Bachelorarbeit

Verbesserungen der virtuellen Arbeitskultur durch Gamification

Zürcher Hochschule für Angewandte Wissenschaften
School of Management and Law
Studiengang Wirtschaftsinformatik

vorgelegt von:
Marcel Mettler
[redacted]

Betreut von:
Benjamin Kühnis

Datum der Abgabe:
09.06.2021

Management Summary

Durch die Digitalisierung, besseren Netzwerkverbindungen und wachsende Infrastruktur wird es immer einfacher von virtuellen Arbeitsplätzen aus zu arbeiten. Virtuelle Arbeitsformen wie Homeoffice, Remote Work oder hybride Versionen davon nehmen stetig zu. Gerade während der Covid-19 Pandemie wurden mehr Arbeitnehmer und Arbeitgeber mit dieser Thematik konfrontiert, da viele Arbeitnehmer gesetzlich im Homeoffice arbeiten mussten und somit erste Erfahrungen mit dieser Arbeitsform sammeln konnten. Erste Umfragen zeigen, dass diese Form der Arbeit bei vielen beliebt ist und es sich nur wenige Personen vorstellen können, wieder in einen absoluten Büroalltag zurückzukehren. Jedoch bringt das Arbeiten zuhause auch Nachteile mit sich. Arbeitnehmer beklagen sich über die Isolation und die dadurch fehlenden sozialen Interaktionen, was zu schwerwiegenden psychischen Folgeerkrankungen führen kann.

Daraus entwickelt sich die Frage, ob eine Applikation, welche einen virtuellen Arbeitsplatz simuliert, dazu beitragen kann dieses Problem zu lösen und somit das Isolationsgefühl mindert. Zusätzlich soll die Arbeitsmoral gesteigert werden, ohne, dass sich durch diese Applikation Arbeitnehmer überwacht vom Arbeitgeber fühlen.

Für die Beantwortung dieser Frage wurden zuerst mithilfe von Literaturrecherche die theoretischen Grundlagen zu virtuellen Arbeitsformen und die gesetzlichen Grundlagen zum Sammeln und Speichern von Daten erarbeitet. Anhand der gesammelten Informationen wurde in der vorliegenden Arbeit eine solche Applikation geplant und anschliessend der Backend dieser Applikation umgesetzt. In einer weiteren Arbeit wurde zeitgleich der Frontend implementiert, um somit ein Minimal Viable Product zu erhalten. Die Applikation simuliert ein virtuelles Office, welches verschiedene Räume wie ein Grossraumbüro, einen Meeting Raum und eine Küche enthält. Jeder User hat einen eigenen virtuellen Avatar, mit dem er sich im virtuellen Office frei bewegen kann. Zudem wurden verschiedene Gamification Elemente, wie Quests und ein Leaderboard, eingebaut.

Mithilfe des erstellten Minimal Viable Product konnte die Akzeptanz, sowie die Lösung des Problems, mittels dieser Applikation geprüft werden. Die Validierung hat ergeben, dass sich Personen, welche die Applikation verwendet haben, verbundener mit ihren Kollegen im Office gefühlt haben. Die implementierten Gamification Elemente führten zu

einer Abwechslung im Alltag, was die Motivation steigern konnte. Zudem gab es eine Möglichkeit sich mit seinen Kollegen zu messen und zu vergleichen.
Da sich User frei im Office bewegen konnten, hatten sie die Möglichkeit, selbst zu entscheiden, wie viele Informationen sie über ihren aktuellen Status preisgeben wollten. Dadurch entstand kein Überwachungsgefühl.

Inhaltsverzeichnis

Tabellenverzeichnis

Abbildungsverzeichnis

Abkürzungsverzeichnis

API	Application Programming Interface
CD	Continuous Delivery
CI	Continuous Integration
MVP	Minimal Viable Product
JWT	JSON Web Token

1 Einleitung

In der Einleitung wird zuallererst auf die Ausgangslage und die Problemstellung eingegangen. Danach wird die Forschungsfrage gebildet und die Ziele der Arbeit definiert.

1.1 Ausgangslage

Virtuelle Arbeitsformen wie Remote Work oder eine Mischform zwischen Homeoffice und Büroalltag nehmen immer häufiger zu. Die momentane Covid-19 Krise hat die Anzahl an Personen, welche in der Schweiz von Zuhause aus arbeiten, verdoppelt (Deloitte, 2020). Gemäss einer Umfrage von Slack (Slack, 2020) wollen nur 12% der Befragten Wissensarbeiter permanent zurück in einen normalen Büroalltag. 72% der Befragten geben an, eine hybride Lösung zu bevorzugen (Slack, 2020). Es ist noch unklar, wie die zukünftigen Arbeitsweissen aussehen werden, jedoch ist klar, dass nach Covid-19 mehr Personen in einer virtuellen Arbeitsposition sein werden.

1.2 Problemstellung

Ein grosser Teil der Personen, welche neu im Homeoffice arbeiten, erleben einen Mangel an persönlichen Gesprächen und fühlen sich isoliert (Deloitte, 2020). Dieser kann einen grossen Einfluss auf die psychische Gesundheit haben (Deloitte, 2020). Wenn es zu Ablenkungen kommt, haben gerade Personen mit Familie die grössten Probleme. Diese Gruppe gibt auch an, dass die Ablenkung von Kindern eine der grössten Herausforderungen im Homeoffice ist (Deloitte, 2020).

Homeoffice führt aber auch dazu, dass Arbeitgeber weniger Transparenz haben, wie viel und zu welchen Zeitpunkten wirklich von Mitarbeitern gearbeitet wird. Das Erfassen von Informationen der Mitarbeiter ist aber schwierig, da die Erfassung von persönlichen Daten, sowie die Überwachung von Mitarbeitern, gesetzlich geregelt ist. Zudem kann es auch schnell zu Misstrauen führen, wenn sich Mitarbeiter überwacht fühlen.

1.3 Forschungsfrage

Diese Arbeit untersucht, inwiefern eine Applikation mit Hilfe von Gamification Elementen die virtuelle Arbeitskultur verbessern kann. Dafür soll folgende Forschungsfrage beantwortet werden:

Wie kann eine Applikation, welche einen virtuellen Arbeitsplatz simuliert, dazu beitragen, die Arbeitsmoral bei virtuellem Arbeitsformen zu steigern, ohne dass sich Mitarbeitende dabei überwacht fühlen?

Teilfragen:
Welche personenbezogenen Daten dürfen von der Applikation rechtlich erhoben und bearbeitet werden?

Wie kann die Datenerfassung transparent gestaltet werden, damit Mitarbeiter bereit sind ihre persönlichen Daten freiwillig zu teilen?

1.4 Ziele

Das Ziel dieser Arbeit ist Remote Work als Arbeitsform wirksamer zu etablieren. Dazu wird eine Applikation erstellt, welche einen virtuellen Arbeitsplatz simuliert. Diese soll Remote Work für Arbeitnehmer und Arbeitgeber ansprechender gestalten und dessen Attraktivität steigern. Mit Hilfe dieser Applikation soll die Arbeitsmoral verbessert werden, ohne dass sich Mitarbeiter dabei überwacht fühlen.

1.5 Abgrenzung

Der Schwerpunkt dieser Arbeit liegt bei der Implementierung eines Backend für die Applikation. Eine weitere Bachelorarbeit erstellt zeitgleich den Frontend der Applikation.

2 Vorgehen und Methoden

In diesem Kapitel wird erläutert, welches Vorgehen und welche Methoden verwendet wurden, um die Fragestellung zu beantworten. In einem ersten Teil wird beschrieben, wie die Ausgangslage analysiert wurde und in einem zweiten Teil wird die eigentliche Umsetzung der Applikation beschrieben.

2.1 Analyse der Ausgangslage

Zu Beginn der Arbeit wird als erstes eine Analyse der Ausgangslage durchgeführt. Dazu wurde eine theoretische Untersuchung von virtuellen Arbeitsformen und den rechtlichen Aspekten zur Erfassung von Daten mithilfe von Literaturrecherche durchgeführt.

Nach Abschluss der Analyse wird in einem zweiten Schritt mit der Erstellung der Anforderung an die Applikation begonnen. Dafür werden als erstes User-Stories erstellt. User-Stories beschreiben eine Anforderung an die Software aus der Sicht eines Users. In einem weiteren Schritt werden basierend auf den User-Stories Benutzerabläufe erstellt, welche genauer aufzeigen, wie der User mit der Software interagiert, um einen Prozess zu durchlaufen. Zusätzlich werden auch die nicht-funktionalen Anforderungen an die Software erhoben.

Anhand der Analyse und der erarbeiteten Anforderungen wird die Systemarchitektur entworfen. Dazu werden die einzusetzenden Technologien definiert sowie der Aufbau der Software und die Planung einer DevOps Pipeline.

Das MVP wird durch ausgewählte Testpersonen sowie den Ersteller des Frontend und den Autoren validiert. Das Ziel ist es, das MVP auf seine Akzeptanz und der Einsetzbarkeit zu überprüfen.

2.2 Iterative Umsetzung

Für die Umsetzung der Applikation wurde ein iteratives Vorgehen gewählt. Bei einem iterativen Vorgehen wird sich der Lösung schrittweise angenähert. Dabei werden in jeder Iteration alle Projektphasen durchlaufen und eine beendete Iteration liefert eine funktionierende Applikation, welche sich mit jeder Iteration verbessert (Hämmerli, 2011). Diese flexible Arbeitsweise ist optimal, da in einer zweiten Arbeit der Frontend der Applikation umgesetzt wird und somit die Arbeitspakete klein bleiben. Der Ersteller des Frontend und des Backend können sich fortlaufend absprechen.
Das Ziel ist es, mit jeder Iteration einem MVP näher zu kommen, welches verwendet werden kann, um zu überprüfen, ob dieses zur Lösung des Problems beiträgt. Während der Arbeit soll kein praxisreifes Produkt entstehen.

Für das Projektmanagement wird die Software Jira und Confluence eingesetzt. Jira wird verwendet, um ein Kanban-Board zu erstellen. Confluence bietet die Möglichkeit, online an Dokumenten zu arbeiten und diese direkt mit Jira zu verknüpfen. In Confluence werden Dokumente für die Planung der Applikation, wie die Anforderungen, festgehalten.

3 Analyse

In diesem Kapitel werden die grundlegenden Theorien analysiert, welche benötigt werden, um ein grundlegendes Wissen aufzubauen und die Anforderungen an die Applikation zu erheben.

3.1 Virtuelle Arbeitsformen

Verschiedene technische und nicht technische Treiber verändern die Arbeitswelt. Es verändern sich Arbeitsinhalte, Arbeitsstrukturen und Tätigkeiten (Neuburger, 2020, S. 1). Klassische hierarchische Strukturen, bei denen Mitarbeiter keinen Einfluss haben, werden weniger akzeptiert. Mitarbeiter setzen neu mehr Wert auf Selbständigkeit, Individualität und streben nach sinnvollen und inhaltlich spannenden Aufgaben (Neuburger, 2020, S. 3). Neue digitale Technologien, verändern Prozesse in Unternehmen und sorgen für steigende Effizienz (Neuburger, 2020, S. 2). Digitale Technologien ermöglichen es Unternehmen sich stärker zu vernetzen (Neuburger, 2020, S. 2). Die Vernetzung führt dazu, dass einerseits physische Geräte wie Maschinen oder Roboter aus der Ferne in Echtzeit gesteuert werden können, aber sich auch Personen einfacher und schneller austauschen können (Neuburger, 2020, S. 2). Durch diese Änderungen und Umstellungen in Unternehmensstrukturen entstehen immer häufiger virtuelle Arbeitsplätze.

Der Virtuelle Arbeitsplatz bietet die Möglichkeit, dass keine physische Anwesenheit im Büro Notwendig ist. Angestellte arbeiten aber mobile oder an anderen Standorten, als wären sie im Unternehmen anwesend (Neuburger, 2020, S. 6). Sie befinden sich im Homeoffice, im Büro der Kundschaft, unterwegs oder in einem Café (Neuburger, 2020, S. 6). Die einzige Bedingung, die erfüllt sein muss, ist ein Zugang zum Internet, um auf Unternehmensdaten zuzugreifen (Neuburger, 2020, S. 6).

Virtuelle Arbeitsplätze bringen viele Vorteile mit sich. Unter anderem führt die Digitalisierung des Arbeitsplatzes zu höherer Flexibilität der Mitarbeiter (Krämer, 2019, S. 26), welche ihre Arbeitszeit ihren individuellen Ansprüchen anpassen können (Neuburger, 2020, S. 8). Das verlagern des Arbeitsortes in einen Virtuellen Arbeitsplatz kann auch Kosteneinsparungen mit sich bringe. Es wird weniger Bürofläche benötigt und Reisekosten für Meetings können eingespart werden, wenn diese online stattfinden (Neuburger, 2020, S. 9). Jedoch gerade in der Umstellungsphase von einem lokalen in einen virtuellen

Arbeitsplatz können die kosten höher sein. Denn in der anfangs Phase müssen Investitionen getätigt werden und Mitarbeiter und Führungskräfte ausgebildet (Krämer, 2019, S. 39).

Gesetzliche Regeln müssen vom Arbeitgeber auch im Homeoffice auch eingehalten werden (Krämer, 2019, S. 39). Das Gesetzt unterscheidet nicht zwischen normalen Arbeiten im Büro und Homeoffice, deshalb gelten die normalen Gesetze auch im Homeoffice (Staatssekretariat für Wirtschaft, 2019). Wenn ein Mitarbeiter im Homeoffice arbeitet, ist der Arbeitgeber nicht von seinen Pflichten entbunden und muss weiterhin den Gesundheitsschutz, sowie den Arbeits- und Ruheschutz gewährleisten (Staatssekretariat für Wirtschaft, 2019).

