In dieser Übung betrachten wir die Anforderungen der Software *Sidequest*. Das Ziel von *Sidequest* ist es Menschen zu helfen ihre Aufgaben zu organisieren.

1 Aufgabe 1

Ergänzen Sie pro Gruppenmitglied 1-2 Anwendungsfälle. Hierzu identifizieren Sie Features, in denen Anwendungsfälle sinnvoll sind. Dies sind vor allen die Features, bei denen der Ablauf nicht offensichtlich oder nicht trivial ist. Oft betreffen gute Anwendungfälle mehrere Anforderungen.

2 Aufgabe 2

Spezifizieren Sie ein optionales Feature. Die Whatsapp Integration (O1) dient als Beispiel Anforderungen an das Feature sind: (1) Das Feature soll einen echten Mehrwert liefern mit der realistischerweisen Option, dass Nutzer die Anwendung wegen dieses Features verwenden oder kaufen. (2) Das Feature soll realistisch umsetzbar sein.

3 Vision

Sidequest unterstützt Individuen und Gruppen wie z.B. Familien oder Vereine ihre Alltags- und Verwaltungsaufgaben im Griff zu behalten. Durch Künstliche Intelligenz lernt das System, welche Schritte zur Erreichung eines Ziels erforderlich sind und verfeinert Aufgaben (genannt Quests) automatisch zu Schritten (genannt Subquests). *Sidequest* kann erweitert werden, um nach und nach immer mehr Verwaltungsaufgaben automatisch zu übernehmen.

4 Ziele

- Ziel 1: *Sidequest* ist ein Mehrbenutzer System, dass es erlaubt, Aufgaben anzulegen, zu Unteraufgaben zu verfeinern und den Status der Aufgaben zu verfolgen.
- Ziel 2: *Sidequest* erlaubt es, Aufgaben oder Teilaufgaben einer Aufgabe an andere Personen zu delegieren.
- Ziel 3: *Sidequest* erkennt, wenn Nutzer wiederkehrend semantisch ähnliche Aufgaben anlegen, und nutzt dies, um automatisch Subquests vorzuschlagen. Diese Erkennung funktioniert sowohl Nutzerbezogen, d.h. das System erkennt, dass ein Nutzer Aufgaben immer auf die gleiche Art erledigt, als auch über Nutzer hinweg, d.h. das System kann eine Aufgabe zerlegen, weil andere Nutzer sie bereits sinnvoll zerlegt haben.

5 Fragen und Probleme

- Frage 1 Wie soll mit Deadlines umgegangen werden?
- Frage 2 Wie soll mit Deadlines umgegangen werden?

6 Funktionale Anforderungen

Systemqualität	sehr gut	gut	normal	nicht relevant
Funktionalität		x		
Zuverlässigkeit			x	
Benutzbarkeit	x			
Effizienz			x	
Wartbarkeit		x		
Portabilität	x			

6.1 Begriffslexikon

Begriff: Benutzer

Bedeutung: Account eines Benutzers, der von mehreren Endgeräten aus benutzt werden kann.
Gültigkeit: Ein *Benutzer* existieren von der Registrierung (siehe R X) eines Benutzers bis zu seiner Löschung (R Y).
Identifikation: *Benutzer* wird identifiziert über eine EMail-Adresse. Hierbei findet nur ein Sanity-Check statt (Eingabestring entspricht EMail-Format), aber eine Überprüfung der Existenz eines passenden EMail-Kontos.

Begriff: Benutzerprofil
Bedeutung: Konfigurierbar ist: Zeitraum der automatischen Löschung, ...
Gültigkeit: Die Benutzereinstellungen existieren von der Anlage eines Benutzers bis zu seiner Löschung.
Identifikation: Ist 1:1 mit dem Benutzer verknüpft.
Querverweise: *Benutzer*

Begriff: Quest
Bedeutung: Eine *Quest* repräsentiert eine Aufgabe, die ein *Benutzer* lösen möchte. Es gibt für jede *Quest* einen Besitzer (Standart: Ersteller) und einen Bearbeiter (Standart: Besitzer).
Abgrenzung: Quest Template
Gültigkeit: Eine *Quest* existiert von ihrer Anlage bis zu ihrer Löschung.
Identifikation: Eine *Quest* wird über eine interne ID identifiziert.
Querverweise: *Quest Status*, *Haupquest*, *Subquest*

Begriff: Hauptquest
Bedeutung: Eine *Hauptquest* ist eine *Quest* ohne übergeordnete *Quest*.

