmet und haben uns am Ende auch noch einmal alle bisherigen Ergebnisse/Artefakte angeguckt und noch einmal durchgearbeitet und bewertet was auch nicht ganz Scrum entspricht. Daher würden



wir sagen, dass wir Scrum ähnlich gearbeitet haben aber manchmal von dem Prinzip abgewichen sind.  
(Quelle Bild vergleiche [9])

Aber wir haben wie bei Scrum zuerst die Anforderungen erhoben, dann einige Elemente aus dem Backlog bearbeitet, Teil des Backlogs waren alle Wochenaufgaben und Artefakte, die es zu erstellen und zu erledigen galt, dann haben wir jede Woche aufgaben aus dem Backlog entsprechend der Wochenaufgabe her rausgenommen und bearbeitet, aber noch nicht bewertet und analysiert, da wie gerade am Anfang wussten, dass sich die Teilergebnisse noch stark verändern werden würden und wir am Ende noch einmal vieles überarbeiten müssen, da es unser erstes Software Projekt war und wir die Techniken erst über das Semester gelernt haben, war uns bewusst, dass diese nicht astrein seien würden und wir diese deshalb ganz am Ende noch einmal verändern müssten. Bis dann das Projekt fertig war und wir alles am Ende noch einmal analysiert und bewertet haben.

## • 13 Integration der Diskussionsergebnisse zu „Wissenschaftlichen Artikeln“ mit Bezug zum SWE2-Projekt (Maeva Lekeufack)

Im Kontext eines Terminbuchungs- und Verwaltungssystems für ein Umzugsunternehmen können unser selbst gewähltes SWE-Thema "User-Centered Design" und Konzepte auf verschiedene Weisen relevant sein:

Videos als gestalterisches Medium im User-Centered Design:

Die Integration von Videos als gestalterisches Medium im User-Centered Design eines Terminbuchungs- und Verwaltungssystems für ein Umzugsunternehmen könnte eine transformative Wirkung auf die Interaktion zwischen Benutzern und dem System haben. Kurze Videos könnten nicht



nur dazu verwendet werden, den Umzugsprozess visuell darzustellen, sondern auch als kreative Werkzeuge dienen, um den Buchungsprozess für Kunden und Mitarbeiter gleichermaßen zu erleichtern. Diese Videos könnten in verschiedene Phasen des Umzugsprozesses unterteilt werden, angefangen bei der Vorbereitung und Verpackung bis hin zur Lieferung und Auspacken am neuen Ort. Solche Videos könnten nicht nur als informative Anleitungen dienen, sondern auch als kreative Inspirationsquelle für Kunden, um individuelle Anforderungen und Wünsche besser zu kommunizieren. Darüber hinaus könnten Videos genutzt werden, um komplexe Verfahren zu erklären, wie beispielsweise die sichere Handhabung von empfindlichen Gegenständen oder die Organisation von internationalen Umzügen. Die Integration von Videos in die Buchungsanwendung könnte eine umfassende Unterstützung bieten, die über traditionelle textbasierte Anleitungen hinausgeht.

#### Allgegenwärtige Mensch-Computer-Interaktion:

Die Bedeutung der Benutzerfreundlichkeit und Usability im Kontext des Terminbuchungs- und Verwaltungssystems eines Umzugsunternehmens kann nicht genug betont werden. Eine benutzerfreundliche Oberfläche und eine intuitive Navigation könnten dazu beitragen, dass sowohl Kunden als auch Mitarbeiter das System effektiv nutzen können. Um sicherzustellen, dass die Interaktion nahtlos und reibungslos verläuft, könnte die Buchungsanwendung für verschiedene Plattformen und Geräte optimiert werden, einschließlich Webbrowsern und mobilen Apps. Ein systematischer Ansatz zur Usability könnte die Eingabe von Kundendaten und Umzugsdetails vereinfachen und eine klare Übersicht über verfügbare Termine bieten. Darüber hinaus könnten personalisierte Benutzerprofile erstellt werden, um den Buchungsprozess zu beschleunigen und wiederholte Eingaben zu minimieren. Die Möglichkeit, individuelle Präferenzen und Spezifikationen zu speichern, könnte den Kundenservice verbessern und die Zufriedenheit der Kunden steigern.