3.2 Erfassung personenbezogener Daten

Damit ein Virtuelles Office umgesetzt werden kann, müssen Daten der Nutzer erhoben werden. Beim Sammeln von Daten wird zwischen Personendaten und Sachdaten unterschieden (Kaufmann, 2021). Personendaten sind Angaben, Informationen und Aussagen welche sich auf eine bestimmte oder bestimmbare Person beziehen (Eidgenössischer Datenschutz- und Öffentlichkeitsbeauftragter, o. J.-b). Bei Sachdaten handelt es sich um Daten, welche sich nicht auf eine bestimmte oder bestimmbare Person beziehen (Kaufmann, 2021). Personendaten unterliegen dem Datenschutz (Kaufmann, 2021). Datenschutz beschreibt den Schutz der Privatsphäre bei der Datenverarbeitung von Personendaten (Eidgenössischer Datenschutz- und Öffentlichkeitsbeauftragter, o. J.-a). Der Datenschutz schützt persönliche Daten vor der wirklichen Verwendung. Er legt fest, dass nur so viele Personendaten, wie benötigt, gesammelt und bearbeitet werden dürfen (Eidgenössischer Datenschutz- und Öffentlichkeitsbeauftragter, o. J.-a). Zudem regelt er das die betroffene Person die Möglichkeit hat, seine persönlichen Daten zu kontrollieren und das Inhaber von Datensammlungen Auskunft geben müssen, welche persönlichen Daten gesammelt werden und für welchen Zweck diese verarbeitet werden. (Eidgenössischer Datenschutz- und Öffentlichkeitsbeauftragter, o. J.-a). Der Datenschutz unterliegt dem Datenschutz Gesetz (DSG). Das DSG baut auf Art. 13 der Bundesverfassung auf welcher die Achtung sowie den Schutz vor Missbrauch von persönlichen Daten festlegt (Eidgenössischer Datenschutz- und Öffentlichkeitsbeauftragter, o. J.-a).

Gemäss DSG Art. 4 dürfen personenbezogene Daten nur für den Zweck verwendet werden, welcher bei der Beschaffung der Daten angegeben wurde. Beim beschaffen von personenbezogenen Daten, muss der betroffenen Person klar ersichtlich sein, welche Daten erhoben und diese bearbeitet werden. Die Einwilligung der betroffenen Person zur Bearbeitung der Daten ist erst gültig, wenn die betroffene Person nach angemessener Information diese freiwillig erteilt.

Nach der Zweckerfüllung dürfen Personendaten nicht mehr weiterbearbeitet werden und müssen gelöscht werden (Dohr, 2021). Damit die Informationen der Personendaten nicht verloren gehen, können diese Anonymisiert werden (Dohr, 2021). Wenn Personendaten anonymisiert werden, unterstehen sie nicht mehr dem Datenschutzgesetzt (Eidgenössischer Datenschutz- und Öffentlichkeitsbeauftragter, o. J.-b). Bei einer Anonymisierung von Daten, werden die Personenbezüge der Daten irreversibel aufgeboben, damit keinen Bezug mehr auf die Person hergestellt werden kann. Wenn jedoch mithilfe einer Übersetzungstabelle der Personenbezug wieder hergestellt werden kann, wurden die Daten nur Pseudonymisiert und unterliegen weiterhin dem Datenschutz (Kaufmann, 2021). Gemäss Dohr (Dohr, 2021) gibt verschiedene Methoden zur Anonymisierung von Daten. Eine erste Möglichkeit wäre, die direkten personenbezogenen Daten zu löschen und nur die übrig gebliebenen Daten zu behalten. Eine weitere Methode zur Anonymisierung ist das Clustering. Hierfür werden die Personendaten in vordefinierte Cluster überführt (Dohr, 2021). Als Beispiel können Altersklassen anstatt Geburtsdatum gespeichert werden (Dohr, 2021). Wichtig ist jedoch, dass man nach dem Clustering die Person nicht mehr identifizieren kann (Dohr, 2021). Nach dem Anonymisieren der Daten, können diese weiterhin, auch nach Ablauf der Zweckbindung, für analysezwecke verwendet werden sowie an Dritte weitervergeben (Dohr, 2021).

Das DSG wird derzeit Überarbeit. Am 25. September 2020 hat das Schweizer Parlament die totale Revision entschieden (Fanger, 2020). Die neue DSG wird der europäischen Datenschutzverordnung (GDPR) angeglichen, wird aber weiterhin die eigene Grundkonzeption behalten (Meyerlustenberger Lachenal, 2020). Es wird erwartet, dass das überarbeitete, DSG jedoch nicht vor 2022 in Kraft treten wird.

4 Anforderungen

In diesem Kapitel werden die Anforderungen an die Applikation beschrieben. In einem ersten Schritt wurden die User-Stories definiert. Basierend auf den User-Stories wurden Benutzerabläufe erstellt, damit es verständlicher wird, wie ein User mit der Software interagiert. Die erstellten User-Stories sowie die Benutzerabläufe wurden zusammen mit dem Ersteller des Frontend definiert. Beim Aufstellen der Anforderungen wurde bemerkt, dass die Erstellung eines Glossars notwendig ist, damit Begriffe einheitlich verwendet werden und immer dieselbe Bedeutung haben. Das Glossar ist in Anhang A verfügbar.
In einem letzten Schritt wurden in Kapitel 4.3 die nicht-funktionalen Anforderungen erhoben.

4.1 User-Stories

In diesem Teil werden die definierten User-Stories aufgelistet. Die User-Stories wurden in verschiedene Komponenten aufgeteilt. Diese Komponenten repräsentieren einerseits die verschiedenen Hauptfunktionen der Applikation, andererseits wurden diese Komponenten auch verwenden, um verschiedene Arbeitstakte festzulegen. Damit die User-Stories einheitlich formuliert werden, wurde direkt am Anfang eine einheitliche User-Stories Struktur definiert:

> *«Als [Rolle] möchte ich [Funktion], um [Nutzen]…».*

Um sicherzustellen, dass auch der genannte User einheitlich ist, wurden drei verschiedene Rollen für die Applikation definiert. Diese Rollen werden auch verwendet, um ein einfaches Berechtigungssystem im Office einzubauen.

User: Ein User repräsentiert eine Person, welche die Applikation benutzt.

Office-User: Ein Office-User ist der Standard User in einem Office. Er kann grundlegende Aktivitäten in einem Office ausführen.

Office-Admin: Ein Office-Admin ist ein Administrator eines Office. Er hat alle Berechtigungen in einem Office. Ein Office-Admin ist immer auch ein Office-User.

Jeder User-Story wurde zusätzlich auch eine Priorität zugewiesen, damit sichergestellt werden konnte, dass der Fokus auf der Erstellung der wichtigen Komponenten lag, welche für das MVP unverzichtbar waren.

Benutzer-Verwaltung: Die Benutzer-Verwaltung beinhaltet die Erstellung, Verwaltung und Authentifizierung eines Benutzers.

Tabelle 1 Benutzer-Verwaltung User-Stories

Nr.	Requirement	User-Story	Priorität
VW-50	Registrieren	Als User möchte ich mich bei der Plattform registrieren.	Hoch
VW-36	Login	Als User möchte ich mich einloggen.	Hoch
VW-90	Passwort zurücksetzen	Als User möchte ich mein Passwort zurücksetzen, um wieder Zugriff auf meinen Account zu erhalten.	Hoch
VW-51	«Remember me» Funktion	Als User möchte ich länger eingeloggt bleiben, um mich nicht jedes Mal einloggen zu müssen.	Tief
VW-45	Account Daten editieren	Als User möchte ich meine gespeicherten Daten verändern.	Hoch
VW-53	Account löschen	Als User möchte ich meinen Account löschen.	Mittel

Office-Verwaltung: Die Office-Verwaltung umfasst alle Aktivitäten, welche benötigt werden, ein Office zu erstellen und dieses zu verwalten.

Tabelle 2 Office-Verwaltung User-Stories

Nr.	Requirement	User-Story	Priorität
VW-55	Office erstellen	Als User möchte ich ein neues Office erstellen.	Hoch
VW-57	User verwalten	Als Office-Admin möchte ich User einladen/entfernen.	Hoch
VW-59	Office verändern	Als Office-Admin möchte ich das Office auf die Gegebenheiten meines Unternehmens anpassen.	Mittel

VW-62	Office löschen	Als Office-Admin möchte ich ein Office löschen.	Tief

User-Verwaltung: Die User-Verwaltung enthält die Funktionen, den eigenen User zu individualisieren.

Tabelle 3 User-Verwaltung User-Stories

Nr.	**Requirement**	**User-Story**	**Priorität**
VW-63	Avatar verwalten	Als User möchte ich meinen Avatar anpassen.	Hoch
VW-69	User-Daten verwalten	Als User möchte ich meine dargestellten Daten verändern.	Hoch
VW-68	Badges verwalten	Als User möchte ich meine Badges verwalten.	Mittel
VW-67	Items einsehen	Als User möchte ich meine Items einsehen.	Mittel
VW-66	Items verwalten	Als User möchte ich meine Items verwalten.	Mittel

Office-Ansicht: Die Office-Ansicht gibt eine Übersicht über alle User, welche sich momentan im Office befinden und zeigt ihren aktuellen Standort.

Tabelle 4 Office-Ansicht User-Stories

Nr.	**Requirement**	**User-Story**	**Priorität**
VW-65	Online User anzeigen	Als User möchte ich alle Avatare im Office sehen, welche online sind.	Hoch
VW-64	Aktivitäten ausführen	Als User möchte ich Aktivitäten in verschiedenen Räumen ausführen.	Hoch

Pinnwand: Die Pinnwand dient der Ansicht und Verwaltung der User Quests.

Tabelle 5 Pinnwand User-Stories

Nr.	**Requirement**	**User-Story**	**Priorität**
VW-60	Quest erstellen	Als Office-User möchte ich Quests erstellen.	Hoch

VW-52	Quests ansehen	Als Office-User möchte ich auf alle Quests auf dem Quest-Board zugreifen.	Hoch
VW-58	Quest bearbeiten	Als Office-User möchte ich Quests bearbeiten.	Hoch
VW-56	Quest löschen	Als Office-User möchte ich Quests löschen.	Hoch
VW-54	Quest erledigen	Als Office-User möchte ich Quests erledigen, um eine Belohnung zu erhalten.	Hoch

Market: Der Marktplatz enthält Funktionen, um neue Items für den Avatar zu kaufen.

Tabelle 6 Market User-Stories

Nr.	**Requirement**	**User-Story**	**Priorität**
VW-49	Items ansehen	Als User möchte ich alle kaufbaren Items sehen.	Hoch
VW-44	Items erwerben	Als User möchte ich mit meinen Coins Items erwerben.	Hoch

Leaderboard: Das Leaderboard zeigt User und ihre Ränge auf.

Tabelle 7 Leaderboard User-Stories

Nr.	**Requirement**	**User-Story**	**Priorität**
VW-49	Leaderboards ansehen	Als User möchte ich verschiedene Leaderboards ansehen, um meinen Rang einzusehen.	Hoch
VW-48	Übersicht über Leaderboard	Als User möchte ich eine Übersicht über alle Leaderboards haben.	Tief

4.2 Benutzerabläufe

In diesem Kapitel werden die Benutzerabläufe der Applikation beschrieben. Dafür wurde für jede Aktivität ein Diagramm erstellt, um den ganzen Prozess verständlicher aufzuzeigen.

4.2.1 Registrieren

Der erste Schritt, um die Applikation zu verwenden, ist das Registrieren eines neuen Accounts. Dafür muss, wie in Abbildung 1 dargestellt, der Nutzer die Webseite erst besuchen und das Registrierungsformular muss ausgefüllt werden. Nachdem der Nutzer sich erfolgreich registriert hat, wird dieser zum Login weitergeleitet.

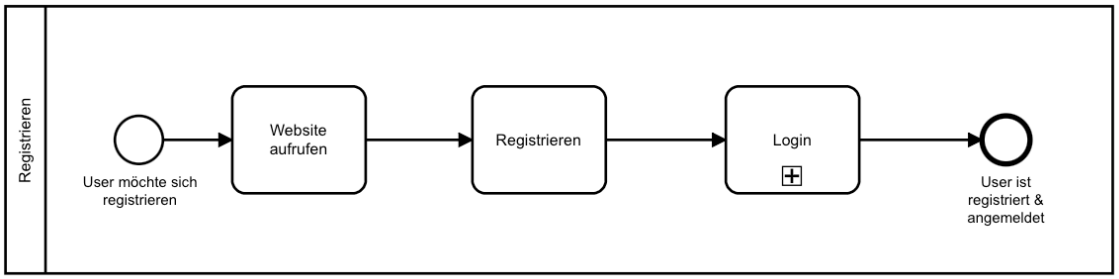

Abbildung 1 Registrieren

4.2.2 Login

Ein bereits registrierter User kann sich mithilfe seiner E-Mail und seinem Passwort einloggen. Nach einem Login wird überprüft, ob sich der User bereits in mindestens einem Office befindet. Sollte dies der Fall sein, wird der User direkt in seine Office-Ansicht weitergeleitet. Wenn ein User noch kein Office besitzt, wird dieser aufgefordert ein neues Office zu erstellen. Dieser Ablauf ist in Kapitel 4.2.4 genauer beschrieben. Nach erfolgreichem erstellen des Office, wird dieser dann auch auf die Office-Ansicht weitergeleitet und der User ist eingeloggt. In Abbildung 2 ist der Login Vorgang abgebildet.