Begriff: Besitzer
Bedeutung: Der *Besitzer* einer Quest ist ein *Benutzer*, der für eine *Quest* verantwortlich ist und Rechte an dieser in vollem Umfang hat. Es gibt für jede *(Sub-)Quest* nur einen Besitzer.
Abgrenzung: Bearbeiter
Gültigkeit: Ein *Besitzer* existiert für jede *(Sub-)Quest* von Anfang bis zur Löschung.
Identifikation: Der *Besitzer* einer Quest ist ein *Benutzer*.
Querverweise: *Bearbeiter*

Begriff: Bearbeiter
Bedeutung: Der *Bearbeiter* einer Quest ist ein *Benutzer*, der berechtigt ist, eine *(Sub-)Quest* zu bearbeiten, d.h. er kann die *Quest* ändern oder ihren Status ändern.
Abgrenzung: Besitzer
Identifikation: Der *Bearbeiter* einer Quest ist ein *Benutzer*.
Querverweise: *Besitzer*

Begriff: Gruppe
Bedeutung: Eine *Gruppe* ist ein spezieller *Benutzer*, der stellvertretend für eine Menge konkreter *Benutzer* sowohl *Besitzer* als auch *Bearbeiter* sein kann. Anstelle dieses Gruppenbenutzers kann jedes Gruppenmitglied gleichermaßen *Besitzer*- u. *Bearbeiter* verändern und *(Sub-)Quests*, die speziell für diese *Gruppe* erstellt wurden, an Nichtmitglieder delegieren.
Abgrenzung: Benutzer
Gültigkeit: Eine *Gruppe* existiert von der Erstellung bis zur Löschung.
Identifikation: Die *Gruppe* ist ein spezieller *Benutzer* und wird daher auch durch eine interne ID identifiziert.
Querverweise: *Benutzer*

Begriff: Questhierarchie
Bedeutung: Eine *Questhierarchie* ist eine Menge an *Quests*, die untereinander verbunden sind. Zur *Questhierarchie* einer *Quest* gehören alle *Subquests*, deren *Subquests* und so weiter sowie alle übergeordneten *Quests* und deren *Questhierarchien*.
Abgrenzung: Quest
Querverweise: *Quest*, *Haupquest*, *Sidequest*

Begriff: Subquest
Bedeutung: Eine *Subquest* ist eine *Quest*, die einer anderen *Quest* untergeordnet ist. Um eine *Quest* zu erfüllen, müssen zunächst alle *Subquests* erfüllt werden.

Begriff: Quest Status
Bedeutung: Eine *Quest* kann verschiedene Status haben. Neu angelegt ist der Status "offen". Wird die *Quest* erledigt, ist der Status "erledigt". Sind alle Quests einer *Questhierarchie* erledigt, so wird der Status der Hierarchie "inaktiv', d.h. alle darin enthaltenen Quests werden nicht mehr angezeigt'. Wird eine Quest entfernt, so bekommt sie den Status "gelöscht".
Querverweise: Quest, Questhierarchie

Begriff: Quest Template
Bedeutung: Ein *Quest Template* repräsentiert ein wiederkehrendes Ziel, dass *Benutzer* lösen möchten. Durch *Quest Templates* können *Quests* automatisch mit *Subquests* befüllt werden.
Abgrenzung: Quest
Gültigkeit: Ein *Quest Template* existiert als Programmbestandteil ständig.
Identifikation: Eine *Quest Template* wird über eine interne ID identifiziert.
Querverweise: Quest, Hauptquest, Sidequest