#### User Experience Design und Organisationsentwicklung:

Bei der Entwicklung des Terminbuchungs- und Verwaltungssystems für das Umzugsunternehmen könnte ein tief verwurzelter Ansatz des User Experience (UX) Designs angewendet werden. Durch die Einbindung der Nutzer von Anfang an könnten deren Bedürfnisse, Anforderungen und Feedback kontinuierlich berücksichtigt werden. Dies könnte in Form von Fokusgruppen, Umfragen und Usability-Tests geschehen. Die gewonnenen Erkenntnisse könnten in die Gestaltung der Buchungsanwendung einfließen, um eine maßgeschneiderte Lösung zu schaffen, die den Kundenanforderungen optimal gerecht wird. Zusätzlich könnte die Berücksichtigung von Innovation und Kreativität bei der Gestaltung der Benutzeroberfläche zu einem einzigartigen und ansprechenden Buchungserlebnis führen. Die Integration von Gamification-Elementen, wie etwa Belohnungen für wiederholte Buchungen oder das Erreichen von Meilensteinen, könnte die Benutzerbindung stärken und die Anwendung attraktiver gestalten. Eine menschenzentrierte Organisationsentwicklung könnte sich in einer verstärkten Kundenorientierung des Unternehmens widerspiegeln, was letztendlich zu einer nachhaltigen Steigerung der Kundenzufriedenheit und -bindung führen könnte.

#### Scenarios in User-Centered Design:

Die Anwendung von Szenarien im Designprozess des Terminbuchungs- und Verwaltungssystems könnte dazu beitragen, potenzielle Herausforderungen frühzeitig zu identifizieren und geeignete Lösungen zu entwickeln. Ein Szenario könnte die Situation simulieren, in der ein Kunde kurzfristig seinen Umzugstermin ändern muss. Dies könnte dazu dienen, die Flexibilität und Anpassungsfähigkeit des Buchungsprozesses zu testen. Ein weiteres Szenario könnte die Handhabung von Sonderanfragen wie beispielsweise sperrigen oder besonders empfindlichen Gegenständen behandeln. Durch solche Szenarien könnten die Designentscheidungen kontextualisiert und auf die tatsächlichen Bedürfnisse der Benutzer zugeschnitten werden. Dies könnte auch dazu beitragen, mögliche Engpässe oder Engstellen im Prozess zu erkennen und zu beseitigen, um einen reibungslosen Ablauf sicherzustellen.

Softwarequalität aus Nutzersicht und ihre wirtschaftliche Bewertung:

Die Betonung der Softwarequalität aus Nutzersicht im Kontext des Terminbuchungs- und Verwaltungssystems könnte erhebliche wirtschaftliche Auswirkungen haben. Fehlerhafte Software und ineffiziente Buchungsprozesse könnten zu Kundenunzufriedenheit, verpassten Geschäftschancen und zusätzlichen Kosten führen. Durch eine umfassende Qualitätssicherung, einschließlich automatisierter Tests und manueller Prüfungen, könnten mögliche Fehlerquellen minimiert und die Zuverlässigkeit des Systems erhöht werden. Die kontinuierliche Überwachung der Softwareleistung und -stabilität könnte dazu beitragen, potenzielle Probleme frühzeitig zu erkennen und zu beheben, bevor sie zu größeren Störungen führen. Die wirtschaftliche Bewertung der Softwarequalität könnte die Einsparungen durch eine reibungslose Buchung und Durchführung von Umzügen quantifizieren. Dies könnte die Grundlage für Investitionen in die Softwareentwicklung bilden und langfristige Wettbewerbsvorteile sichern.

Usability Testing: Affektive Interfaces und Eyetracking-Technologie könnten in Usability-Tests verwendet werden, um das Verhalten der Nutzer bei der Nutzung der Buchungsanwendung zu analysieren. Dies könnte Aufschluss über die Bereiche geben, die für die Nutzer besonders ansprechend sind, sowie über potenzielle Schwierigkeiten oder Unklarheiten. Basierend auf diesen Erkenntnissen könnten gezielte Verbesserungen an der Benutzeroberfläche vorgenommen werden.

Insgesamt könnten diese detaillierten Anwendungen der Konzepte dazu beitragen, ein maßgeschneidertes Terminbuchungs- und Verwaltungssystem für das Umzugsunternehmen zu entwickeln. Dieses System könnte nicht nur den Ablauf des Unternehmens verbessern, sondern auch zu einer höheren Zufriedenheit bei Kunden und Mitarbeitern führen.