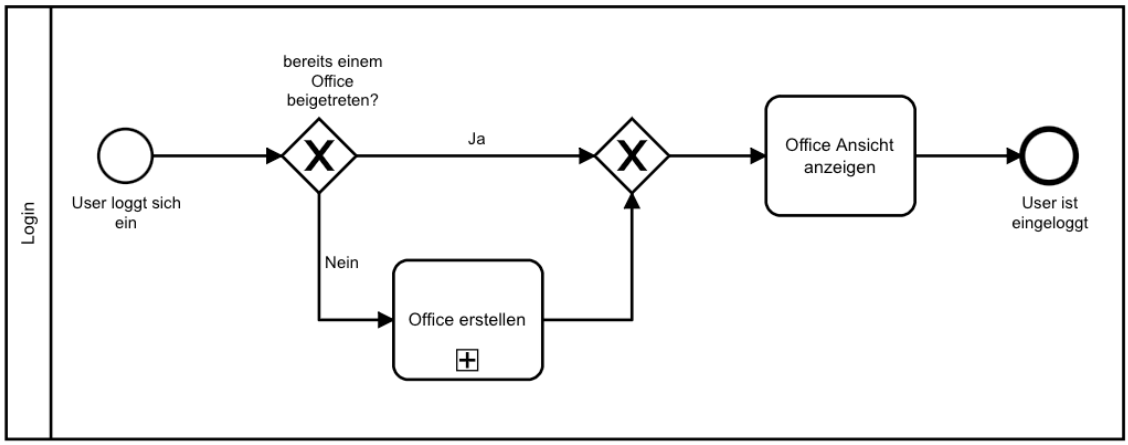

Abbildung 2 Login

4.2.3 Passwort vergessen

In Abbildung 3 wird der «Passwort vergessen» Prozess abgebildet. Gestartet wird der Prozess mit dem Aufrufen der «Passwort vergessen» Seite. Dort wird nach einer E-Mail verlangt, welcher der User eingeben muss. Die Applikation überprüft, ob die angegebene E-Mail im System existiert. Wenn die Überprüfung nicht erfolgreich ist, wird dem Nutzer eine Fehlermeldung angezeigt und der Prozess ist fehlgeschlagen und somit beendet. Bei einer erfolgreichen Überprüfung wird ein zufälliger userspezifischer resetToken generiert. Der User erhält eine E-Mail mit einem Link zum Zurücksetzen seines Passwortes. Der Link enthält eine Query, welche den resetToken enthält. Der User muss den Link in der E-Mail anklicken und wird auf eine Webseite weitergeleitet, auf der er ein neues Passwort festlegen kann. Mit dem Absenden dieses Formulars wird das neue Passwort gespeichert und der Prozess beendet. Der User kann sich nun mit seinem neuen Passwort wieder einloggen.

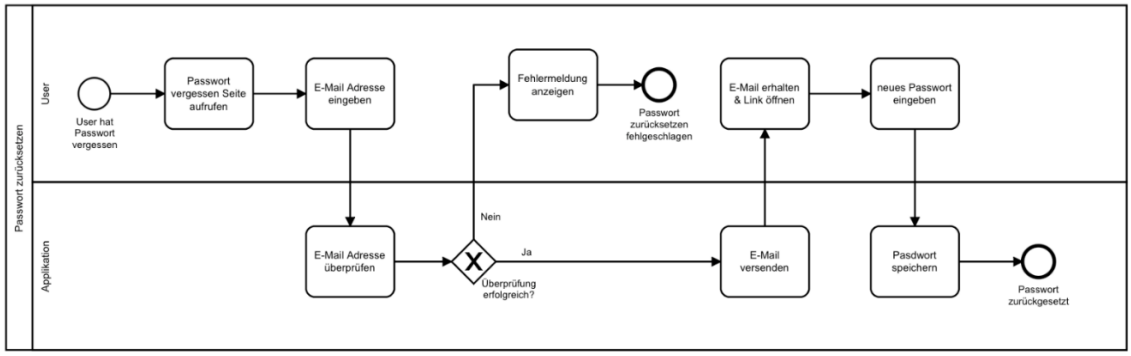

Abbildung 3 Passwort vergessen

4.2.4 Office erstellen

Ein User, welcher ein neues Office erstellen will, muss als erstes einen Namen für das Office eingeben. Dadurch wird ein neues Office von der Applikation erstellt. Der User kann in einem nächsten Schritt seinen neuen Office-User personalisieren. Um den Prozess zu beenden, müssen noch die persönlichen Daten vom User eingetragen werden. Der Prozess ist in Abbildung 4 abgebildet.

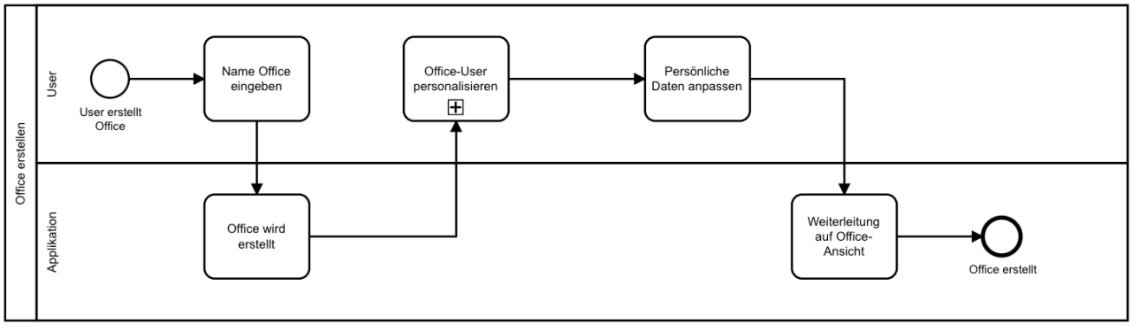

Abbildung 4 Office erstellen

4.2.5 User einladen

In Abbildung 5 wird der Prozess des Einladens von Usern abgebildet. Er beginnt mit dem Anklicken des «Einladen» Button und dem Eingeben der E-Mail der einzuladenden Person. Die Applikation sendet als nächstes eine Einladung an die angegebene E-Mail. Im Falle das die eingeladene Person auf die E-Mail nicht reagiert, endet der Prozess. In der E-Mail befindet sich ein Link zur Webseite. Existiert in der Applikation bereits ein User mit dieser E-Mail wird der User zur Login Seite geleitet. Wenn noch kein User mit dieser E-Mail existiert, auf die Registrieren Seite. Nach dem Login oder der Registrierung wird der inviteToken, welcher sich in einer Query befindet, von der Applikation validiert und wenn die Validierung erfolgreich war, ein neuer User im Office angelegt. Der User wird nach der Validierung auf eine Seite weitergeleitet, um seinen User einzurichten. Dieser Prozess ist in Abbildung 6 abgebildet. Der User wird zuerst aufgefordert seinen Avatar zu personalisieren. Danach muss dieser noch seine persönlichen Informationen angeben und nach dem Abschluss der Konfiguration seines Users, wird dieser in das eingeladene Office weitergeleitet.

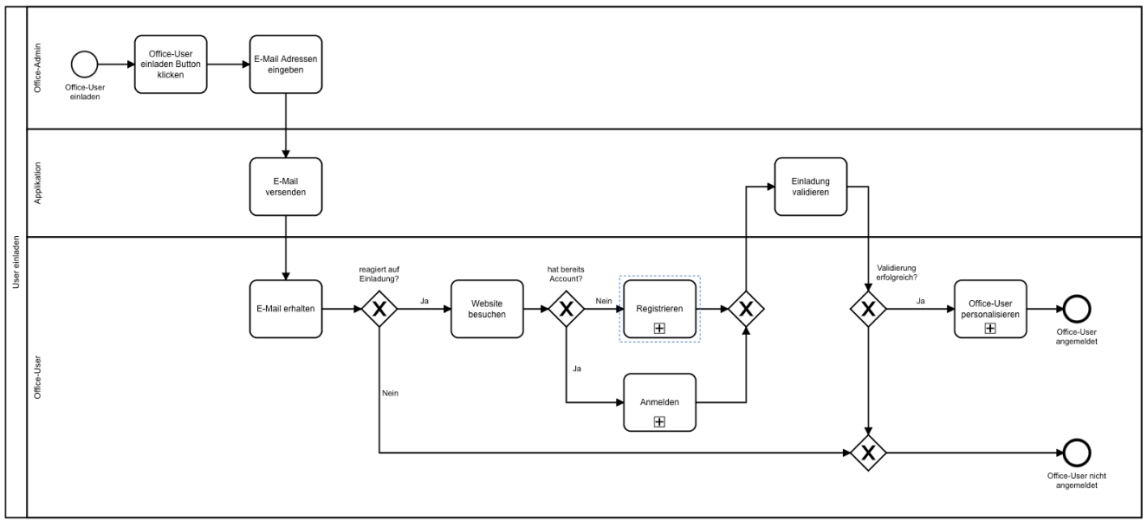

Abbildung 5 User einladen

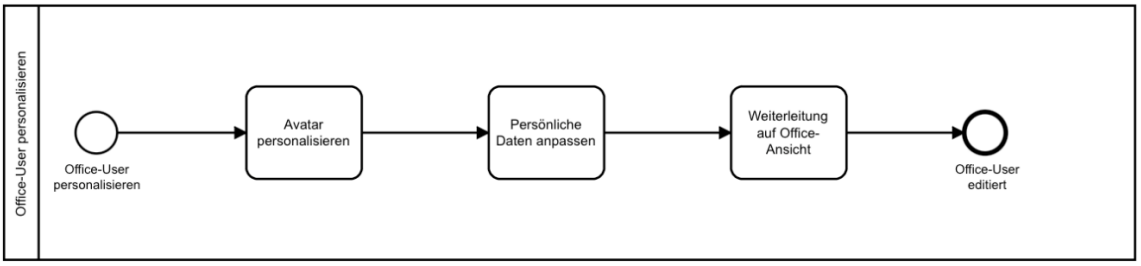

Abbildung 6 User personalisieren

4.2.6 Im Office bewegen

Der Prozess, um sich in einem Office zu bewegen, ist in Abbildung 7 ersichtlich. Der User muss als erstes die Office Ansicht aufrufen. Er wählt den Raum, in welchen er sich bewegen will, und klickt auf die gewünschte Aktivität. Dies führt dazu, dass sich der User in den gewählten Raum bewegt.

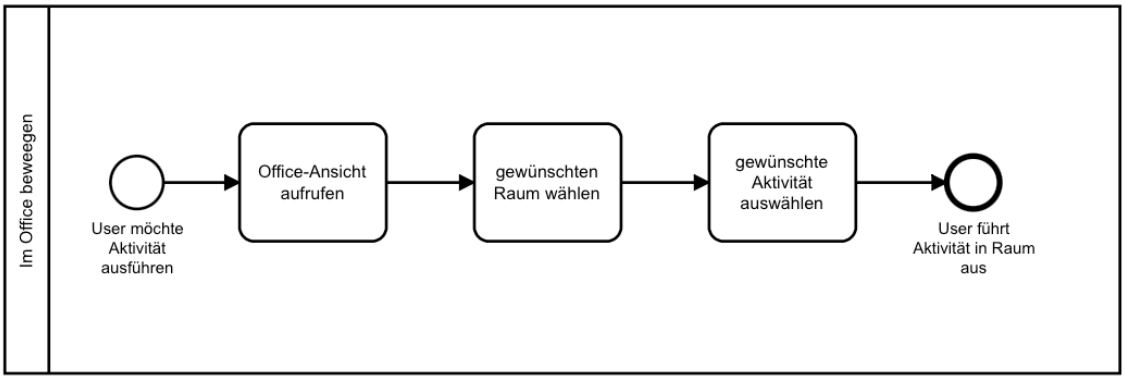

Abbildung 7 Im Office bewegen

4.2.7 Office anpassen

Ein User kann Office Information anpassen, indem er in das Settings Menu navigiert. Dort kann er den Namen des Office, die Raumnamen und die Raumbeschreibungen editieren, wie in Abbildung 8 ersichtlich.

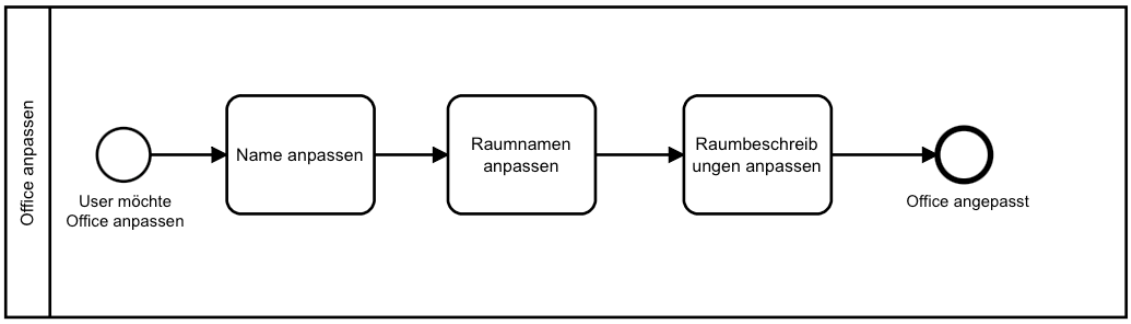

Abbildung 8 Office anpassen

4.2.8 Quest hinzufügen

Abbildung 9 zeigt auf, wie eine Quest erstellt werden kann. Dafür muss der User als erstes den «Quest hinzufügen» Button anklicken. Es öffnet sich ein Modal, in welchem der User die Quest Informationen, wie Titel, Beschreibung, Datum und den Schwierigkeitsgrad, festlegen kann. Als letztes klickt der User auf Quest hinzufügen, um die Quest zu erstellen.