Begriff: Weiches Löschen
Bedeutung: Es gibt zwei Arten des Löschens. Beim "weichen Löschen" (auch Entfernen genannt) wird der Status der Quest auf "gelöscht" gesetzt.
Abgrenzung: Hartes Löschen
Querverweise: Quest

Begriff: Hartes Löschen
Bedeutung: Es gibt zwei Arten des Löschens. Beim "harten Löschen" wird die Quest vollständig aus dem System entfernt.
Abgrenzung: Hartes Löschen
Querverweise: Quest

6.2 Anforderungen (Requirements)

6.2.1 R1 Quest Anlage

- R1.1 *Benutzer* können *Quests* für sich selber anlegen. Hierzu wird ein Titel und eine optionale Beschreibung eingegeben, *Besitzer* und *Bearbeiter* sind dann der Benutzer selber.
- R1.2 *Benutzer* können *Quests* für eine *Gruppe* anlegen. *Besitzer* und *Bearbeiter* ist dann die *Gruppe*.
- R1.3 *Benutzer* können eine soeben angelegte *Quests* mit *Subquests* befüllen. Das Befüllen einer *Quests* mit potentiell vielen *Subquests* wird in der Benutzerführung so komfortabel wie möglich

umgesetzt.

- R1.4 Quests sind grundlegend hierarchisch, d.h. *Subquests* können wieder *Subquests* enthalten. Die Anlage von auch tief gestaffelten *Quests* wird so komfortabel wie möglich umgesetzt.
- R1.5 Bei der Anlage einer *Quests* verwendet das System alle von diesem *Benutzer* bereits verwendeten Titel zur Autocompletion.
- R1.6 Wenn ein *Benutzer* eine *Quests* erstellt, für die ein *Questtemplate* existiert, so schlägt das System die *Subquests* des *Questtemplate* automatisch vor. Ein *Questtemplate* existiert dann, wenn der Titel einer *Quests* dem Titel einer bereits von diesem *Benutzer* erstellten *Quests* exakt entspricht.

6.2.2 R2 Quest Erledigen

- R2.1 *Benutzer* können ihnen als *Bearbeiter* zugeordnete *Quests* als "erledigt" markieren.
- R2.2 *Quests* können erst erledigt werden, wenn alle zugehörigen *Subquests* erledigt sind.
- R2.3 Sind alle *Subquests* einer *Quests* erledigt, wird vorgeschlagen, diese ebenfalls als erledigt zu markieren.
- R2.4 Sind in einer *Hierarchischen Quests* alle *Quests* erledigt, so heisst die gesamte Hierarchie *inaktiv* und wird ausgeblendet.

6.2.3 R3 Quest Delegieren

- R3.1 *Besitzer* von *Quests* können an andere *Benutzer* oder *Gruppen* delegieren. Wird eine *Quests* delegiert, so werden die zugehörigen *Subquests* ebenfalls delegiert. Es gibt zwei Arten von delegieren: wird die Bearbeitung delegiert, so ändert sich der *Bearbeiter* einer *Quest*. Wird die Zuständigkeit delegiert, so ändert sich der *Besitzer*. Delegierung der Bearbeitung ist der Standardfall, dieser muss so komfortabel wie möglich umgesetzt werden.
- R3.2 Wird eine *Quest* an einen *Benutzer* delegiert, so muss dieser sie erst annehmen bevor sie ihm zugewiesen wird.
- R3.3 Ist eine *Gruppen* der *Bearbeiter* einer *Quest*, so kann ein Mitglieder der *Gruppen* die *Quest* oder eine *Subquests* an sich selber delegieren ("ich übernehme das").

