Duale Hochschule Baden-Württemberg Mannheim

Seminararbeit

Konzeption und prototypische Implementierung der Impfplattform „my.vay“

Studiengang Wirtschaftsinformatik

Studienrichtung E-Government / E-Health

Verfasser:	Yasmin Braun, Valentin Müller, Karen Pagnia Pia Schramm, Tristan Schwarzer
Kurs:	WWI18DSA
Studiengangsleiter:	Prof. Dr.-Ing. habil. Dennis Pfisterer
Wissenschaftlicher Betreuer:	Dipl. Ing. Mirjana Radonjic-Simic
Bearbeitungszeitraum:	14.05.2021 – 30.07.2021

Hinweise für diese Arbeit

Aus Gründen der Lesbarkeit wird in dieser Arbeit die männliche Flexionsform genutzt und stellt keinerlei Ungleichberechtigung des weiblichen und diversen Geschlechts dar.

Kurzfassung

Impfungen zählen zu den wichtigsten präventivmedizinischen Maßnahmen. Daher ist es von großer Wichtigkeit, dass für Epidemiologen und Virologen aktuelle und aussagekräftige Daten zum Impfschutz der Bevölkerung erhoben werden können. Zu diesem Zweck wird der analoge gelbe Impfpass genutzt, der Impfungen nachweist, die ein Patient erhalten hat. Dessen Beschaffenheit sowie der händische Umgang mit diesem stellen den gelben Impfpass vor einige Herausforderungen, sodass Impfungen teils nicht nachvollzogen werden können, da sie falsch dokumentiert werden. Zudem ist aus ihm oftmals nicht ersichtlich, wann ein Patient eine Auffrischungsimpfung des bereits geimpften Vakzins benötigt. Aus diesem Grund und vor allem durch die anhaltenden Impfkampagnen der fortschreitenden COVID-19-Pandemie wird der Ruf nach einem digitalen Impfpass immer lauter.

Im Rahmen der vorliegenden Arbeit wird ein Konzept vorgestellt, das eine Impfplattform namens my.vay abbildet. Mit my.way soll das nachhaltige digitale Abbilden der bisherigen analogen Impfdokumentation samt Erinnerungsfunktion an anstehende Impfungen ermöglicht werden. Ein weiteres Ziel der Plattform ist, dass verschiedene Nutzergruppen über Impfstoffe sowie deren Nebenwirkungen informiert werden können. Für jedes dieser digitalen Impfeinträge soll ein elektronischer Nachweis erstellt werden können. Damit die Ziele auf einer Plattform vereint werden können, wurden unter Heranziehen europäischer Richtlinien funktionale sowie nicht-funktionale Anforderungen definiert. Diese Anforderungen wurden genutzt, um nach dem „4+1-View-Model“ my.vay aus verschiedensten Sichten darzustellen und zu modellieren.

Die Architektur von my.vay setzt sich aus für jede Nutzergruppe eigenen *Frontends* zusammen. Die darauf aufbauende Datenspeicherung ist als dezentrales System konzipiert.

Ein für Anschauungszwecke implementierter Prototyp enthält die wichtigsten Kernfunktionalitäten, die my.vay für seine Nutzergruppen bietet und womit das bisherige Impfsystem revolutioniert werden kann. Hierzu wurde eine auf der Programmiersprache „Python“ basierende zentralisierte Webapplikation realisiert. Als zugrundeliegendes Datenbankmanagementsystem kommt „PostgreSQL“ zum Einsatz.

Schlagworte: Digitaler Impfpass, Digitaler Impfnachweis, Impfplattform

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Quelltextverzeichnis

Abkürzungsverzeichnis

ANF-IA	Anforderungen der Issuer Application
ANF-PA	Anforderung der Patient Application
ANF-RA	Anforderungen der Researcher Application
ANF-VA	Anforderungen der Verifier Application
API	Application Programming Interface
CBOR	Concise Binary Object Representation
COVID-19	Coronavirus disease 2019
CSS	Cascading Style Sheets
DGC	Digital Green Certificate
DGCG	Digital Green Certificate Gateway
DMS	Doctors Management System
DSC	Document Signer Contract
DSGVO	Datenschutz-Grundverordnung
eImpfpass	elektronischer Impfpass
EU	Europäischen Union
HCERT	Electronic Health Certificate Container Format
HTML	Hypertextmarkuplanguage
IATA	International Air Transport Association
ID	Identifikationsnummer
IfSG	Infektionsschutzgesetz
NANF-DS	nicht-funktionale Anforderungen Datenschutz und Sicherheit
NANF-HC	nicht-funktionale Anforderungen HCERT
NHS	National Health Service
Nr.	Nummer
ORM	Object-Relational Mapping
PEI	Paul-Ehrlich-Institut

PNG	Portable Network Graphics
PVS	Praxisverwaltungssystem
RKI	Robert-Koch-Institut
QR	Quick Response
SQL	Structured Query Language
STIKO	ständige Impfkommission
TAN	Transaktionsnummer
UML	Unified Modeling Language
WHO	World Health Organisation

Glossar

Begriff	Erklärung
Digitaler Impfeintrag	Enthält alle Daten einer Impfung, die nach Paragraph 22 des Infektionsschutzgesetzes vorgeschrieben sind
Digitaler Impfpass	Enthält mind. eine individuell ausgestellte digitale Impfung
Impfdaten	Teilmenge an Daten einer digitalen Impfung, die laut World Health Organisation (WHO) vorgeschrieben sind und für die Ausstellung eines Impfnachweises benötigt werden
Impfnachweis	Individueller Nachweis einer durchgeführten Impfung, welcher anhand der Impfdaten erstellt und in Form eines QR-Codes zur Verfügung gestellt wird. Der Impfnachweis muss von einem Issuer ausgestellt werden, um rechtskräftig zu sein.
Issuer	Aussteller des Impfnachweises, welche über eine medizinische Grundausbildung verfügt und dazu berechtigt ist zu Impfen.
Patient	Halter des Impfnachweises und Person, die medizinische Leistung in Anspruch nimmt.
Verifier	Institution die Impfnachweise auf Zulässigkeit gemäß der geltenden Vorschriften überprüft.
Researcher	Wissenschaftlicher Angestellter in einer offiziellen Forschungseinrichtung, im Bereich der Virologie und Prävention.

1 Einleitung

1.1 Motivation

Impfungen stellen eine der bedeutendsten und häufigsten Präventionsmaßnahmen dar. Insbesondere in Zusammenhang mit der aktuellen Coronavirus disease 2019 (COVID-19)-Pandemie erfahren Impfungen ein bisher selten dagewesenes Maß an Aufmerksamkeit. Seit Anfang 2020 bestimmt COVID-19 die Lebensumstände der weltweiten Bevölkerung. Die starke Ausbreitung der Infektion mit teilweise schweren Krankheitsverläufen und Todesfällen ließ sich im Laufe des Jahres innerhalb Deutschlands zwar verlangsamen, jedoch nicht gänzlich ausbremsen [1]. Infolgedessen stiegen die Infektionszahlen innerhalb der Wintermonate und aggressivere Varianten des Virus traten zu Tage [1].

Um die weitere Ausbreitung des Virus sowie schwere Verläufe der Krankheit zu verhindern, wird eine sogenannte Herdenimmunität benötigt [2]. Davon wird in der Epidemiologie gesprochen, wenn ein ausreichend hoher Prozentsatz einer Bevölkerung durch Infektion oder Impfung Immunität erlangt hat [2].

Üblicherweise, zum Beispiel bei Mumps, Kinderlähmung, Pocken und einigen weiteren Krankheiten, liegt die empfohlene Herdenimmunität bei circa 75-85% [3]. Da die Impfstoffe gegen COVID-19 nicht zu 100% wirksam und die neuen Virus-Varianten Alpha, Delta und Lambda ansteckender als die Ursprungsform sind, wird eine Immunität von 85% der 12-59-Jährigen und von 90% der ab 60-Jährigen seitens des Robert-Koch-Institut (RKI) empfohlen [4].

Jedoch existiert eine maßgebliche Barriere, die diese Zahlen unmöglich erscheinen lässt: die fehlende Impfbereitschaft einer Vielzahl von Bürgern. Die Gründe hierfür sind vor allem Misstrauen aufgrund fehlender Informationen über die Inhaltsstoffe und die langfristig möglichen Nebenwirkungen der einzelnen angebotenen Impfstoffe [5]. Es bedarf dementsprechend der Aufklärung über die Inhaltsstoffe und realistisch mögliche Nebenwirkungen, um genau diese Bevölkerungsgruppe mittels vorhandener Transparenz ebenfalls zur Impfbereitschaft zu bewegen. Ein weiteres Problem besteht darin, dass ein Großteil der Bevölkerung ihre Impftermine vergisst, da in den meisten Fällen nicht alle Auffrischungsintervalle

der gängigen Impfungen gleich sind, beziehungsweise diese nicht gleichzeitig vorgenommen wurden [5].

Während die unterschiedlichsten personalisierten Nachweise, wie beispielsweise Terminbuchungen, Eintrittskarten oder COVID-19-Testergebnisse längst per Smartphone validiert und akzeptiert werden können, scheint die notwendige Digitalisierung im Gesundheitswesen größtenteils zu stagnieren [6]. Gerade in Zeiten der Pandemie würde ein digitales Abbild des analogen Impfpasses zur Automatisierung der Abläufe innerhalb der Gesundheitsbranche beitragen. Ebenfalls spielte die Fälschungssicherheit bei den bisherigen Impfpässen in Papierform eine Rolle, da diese leicht nachgebildet werden können [7].

1.2 Zielsetzung

Ziel dieser Arbeit soll es sein, im Hinblick auf die weltweit herrschende Pandemie-Situation, ein Konzept zur Verbesserung des aktuellen Impfsystems zu entwerfen. Das Konzept soll demzufolge eine umfassende Lösung darstellen, welche den aktuellen Impfpass weitestgehend digitalisiert und um zusätzliche Funktionen erweitert. Diesbezüglich soll im Rahmen dieser Arbeit die Forschungsfrage beantwortet werden, welche Funktionalitäten ein digitaler Impfpass für die bestmögliche Unterstützung des Impfprozesses bieten sollte und wie diese umzusetzen sind.