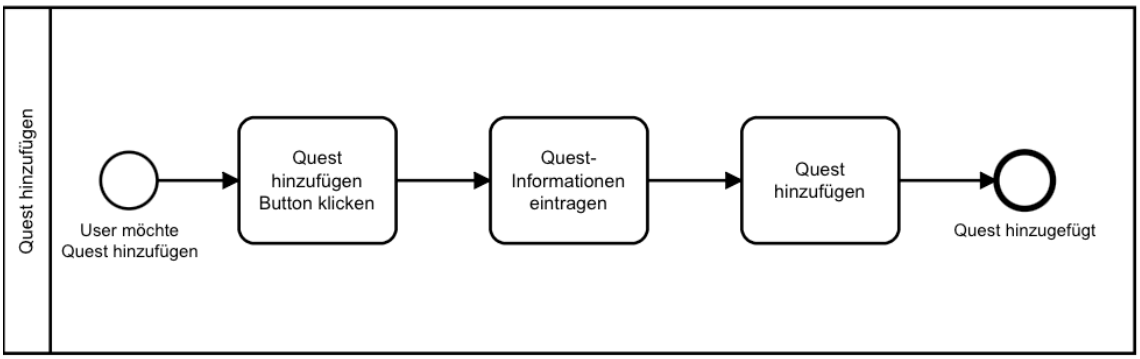

Abbildung 9 Quest hinzufügen

4.2.9 Quest bearbeiten

Um eine Quest zu bearbeiten, muss diese zuerst ausgewählt werden. Ein Modal öffnet sich und die vorhandenen Quest Informationen können angepasst werden. Nach dem Editieren der Daten müssen diese noch gespeichert werden. Der ganze Prozess ist auch in Abbildung 10 ersichtlich.

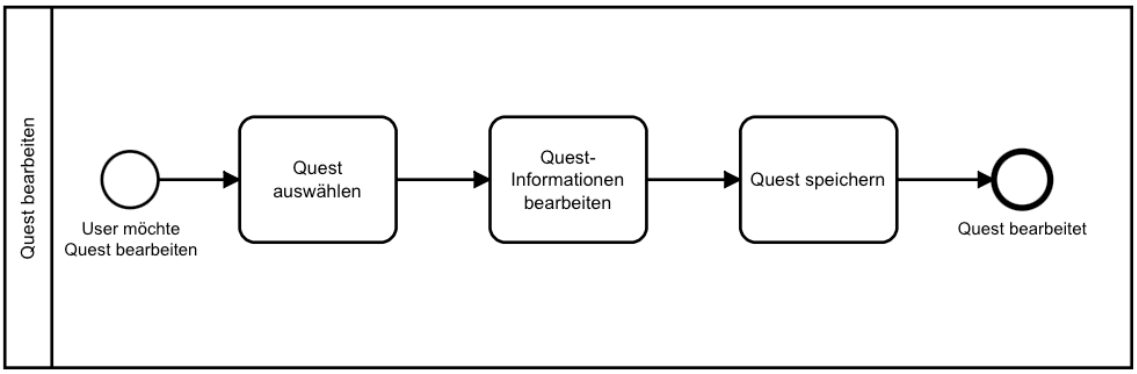

Abbildung 10 Quest bearbeiten

4.2.10 Quest abschliessen

Nachdem ein User die Aufgabe einer Quest erledigt hat, kann dieser die Quest abschliessen. Dazu muss der User die abzuschliessende Quest als erstes auswählen. Dadurch öffnen sich die Quest Informationen, welche auch einen Button zum Abschliessen der Quest beinhaltend. Dieser Button muss vom User gedrückt werden. In einem letzten Schritt wird dem User die Quest Belohnung hinzugefügt. Dieser Prozess ist in Abbildung 11 dargestellt.

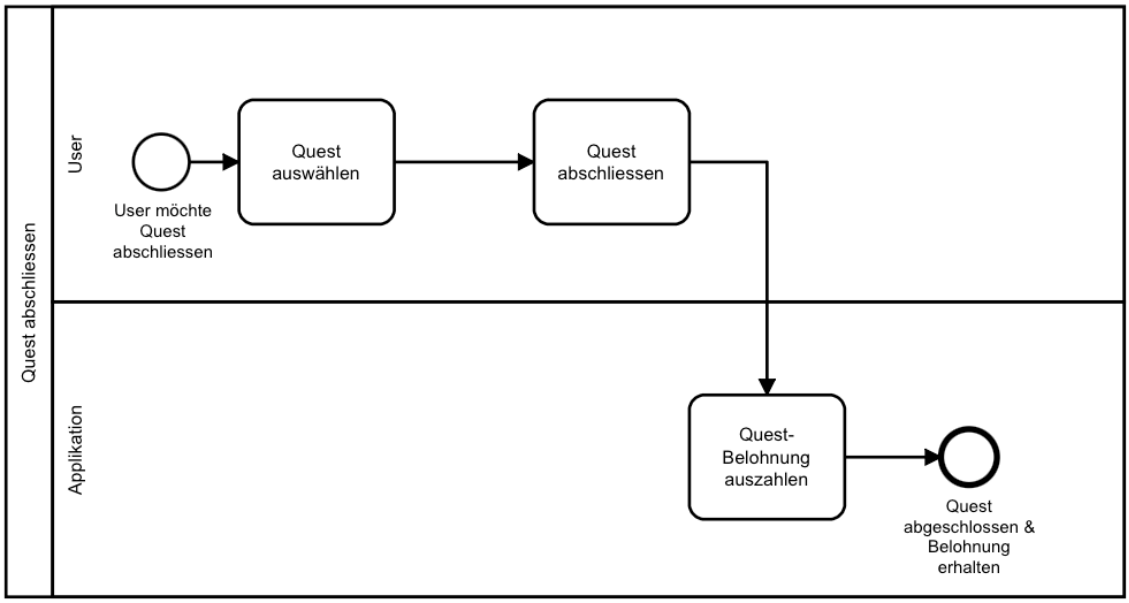

Abbildung 11 Quest abschliessen

4.2.11 Quest löschen

Eine Quest kann jederzeit gelöscht werden, falls sie nicht mehr gebraucht wird. Der Ablauf dazu ist in Abbildung 12 ersichtlich. Nachdem der User eine Quest ausgewählt hat, kann dieser den «Löschen» Button anklicken und die Quest wird gelöscht.

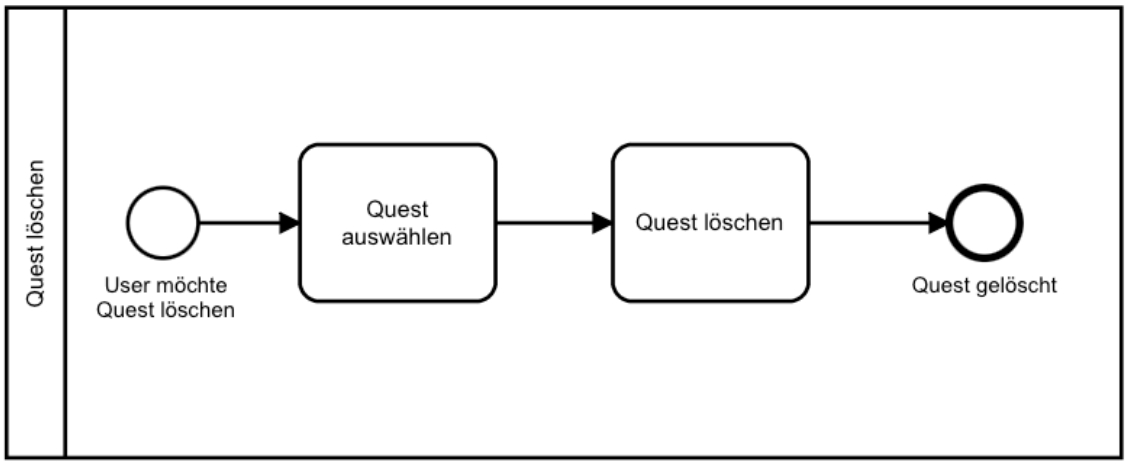

Abbildung 12 Quest löschen

4.2.12 Items kaufen

In Abbildung 13 wird der der Benutzerablauf, um Items zu kaufen, abgebildet. Der Prozess startet mit dem Aufrufen der «Market» Seite. Auf dieser Seite kann der User Items zum Kaufen auswählen. Er startet den Kaufvorgang mit einem Klick auf den «Kaufen» Button. Die Applikation überprüft, ob der User genügend Coins hat, um den Kauf zu tätigen, sowie ob der User die Items bereits besitzt. Sollte die Validierung fehlschlagen wird dem User eine Fehlermeldung angezeigt und der Kaufprozess scheitert. Bei einer erfolgreichen Validierung erhält der User die neu gekauften Items und es wird ihm eine Erfolgsnachricht angezeigt.

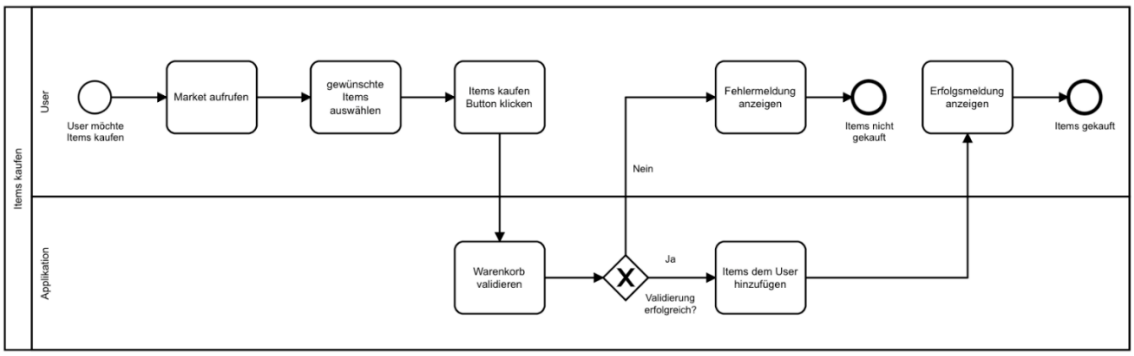

Abbildung 13 Items kaufen.

4.2.13 Daten verändern

Damit ein User seine Daten verändern kann, muss dieser zuerst auf die Settings Seite navigieren. Auf dieser Seite hat er die Möglichkeit, seine Benutzerdaten zu editieren. Nachdem er die editierten Daten abgespeichert hat, ist der Prozess beendet. Der ganze Prozess ist in Abbildung 14 ersichtlich.

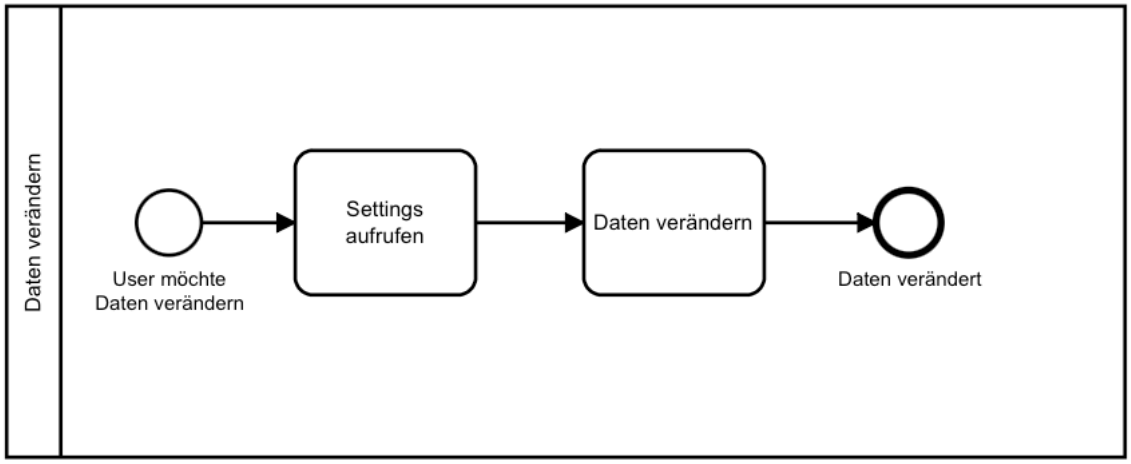

Abbildung 14 Daten verändern

4.3 Nicht-funktionale Anforderungen

In Tabelle 8 sind die definierten nicht-funktionalen Anforderungen erfasst.

Tabelle 8 Nicht-funktionale Anforderungen

Nr.	**Anforderung**
NFA-1	Das MVP soll online für jeden erreichbar sein.
NFA-2	Die Applikation soll verständlich sein und leicht zu bedienen.
NFA-3	Anfragen an den API Gateway sollen in angemessener Zeit eine Rückmeldung erhalten.

5 System-Dokumentation

In diesem Kapitel wird beschrieben, wie die Software aufgebaut wurde, welche zur Lösung des Problems führen soll. In Abschnitt 5.1 wird die Systemarchitektur erklärt. Es wird aufgezeigt, welche Technologien eingesetzt werden und wie diese miteinander interagieren. Anschliessend wird der API-Gateway in Kapitel 5.2 und die Microservices in Kapitel 5.3 genauer beschrieben. Zum Abschluss dieses Kapitel werden in Abschnitt 5.4 noch die verwendeten DevOps Praktiken vorgestellt.

5.1 Systemarchitektur

Beim Erstellen der Systemarchitektur wurde als erstes festgelegt, dass die Applikation als eine Webapplikation umgesetzt wird. Der Frontend soll mithilfe des Vue Framework erstellt und später mithilfe von nginx bereitgestellt werden.

Beim Backend wurde sich für eine Microservice Architektur entschieden. Dies hat den Vorteil das die einzelnen Aufgaben auf eigene Services aufgeteilt sind und so eine Seperation of Concerns entsteht. Die einzelnen Microservices verfügen über ihre eigene Datenbank und speichern dort die Daten, welche sie verwalten. Damit Daten nicht redundant gespeichert werden, kommunizieren die Microservices, wenn nötig, um an Daten zu gelangen, für welche sie selbst nicht zuständig sind. In Abbildung 15 ist die ganze Systemarchitektur abgebildet.