6.2.4 R4 Quest Sichtbarkeit

- R4.1 *Benutzer* bekommen alle *Quests* angezeigt, wenn sie entweder *Besitzer* oder *Bearbeiter* sind, entweder direkt oder indirekt über eine *Gruppe*.
- R4.2 Normalerweise werden *Quests* nur angezeigt, wenn sie im Status "offen" oder "erledigt" sind. *Quests* aus inaktiven *Hierarchien* oder gelöschte Quests werden nicht angezeigt (Ausnahme: siehe R4.4).
- R4.3 *Quests* können in mehreren Ansichten angezeigt werden.
  - R4.3.1 Ansicht nach Priorität
  - R4.3.2 Hierarchische Ansicht, gegliedert nach *Hauptquests*
  - R4.3.3 Gegliedert nach Besitzer (*Benutzer* selbst oder Gruppen)
- R4.4 *Benutzer* können einen vollständigen Verlauf ihrer *Quests* anzeigen, dann werden auch Quests im Status "inaktiv" oder "gelöscht" angezeigt.

6.2.5 R5 Quest Löschen

- R5.1 *Benutzer* können *Quests* löschen, wenn sie *Besitzer* sind. Eine *Quest* kann weich gelöscht werden. Das weiche Löschen ist der Standardfall und soll so komfortabel wie möglich umgesetzt werden. Wird eine *Quest* gelöscht, so werden alle *Subquests* ebenfalls gelöscht.
- R5.2 *Quests* können hart gelöscht werden.

6.2.6 R6 Quest Bearbeiten

- R6.1 *Benutzer* können alle Attribute von *Quests* bearbeiten. Das beinhaltet Titel, Beschreibung, Status.
- R6.2 *Benutzer* können nachträglich *Subquests* ergänzen.
- R6.3 *Benutzer* können *Subquests* hinsichtlich der Reihenfolge innerhalb einer *Quest* nachträglich umsortieren
- R6.4 *Benutzer* können *Subquests* nachträglich einer anderen *Quest* zuordnen.

6.2.7 R7 Benutzerverwaltung

- R 7.1 *Benutzer* müssen mit Email Adresse und Namen angelegt werden können.
- R 7.2 *Benutzer* müssen harrt gelöscht werden können.
- R 7.3 Alle Attribute eines *Benutzers* müssen sich ändern lassen können.
- R 7.4 Wird ein *Benutzer* gelöscht, so müssen ihm zugewiesene *Quests* an einen anderen noch existierenen *Benutzer* übertragen werden können. Dieser muss solche *Quests* annehmen können, andernfalls werden sie gelöscht.

6.2.8 R8 Gruppenverwaltung

- R 8.1 *Benutzer* können *Gruppen* anlegen. Der *Benutzer* ist dann Mitglieder dieser *Gruppe*.
- R 8.2 Bei Anlage einer *Gruppe* können weitere *Benutzer* zugefügt werden. Die neu angelegte *Gruppe* bekommt einen automatisch aus den Namen der Mitglieder gebildeten Namen.
- R 8.3 *Gruppen* können umbenannt werden.
- R 8.4 *Gruppen* können wieder Mitglieder in *Gruppen* sein. Es sind keine Zirkel möglich.

6.2.9 O1 Whatsapp Integration (optional, Tim Dahmen)

- O 1.1 Eine Sidequest *Gruppe* kann vom Besitzer der *Gruppe* mit einer Whatsappgruppe verknüpft werden.
- O 1.2 Ist eine *Gruppe* mit Whatsapp verknüpft, so sendet das System automatisch Nachrichten in die Whatsappgruppe, wenn in der *Gruppe* eine in der Quest erledigt, geändert, oder angelegt wird.
- O 1.3 Ist eine *Gruppe* mit Whatsapp verknüpft, so sendet das System automatisch Nachrichten in die Whatsappgruppe, wenn ein Benutzer einer Gruppe beitritt oder diese verlässt.
- O 1.4 Die Verknüpfung einer Sidequest *Gruppe* mit einer Whatsappgruppe kann vom Besitzer der *Gruppe* wieder aufgelöst werden.
- O 1.5 Ein *Benutzer* kann die Whatsapp Benachrichtigungen pro Quest oder pro Gruppe stumm schalten und die Stummschaltung wieder aufheben. Die Stummschaltung betrifft dann nur diesen Benutzer.