Dennoch gilt es zu beachten, dass bei der Konzeption der Lösung die Einhaltung nationalrechtlicher Rahmenbedingungen sowie die seitens der Europäischen Union (EU) gewährleistet werden müssen. Neben der Ausarbeitung des Konzepts soll als Ergebnis dieser Arbeit zusätzlich ein Prototyp entwickelt werden.

1.3 Aufbau dieser Arbeit

Die vorliegende Arbeit wird in insgesamt acht Kapitel unterteilt. Im zweiten Kapitel werden die für das Verständnis der Arbeit notwendigen Grundlagen vorgestellt. Diese beinhalten eine Vorstellung des gegenwärtigen Impfsystems in Deutschland und des digitalen Impfpasses. Des Weiteren werden zur Konzeption benötigte technische Grundlagen erläutert. Das dritte Kapitel soll den weltweit herrschenden Status Quo und somit die Ausgangssituation

im Bezug auf die zu entwerfende Plattform aufzeigen. Anschließend werden im vierten Kapitel die Zielsetzung an die Plattform festgelegt und hieraus resultierend die Anforderungen erhoben. Diese Anforderungen werden im nachfolgenden fünften Kapitel als Grundlage für die Erstellung des Konzepts verwendet. Zur umfänglichen Darstellung des Konzepts, wird dieses mittels verschiedener Unified Modeling Language (UML)-Diagramme aus fünf unterschiedlichen Perspektiven beschrieben. Im sechsten Kapitel wird das zuvor erarbeitete Konzept prototypisch umgesetzt sowie dessen Funktionalitäten aus Sicht der Anwender beschrieben. Nachfolgend werden die Ergebnisse dieser Arbeit im siebten Kapitel in einer kritischen Würdigung reflektiert und abschließend im achten Kapitel zusammengefasst sowie ein Ausblick gegeben. Ein darin gezogenes Fazit beantwortet zudem die in der Einleitung definierte Forschungsfrage. Eine Visualisierung des Aufbaus der Arbeit ist Abbildung 1.1 zu entnehmen.

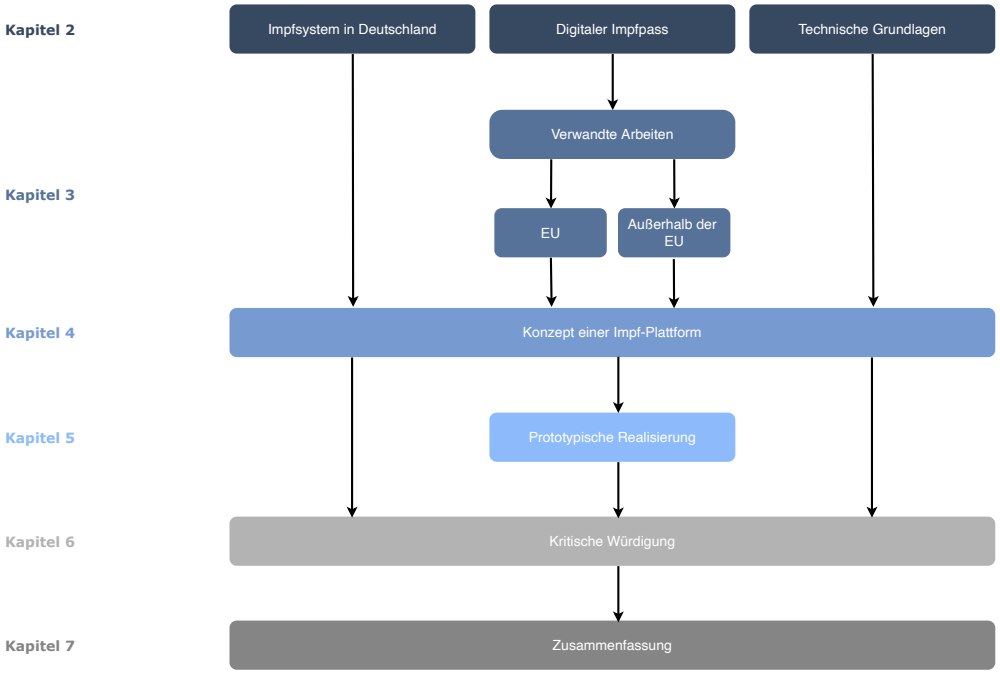

Abbildung 1.1: Aufbau der Arbeit

2 Grundlagen

In diesem Kapitel werden alle theoretischen Grundlagen vorgestellt, welche zum weiteren Verständnis dieser Arbeit notwendig sind. Dazu widmet sich der erste Abschnitt des Kapitels dem aktuellen Impfsystem in Deutschland sowie dem international gültigen, analogen Impfpass der WHO. Im Anschluss daran wird auf das „digitale COVID-19-Zertifikat der EU“ eingegangen, welches ein erster Schritt in Richtung digitaler Impfpass in Europa ist. Der Fokus liegt dabei insbesondere auf die dem Zertifikat zugrundeliegende technische Architektur. Zum Abschluss des Kapitels werden noch einige technische Grundlagen definiert, die für die spätere Umsetzung des Prototypen von Relevanz sind. Hierzu zählen unter anderem die Begriffe „Flask“, „SQLAlchemy“ und QR-Code.

2.1 Impfsystem in Deutschland

Im folgenden Unterkapitel wird ein Einblick in das teildigitalisierte deutsche Impfsystem gegeben. Dazu wird zunächst ausgeführt, wozu Impfungen gebraucht werden, welche Voraussetzungen diese erfüllen müssen und wann diese empfohlen werden. Des Weiteren wird vorgestellt, wie Impfungen dokumentiert werden und ein Überblick über den aktuellen gelben Impfpass gegeben. Zusätzlich wird auf die Herausforderungen des analogen Impfpasses eingegangen.

2.1.1 Impfungen

Zweck einer Impfung

Infektionskrankheiten sind eine große Gefahr für die Gesundheit, da sich diese schnell ausbreiten und zu schweren Krankheitssymptomen führen können. Einen wirksamen Schutz gegen Infektionskrankheiten stellen Impfungen dar. Durch eine Impfung kann ein Patient sich nicht nur selbst, sondern auch das eigene persönliche Umfeld vor einer Erkrankung schützen. Dies wird auch Gemeinschaftsschutz genannt [8]. Je größer der Anteil der Geimpften in der Bevölkerung ist, desto effizienter kann eine Krankheit eingedämmt und gegebenenfalls sogar eliminiert werden [9]. International wird aktuell beispielsweise die Elimination

von Masern angestrebt. Zwar gilt in Deutschland das Infektionsschutzgesetz (IfSG), welches nach § 1 IfSG den Zweck hat, ansteckende Krankheiten frühzeitig mit entsprechenden Maßnahmen einzudämmen, jedoch ist eine Impfung bis auf wenige Ausnahmefälle, zum Beispiel aufgrund der Berufsgruppe, gesetzlich nicht verpflichtend.

Impfarten

Generell kann es sich bei einem Impfstoff um zwei verschiedene Arten handeln, entweder ein Lebend- oder ein Totimpfstoff. Bei Lebendimpfstoffen handelt es sich um abgeschwächte, vermehrungsfähige Viren oder Bakterien, wie beispielsweise bei Masern oder Röteln. Lebendimpfstoffe können gleichzeitig mit anderen Impfstoffen verabreicht werden. Erfolgt die Impfungen jedoch nicht gleichzeitig, sollte vor einer neuen Impfung ein Abstand von mindestens vier Wochen eingehalten werden. Bei Totimpfstoffen muss kein Mindestabstand zu einer anderen Impfung beachtet werden, außer wenn der Patient stark auf die Impfung reagiert. Wird der ärztlich empfohlene Impfabstand unterschritten, kann dies die Wirksamkeit des Impfstoffes beeinflussen. Überschreitung der Abstände sind jedoch meistens unproblematisch, da der Beginn der Immunität lediglich nach hinten verzögert wird [10].

Impfzulassung und -durchführung

Impfstoffe sind heute im Allgemeinen gut verträglich, da diese vor einer Zulassung umfassend erforscht und in Studien geprüft werden [11]. Wie bei jedem anderen Arzneimittel sind allerdings auch bei Impfungen Nebenwirkungen nicht ausgeschlossen, weswegen es zu Impfverweigerungen in der Bevölkerung kommen kann. Eine Impfung kann eine Erkrankung außerdem nicht immer verhindern. Die Erkrankungswahrscheinlichkeit wird jedoch erheblich gesenkt und in den meisten Fällen kann ein schwerer Krankheitsverlauf verhindert werden. Das Risiko der Impfung sollte somit immer mit dem Risiko einer Nicht-Impfung verglichen werden, da hierbei nicht nur die eigene Erkrankung droht, sondern auch Immunitätslücken in der Bevölkerung entstehen [12, 13].

Jeder Impfstoff muss bei der Herstellung bestimmte Anforderungen erfüllen, welche von der WHO definiert werden [12]. Die Zulassung des Impfstoffs erfolgt anschließend entweder über die Kommission der Europäischen Gemeinschaft oder in Deutschland über das Paul-Ehrlich-Institut (PEI). Das PEI ist das Bundesinstitut für Impfstoffe und biomedizinische Arzneimittel, welches für die Zulassung von Impfstoffen zuständig ist. Sollten nach der

Zulassung eines Impfstoffes zu viele Verdachtsfälle in Form von Erkrankungen oder starken Nebenwirkungen auftreten, kann das PEI die Situation neu bewerten und eine Zulassung wieder verweigern [14]. Des Weiteren kann das PEI über die individuelle Chargennummer auch Impfstoffe und Arzneimittel einzeln überprüfen. Bei der Chargennummer handelt es sich um eine eindeutig identifizierbare Kennziffer für eine gewisse Impfdosis, über welche der Lieferweg und Produktionsort zurückverfolgt werden kann [11].