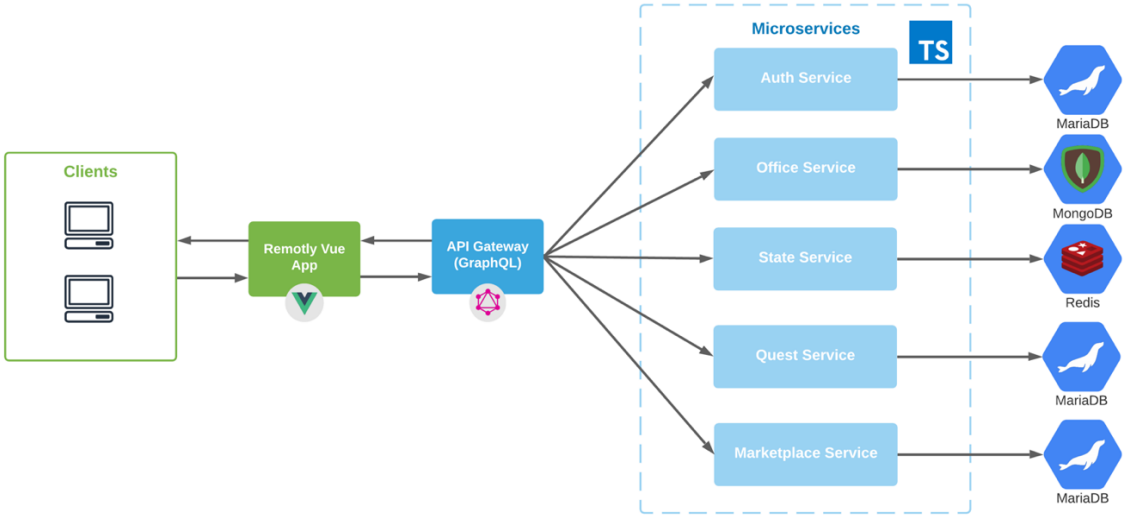

Abbildung 15 Systemarchitektur

Als Schnittstelle zwischen dem Frontend und den Microservices gibt es einen API Gateway. Der API Gateway veröffentlich einen GraphQL Endpunkt, über welchen der Frontend Anfragen senden kann, um an Daten zu gelangen und diese zu editieren.

Der ganze Software Stack wird in einem Kubernetes Cluster in Docker Containern bereitgestellt. Dafür wurde ein Kubernetes Cluster in der Hetzner Cloud eingerichtet. Für das Erstellen des Kubernetes Cluster wurde Terraform verwendet. Terraform ist eine Infrastruktur als Code Software, welche es ermöglicht mithilfe von Konfigurationsdateien Infrastruktur zu erstellen oder editieren. Der Frontend ist unter https://remotly.ch erreichbar. Der einkommende Traffic wird von einem Loadbalancer entgegengenommen und an die entsprechenden Systeme im Cluster weitergeleitet. Die GraphQL Schnittstelle des API Gateway ist unter https://api.remotly.ch erreichbar und der Websocket unter wss://api.remotly.ch. Alle Applikationen im Cluster befinden sich in einem privaten Netzwerk und sind nur über den Loadbalancer erreichbar. Dadurch können die Microservices nicht von ausserhalb aufgerufen werden und somit kann nur der API Gateway mit den Microservices kommunizieren.

Für das Zwischenspeichern von Dateien und für die Verbesserung der Performance und Sicherheit wurde Cloudflare vor den Loadbalancer geschaltet. Dadurch wird zwischen dem Client und Cloudflare ein SSL Zertifikat von Cloudflare verwendet und zwischen den Cloudflare und dem Loadbalancer das Zertifikat des Kubernetes Cluster.

Für die Authentifizierung von Usern wird ein Stateless Ansatz mithilfe von Json Web Tokens (JWT) angewendet. Für authentifizierte Anfragen muss der Client immer einen Auth Header mit einem gültigen JWT mitsenden. Der API Gateway nimmt diesen Header entgegen und übernimmt ihn, um weitere Anfragen an die Microservices zu senden. Die Microservices selbst, sind dafür verantwortlich den JWT auf seine Gültigkeit zu Prüfen.
Für das Erstellen des MVP ist dieser Ansatz ausreichend. Für die Erstellung einer produktionsfertigen Version, sollte die Authentifizierung mittels OAuth2 umgesetzt werden und ein JWT sollte nur noch im internen System für die Authentifizierung verwendet werden.

5.2 API Gateway

Der API Gateway dient als Schnittstelle zwischen dem Frontend und den Microservices. Der API Gateway ist ein GraphQL Server, welcher es dem Frontend ermöglicht GraphQL Anfragen zu senden. GraphQL ist eine Sprache für die Abfrage und Manipulation von Daten. Sie ermöglicht es Datenstrukturen mithilfe Typen in Schemas zu beschreiben. Beim Senden einer GraphQL Anfrage kann der Client genau angeben welche Daten er benötigt und es werden nur diese Daten zurückgegeben. Dadurch kann die Anzahl der gesendeten Daten auf das Minimum reduziert werden. GraphQL ermöglicht es somit auch auf mehrere Ressourcen mit nur einem Request zuzugreifen. Im Falle der hier beschriebenen Applikation können somit Daten von mehreren verschiedenen Microservices in nur einem Request aufgerufen oder verändert werden.

Mithilfe von Schemas und Resolvern wird im API Gateway definiert, welche Daten er zur Verfügung stellt und von wo er diese bekommt. In Schemas werden die verschiedenen Queries und Mutations, welche zur Verfügung stehen, definiert. Zudem wird angegeben, welche Daten von einer Query angefordert werden können und welchen Typen diese Daten haben. Die standartmässigen Typen sind Strings, Integers, Floats, Boolean und ID. Für die Applikation wurde noch ein Typ DateTime gebraucht. Deshalb wurde dieser speziell als eigener Datentyp hinzugefügt. Damit die Übergabe des Datums einheitlich läuft und keine Probleme bereitet, wurde entschieden das Daten immer nach der ISO-8601 Norm übergeben werden müssen. Die Resolver legen fest, wie auf die im Schema definierten Daten zugegriffen werden kann. Dies kann als Beispiel über direkte Datenbankabfragen oder durch Aufrufen von REST Servern geschehen. Beim Aufrufen von REST Servern wird auch der Cache Header beachtet. Sollte die Response einen enthalten, speichert der GraphQL Server diesen lokal, um in Zukunft eine schnellere Responsezeit zu liefern. Eine weitere Aufgabe von Resolvern ist die Formatierung der erhaltenen Daten auf die Struktur des Schemas, wenn dies nötig sein sollte.

GraphQL bietet neben Queries und Mutations auch die Möglichkeit Subscriptions zu erstellen. Der API Gateway ist somit auch für das Abhandeln von Subscriptions zuständig, welche auf Websockets basieren. Subscriptions werden wie Queries und Mutations in Schemas definiert und haben einen Resolver, welcher für das Abhandeln der Daten zuständig ist. Clients haben die Möglichkeit eine Subscriptions zu erstellen und der API Gateway kann dadurch Informationen an den Client senden, wenn sich beispielsweise Informationen ändern. Somit kann der Client diese in Echtzeit aktualisieren.

5.3 Microservices

Alle Microservices sind einheitlich aufgebaut und haben dieselbe Struktur. Die Microservices können über eine REST API aufgerufen werden. Sie wurden mithilfe von Typescript programmiert und bauen alle auf dem Express.js Framework auf. Express.js ist ein minimales und flexibles web Framework. Die Microservices verwenden ihre eigene Datenbank und sind grösstenteils unabhängig voneinander. Deshalb können einzelne Microservice auch einfach ersetzt werden und wenn nötig in einer beliebigen anderen Programmiersprache entwickelt werden.

In Kapitel 5.3.1 wird die Struktur eines Microservice genauer aufgezeigt und in Kapitel 5.3.2 werden die einheitlichen Komponenten der Microservices vorgestellt. In den folgenden Kapiteln wird dann auf die eigentlichen Microservices eingegangen und ihre Besonderheiten und Aufgaben hervorgehoben.

5.3.1 Struktur eines Microservice

Die Struktur eines Microservices ist immer gleich aufgebaut und unterscheidet sich nur an den unterschiedlichen Dokumenten der dazugehörigen Datenbanken. In Abbildung 16 ist die Ordnerstruktur des Auth Service zu sehen.

Der «logs» Ordner beinhaltet die generierten Log Dateien.
Im Ordner «src» befinden sich alle Code Dateien. Er ist unterteilt in fünf weitere Ordner.
Der «Components» Ordner beinhaltet weitere Ordner, welche den einzelnen Endpunkten zugeordnet werden können. Diese beinhalten eine Controller Datei, welche die Funktionen der einkommenden Anfragen behandelt. Die «routes» Datei definiert die URL Pfade des Endpunktes. In der «interface» Datei werden die benötigten Interfaces und Types für den Endpunkt definiert und die Validierungsdatei beinhaltend die Request Validierungsfunktionen.
Konfigurationsdateien werden im «config» Ordern abgespeichert.
Der Middleware Ordner enthält Dateien welche Anfragen beeinflussen. Darunter befinden sich Error, Authentifizierung und logging Handler.
Im «utils» Ordner sind Helferklassen zu finden, welche an verschiedenen Orten im Code benötigt werden.

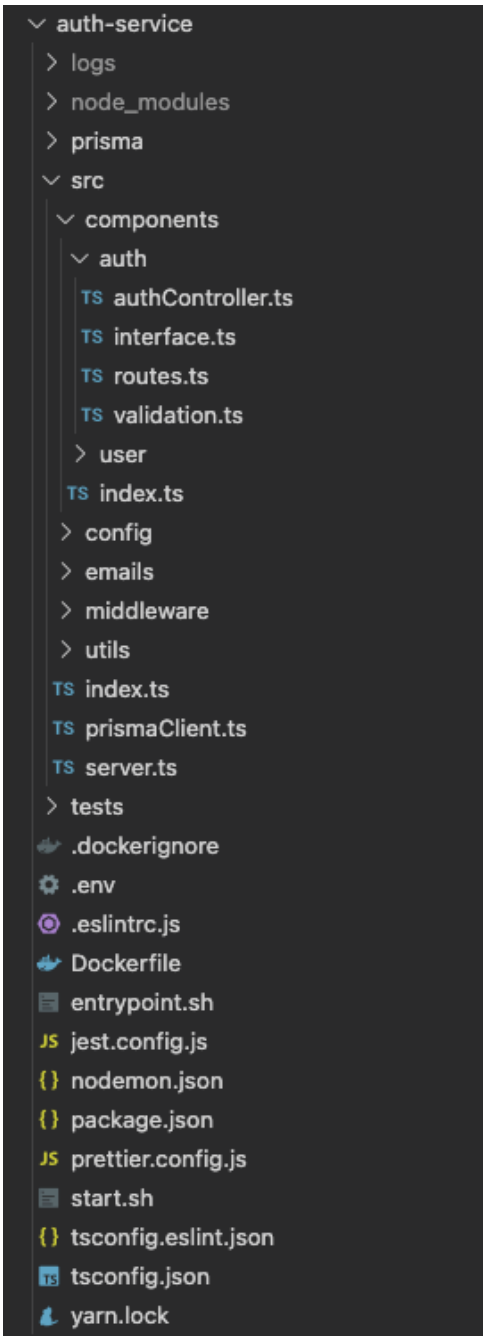

Abbildung 16 Struktur eines Microservice

5.3.2 Einheitliche Komponenten

In diesem Teil werden die einheitlichen Komponenten der Microservices vorgestellt. Diese Komponenten sind in allen Microservices zu finden und auf die einzelnen Microservices angepasst.

5.3.2.1 Logging

Für das Logging wurde ein eigener Logger mithilfe des Packages «winston» erstellt. Dafür wurden eigene Log Levels wie «error», «warn» und «info» definiert. Die auszugebende Nachricht wurde mit einer aktuellen Zeitangabe ergänzt und es wurden zwei Logfiles, eine für «error» Meldungen und eine für alle Meldungen, festgelegt. Dadurch werden die Log Nachrichten einerseits in der Konsole ausgegeben, andererseits auch in die Logfiles geschrieben. Die REST Endpoints wurden so definiert, dass auch alle ankommenden Anfragen geloggt werden. Diese Methode, einen eigenen benutzerdefinierten Logger zu erstellen, hilft beim Debugging der Applikation in der Entwicklung, aber auch wenn diese in einer produktiven Umgebung betrieben wird.

5.3.2.2 Http Errors & Error Codes

Um von Anfang an mit fehlerhaften Anfragen einheitlich umzugehen, wurde verschiedenen Fehlern ein http Error Code zugewiesen.

401 Unauthorized: Der «401» Error wird verwendet, wenn Anfragen eingehen, welche einen JWT benötigen, aber keinen JWT im Request Header enthalten.

403 Forbidden: Wenn ein Request versucht Daten zu verändern, der Nutzer jedoch keine Berechtigung dazu hat, wird ein «403» Error verwendet.

404 Not Found: Der «404» Code wird verwendet, wenn Ressourcen angefordert werden, welche nicht existieren. Er wird auch eingesetzt, wenn ein User eine Ressource anfordert, auf welche er keinen Zugriff hat, aber auch nicht wissen soll, dass diese Ressource existiert.

422 Unprocessable Entity: Ein «422» Error wird ausgelöst, wenn der einkommende Request verstanden wird, aber die übergebenen Daten nicht den Erwartungen entsprechen. Damit im Frontend genauer angezeigt werden kann, welche Daten ungültig sind, wird der Response auch eine detaillierte Fehlernachricht angehängt. Diese besteht aus einem Array, welche alle fehlerhaften Daten, in Form eines Objektes, enthält. Dieses Objekt besteht aus drei Feldern: die angeforderte Ressource, das zu verändernde Feld und ein Error Code, welcher über den genauen Fehler informiert. In Abbildung 17 ist zu sehen, wie ein «422» Error zurückgegeben wird, wenn beim Erstellen einer Einladung die angegebene E-Mail bereits eingeladen ist.

```
// Check if email is already invited
if (office.invites.some((x) => x.email === body.email)) {
  return Promise.reject(
    new HTTP422Error({
      message: 'Validation Failed',
      errors: [{ resource: 'Office', field: 'invites', code: ErrorCode.ALREADY_EXIST }],
    }),
  );
}
```

Abbildung 17 Http «422» Error Beispiel

5.3.2.3 Validierung von Daten

Bevor die eingehenden Anfragen der bearbeitenden Funktion übergeben werden, wird eine pfadspezifische Validierungsfunktion aufgerufen. In der Validierungsfunktion wird überprüft, ob der «body» der Anfrage die benötigten Werte enthält und ob diese auch im richtigen Format ankommen. Sollte dies nicht der Fall sein, wird ein Validierungsfehler zurückgegeben. Dadurch wird sichergestellt, dass die einkommenden Anfragen fehlerfrei sind.