Allgemein haben uns die Kenntnisse über User Centered Design geholfen, unsere Schwerpunkte für das System zu finden und auch effektiv an dem Projekt zu arbeiten, dadurch hatten wir immer einen weiteren Punkt das System und unsere Ideen zu bewerten, inwiefern macht es Spaß das System zu bedienen und ist es intuitiv, wenn nicht wie können wir es weiter entwickeln und verbessern, damit es denn Benutzern als positive Erfahrung und nicht als Grauen in Erinnerung bleibt.

## • 14.Referenzen, Literatur- und Quellenverzeichnis

[1] - UTF-8 (Abkürzung für 8-Bit UCS Transformation Format, wobei UCS wiederum Universal Coded Character Set abkürzt) ist die am weitesten verbreitete Kodierung für Unicode-Zeichen (Unicode und UCS sind praktisch identisch). Die Kodierung wurde im September 1992 von Ken Thompson und Rob Pike bei Arbeiten am Plan-9-Betriebssystem festgelegt.

[2] - SQL-Injection ist das Ausnutzen einer Sicherheitslücke in Zusammenhang mit SQL-Datenbanken. Die Sicherheitslücke entsteht durch einen Programmierfehler in einem Programm, das auf die Datenbank zugreift. Durch diesen Programmierfehler kann ein Angreifer Datenbankbefehle einschleusen und abhängig vom Einzelfall weitere Daten aus der Datenbank auslesen, Daten unberechtigt ändern oder löschen oder sogar die Kontrolle über den kompletten Datenbankserver übernehmen.

[2]Vorlesung Anforderungen erheben – Projektidee entwickeln

[https://elli.hs-bremerhaven.de/goto.php?target=file\\_309324\\_download&client\\_id=elli](https://elli.hs-bremerhaven.de/goto.php?target=file_309324_download&client_id=elli)

Zugriff am 18.08.2023 21:38

[3]Vorlesung Interviewmethodik,

[https://elli.hs-bremerhaven.de/goto.php?target=file\\_308662\\_download&client\\_id=elli](https://elli.hs-bremerhaven.de/goto.php?target=file_308662_download&client_id=elli)

Zugriff am 15.08.2023 12:18 Uhr

[3.1]Vorlesung Benutzungsschichten – User Stories

[https://elli.hs-bremerhaven.de/goto.php?target=file\\_308664\\_download&client\\_id=elli](https://elli.hs-bremerhaven.de/goto.php?target=file_308664_download&client_id=elli)

Zugriff am 18.08.2023 21:51 Uhr

[3.2]Vorlesung Personas

[https://elli.hs-bremerhaven.de/goto.php?target=file\\_308663\\_download&client\\_id=elli](https://elli.hs-bremerhaven.de/goto.php?target=file_308663_download&client_id=elli)

Zugriff am 18.08.2023 21:56 Uhr

[4]Vorlesung Testfälle,

[https://elli.hs-bremerhaven.de/goto.php?target=file\\_313373\\_download&client\\_id=elli](https://elli.hs-bremerhaven.de/goto.php?target=file_313373_download&client_id=elli)

Zugriff am 14.08.2023 17:00 Uhr

[4.1]Vorlesung Sollkonzept: Vom Bedarf zu den Features Funktionale Anforderungen als Use Cases modellieren

[https://elli.hs-bremerhaven.de/goto.php?target=file\\_311130\\_download&client\\_id=elli](https://elli.hs-bremerhaven.de/goto.php?target=file_311130_download&client_id=elli)

Zugriff am 18.08.2023 21:57 Uhr

[5]Vorlesung Storyboarding,

[https://elli.hs-bremerhaven.de/goto.php?target=file\\_312792\\_download&client\\_id=elli](https://elli.hs-bremerhaven.de/goto.php?target=file_312792_download&client_id=elli)

Zugriff am 10.08.2023 19:30 Uhr

[6]Vorlesung Aktivitätsdiagramme,

[https://elli.hs-bremerhaven.de/goto.php?target=file\\_313230\\_download&client\\_id=elli](https://elli.hs-bremerhaven.de/goto.php?target=file_313230_download&client_id=elli)

Zugriff am 25.07.2023 15:00 Uhr

[7]Vorlesung Anforderungsanalyse,

[https://elli.hs-bremerhaven.de/goto.php?target=file\\_309935\\_download&client\\_id=elli](https://elli.hs-bremerhaven.de/goto.php?target=file_309935_download&client_id=elli)