Die Impfung selbst darf von jedem niedergelassenen Arzt durchgeführt werden [15]. Vor einer Impfung muss der Patient in einem Aufklärungsgespräch entsprechend über den Nutzen der Impfung, die zu verhütende Krankheit, die Durchführung, das Verhalten danach, mögliche Nebenwirkungen und eventuell nötige Folgeimpfungen informiert werden. Anschließend muss für die Impfung ausdrücklich eingewilligt werden [16]. Die Vergütung der medizinischen Einrichtung erfolgt über die Krankenkasse des Patienten. Jede Impfung muss im Anschluss entsprechend § 22 IfSG dokumentiert werden. Häufig besteht jedoch das Problem, dass die Dokumentation vernachlässigt wird und somit der Nachweis verloren geht [17].

Impfempfehlungen

Da bei der Impfung selbst verschiedene Voraussetzungen beachtet werden müssen, veröffentlicht die ständige Impfkommission (STIKO), bei der es sich um ein Gremium von Experten aus dem Medizin- und Forschungsbereich handelt, Empfehlungen zu Impfzeitpunkten in Deutschland. In diesen Empfehlungen wird zum Beispiel definiert, in welchem Alter, welche Impfung durchgeführt werden sollte, welche Abstände zwischen Impfungen eingehalten werden müssen und ob eine Impfung nach einer bestimmten Zeit aufgefrischt werden muss [18]. Die Einhaltung einer Auffrischung ist wichtig, da nur so die bestmögliche Wirksamkeit erzielt werden kann [19]. Wird diese vernachlässigt, kann der Impfschutz nicht länger garantiert werden. Manche Impfungen werden zusätzlich nur in bestimmten Situationen empfohlen. Eine hiervon ist beispielsweise vor einer Reise ins Ausland, da in dem angestrebten Reiseziel eventuell ein höheres Ansteckungsrisiko einer Infektionskrankheit vorliegt und bei der Einreise ein Impfnachweis erforderlich sein könnte.

Impfungen werden meist in einem bestimmten Alter mit spezifischen Impfabständen empfohlen. Wie diese Empfehlungen lauten, ist im Impfkalender des RKI dargestellt. Tabelle 2.1 zeigt den Impfkalender ausschnittsweise [20]. Die Abkürzung *G* steht dabei für Grundimpfung, *A* für Auffrischung und *S* für Standardimpfung. Die Nummer hinter den Abkürzungen

zeigt, wie oft die Impfung bis zu diesem Alter bereits für eine Immunisierung durchgeführt sein sollte. Eine Auffrischung ohne Zahl steht in der Tabelle dafür, dass diese ab einem gewissen Alter in regelmäßigen Abständen durchgeführt werden muss. Bei Tetanus sind dies beispielsweise alle zehn Jahre ab einem Alter von 18 Jahren.

Impfungen	**0-2**	**2-6**	**6-18**	**18-60**	**ab 60**
Rotaviren	G1-3				
Tetanus	G1-3	A1	A2	A	A
Diphtherie	G1-3	A1	A2	A	A
Hib.	G1-3				
Hepatitis B	G1-3				
Pneumokokken	G1-3				S
Masern	G1-2			S	
Mumps, Röteln	G1-2				
HPV			G1-2		
Influenza					S

Tabelle 2.1: Ausschnitt des Impfkalenders des RKIs (Quelle: In Anlehnung an [20])

2.1.2 Aufbau des analogen Impfpasses

Wie im vorherigen Kapitel 2.1.1 beschrieben, müssen Impfungen entsprechend dokumentiert werden, da sie nur so als offiziell gültig anerkannt werden können [21]. Um diese Dokumentation zu vereinheitlichen, werden alle Impfungen analog im gelben Impfpass eingetragen. Hierbei handelt es sich um ein persönliches Dokument, welches zu jedem Impftermin mitgebracht werden sollte, um schrittweise vom Arzt ausgefüllt zu werden. Mithilfe des Impfpasses kann jede befugte Person schnell nachvollziehen, welche Impfungen bisher durchgeführt wurden [22].

Ein neuer Impfpass kann kostenlos von jedem Arzt bezogen werden. Die ältere Version des Impfpasses ist der weiße Falt-Impfausweis, welcher von der Form zwar veraltet ist, jedoch den gleichen Inhalt wie die neuere Version umfasst. Daher wird dieser auch weiterhin als Nachweis in Deutschland anerkannt. Die neuste Version ist der gelbe Impfpass, welcher durch die Einhaltung der Richtlinien der WHO international gültig ist. Auf dem Deckblatt des gelben Impfpasses ist aus diesem Grund auch das Zeichen der WHO zu sehen. Zudem müssen auf dem Deckblatt die grundlegenden Daten des Besitzers angegeben werden. Hierzu zählen neben dem vollständigen Namen auch das Geburtsdatum, der Geburtsort, die derzeitige Adresse und bei Bedarf die Nummer des Personalausweises. Abbildung 2.1

zeigt das Deckblatt des gelben Impfpasses [23]. Der ausgestellte gelbe Impfpass enthält in Deutschland auf jeder Seite Beschreibungen in Deutsch, Englisch und Französisch. Je nach Art der Impfung muss der Arzt diese im Impfpass auf einer entsprechenden Seite eintragen. Bei der Dokumentation einer Impfung müssen außerdem bestimmte Daten angegeben werden. So muss neben dem Handelsnamen des Impfstoffes auch die Chargennummer dokumentiert werden, über welche die geimpfte Dosis beim Hersteller zurückverfolgt werden kann. Diese beiden Informationen werden meist mit Hilfe eines Aufklebers vermerkt. Des Weiteren muss der Arzt mit seiner Unterschrift, dem Stempel der impfenden medizinischen Einrichtung und der Angabe des Impfdatums die Echtheit des Eintrags beweisen [24].

Neben der Dokumentation der durchgeführten Impfungen dient der gelbe Impfpass außerdem als Informationsquelle. Auf der letzten Seite können dazu in Form eines Impfkalenders die empfohlenen Impfungen für Kinder und Jugendliche eingesehen werden, wie sie bereits in Kapitel 2.1.1 vorgestellt wurden.

Abbildung 2.1: Internationaler gelber Impfpass der WHO (Quelle: [23])

2.1.3 Herausforderungen des analogen Impfpasses

Als Nachweis über den aktuellen Impfschutz eines Patienten dient in Deutschland bislang ausschließlich der zuvor erläuterte analoge Impfpass. Die in dem Impfpass getätigten handschriftlichen Einträge sowie die papierbasierte Dokumentation birgt gewisse Herausforderungen und Risiken, die nachfolgend dargelegt werden.

Eine, vor allem im Kontext der COVID-19-Krise, stark diskutierte Herausforderung stellt die nicht vorhandene Fälschungssicherheit dar. Diese resultiert aus den manuellen und damit unverschlüsselten Eintragungen [7, 25]. Hinzu kommen fehlende Sicherheitsmerkmale wie beispielsweise ein Wasserzeichen oder ein Hologramm bei dem Etikett des verabreichten Impfstoffes sowie der Unterschrift und des Stempels des impfenden Arztes. Demzufolge können Fälschungen durch das Kopieren von Eintragungen einfach angefertigt werden, sodass sich diese nicht von einem Originaldokument unterscheiden lassen [26]. Vorrangig in aktuellen Zeiten der Pandemie häuft sich das Fälschen von Impfpässen, in der Hoffnung auf Freiheiten in Bezug auf die aktuellen Einschränkungen wie beispielsweise die Quarantänepflicht nach Reisen in ein Risikogebiet. Das Fälschen von Impfpässen sowie der Gebrauch von Fälschungen wird mit dem Tatbestand der Urkundenfälschung geahndet, wodurch mit einer Freiheitsstrafe von bis zu zwei Jahren zu rechnen ist [27].

Darüber hinaus ist der Aspekt der dauerhaft fehlenden Sicherung des analogen Dokuments eine tragende Herausforderung. Bereits im Säuglingsalter werden Impfungen im Impfpass dokumentiert. Nicht selten kommt es im Laufe eines Lebens zu dessen Verlust. Wer den Impfpass und damit den Nachweis aller jemals erhaltenen Impfungen verliert, hat die Möglichkeit, Impfungen durch eine befugte Person in einem neuen Dokument nachtragen zu lassen. Hierzu werden die ärztlichen Unterlagen eines Patienten benötigt, die durch den impfenden Arzt mindestens zehn Jahre aufbewahrt werden müssen. Sind frühere Impfungen jedoch nicht mehr nachweisbar, müssen diese nachgeholt oder ergänzt werden [28].

Neben dem potentiellen Verlust ist die Papierform des analogen Impfpasses nachteilig, denn grundsätzlich entstehen mit der Verwendung von Papier im digitalen Zeitalter Medienbrüche. Dies hat zur Folge, dass im Prozess der Impfdokumentation nach § 22 IfSG teils Daten nicht rechtmäßig in den Impfass übertragen werden. Eine mangelnde Impfdokumentation führt dazu, dass eine Impfung als nicht durchgeführt gilt [29]. Hinzu kommen oftmalige Unleserlichkeiten der Einträge. Es ist für einen Patienten durch die teils unleserlichen beziehungsweise falsch eingetragenen Einträge oftmals schwer nachvollziehbar, ob die Impfungen gemäß dem für den Impfstoff vorgesehenen Zeitraum durchgeführt wurden [30]. Des Weiteren besteht durch die Papierform die Herausforderung, die Daten zum Impfstatus für die Forschung freizugeben, da hierfür aktuell kein einheitliches System existiert. Die Impfdaten könnten der Forschung jedoch wichtige Erkenntnisse zur Impfakzeptanz bieten und zu einer besseren Verfolgung von Nebenwirkungen beitragen [31]. Zusammengefasst lässt sich festhalten, dass die gegebenen Möglichkeiten, eine Impfung einfach, schnell und unkompliziert zu dokumentieren, zu mehreren Herausforderungen führen, die in Bezug auf

den analogen Impfpass optimiert werden können.