5.3.2.4 Linting

ESLint wird verwendet, um Codeanalysen durchzuführen. Neben den Standardregeln von ESLint wird auch der «Airbnb JavaScript Style Guide» verwendet. Dadurch kann sichergestellt werden, dass der Code den Best Practices folgt, keine offensichtlichen Fehler enthält und einheitlich gestaltet wird. Visual Studio Code enthält eine ESLint Integration, welche es ermöglich, Feedback in Echtzeit zu erhalten und Funktionen wie «lint on save» verfügt.

5.3.2.5 Authentifizierung

Für die Authentifizierung von Anfragen wird überprüft, ob der Request einen JWT im Auth Header enthält. Der JWT wird ausgelesen und auf seine Gültigkeit überprüft. Sollte der JWT nicht vorhanden oder ungültig sein, wird ein «401» Error zurückgegeben. Der JWT enthält zusätzliche Informationen über den User, welcher die Anfrage sendet. Diese werden dem Request angehängt, damit sie später beim Ausführen der Funktion verwendet werden können.

5.3.2.6 REST Pfade Dokumentation

Alle REST Pfade der Microservices wurden mithilfe von Swagger dokumentiert. Die Dokumentation der Pfade wurde direkt in den «routes» Dateien hinterlegt. Swagger durchsucht alle hinterlegten Orte nach mit «@Swagger» gekennzeichneten Dokumentationen. Diese können dann in einer formatierten Version auf einer lokalen Webseite betrachtet werden. Diese Möglichkeit bietet den Vorteil, dass die Pfaddokumentation direkt im Code erstellt beziehungsweise angepasst werden kann.

5.3.2.7 Dockerfile

Das Dockerfile unterteilt sich in einen Development und einen Staging Teil. Der Development Teil ist für die Erstellung des Docker Images für die Entwicklungsumgebung zuständig. Dementsprechend ist der Staging Teil für die Erstellung des Staging Images verantwortlich. In den beiden Teilen werden auch die benötigten Environment Variablen gesetzt.

5.3.2.8 Testing

Zum Testen der Applikation werden Functional Tests verwendet. Dazu werden in den Testdateien Anfragen an den Microservice definiert. Jeder REST Pfad soll mit gültigen und ungültigen Daten aufgerufen werden, um festzustellen ob die erwarteten Werte zurückkommen. Beim Testen wird der Microservice, sowie die weiteren benötigten Applikationen, wie Datenbanken oder andere Microservices, gestartet. Das Ziel dabei ist es, sicherzustellen, dass durch Updates keine Fehler auftreten und der API Gateway immer die erwarteten Daten erhält.

5.3.2.9 Prisma

Das Package «prisma» wird für die Interaktion mit den SQL-Datenbanken verwendet. Prisma bietet die Möglichkeit, Datenbanken Schemas in Dateien zu erstellen, welche mithilfe eines Migration Command angewendet werden können. Jede Änderung am Datenbank Schema wird mit dem Ausführen des Migration Command an der Datenbank vorgenommen und die Migrationen werden in einem Log gespeichert. Dies bietet den Vorteil, dass die Datenbanken problemlos angepasst werden können und bei einem Update können die gespeicherten Migrationen automatisch auf neuen Systemen angepasst werden. Somit muss bei einem Update nicht jede Datenbank von Hand angepasst werden und es wird sichergestellt, dass die Datenbanken die gleichen Strukturen haben.

5.3.3 Auth Service

Der Auth Service ist für die Verwaltung und Authentifizierung der Accounts zuständig. Dazu hat der Auth Service eine User Kategorie, welche Funktion zum Erstellen, Aufrufen und Bearbeiten von Accounts zur Verfügung stellt. Zusätzlich steht auch eine Funktion zum Zurücksetzen des Passworts in dieser Kategorie zu Verfügung. In Abbildung 18 ist die REST API des Auth Services abgebildet. Für die Authentifizierung von Accounts gibt es einen Login Pfad in der Auth Kategorie. Nach dem Login wird dem User ein JWT zugesendet, welchen er in zukünftigen Anfragen als Auth Header anhängen muss.

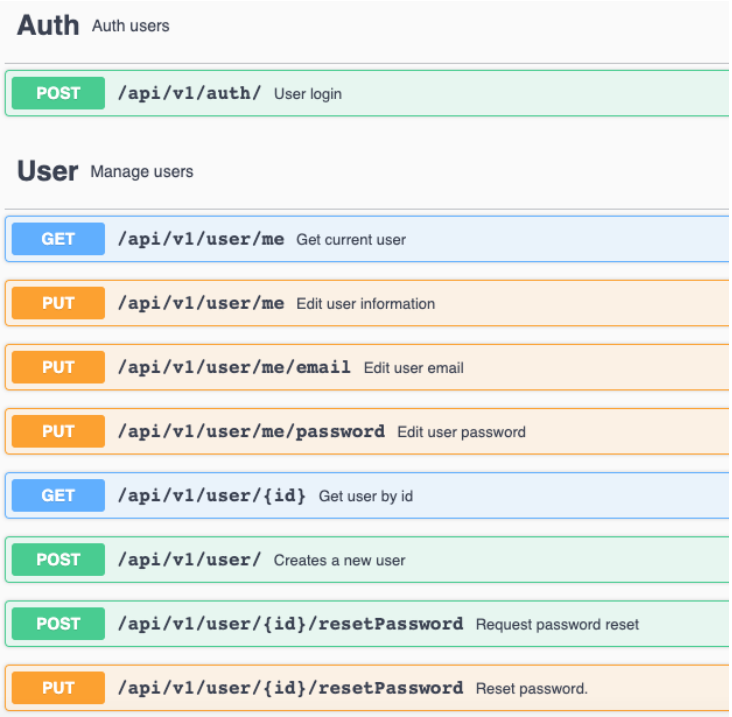

Abbildung 18 Auth Service REST API

Zum Speichern der Daten des Auth Service wird eine rationale Datenbank verwendet, da die zu speichernden Daten strukturiert sind. Als Datenbank kommt MariaDB zum Einsatz. Die Struktur der Daten in der Datenbank ist in Abbildung 19 ersichtlich.

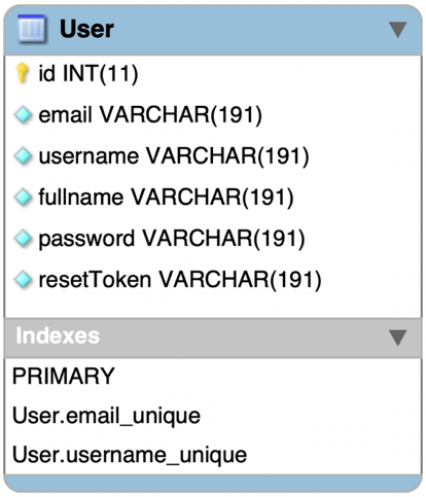

Abbildung 19 Auth Service Datenstruktur

5.3.4 Office Service

Der Office Service verwaltet Offices, Office-Users und die Leaderboards. Die Office Komponente, sowie die Office-User Komponente, enthalten grössere unstrukturierte und verschachtelte Datenstrukturen. Deshalb wird im Office Service im Gegensatz zu den meisten anderen Microservices die NoSQL Datenbank MongoDB eingesetzt. Für die Interaktion mit der Datenbank wird im Office Service das Package «mongoose» verwendet. Mongoose ermöglicht es, Schemas der zu speichernden Daten zu erstellen. Dies ist hilfreich, um einen besseren Überblick über die Datenstruktur zu erhalten. Zudem hilft Mongoose auch mit der Serialisierung der Daten. Es wurden für das Speichern von Offices und von Office-Usern jeweils ein Schema angelegt. In Abbildung 20 ist die Datenstruktur der beiden Schemas ersichtlich.

Der Office Service ist in drei Komponenten unterteilbar. In Abbildung 21 sind die API-Pfade dargestellt.

Office: Die Office Komponente ist für die Verwaltung der Offices zuständig. Neben den standardmässigen CRUD Operationen eines Office beinhaltet diese auch Funktionen zum Erstellen und Annehmen von neuen Office-User Einladungen, sowie Optionen für die Verwaltung der Räume in einem Office.

Office-User: Die Office-User Komponente verwaltet die User in einem Office. Es besteht die Möglichkeit User anzufordern, zu editieren oder zu löschen. Wenn ein User gelöscht wird, wird die «accountID» sowie die Relation zum Office gelöscht. Somit sind

die Informationen des Users anonymisiert. Die restlichen Daten werden nicht gelöscht und können zu einem späteren Zeitpunkt für analytische Zwecke verwendet werden.

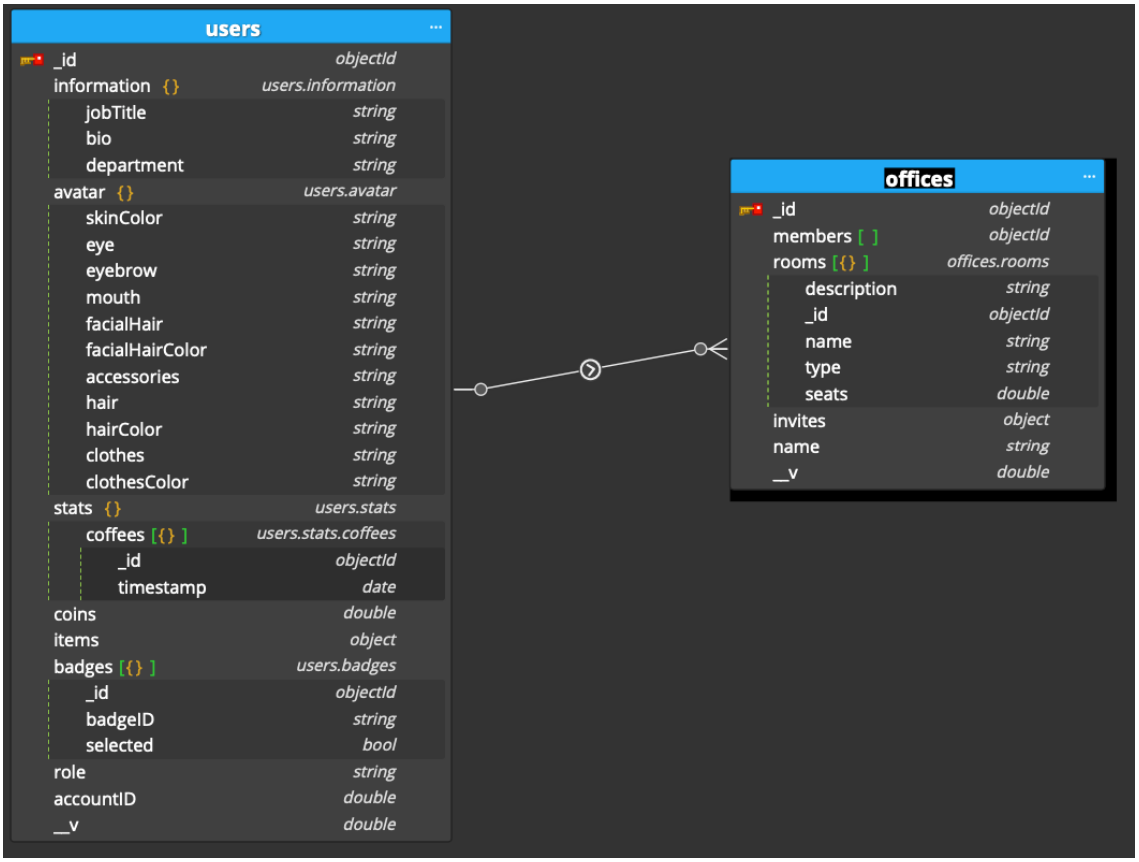

Abbildung 20 Office Service Datenstruktur

Leaderboard: Die Leaderboard Komponente enthält zwei GET Pfade, um die beiden Leaderboards «Coins» und «Coffee-Breaks» abzurufen. Beim Abrufen des Coffee-Break Leaderboards kann ein Queryparameter «period» mitgegeben werden. Dieser kann den Wert «permanent» haben, um das Leaderboard des gesamten Zeitraumes aufzurufen oder den Wert «weekly», um das wöchentliche Leaderboard zu erhalten. Das Coins Leaderboard kann nur permanent abgerufen werden.

Der Response eines Leaderboard Requests wird immer ein «Cache-control» Header mit dem Inhalt «public, max-age=60» gesetzt. Dadurch weiss der API-Gateway, dass er diesen Request im Cache speichern soll. Somit kann sichergestellt werden, dass Leaderboard Requests nur maximal einmal pro Minute ausgeführt werden, da diese Anfragen durch die Berechnungen und das Sortieren ressourcenaufwendiger sind.

Abbildung 21 Office Service REST API

5.3.5 State Service

Im State Service werden die aktuellen User eines Office, welche gerade online sind, zwischengespeichert. Der State Service ist somit zuständig, Information zu überliefern, welche User sich gerade in einem Office befinden, in welchem Raum sie sich aufhalten und was ihr aktueller Status ist. Da sich diese Informationen ständig ändern und nicht langfristig abgespeichert werden sollen, wird dafür die In-Memory-Datenbank Redis verwendet. Ein weiterer Grund eine In-Memory-Datenbank zu verwenden, ist die Sicherstellung, dass Mitarbeiter nicht langfristig überwacht werden können.
Der State Service bietet Schnittstellen, um alle Online-User eines Office zu erhalten, die Position eines Users zu verändern, sowie das Entfernen eines Users, sobald dieser offline geht. Eine genaue Übersicht der REST API des State Service ist in Abbildung 22 aufgeführt.