Zugriff am 13.07.2023 17:00 Uhr

[8]Vorlesung Anforderungserhebung,

[https://elli.hs-bremerhaven.de/goto.php?target=file\\_309324\\_download&client\\_id=elli](https://elli.hs-bremerhaven.de/goto.php?target=file_309324_download&client_id=elli)

Zugriff am 10.07.2023 17:00 Uhr

[9] SCRUM, Bild und Erklärung;

[https://omr.com/de/daily/glossary/scrum/;](https://omr.com/de/daily/glossary/scrum/)

Zugriff am 17.08.2023 18:44 Uhr

[10]Vorlesung Personas

[https://elli.hs-bremerhaven.de/goto.php?target=file\\_308663\\_download&client\\_id=elli](https://elli.hs-bremerhaven.de/goto.php?target=file_308663_download&client_id=elli)

Zugriff am 01.07.2023 17:00 Uhr

[11]Vorlesug UserStories

[https://elli.hs-bremerhaven.de/goto.php?target=file\\_308664\\_download&client\\_id=elli](https://elli.hs-bremerhaven.de/goto.php?target=file_308664_download&client_id=elli)

Zugriff am 06.08.2023 20:34 Uhr

[12]Übung Ue\_02\_Wochenaufgabe\_PM\_undRecherche

[https://elli.hs-bremerhaven.de/goto.php?target=file\\_308132\\_download&client\\_id=elli](https://elli.hs-bremerhaven.de/goto.php?target=file_308132_download&client_id=elli)

Zugriff am 11.04.2023 12:000 Uhr

[13]Übung Ue\_06\_OSM-undJS\_Uebung-Storyboarding

[https://elli.hs-bremerhaven.de/goto.php?target=file\\_312790\\_download&client\\_id=elli](https://elli.hs-bremerhaven.de/goto.php?target=file_312790_download&client_id=elli)

Zugriff am 18.08.2023 22:07 Uhr

[14]Vorlesung Von den Anforderungen zum Entwurf

[https://elli.hs-bremerhaven.de/goto.php?target=file\\_316314\\_download&client\\_id=elli](https://elli.hs-bremerhaven.de/goto.php?target=file_316314_download&client_id=elli)

Zugriff am 18.08.2023 22:10 Uhr

[15]Vorlesung Sequenzdiagrammen

[https://elli.hs-bremerhaven.de/goto.php?target=file\\_317164\\_download&client\\_id=elli](https://elli.hs-bremerhaven.de/goto.php?target=file_317164_download&client_id=elli)

Zugriff am 18.08.2023 22:18 Uhr

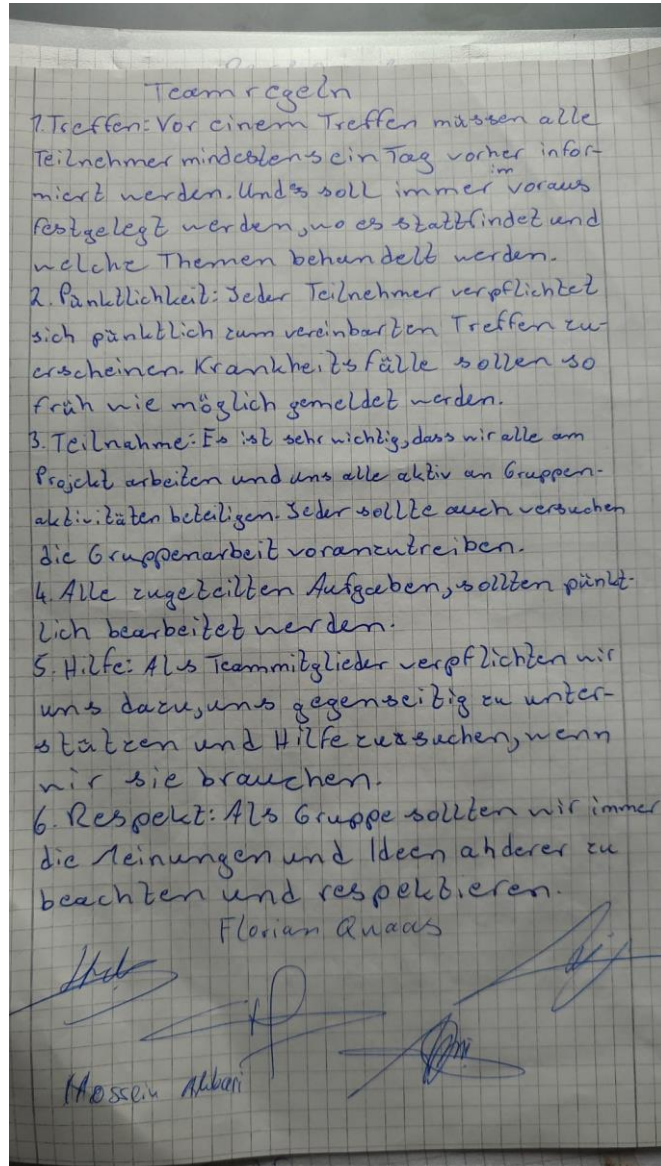
[16]Vorlesung Qualitätssicherung und Prototyp Test