2.2 Digitaler Impfpass

Trotz der im vorherigen Abschnitt beschriebenen Herausforderungen eines analogen Impfpasses und der fortschreitenden digitalen Transformation des Gesundheitswesens gab es bis vor kurzem nur wenige Bestrebungen den Impfpass zu digitalisieren. Angesichts der COVID-19-Pandemie ist der Impfpass allerdings seit Ende 2020 in den Fokus der allgemeinen Aufmerksamkeit gerückt. Die Ursache dafür ist, dass nahezu alle Nationen weltweit Reise- und Bewegungsbeschränkungen zur Eindämmung der Pandemie erlassen haben. Diese Einschränkungen werden jedoch für den Einzelnen mit der Erbringung eines Nachweises über dessen Impfung gegen COVID-19 teilweise oder gänzlich aufgehoben [32, 33, 34].

Damit ein solcher Impfnachweis sicher und leicht erfolgen kann, haben einige Staaten mit der Entwicklung eines digitalen Nachweises für eine COVID-19-Impfung begonnen. Um zu verhindern, dass in jedem EU-Mitgliedsstaat eine Individuallösung entsteht, trafen sich schon im November 2020 Gesundheitsexperten der Europäischen Kommission, um zu diskutieren, wie künftig Impfnachweise europaweit digital transferiert werden könnten [35]. Der Austausch führte dazu, dass am 17.03.2021 eine Verordnung des europäischen Parlaments und des Rates vorgeschlagen wurde, welche die Rahmenbedingungen eines digitalen Impfnachweises für die EU definiert [36, 37]. In den folgenden Monaten erarbeitete die Europäische Kommission Leitlinien für einen europaweiten Impfnachweis namens „digitales COVID-19-Zertifikat der EU“ oder „Digital Green Certificate (DGC)“ [36, 38].

2.2.1 Digitales COVID-19-Zertifikat der EU

Das digitale COVID-19-Zertifikat der EU dient nicht nur als Nachweis, dass ein Patient gegen COVID-19 geimpft ist, sondern belegt alternativ auch eine Genesung von COVID-19 oder das Vorliegen eines negativen Testresultats [36, 38]. Seit dem 01.07.2021 sind alle EU-Mitgliedstaaten dazu verpflichtet, dieses Zertifikat auszustellen und es als Nachweis für die soeben angesprochenen Sachverhalte anzuerkennen [36, 39]. Zur Umsetzung dieses Vorhabens wurden, wie bereits zuvor erwähnt, technische Spezifikationen von der EU definiert [38]. In insgesamt zwölf Leitlinien (Stand: 19. Juni 2021) definierte die EU maßgeblich

das Zertifikat selbst und den Austausch der Zertifikate zwischen den teilnehmenden Staaten [40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51]. Im Folgenden sollen mit dem „Digital Green Certificate Gateway (DGCG)“ und dem „Electronic Health Certificate Container Format (HCERT)“ die für diese Arbeit relevanten Aspekte der Spezifikationen beschrieben werden.

Digital Green Certificate Gateway

Im Mittelpunkt der technischen Leitlinien der EU steht der Entwurf einer Architektur, die eine europaweit einheitliche Ausstellung und Verifizierung von digitalen COVID-19-Zertifikaten ermöglicht. Zur Wahrung der nationalen Autonomie bei gleichzeitiger Aufrechterhaltung der Interoperabilität zwischen den Nationalstaaten, wurde diese Architektur teilweise auf Ebene der Nationalstaaten und teilweise auf EU-Ebene umgesetzt. Der Aufbau der Architektur ist in Abbildung 2.2 dargestellt [43, 46].

Kernelement der Architektur ist das DGCG, welches die Interoperabilität auf der EU-Ebene sicherstellt. Dazu umfasst das DGCG einen verteilten Datenspeicher, der als Vertrauensanker für alle anderen Komponenten dient. In diesem werden zu Anfang vorwiegend sogenannte *Document Signer Contracts (DSCs)* gespeichert. *DSCs* sind Zertifikate, die von einer „nationalen Wurzelzertifizierungsstelle für das Signieren“ ausgestellt werden und es einem System erlauben, digitale COVID-19-Zertifikate auszustellen [46, 52]. Der Zugriff auf die im DGCG gespeicherten *DSCs* erfolgt durch ein *Application Programming Interface (API)*. Darüber können mittels des *Publish-and-Subscribe-Mechanismus* neue *DSC* in den Datenspeicher geladen und die aktuellen *DSC* ausgegeben werden [46].

Da das DGCG, wie bereits erläutert, als Vertrauensanker dient, darf nicht jede beliebige Applikation die *API* nutzen, um *DSCs* hochzuladen. In Folge dessen gibt es einen *Onboarding*-Prozess, den die Applikation eines Mitgliedsstaates durchlaufen muss, bevor sie die *API* nutzen darf. Registrierte Anwendungen werden als „National Backend“ bezeichnet. Während die konkrete Ausgestaltung der National Backends den EU-Mitgliedsstaaten überlassen wird, verfügen alle Applikationen über einen sogenannten *Local Trust Store* sowie einen Issuer [43, 46].

Der *Local Trust Store* ist lediglich eine nationale Kopie des Datenspeichers im DGCG und wird von Verifiern genutzt, um die Gültigkeit von digitalen COVID-19-Zertifikaten zu bestimmen. Diesbezüglich überprüfen die Verifier, ob ein digitales COVID-19-Zertifikat mit einem *DSC* erstellt wurde, das in dem *Local Trust Store* gespeichert ist. Ist dies der Fall, so

ist das digitale COVID-19-Zertifikat gültig und wurde von einer vertrauenswürdigen Instanz ausgestellt [46].

Solche Instanzen werden als Issuer bezeichnet und sind, wie bereits erwähnt, ebenfalls Bestandteil eines jeden National Backends. Ein Issuer verfügt über ein DSC und ist somit zur Ausstellung von digitalen COVID-19-Zertifikaten berechtigt. [46]

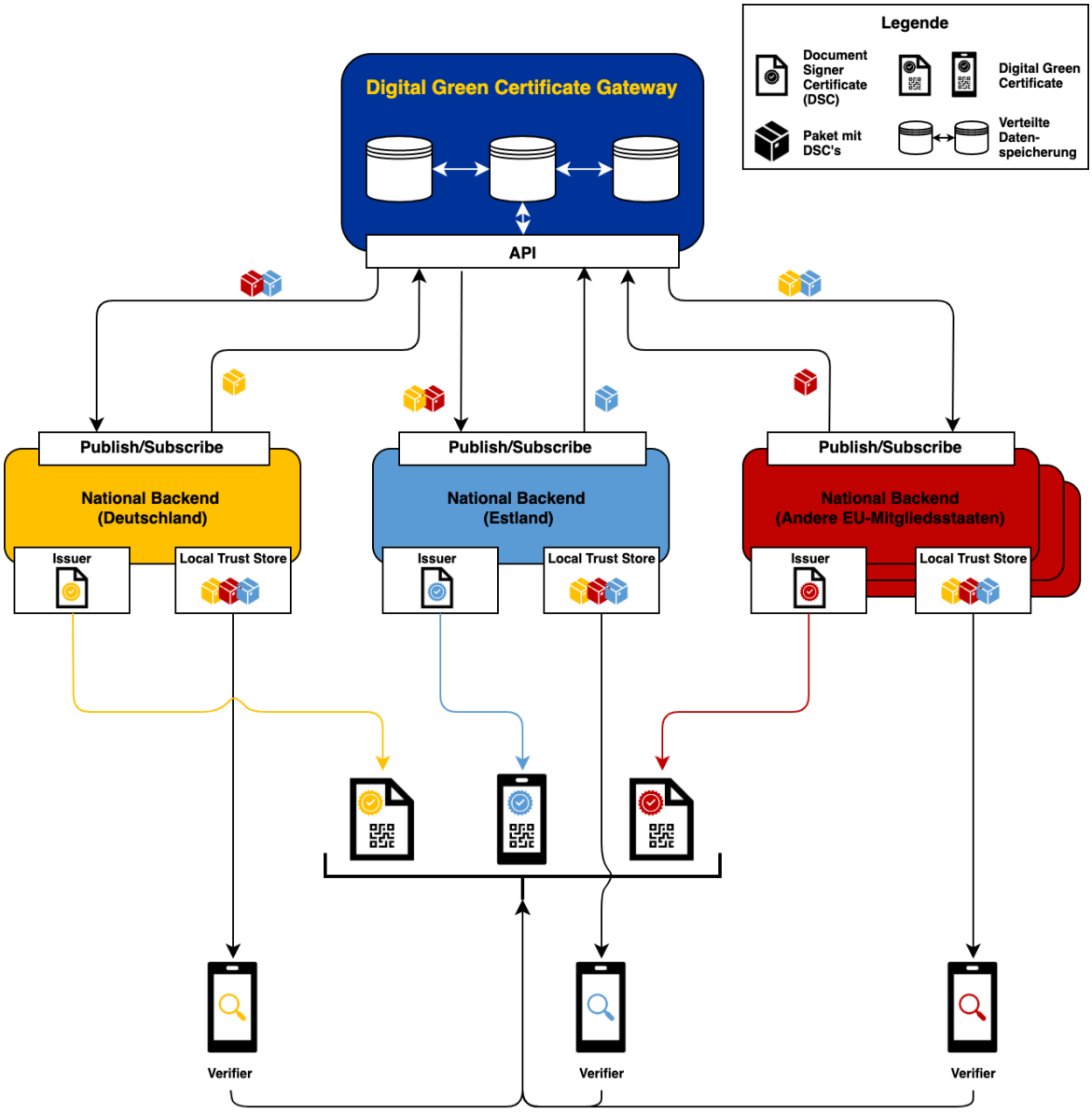

Abbildung 2.2: Veranschaulichung des Digital Green Certificate Gateway's (Quelle: Angelehnt an [43, 46])