- O 1.6 Ein *Benutzer* kann sich mit seinen Whatsapp Account verknüpfen und dies wieder aufheben.
- O 1.7 Ist ein *Benutzer* mit einem Whatsapp Account verknüpft, so erhält der Whatsapp Benutzer Benachrichtigungen wie in O 1.2 und O 1.3 für alle *Quest* und *Gruppen*, mit denen der *Benutzer* verknüpft ist.

6.3 Use-Cases

Numbers: SQ-1
Name: Quest abschließen
Actors: *Besitzer* der *Quest*, *Bearbeiter* der letzten *Subquest*
Trigger: Die letzte offene *Subquest* einer *Quest* wird erledigt.
Preconditions: Eine *Quest* (Q_{parent}) hat den Status "offen" und genau eine *Subquest* (Q_{child}), hat ebenfalls den Status "offen".
Postconditions / Goal: Der Status von Q_{parent} ist "erledigt".
Postconditions Special Cases:

3a Der Status von Q_{parent} bleibt "offen".

4a Der Status der übergeordneten Quests wird ebenfalls auf "erledigt" gesetzt.

5a Der Status der gesamten Questhierarchie wird auf "inaktiv" gesetzt.

Steps Default Case:

1 *Bearbeiter* der letzter offener *Subquest* setzt deren Status auf "erledigt".

2 Der *Besitzer* von Q_{parent} bekommt eine Nachricht über die Statusänderung und wird gefragt, ob Q_{parent} ebenfalls als "erledigt" gesetzt werden soll.

3 Der *Besitzer* Q_{parent} bestätigt die vollständige Bearbeitung der übergeordneten *Quest*.

4 Es wird überprüft, ob Q_{parent} wiederum eine übergeordnete *Quest* besitzt, die die Precondition von SQ-1 erfüllt, so dass sich der Use-Case rekursiv fortsetzt.

5 Es wird überprüft, ob in der gesamten *Questhierarchie* von Q_{child} keine *Quest* mehr den Status "offen" hat (in diesem Fall 5a).

Special Cases:

3a Besitzer lehnt es ab Q_{parent} auf "erledigt" zu setzen.

4a Weiter mit Schritt 1, wobei Q_{parent} die Rolle von Q_{child} einnimmt.

5a Die gesamte *Questhierarchie* wird auf den Status "inaktiv" gesetzt und ausgeblendet.

7 Nicht Funktionale Anforderungen

7.0.1 NF1 Systemumgebung

- NF1.1 Das System soll eine unter Windows lauffähige App aufweisen.
- NF1.2 Das System soll eine Android App aufweisen (optional, Tim Dahmen).
- NF1.3 Das System soll eine Apple App aufweisen (optional, Tim Dahmen).

7.0.2 NF2 Skalierbarkeit

- NF2.1 Ein *Benutzer* kann hunderte von *Quests* verwalten, die nicht im Status “inaktiv” sind.
- NF2.2 Ein *Benutzer* kann tausende von *Quests* verwalten, die im Status “inaktiv” sind.
- NF2.3 Das System kann Millionen *Quests* verwalten.
- NF2.4 Eine *Gruppe* kann hunderte *Benutzer* enthalten.
- NF2.5 Ein *Quests* kann maximal hundert direkte *Subquests* enthalten.
- NF2.6 Ein *Questhierarchie* kann tausende *Quests* enthalten.
- NF2.7 Das System kann tausende *Benutzer* enthalten.

7.0.3 NF3 Performanz

- NF3.1 Das System ist jederzeit ohne als störend wahrgenommene Verzögerung bedientbar. Es gibt die Vermutung, dass dies bei einer Antwortzeit ¡1500ms für Transaktionen der Fall ist.
- NF3.2 Das System kann bis zu 10 Änderungen pro Sekunde verarbeiten.
- NF3.3 Das System kann mit maximal tausend gleichzeitg eingeloggten *Benutzer* umgehen.