[https://elli.hs-bremerhaven.de/goto.php?target=file\\_317799\\_download&client\\_id=elli](https://elli.hs-bremerhaven.de/goto.php?target=file_317799_download&client_id=elli)

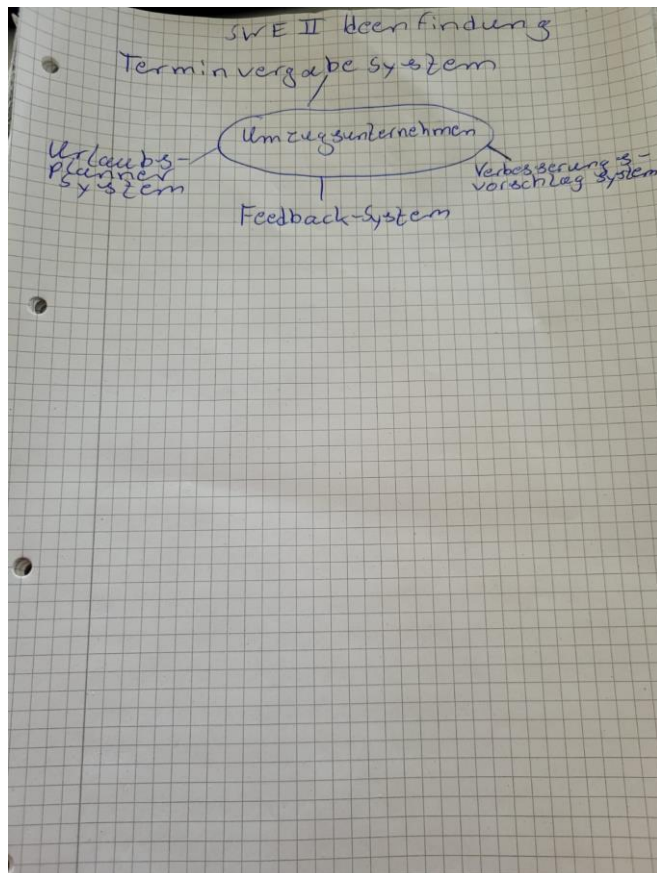
Zugriff am 18.08.2023 22:27 Uhr

## 15. Anhang

### 15.1 Teamregeln



## 15.2 Ideenfindung



## 15.3 Interviewleitfaden

### 1. Begrüßung

### 2. Zweck des Interviews erläutern->Ziel Anforderungserhebung

### 3. Fragen stellen zur Erhebung der Anforderungen an das System (sowohl funktional als auch nicht funktional)

1. Was wird gewünscht?

2. Was soll es können?

3. Wie läuft der Prozess bisher?

4. Was sind die Kapazitäten?

5. Was sind die buchbaren Zeiten?

6. Was sind spezielle Anforderungen der Kunden?

7. Wie werden ihre Mitarbeiter informiert?

8. Was für Funktionen soll sie haben?

9. Was ist Ihnen wichtig für die Benutzeroberfläche?

10. Was stellen Sie sich vor, wie sie die Anwendung nutzen würden?

11. Was stellen Sie sich für Nutzung des Systems und deren Nutzung vor?

12. Was ist ihren Kunden besonders wichtig?

13. Was ist die Zielgruppe?

14. Was ist der Bedarf?

### 4. Interview beenden

1. Aufklärung über weiteren Ablauf

2. Bedanken und Nummern Austausch