Zusammenfassend ist jeder EU-Mitgliedsstaat für die Bereitstellung des National Backends

eigenverantwortlich. Damit übernehmen die Staaten die Verantwortung über das Ausstellen und Verifizieren der digitalen COVID-19-Zertifikate. Um trotzdessen ein einheitliches Zertifikat auf EU-Ebene zu ermöglichen, wird das DGCG genutzt. Dieses stellt einen vertrauenswürdigen europäischen Datenspeicher bereit, indem alle DSC vorhanden sind, die zur Ausstellung eines digitalen COVID-19-Zertifikats berechtigt sind. [43, 46]

Electronic Health Certificate Container Format

Innerhalb der EU-Leitlinien zum digitalen COVID-19-Zertifikat wurde nicht nur das DGCG konzipiert, sondern auch ein neues Datenformat namens HCERT. Das Datenformat wird erstmalig für die COVID-19-Zertifkate der EU eingesetzt. Zukünftig soll es jedoch zum Standard für digitale Gesundheitszertifikate in der EU werden, weshalb bei der Spezifikation des HCERT auf Interoperabilität geachtet wird. [42]

Infolgedessen wird das HCERT unter anderem in dem gängigen Datenformat Concise Binary Object Representation (CBOR) nach „Request for Comments“ 8949 kodiert.[53] Die Datenfelder des HCERTs sind ebenfalls standardisiert, wobei diese nachfolgend aufgelistet sind:

- ***Protected Header***
- ***Signature Algorithm:*** Definiert den Signatur-Algorithmus, der für das Erstellen der Signatur genutzt wurde
- ***Key Identifier:*** Gibt das DSC an, das den öffentlichen Schlüssel enthält, der von der Prüfstelle zur Überprüfung der Korrektheit der digitalen Signatur verwendet werden soll
- ***Payload:*** Nutzdaten, die während einer Kommunikation übertragen werden
- ***Issuer:*** Zeichenkette, die das Herkunftsland des Issuers des *Health Certificates* angibt
- ***Issued At:*** Datum und Uhrzeit, zu der das *Health Certificates* ausgestellt wurde
- ***Expiration Time:*** Datum und Uhrzeit, zu der die Gültigkeit des *Health Certificates* abläuft
- ***Health Certificate:*** JSON-Objekt, welches das *Health Certificate* wie beispielsweise das digitale COVID-19-Zertifikat der EU enthält
- ***Signature:*** Signatur des *Health Certificates*

2.3 Technische Grundlagen

Neben den Kenntnissen über das analoge deutsche Impfsystem und den Vorgaben der EU bezüglich des digitalen Impfpasses ist für das vollumfängliche Verständnis dieser Arbeit ein technisches Verständnis erforderlich. Infolgedessen werden in diesem Unterkapitel die relevanten technischen Grundlagen vermittelt.

2.3.1 Dezentrale Datenspeicherung

Bei einer dezentralen Datenspeicherung werden die entsprechenden Daten lokal auf den Endgeräten der Nutzer abgespeichert, statt sie wie beim zentralen Ansatz auf einem Server zu speichern [54].

Die dezentrale Datenspeicherung hat gegenüber der zentralen Speicherung den Vorteil, dass die Daten durch die Minimierung von Angriffspunkten besser vor unbefugten Zugriff durch Dritte geschützt sind, wodurch der Nutzer die Kontrolle über seine Daten behält [54, 55]. Allerdings muss man beachten, dass ein Nutzer nicht von jedem Endgerät aus auf seine Daten zugreifen kann. Geht das entsprechende Endgerät, auf welchem die Daten gespeichert sind, verloren, sind im Zweifelsfall auch seine Daten weg, die von der Anwendung lokal gespeichert wurden. Dies kann für den Nutzer zu einem schlechten Nutzererlebnis bei der Interaktion führen. Eine weitere Herausforderung des dezentralen Ansatzes ist die Datenintegrität. Die Wahrscheinlichkeit ist bei diesem Ansatz höher als bei dem Zentralen, dass Nutzer die eigenen Daten manipulieren. Des Weiteren ist die Implementierung einer dezentralen Datenspeicherung wesentlich komplexer und aufwändiger [54].

Chancen	**Herausforderungen**
Datensicherheit	Nutzererlebnis
Nutzer hat Gewalt über seine Daten	Implementierungs- & Wartungsaufwand
Leistung	Datenintegrität
Skalierbarkeit	

Tabelle 2.2: Chancen und Herausforderungen der dezentralen Datenspeicherung

Die Chancen und Herausforderungen der dezentralen Datenspeicherung sind in Tabelle 2.2 zusammengefasst. Der dezentraler Ansatz ist gerade bei sensiblen Daten, wie Gesundheitsdaten, empfehlenswert, da der Ansatz unter anderem bei dem Thema Datensicherheit sowie bei der Akzeptanz der Nutzer punkten kann [54].

2.3.2 Python

Hinter dem Begriff „Python“ verbirgt sich eine universelle, objektorientierte Programmiersprache, welche zur Entwicklung eigenständiger Programme und für das Skripten in verschiedenen Anwendungsbereichen verwendet wird. Aufgrund der *Open-Source*-Lizenz der Programmiersprache kann diese im vollen Umfang frei verwendet werden und ist daher sehr beliebt unter den Entwicklern [56]. Python wurde insbesondere mit dem Ziel der Steigerung der Entwicklungsproduktivität, Softwarequalität, Programmportabilität und der optimalen Komponentenintegration entworfen. Als besonders gelten bei Python der Fokus auf die Lesbarkeit des Codes und die Verwendung von Bibliotheken. Programme, welche in Python geschrieben wurden, besitzen die Eigenschaft auf diversen gebräuchlichen Plattformen, wie „Linux“ [57], „Windows“ [58], „macOS“ [59] und zahlreichen weiteren, ausgeführt werden zu können [60].

2.3.3 Flask

Bei Flask handelt es sich um ein webbasiertes *Framework*, welches unter Verwendung der Programmiersprache Python geschrieben wurde [61]. Die Hauptfunktion dieses *Frameworks* liegt vor allem in der Vereinfachung der Web-Anwendungsentwicklung mittels integrierter Funktionen. So können beispielsweise über die integrierte *Template-Engine* „Jinja2“, vordefinierte HTML-Gestaltungsvorlagen direkt als Webseite angezeigt werden [61]. Darüber hinaus lässt sich Flask mit unterschiedlichen Datenbanken verbinden und ist leicht verwendbar aufgrund der umfassenden Dokumentation [62]. Flask ist so konzipiert, dass es im Kern möglichst minimal gehalten ist und keine überflüssigen Funktionen beinhaltet. Sämtliche Funktionen, die bereits von anderen Bibliotheken abgedeckt sind, werden nicht in Flask umgesetzt, sondern lassen sich über die bestehenden Bibliotheken integrieren [61].

WTForms

Als „WTForms“ wird eine Bibliothek für Flask bezeichnet, welche zur Gestaltung von Formularen in Flask-Webanwendungen genutzt werden kann. Unter der Verwendung von WTForms soll dem Anwender eine interaktive Benutzeroberfläche ermöglicht werden [63]. Die Anfrageobjekte werden nach deren Eingabe inklusive der zugehörigen Formularelemente von der Anwenderseite an die Serverseite übermittelt. Des Weiteren existieren innerhalb der

Bibliothek bereits vordefinierte Standardformularfelder, wie beispielsweise Eingabe- oder Passwortfelder [64].

2.3.4 SQLAlchemy

„SQLAlchemy“ ist eine Bibliothek, die die Kommunikation zwischen Python-Programmen und Datenbanken erleichtert [65]. Häufig wird diese Bibliothek als Object-Relational Mapping (ORM)-Tool verwendet, welches Python-Klassen in Tabellen in relationalen Datenbanken übersetzt und Funktionsaufrufe automatisch in SQL-Anweisungen umwandelt [65]. Ferner fungiert SQLAlchemy als Schicht zwischen der Anwendung und der Datenbank. Es legt Objekte der Anwendung innerhalb der relationalen Datenbankstruktur ab. Zu den Besonderheiten von SQLAlchemy gehört die Vereinheitlichung des Zugriffs auf die Datenbank, unabhängig von den Unterschieden zu der genutzten „Structured Query Language (SQL)“ [66]. Außerdem gilt das Modul als extrem leistungsfähig und ermöglicht auch komplexe *Mappings* zwischen einer Datenbank und Python-Objekten [67]. Es unterstützt eine Reihe von relationalen Datenbanken, unter anderem auch „PostgreSQL“.

PostgreSQL

Bei PostgreSQL handelt es sich um ein objektrelationales Datenbankmanagementsystem, dessen Software ebenfalls frei verfügbar ist und plattformunabhängig einsetzbar ist [68]. Beispielsweise kann es auf Linux-, Windows- und macOS-Systemen genutzt werden. PostgreSQL basiert auf einem Client-Server-Modell, bei dem der Server für die Verwaltung der Datenbanken und die Verarbeitung und Beantwortung der Client-Anfragen zuständig ist. Server und Client können hierbei auf verschiedenen Systemen laufen [69]. Ein besonderes Merkmal von PostgreSQL stellt die Möglichkeit zur hochgradigen Modifizierung, Erweiterung und Anpassung dar [68]. Folglich können beispielsweise neue Funktionen, Operatoren oder Datentypen hinzugefügt werden. Des Weiteren existieren keine Beschränkungen bezüglich der Datenbankgröße oder der Verarbeitung komplexerer Anfragen [70].