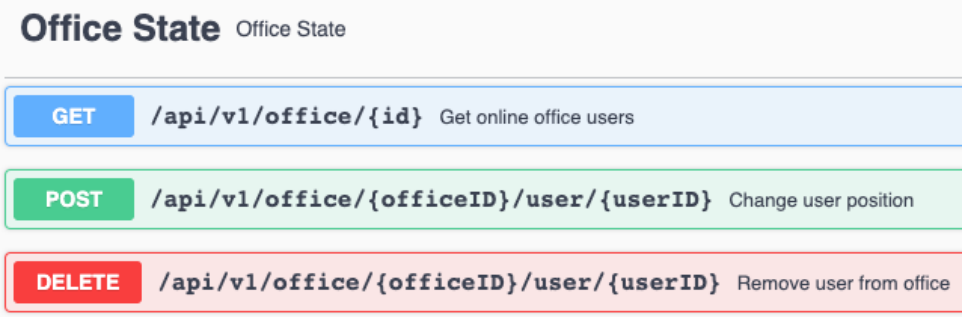

Abbildung 22 State Service REST API

5.3.6 Market Service

Der Market Service verfügt über zwei Endpunkte, wie in Abbildung 23 ersichtlich. Der GET Request ist zuständig, um eine Liste mit allen kaufbaren Items und deren Preisen zu erhalten. Für das Speichern der Daten wird wieder MariaDB verwendet. In Abbildung 24 ist die Datenstruktur der Market DB aufgeführt. Der zweite Request des Market Service wird verwendet, um neue Items zu kaufen. Dazu kommuniziert der Market Service mit dem Office Service, um abzugleichen, ob der Office-User genügend Coins hat und um danach dem Office-User die Items hinzuzufügen und die Coins abzuziehen.

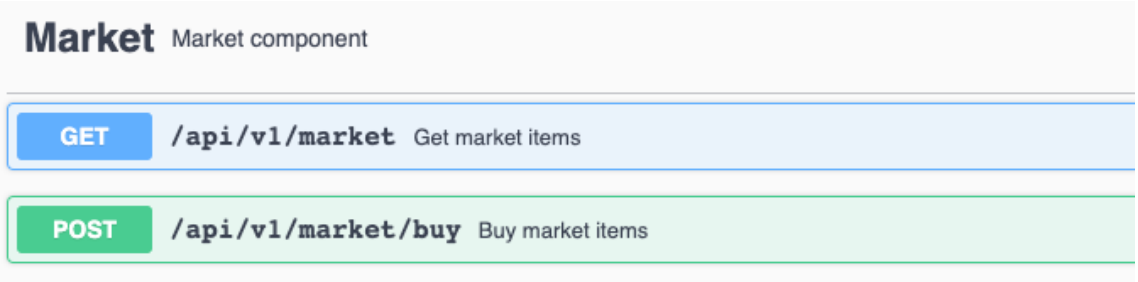

Abbildung 23 Market Service REST API

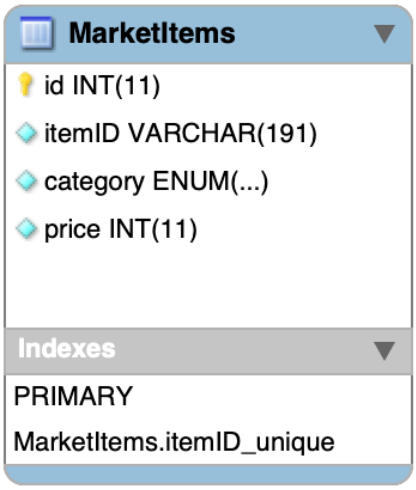

Abbildung 24 Market Service Datenstruktur

5.3.7 Quest Service

Der Quest Service ist zuständig für die Abwicklung von Quests. Er verfügt über Pfade, um Quests aufzurufen, zu erstellen, zu editieren, abzuschliessen und zu löschen. Die genannten Pfade sind in Abbildung 25 ersichtlich.

Quests können in drei verschiedenen Schwierigkeitsstufen erstellt werden. Je nach Schwierigkeitsstufe erhält der Nutzer eine unterschiedliche Anzahl an Coins für das Abschliessen einer Quest. Für einfache Quests sind es 50, für mittlere Quests 100 und für schwere Quests gibt es 200 Coins.

Damit der User die Coins gutgeschrieben bekommt, kommuniziert der Quest Service mit dem Office Service über die REST Schnittstelle.

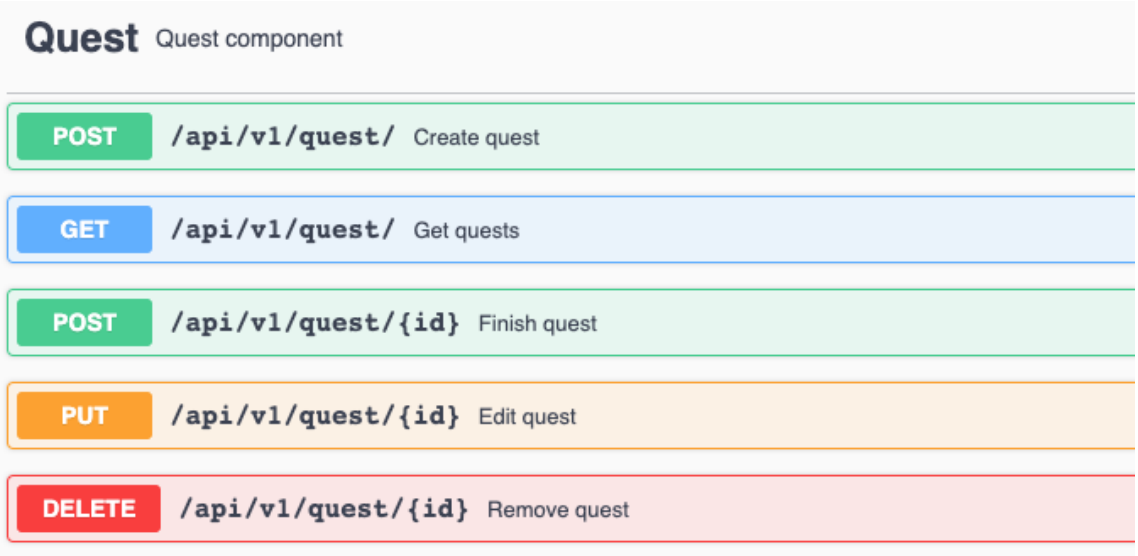

Abbildung 25 Quest Service REST API

Für das Abspeichern der Daten verwendet der Quest Service auch MariaDB. Die Datenstruktur des Quest Service ist in Abbildung 26 ersichtlich. Für das Feld «difficulty» wurde ein Enum mit den Werten «EASY», «MEDIUM» und «HARD» erstellt.

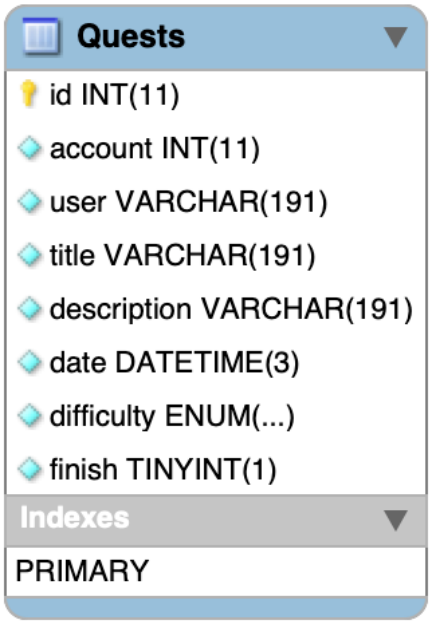

Abbildung 26 Quest Service Datenstruktur

5.4 DevOps

Beim Aufsetzen der DevOps Pipeline wurde darauf geachtet, das am Frontend, sowie am Backend, einwandfrei gearbeitet werden kann, ohne dass eine Abhängigkeit der beiden Komponenten besteht. Für die Versionsverwaltung des Codes wird Git verwendet. Der Code wird online in einer Repository auf Github gespeichert. Der Frontend sowie das Backend sind zusammen in einer Monorepository gespeichert. Das Repository hat zwei Branches. Einerseits den «master» Branch, welcher den Code des Development Environments enthält und andererseits einen «staging» Branch, welcher den Code des Staging Environments enthält. Für den Aufbau einer CI/CD Pipeline wurde Github Actions verwendet. Die darin erstellten Docker Images werden auf Docker Hub gespeichert.

5.4.1 Environments

Für die Entwicklung des Prototyps wurden zwei Environments verwendet. Ein Development Environment für die Entwicklung und ein Staging Environment für die Validierung des Prototyps. Damit die gleiche Codebase für beide Environments benutzt werden konnte, wurden Environment Variablen verwendet, um sich ändernde Daten, wie URLs, zu speichern.

Development Environment: Das Development Environment wird verwendet, um die Applikation auf dem lokalen Computer zu entwickeln. Für den Frontend, sowie die Microservices im Backend, wurden Dockerfiles erstellt. Mithilfe einer «docker-compose» Datei konnten somit alle benötigten Programme, sowie Datenbanken, gestartet werden. Durch die Verwendung von Containern mussten keine lokalen Datenbanken von Hand aufgesetzt werden und der Prototyp kann mithilfe von nur einem Befehl auf jedem beliebigen Computer gestartet werden.

Um die API-Gateway sowie die REST Endpoints zu testen, wurde die Software Postman verwendet. Diese bietet die Möglichkeit Http Anfragen zu senden, sowie diese auch abzuspeichern. Für das Projekt wurde ein Teamworkspace erstellt. In diesem wurden alle möglichen Anfragen abgespeichert. Eine Beispielsanfrage an den API-Gateway sowie die gespeicherten Anfragen sind in Abbildung 27 einsehbar.

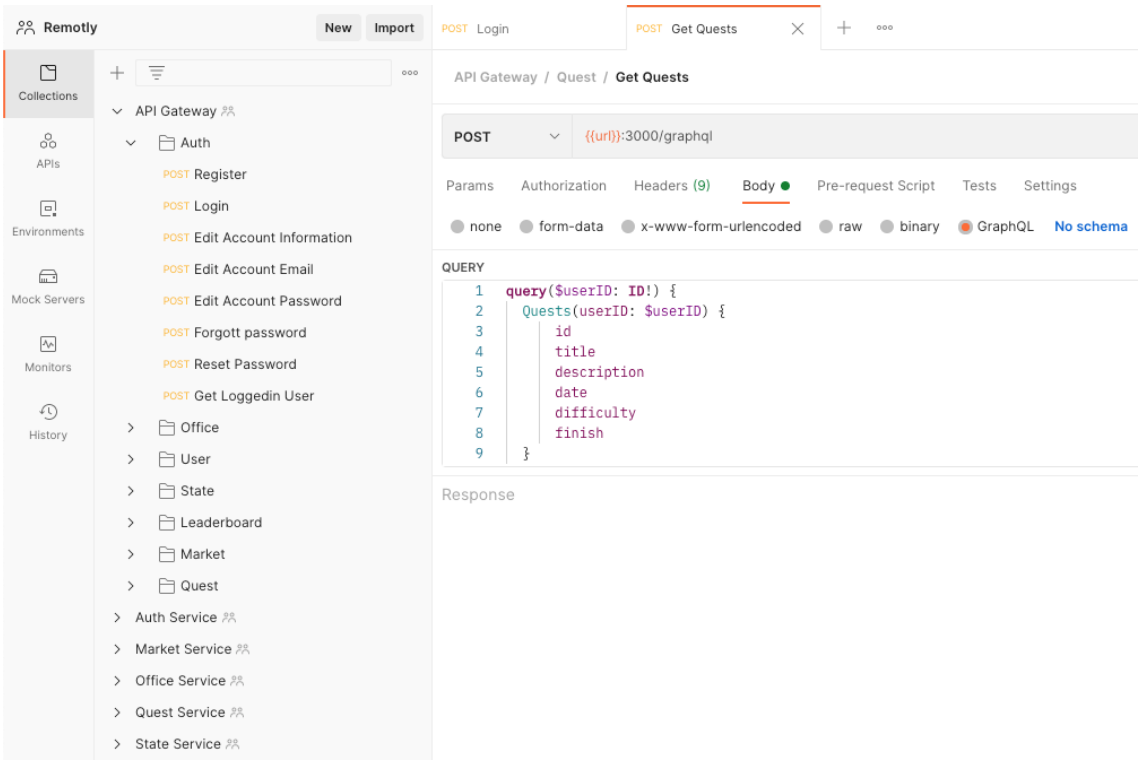

Abbildung 27 Postman Anfragen

Staging Environment: Das Staging Environment wird verwendet, um den Prototypen zu valideren. Der Prototyp läuft in einem Kubernetes Cluster in der Hetzner Cloud, wie es in der Systemarchitektur beschrieben ist. Das Staging Environment ist öffentlich zugänglich.

5.4.2 Continuous Integration

Damit eine fortlaufend gute Codequalität sichergestellt werden kann, wurde mithilfe von Github Actions eine CI Pipeline erstellt. Bei jedem «push» in den Master Branch, sowie bei jedem «pull» Request in einen Branch, wird die CI Pipeline ausgeführt. Für jeden Komponenten der Monorepository wurde ein eigener CI Workflow erstellt. Dadurch sind nur Komponenten bei der Durchführung betroffen, welche auch durch Commits verändert wurden.