2.3.5 Quick Response Code (QR)

Unter einem QR-Code versteht man eine zweidimensionale besondere Form des Barcodes. Er ist der Gruppierung der Matrixcodes zugehörig und wird durch schwarze und weiße

Quadrate definiert, welche die kodierten Daten binarisch darstellen und durch ein Gerät mit spezieller Sensortechnik, beispielsweise einem Smartphone, ausgelesen werden kann. Bei einem Matrixcode werden die Dateninhalte über die Position der dunkleren Elemente gleichmäßig verteilt [71]. Die maximale Anzahl an Stellen, die ein QR-Code speichern kann, ist auf 7089 numerische und 4297 alphanumerische Zeichen begrenzt [72]. Dabei können Zeichen sowohl vertikal als auch horizontal gespeichert werden, was den Platzbedarf des Codes reduziert. Verschiedenste Versionen ermöglichen es, die Anzahl an Modulen (Quadraten) eines QR-Codes festzulegen. Diese Module definieren die sogenannte *Encoding Region* und *Function Patterns*, welche in Abbildung 6.4 dargestellt sind. Diese setzen sich aus „Hauptpositionsmarkierungen, Zeitraster, Seperator und Ausrichtungsmuster“ [72] zusammen.

Die *Function Patterns* werden benötigt, um den QR-Code korrekt auslesen zu können. Hierzu werden folgende Formen in bestimmten Bereichen des QR-Codes platziert:

- ***Finder Patterns:*** Drei Blöcke in den oberen beiden und der linken unteren Ecke des QR-Codes
- ***Separators:*** Leerraum neben den *Finder Patterns*
- ***Allignment Patterns:*** Diese sind ähnlich der *Finder Patterns* nur lediglich kleiner und im gesamten QR-Code verteilt
- ***Timing Patterns:*** Gepunktete Linien, welche die *Finder Patterns* verbinden
- ***Dark Module:*** Schwarzes Modul, dass immer neben den *Finder Patterns* unten links platziert wird [73]

Die *Encoding Region* enthält die reinen Daten und das Datenformat [73].

Das Auslesen eines QR-Codes verläuft grundlegend in zwei Schritten. Für die Dekodierung der Daten wird lediglich ein Lesegerät, beispielsweise ein Smartphone, benötigt. Mit der Kamera wird der QR-Code grafisch erfasst und es wird ein digitales Bild der kodierten Daten erzeugt. Anschließend werden die Formatinformationen samt Dateninhalten ausgelesen [72, 73].

Die weite Verbreitung von Smartphones sowie die fortgeschrittene Entwicklung leistungsfähiger Sensoren in diesen Geräten ermöglicht das Durchdringen von QR-Codes in nahezu allen Lebensbereichen. Gegenwärtige Anwendungsbereiche sind in Bezug auf die Corona-Pandemie beispielsweise das Verlinken zu Internetauftritten von Restaurants durch die

„Luca-App“ [74]. Ebenfalls werden in diesem Kontext QR-Codes bei dem Nachweis des europaweiten gültigen Impf-/Genesenen-/Test-Nachweis eingesetzt.

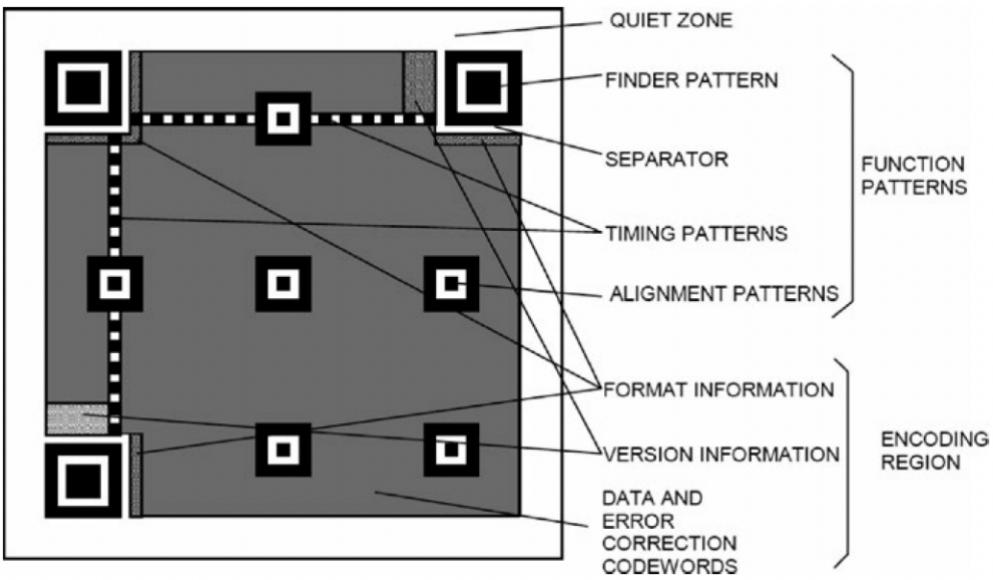

Abbildung 2.3: Struktur eines QR-Codes (Quelle: [73])

3 Verwandte Arbeiten

Im vorherigen Kapitel wurde schon auf das digitale COVID-19-Zertifikat der EU eingegangen. Da den einzelnen EU-Mitgliedsstaaten jedoch Spielraum bei der genauen Ausgestaltung des digitalen Impfnachweises gegeben wurde, werden diese Lösungen in diesem Kapitel näher beleuchtet. Darüber hinaus sind auch andere Staaten außerhalb der EU und internationale Organisationen bestrebt, eine praktikable Lösung zum Nachweis getätigter Impfungen zu realisieren, weshalb auch diese im Folgenden betrachtet werden. Dadurch soll ein umfassender Überblick über verwandte Ansätze und bereits bestehende Lösungen gegeben werden, bevor in dieser Arbeit eine eigene Lösung konzipiert wird.

3.1 Europäische Union

In der Einleitung zu diesem Kapitel wurde bereits ausgeführt, dass die einzelnen EU-Mitgliedsstaaten unterschiedliche Lösungen zum digitalen Nachweis einer COVID-19 Impfung implementiert haben. Während einige Staaten wie Estland bereits zu einem frühen Zeitpunkt der Impfkampagnen einen digitalen Impfnachweis bereitstellen konnten, dauerte es in Deutschland deutlich länger. Doch auch schon vor der COVID-19 Pandemie gab es in Deutschland Ansätze für einen digitalen Impfpass. Im folgenden Abschnitt werden deshalb relevante Arbeiten und Erkenntnisse verschiedener Länder der EU betrachtet.

3.1.1 Deutschland

Seitens der Bundesregierung steht die Entwicklung einer Nachweismöglichkeit seit längerer Zeit im Raum. Dennoch sind auch einzelne gewinnorientierte Unternehmen bestrebt, ihren Kunden eine gesonderte Lösung bereitzustellen. Die nachfolgenden Unterabschnitte thematisieren zum einen den „CovPass“ des Bundesgesundheitsministeriums und zum anderen den „eImpfpass“ der Krankenkasse „AOK Plus“.

CovPass

Bereits im März 2021 hatte das Bundesgesundheitsministerium angekündigt, einen elektronischen standardisierten Impfnachweis zu entwickeln, mit welchem das Aufrufen der Nachweise per Smartphone auch ohne Internetverbindung ermöglicht werden soll [75]. Papierbasierte Nachweise sollen dennoch ihre Gültigkeit behalten. Für die Implementierung der Nachweis-App, der Prüf-App sowie dem Backend-System wurde ein Zusammenschluss aus dem Startup „Ubirch GmbH“, „IBM Deutschland GmbH“, „govdigital-Genossenschaft“ und „Bechtle AG“ beauftragt [75]. Unter der Berücksichtigung juristischer Rahmenbedingungen, wie des bereits in Kapitel 2.2.1 aufgezeigten digitalen COVID-19-Zertifikats der EU, wurde dieses Vorhaben am 01.06.2021 mit der Veröffentlichung der CovPass-App und der Integration des CovPass als Funktion in die bundesweite „Corona-Warn-App“ umgesetzt. Herausgeber und somit verantwortlich für die Ausgestaltung und Prüfung dieser ist das RKI [76].

Zur Nutzung des CovPass muss der digitale Impfnachweis vorab in einer Arztpraxis, einer Apotheke, einem Krankenhaus oder einem autorisierten Impfzentrum nach der vollständig durchgeführten Impfung erstellt werden. Im Anschluss an die Eingabe der Daten wird ein QR-Code generiert, welcher dem Patienten in Papierform ausgehändigt wird. Dieser QR-Code kann anschließend mit der Corona-Warn-App oder der separat nutzbaren CovPass-App eingescannt werden. Sowohl die CovPass-App als auch die Corona-Warn-App sind kostenlos und können in den meisten gängigen Appstores heruntergeladen werden. Beide Alternativen speichern die Impfbescheinigungen lokal auf dem Smartphone, sodass das Aufrufen des Zertifikats ohne Internetverbindung möglich ist. Im Falle eines Hardwarewechsels, muss der QR-Code erneut abgescannt werden. Aus diesem Grund wird empfohlen, den in Papierform ausgehändigten Nachweis sicher aufzubewahren. Ebenso wie der digitale Impfnachweis enthält der physische Nachweis Informationen wie den Impfstatus, den Namen und das Geburtsdatum des Patienten, das Impfdatum, den verwendeten Impfstoff und die Höhe der Dosis [77].

Überprüft werden kann der QR-Code seitens verschiedener Dienstleister und Behörden via einer separaten Prüf-App, welche den QR-Code einscannen und auf dessen Echtheit überprüfen. Parallel hierzu muss die Identität des Patienten mittels eines Ausweisdokuments überprüft werden. Zur Generieung des QR-Codes werden alle digitalen Impfnachweise nur kurzfristig im Impfprotokollierungssystem gespeichert und anschließend gelöscht. Eine dauerhafte Speicherung der Daten erfolgt lediglich lokal auf dem mobilen Endgerät. Ergänzend

können nun auch negative Tests als „Testzertifikat“ und Genesungen als „Genesenenzertifikat“ in beiden Apps registriert werden [77].

eImpfpass der AOK Plus

Ein weiteres verwandtes Projekt stellt der „elektronischer Impfpass (eImpfpass)“ der AOK Plus dar. Bei der AOK Plus handelt es sich um eine gesetzliche Krankenkasse, welche circa 3,4 Millionen Mitglieder aus Sachsen und Thüringen versichert [78]. Im Gegensatz zu der CovPass-App soll der eImpfpass die Verwaltung von Impfdaten weitestgehend revolutionieren. Der eImpfpass soll als digitale Plattform sämtlicher Impfdaten für Arzt und Patient fungieren. Auf dieser Plattform sollen Impfungen erstellt, kontrolliert und verwaltet werden. Auch Impfempfehlungen relevanter Impfkomissionen sollen berücksichtigt und Patienten sowie Ärzte an anstehende Impfungen erinnert werden. Des Weiteren soll der eImpfpass digital verfügbar sein und bei Bedarf auch ausgedruckt werden können. Die AOK Plus bestrebt mit dem Projekt des elektronischen Impfpasses die Entwicklung eines gleichwertigen Impfpasses nach dem IfSG [79].

Bisher befindet sich die Software seit Sommer 2020 im Einsatz und verzeichnet über 20.000 teilnehmende Versicherte der AOK Plus [80]. Die Krankenkasse plant im Laufe des Jahres 2021 den Wegfall einer bisher benötigten zertifizierten Impfmanagement-Software, welche Praxen benötigen, um auf den eImpfpass zugreifen und Impfdaten exportieren zu können. Infolgedessen soll ein Export der Daten aus den bereits vorhandenen Praxisverwaltungssystem (PVS) der Ärzte erfolgen. Auf diesem Weg können Ärzte ohne zusätzlichen Aufwand valide elektronische Impfdaten erzeugen und auf den eImpfpass übertragen [80]. Künftig könnten die Daten des digitalen Impfpasses in die elektronische Patientenakte übernommen werden [79].

3.1.2 Estland

Bezüglich der Entwicklung eines „E-Impfausweises“ gilt Estland als Vorreiter in Sachen Digitalisierung. Bürger, die in Estland eine Impfung gegen COVID-19 erhalten, können im Anschluss ein digitales Impfzertifikat bekommen. Dieses kann man sich dann als App auf das Smartphone herunterladen [81]. Damit sollen Beschränkungen im Alltag wegfallen und beispielsweise Besuche in Fitnessstudios, Kinos oder Theatern wieder möglich

werden. Die App gibt lediglich Auskunft über den Impfstatus und soll so der Datenschutz-Grundverordnung (DSGVO) entsprechen [82]. Entwickelt hat das Projekt das estnische Unternehmen „Guardtime“ in Zusammenarbeit mit der WHO [82]. Bereits seit Januar wird der digitale Impfpass in einem Pilotprojekt getestet.

3.1.3 Österreich

Auch Österreich plant den digitalen Nachweis und die Erfassung von Impfungen [83]. Konkret soll der „e-Impfpass“ in den kommenden Jahren den klassischen Papier-Impfpass ablösen. Genau wie beim klassischen physischen Impfpass werden im e-Impfpass zukünftig alle Impfungen aufgezeichnet, die eine Person erhalten hat. Die Impfungen werden allerdings nicht nur in einem Impfpass aus Papier dokumentiert, sondern ebenfalls sicher in einem elektronischen nationalen Impfregister abgespeichert [83]. Ein zentralisiertes Impfregister ermöglicht eine vollständige und standardisierte Impfdokumentation ohne die Gefahr von Datenverlust für die Möglichkeit epidemiologischer Auswertungen [84]. Mittels einer Handysignatur soll der digitale Impfpass den Bürgern über das elektronische Gesundheitsakte-Portal zugänglich gemacht werden [85].

3.2 Außerhalb der Europäischen Union

Auch außerhalb der EU gibt es einige Länder, die aufgrund der COVID-19-Pandemie bereits einen digitalen Impfpass entwickelt haben. Im Folgenden soll die Umsetzung des digitalen Passes einiger dieser Länder aufgezeigt werden.

3.2.1 Afrika

Die „African Union Commission“ und die „Africa Centres for Disease Control and Prevention“ haben, um die Überprüfung von öffentlichen Gesundheitsdokumenten von Reisenden zu vereinfachen, das COVID-19-Passsystem „My COVID Pass“, auch unter dem Namen „Trusted Travel Pass“ bekannt, entwickelt. Dieses orientiert sich an dem International Air Transport Association (IATA) Travel Pass, welches in einem der nachfolgenden Abschnitte vorgestellt wird. [86, 87, 88]. Das System wird zur Zeit von einigen Fluggesellschaften getestet und soll nach erfolgreicher Testphase in allen Mitgliedstaaten der Afrikanischen

Union eingeführt werden. Bei dem System handelt es sich um ein Online-Portal, in welchem Passagiere ihre COVID-19-Testergebnisse oder ihren Impfstatus hochladen können. Anschließend erhalten die Passagiere einen QR-Code. Das System bietet zusätzlich Informationen zu den aktuell geltenden Reiseanforderungen und -beschränkungen des Ziellandes [88].

3.2.2 China

In China soll es mithilfe eines digitalen internationalen Reise-Gesundheitspasses in Form eines Online-Formulars ermöglicht werden, wieder zu verreisen. Das Formular beinhaltet Daten wie Namen, Passnummer und Impfstatus einer Person. Des Weiteren kann das Ergebnis eines COVID-19-Tests eingetragen werden. Der Impfpass ist als App in das, in China beliebtes und häufig genutztes, soziales Netzwerk „Wechat“ integriert. Sobald die Daten im Pass eingetragen sind, wird ein QR-Code erstellt der beispielsweise beim Verreisen vorgezeigt werden kann und die Echtheit der Daten überprüft [89, 90].

3.2.3 Dachverband der Fluggesellschaften IATA

Der Dachverband der Fluggesellschaften „IATA“ hat den sogenannten „IATA Travel Pass“ in Form einer App veröffentlicht. Reisende können durch diesen Impf- und Testzertifikate fälschungssicher speichern, verwalten und bei Bedarf vorzeigen. Des Weiteren stellt der Pass Informationen zu den Reise-, Test- und Impfanforderungen des entsprechenden Landes bereit und bietet Reisenden die Möglichkeit auch weitere Reisedokumente digital zu verwalten und online einzuchecken. Mithilfe des Passes wird sich das Öffnen der Grenzen auch ohne Quarantäne und damit die Wiederaufnahme des Luftverkehrs erhofft [91].

3.2.4 England

Der digitale Impfpass Englands baut ähnlich wie in Deutschland auf der nationalen Coronawarn-App auf. Mittlerweile fungiert diese als digitaler Impfnachweis, mit welchem man sich über sein Smartphone ausweisen kann [92]. Am 17. Mai führte England den digitalen Ausweis des National Health Service (NHS) für den Grenzübertritt ein. Bürger können hierbei die Anwendung als Nachweis für deren Impfstatus nutzen [93]. Die Implementierung einer Funktion zum Hinterlegen negativer Testergebnisse steht derzeit noch offen. Alternativ zur

App sollen Bürger zusätzlich die Möglichkeit haben, ein Impfzertifikat nach vollständiger Impfung per Brief anzufordern [93].

3.2.5 Israel

In Israel wurde bereits am 21. Februar 2021 der sogenannte „Grüne Pass“ eingeführt, durch welchen Genesene und Geimpfte mehr Freiheiten geboten werden soll. Dadurch wird für eine schrittweise Öffnung des Landes gesorgt. Jeder Genesene oder vollständig Geimpfte kann sich online einen Impfpass erstellen, mit welchem sich diese über eine spezielle App den grünen Pass ausstellen lassen können [94]. Der Grüner Pass bildet neben einem QR-Code den Namen der Person, die Nummer des Personalausweises, das Geburtsdatum sowie die Daten der Impfungen ab. Zusätzlich zum Pass muss der Personalausweis vorgezeigt werden. Aktuell ist der Grüne Pass nur im Inland gültig, jedoch soll ab August eine neue Version verfügbar sein, welche zusammen mit anderen Staaten sowie der WHO entwickelt wird, damit der Pass auch international gültig ist [95].

3.2.6 Fazit

Dies sind nur einige bereits implementierte Lösungen eines Impfnachweises. Die aufgezeigten Lösungen beziehen sich hauptsächlich auf den Genesenen- bzw. Impfnachweis von COVID-19. Außerdem sollen viele der Lösungen das Reisen inklusive des Flugverkehrs ermöglichen und bieten damit ein zusätzliches Management für die Reise. In der folgenden Seminararbeit soll jedoch eine umfassende Impfplattform entwickelt werden, die den Impfass um Informationsmöglichkeiten, Erinnerungsfunktionen und Impfnachweisen ergänzt. Ein Management für Reisedokumente soll dabei nicht integriert werden.

4 Anforderungsmanagement

Nachdem in den vorangegangenen Kapiteln die theoretischen Grundlagen definiert und verwandte Ansätze analysiert wurden, kann an dieser Stelle mit der Konzeption einer eigenen Lösung für den digitalen Impfpass begonnen werden. Dazu werden als erstes die Ziele und Nicht-Ziele der zu entwickelnden Lösung definiert und festgehalten. Im Anschluss daran werden im zweiten Abschnitt des Kapitels auf Basis der Zielsetzung genaue Anforderungen an die Lösung definiert, wobei diese in die Kategorien funktional und nicht-funktional untergliedert werden.

4.1 Zielsetzung

Das Festlegen der Projektziele stellt einen wichtigen Schritt im Softwareentwicklungsprozess dar. Sie definieren den Rahmen des Projekts und gewährleisten eine zielgerichtete Vorgehensweise. Eine exakte Zieldefinition gilt als wesentlicher Bestandteil der Softwarespezifikation, welcher ein bestimmtes und strukturiertes Ergebnis ermöglicht [96]. Anhand der im Kapitel 2 erwähnten rechtlichen und technischen Rahmenbedingungen sowie dem im Kapitel 3 vorgestellten Stand der Technik, können die Projektziele für diese Arbeit abgeleitet werden. Darüber hinaus lassen sich die Nicht-Ziele im Rahmen dieser Arbeit festlegen.