Jede CI Pipeline besteht aus zwei Jobs. Der erste Job ist ein Linter, welcher für die Codequalität zuständig ist. Falls die festgelegten Regeln nicht erfüllt werden, wird ein Fehler ausgegeben und die Pipeline scheitert. Im zweiten Job werden die vordefinierten Tests ausgeführt. Dafür werden, wenn nötig, automatisch temporäre Datenbanken aufgesetzt. Wenn beide dieser Jobs erfolgreich durchgeführt wurden, ist die Komponente bereit für das Deployment. Um einen aktuellen Statusüberblick über alle CI Pipelines zu haben,

sind im «README» Dokument der Repository Statusinformation ersichtlich, wie in Abbildung 28 einsehbar.

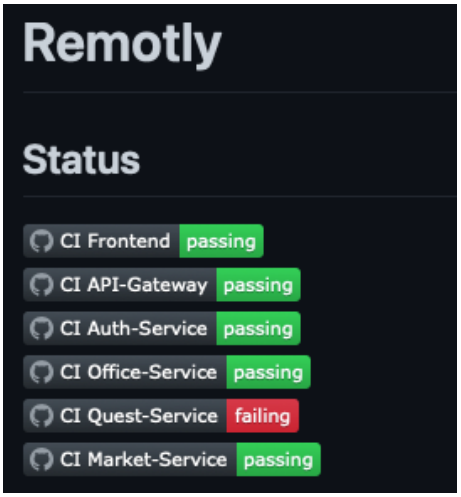

Abbildung 28 Status Übersicht der CI Workflows

5.4.3 Continuous Delivery

Wie bei der CI Pipeline wird auch für die CD Pipeline Github Actions verwendet. Die CD Workflows werden nach einem erfolgreich durchgeführten Pull Request ausgeführt. Damit sichergestellt werden kann das der Code, welcher bereitgestellt werden soll, gut ist, wird dieser immer von einer zweiten Person, welche nicht den Pull Request erstellt hat, überprüft. Auch die CI Workflows werden nochmals durchgeführt und erst nach erfolgreicher Durchführung der CI Workflows sowie der Überprüfung durch eine zweite Person wird der Pull Request angenommen. In Abbildung 29 ist ein Pull Request dargestellt, welcher erfolgreich die CI Tests bestanden hat und als nächstes vom Reviewer angenommen werden muss.

Nach Annahme des Pull Requests erstellt der CD Workflow in einem ersten Schritt ein Dockerimage. Das Dockerimage wird dann mit dem Label «latest» markiert und in die Docker Repository von Dockerhub gepushed. Als letzter Schritt wird das Deployment im Kubernetes Cluster aktualisiert und die neuste Version ist auf dem Staging Environment erreichbar.

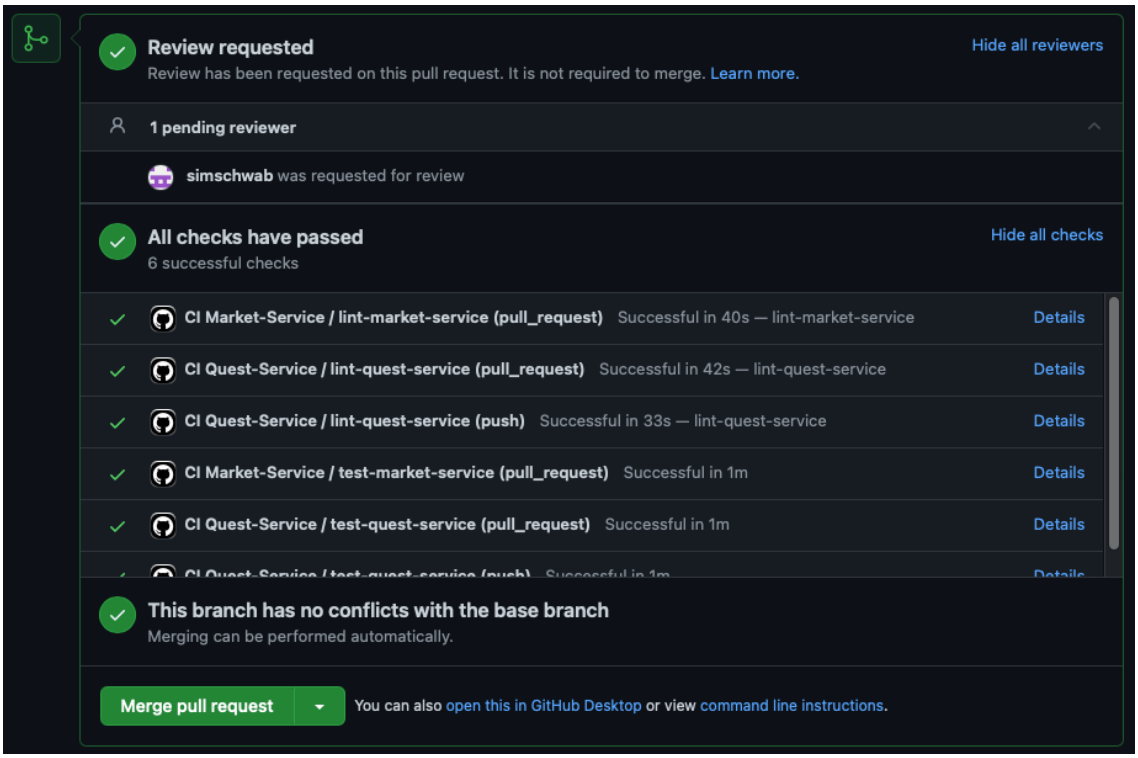

Abbildung 29 Offener Pull Request

6 Schlussteil

In diesem Kapitel wird zuerst die Validierung des MVP erläutert und anschliessend ein Fazit aus der Arbeit gezogen.

6.1 Validierung

Für die Validierung des MVP wurde dieser in der Staging Environment veröffentlicht. In einer zweiwöchigen Testphase wurde ein Office vom Autor erstellt und weitere Personen für die Validierung in das Office eingeladen.

Während der Validierungsphase wurde festgestellt, dass sich durch die Office-Ansicht ein Gemeinschaftsgefühl entwickelt hat und man sich dadurch verbundener fühlt. Die Möglichkeit, sich im Office zu bewegen oder Quests abzuschliessen, bringt eine Abwechslung in den Alltag und führt zu spielerischen Pausen, welche die Arbeitsmotivation erhöhen können. Die Option, den persönlichen Avatar weiter zu personalisieren, motiviert dazu Quests abzuschliessen, um sich mit den erhaltenen Punkten neue Items freizuschalten. Die vielen verschiedenen Avatar Optionen bieten die Möglichkeit einen komplett individuellen und den eigenen Vorstellungen entsprechenden Avatar zu erschaffen.

Bei der Planung der Applikation wurde darauf geachtet, dass User möglichst viel Entscheidungsfreiheit haben, welche Daten diese teilen. Diese Umsetzung gelang gut, denn bei der Benutzung der Applikation fühlten sich User nicht überwacht, da diese sich frei im Office bewegen konnten und selbst entscheiden, wie viele Informationen diese preisgaben. Jedoch könnte die Transparenz der Datensammlung besser gestalten werden, indem dem User genauere Informationen angezeigt werden, um somit ein noch grösseres Vertrauen zu schaffen.

Die Validierung hat jedoch auch ergeben, dass viele Funktionen noch sehr einfach gestaltet sind und diese für eine produktive Nutzung nicht ausreichen. Die Pinnwand und Quests müssten mehr Funktionen enthalten, um diese als wirkliche Aufgabenliste zu verwenden. Das momentane Office ist statisch und kann, abgesehen von Raumnamen und Beschreibungen, nicht editiert werden. Hier müssten ebenfalls Funktionen eingebaut werden, um das Aussehen des Office und die Anzahl an Räumen zu verändern. Letztendlich fehlt auch eine Möglichkeit, direkt im Office miteinander zu kommunizieren. Diese fehlenden Features können in die Applikation selbst oder mithilfe von Drittanbietern, wie zum Beispiel Microsoft Teams, integriert werden.

6.2 Fazit

Für die Beantwortung der Forschungsfrage wurde in dieser Arbeit erfolgreich die Applikation Remotly.ch erstellt. Die gesammelten Informationen im Theorieteil konnten bei der Planung und der Implementation des Backends umgesetzt werden. Das MVP konnte planmässig im erstellten Staging Environment veröffentlich werden.

Das Valideren der Applikation hat aufgezeigt, dass die Applikation eindeutig einen Nutzen mit sich bringt. User der Applikation fühlten sich verbundener mit den Kollegen. Die momentanen Grundfunktionen des MVP reichen aus, um einen groben Überblick der Möglichkeiten solch einer Applikation zu ermöglichen. Jedoch müsste die Applikation noch stark weiterentwickelt werden, um diese produktiv in einem Unternehmen einzusetzen und diese aktiv als Unternehmenssoftware zu verwenden. Gerade Features, wie Kommunikation innerhalb der Applikation, das Individualisieren des gesamten Office, sowie eine weiterentwickelte Version der Pinnwand sind dazu essenziell.

Die im Studium angeeigneten Fähigkeiten im Bereich Projektmanagement wurden effektiv eingesetzt und halfen dabei die Applikation zu planen und die Arbeiten zwischen dem Ersteller des Frontends und des Backends zu koordinieren. Jedoch wurde während dem Planen der Arbeit die Implementationszeit von einigen Tasks stark unterschätzt und während der Erstellung der Applikation mussten immer wieder Features gestrichen werden, um den Zeitplan einhalten zu können. Dies führte auch dazu, dass nicht alle geplanten Aspekte der Applikation umgesetzt und getestet werden konnten.

Literaturverzeichnis

Bundesgesetz über den Datenschutz (DSG) vom 19. Juni 1992 (SR 235.1).

Deloitte. (2020). *How Covid-19 contributes to a long-term boost in remote working*. Abgerufen von der Deloitte-Website https://www2.deloitte.com/ch/en/pages/human-capital/articles/how-covid-19-contributes-to-a-long-term-boost-in-remote-working.html

Dohr, V. (2021). «Daten bitte nicht löschen» – Anonymisierung ist nachhaltiger! *Compliance Kompakt*, *3*.

Eidgenössischer Datenschutz- und Öffentlichkeitsbeauftragter. (o. J.-a). *Datenschutz*. Abgerufen von https://www.edoeb.admin.ch/edoeb/de/home/datenschutz/ueberblick/datenschutz.html

Eidgenössischer Datenschutz- und Öffentlichkeitsbeauftragter. (o. J.-b). *Erläuterung zu Big Data*. Abgerufen von https://www.edoeb.admin.ch/edoeb/de/home/datenschutz/Internet_und_Computer/onlinedienste/erlaeuterungen-zu-big-data/erlaeuterung-zu-big-data.html

Fanger, R. (2020). *Das ist neu am revidierten Schweizer Datenschutzgesetz*. Abgerufen von https://www.netzwoche.ch/news/2020-12-04/das-ist-neu-am-revidierten-schweizer-datenschutzgesetz/0lt0

Hämmerli, M. (2011, März 20). Inkrementelles und iteratives Vorgehen. *SOFTWARE CITY* [Blog-Beitrag]. Abgerufen von https://softwarecity.ch/blog/inkrementelles-und-iteratives-vorgehen

Kaufmann, D. (2021). *Fact Sheet Datenschutz – wichtigste Definitionen & Beispiele*. Abgerufen von https://www.unibe.ch/unibe/portal/content/e809/e962/e963/e6304/e583799/e573822/e583783/pane826303/e826312/DOKU_Fact_Sheet_Datenschutz_-Definitionen_und_Beispiele_20190628_dka_ger.pdf

Krämer, C. F. (2019). Industrie 4.0 – Möglichkeiten und Grenzen von Online-Kommunikation in virtuellen Teams. In A. Ternès & M. Englert (Hrsg.), *Digitale Unternehmensführung: Kommunikationsstrategien für ein exzellentes Management* (S. 25–40). Springer Fachmedien Wiesbaden.

Meyerlustenberger Lachenal. (2020, Oktober 19). *Neues Schweizer Datenschutzrecht: Wichtigste Regelungen der DSG-Revision im Überblick*. Abgerufen von https://www.mll-news.com/neues-schweizer-datenschutzrecht-wichtigste-regelungen-der-dsg-revision-im-ueberblick/

Neuburger, R. (2020). Arbeiten 4.0: Virtuelle Arbeitsplätze. In T. Kollmann (Hrsg.), *Handbuch Digitale Wirtschaft* (S. 1–18). Springer Fachmedien Wiesbaden.

Slack. (2020). *Moving beyond remote: Workplace transformation in the wake of Covid-19*. Abgerufen vonhttps://slack.com/blog/collaboration/workplace-transformation-in-the-wake-of-covid-19

Staatssekretariat für Wirtschaft. (2019). *Arbeiten zu Hause*. Abgerufen von https://www.seco.admin.ch/dam/seco/de/dokumente/Publikationen_Dienstleistungen/Publikationen_Formulare/Arbeit/Arbeitsbedingungen/Broschueren/seco_bro_homeoffice.pdf.download.pdf/seco_broschuere_homeoffice_de.pdf

Anhang A: Glossar

A A-D

Name	Definition	Bemerkung
Aktivität	Tätigkeit welche vom User in Räumen durchgeführt werden.	
Avatar	Virtuelles Abbild einer Person.	
Badge	Auszeichnung welche durch Aktivitäten erreicht werden kann.	
Coin	In-Game Währung	

E E-K

Item	Gegenstand mit welchem der Avatar individualisiert werden kann.	

L L-O

Leaderboard	Rangliste	
Market	Store für Items	
Office	virtuelles Office	

P P-S

Pinnwand	Liste aller Quests	
Quest	Aufgabe	
Raum	Teil/Aufenthaltsort des Office	

T T-Z

User	Benutzeraccount, Resource Owner	Überbegriff, besitzt Email etc.
Office-Admin	Administrator eines Office	Ein Office-Admin ist auch ein User.
Office-User	Individueller Office-User	Ein Office-User ist auch ein User.