4.1.1 Kernziele

Die nachfolgende Aufzählung stellt eine klare Definition der Ziele der zu entwickelnden Plattform dar.

1. my.vay unterstützt eine leicht verständliche Aufklärung der potentiellen Nutzer über Impfstoffe und Impfprozesse.
2. my.vay verkörpert das digitale Abbild des aktuellen physischen Impfpasses, wodurch eine nachhaltige Dokumentation gewährleistet werden kann.
3. my.vay verbessert den Vorgang zum Nachweis einer durchgeführten Impfung.

4. my.vay steigert die durchschnittliche Impfquote durch die Implementierung einer Funktion zur Erinnerung des Nutzers an anstehende Impftermine.

4.1.2 Abgrenzung der Nicht-Ziele

Im Bezug auf das angestrebte Ergebnis dieser Arbeit wurden folgende Abgrenzungen formuliert:

1. my.vay inkludiert keine zusätzlichen Funktionen bezüglich des Reisemanagements.
2. my.vay soll nicht dem Zweck dienen, die Meinungen der Impfgegner zu beeinflussen.
3. my.vay stellt keinen internationalen Lösungsansatz dar und kann nicht in anderen Ländern genutzt werden.
4. my.vay enthält keine Schnittstelle zur potenziellen Anbindung an die Telematikinfrastruktur und deren Anwendungen.

4.2 Anforderungsanalyse

Einen weiteren Teil des Softwareentwicklungsprozesses repräsentiert die Anforderungsanalyse. Das Ziel dieser Analyse ist es, die Anforderungen an die für my.vay zu entwickelnde Plattform zu ermitteln und strukturiert darzustellen [97]. Unter Berücksichtigung der festgelegten Ziele können sowohl funktionale als auch nicht-funktionale Anforderungen erfasst werden. Die funktionalen Anforderungen sagen aus, welche Dienste my.vay anbieten soll [98]. Zusätzlich legen sie fest, welches Verhalten my.vay in bestimmten Situationen aufweisen soll beziehungsweise welche Eingaben, Verarbeitungen und Ausgaben getätigt werden sollen [97]. Die Erfassung der Anforderungen wird aus Sicht der vier Nutzergruppen Patient, Issuer, Verifier und Researcher (siehe Glossar auf Seite) getätigt und gegliedert. Nicht-funktionale Anforderungen sollen aufzeigen, welche Anforderungen an die Qualität und die Benutzbarkeit von my.vay gestellt werden [97].

Für die Formulierung der unterschiedlichen Anforderungen wird die Muss-/Soll-Vorschrift genutzt. Diese beschreibt ein unterschiedliches Ausmaß an den Befolgungsanspruch einer einzelnen Anforderung. Die Bedeutung der zwei Modalverben wird nachstehend weiter ausgeführt [99].

- Muss: Eine Muss-Anforderung gilt in allen Fällen einzuhalten und räumt keinerlei Ermessensspielraum in der Umsetzung ein [99].
- Soll: Eine Soll-Anforderung beschreibt den Regelfall und räumt einen gewissen Grad an Ermessen in der Umsetzung frei [99].

4.2.1 Funktionale Anforderungen (ANF)

Patient Application (PA):

1. Ein Patient muss sich registrieren und anmelden, um my.vay nutzen zu können.
2. Ein Patient muss seine erhaltenen Impfungen in my.vay übersichtlich angezeigt bekommen. Hierunter fallen die Daten nach § 22 IfSG.
3. Ein Patient muss die Daten bisheriger Impfungen mit geringem zeitlichen und organisatorischen Aufwand in my.vay importieren können.
4. Ein Patient muss mithilfe von my.vay seine Impfungen schnell nachweisen können.
5. Ein Patient soll in my.vay seine Impfnachweise bei Bedarf ausdrucken können.
6. Ein Patient muss sich in my.vay über Impfungen und ihre möglichen Nebenwirkungen informieren können.
7. Ein Patient soll sich in my.vay darüber informieren können, welche Impfungen für eine Reise in ein anderes Land benötigt werden.
8. Ein Patient muss in my.vay einsehen können, welche Impfungen, in welchem Alter laut RKI empfohlen werden.
9. Ein Patient muss von my.vay über demnächst fällige Impfauffrischungen erinnert werden.
10. Ein Patient soll seine anstehenden Impftermine in my.vay hinterlegen können, um zu gegebener Zeit an diese erinnert zu werden.
11. Ein Patient soll in my.vay eventuelle Nebenwirkungen nach einer Impfung dokumentieren und einsehen können.
12. Ein Patient soll in my.vay individuell entscheiden können, ob seine Impfdaten anonym mit der Forschung geteilt werden sollen.

13. Ein Patient soll die Möglichkeit haben, ausgewählte Impfungen und deren Daten zur ärztlichen Einsicht in my.vay zu teilen.
14. Ein Patient soll die Daten bisheriger Impfungen von Angehörigen in my.vay importieren können.
15. Ein Patient soll für die Daten seiner Angehörigen jegliche Funktionen von my.vay, die auch für die Verwaltung seiner eigenen Impfdaten zur Verfügung stehen, durchführen können.

Issuer Application (IA):

1. Ein Issuer muss sich registrieren und anmelden, um my.vay nutzen zu können.
2. Ein Issuer muss in my.vay nach einer durchgeführten Impfung dem Patient einen Impfnachweis als QR-Code, gemäß §22 IfSG, ausstellen können.
3. Ein Issuer muss in my.vay nachvollziehen können, ob es sich bei dem individuellen Impfnachweis des Patients um eine Fälschung handelt.
4. Ein Issuer soll sein Praxisverwaltungssystem automatisch mit den Daten eines Patients aus my.vay aktualisieren können.
5. Ein Issuer soll die digitalen Impfdaten des Patients teilweise oder komplett mit dessen Einwilligung vor Ort in my.vay geteilt bekommen können.
6. Ein Issuer soll auf alle Impfinformationen zugreifen können, die auch dem Patient zur Verfügung gestellt werden.

Verifier Application (VA):

1. Ein Verifier muss my.way nutzen können, ohne sich vorher registrieren und anmelden zu müssen.
2. Ein Verifier muss mit my.vay, Impfnachweise eines Patients mit geringem zeitlichen und organisatorischen Aufwand auf Richtigkeit überprüfen können.

Researcher Application (RA):

1. Ein Researcher muss sich registrieren und anmelden, um my.vay nutzen zu können.
2. Ein Researcher soll über my.vay anonymisierte Impfdaten von Patients zur Verfügung gestellt bekommen.
3. Ein Researcher soll in my.vay ein Dashboard für die Auswertung der Impfdaten zur Verfügung gestellt bekommen.

4.2.2 Nicht-Funktionale Anforderungen (NANF):

Benutzerfreundlichkeit (BF):

1. My.vay muss für alle Zielgruppen benutzerfreundlich sein.

Datenschutz und Sicherheit (DS):

1. Personenbezogene Daten müssen in my.vay gemäß der europäischen Datenschutzgrundverordnung 2016/679 verarbeitet werden.
2. Die Eintragung von Impfnachweisen muss in my.vay mit einer Zwei-Faktor-Authentifizierung des Patients abgesichert sein.

Kompatibilität (KO):

1. Die Benutzbarkeit und Funktionalität von my.vay soll in den aktuell verbreitesten Betriebssystemen sichergestellt sein, allerdings geringstenfalls in „iOS 14“ [100] und „Android 11“ [101].
2. Die Benutzbarkeit und Funktionalität von my.vay muss in den aktuell verbreitesten Internet Browsern sichergestellt sein, allerdings geringstenfalls in „Apple Safari 14.1.1“ [102] , „Google Chrome 91“ [103], „Mozilla Firefox 89“ [104] und „Microsoft Edge 46“ [105].
3. my.vay soll ein *Responsive Design* besitzen.

Zuverlässigkeit (ZU):

1. Im Durchschnitt soll my.vay nicht länger als zwei Sekunden laden. Sollte eine längere Wartezeiten auftreten, so muss die Plattform ein Ladesignal anzeigen.
2. my.vay muss bei dem Auftreten eines Fehlers eine Fehlermeldung anzeigen.

HCERT (HC):

1. Der Nachweis für eine Covid-19-Impfung muss gemäß der europäischen Leitlinie zur Formatierung und des Vertrauensmanagements bei digitalen Covid-19-Zertifikaten strukturiert und gespeichert werden [42].
2. Alle Nicht-Covid-19-Impfnachweise sollen an die europäische Leitlinie zur Formatierung und des Vertrauensmanagements bei digitalen Covid-19-Zertifikaten angelehnt werden [42].

5 Konzept einer Impfplattform

Im Mittelpunkt des vorliegenden Kapitels steht der Entwurf eines Konzeptes für die Impfplattform my.vay. Dazu wird auf Basis der zuvor durchgeführten Anforderungsanalyse die Plattform modelliert und in Form von geeigneten UML-Diagrammen dargestellt. Bei der Konzeption wird sich an den Sichten des „4+1 View Models“ orientiert, welches im ersten Abschnitt dieses Kapitels kurz vorgestellt wird. Im Nachgang daran folgen Abschnitte für jede Sicht auf my.vay: Szenarien, Logische Sicht, Entwicklungssicht, Ablaufsicht und Physikalische Sicht.

5.1 4+1 View Model

Das 4+1 View Model ist ein von Phillipe Kruchten im Jahr 1995 entworfenes Modell zur Beschreibung von Systemen mit komplexen Architekturen. Dabei wird das System aus unterschiedlichen Blickwinkeln betrachtet, um die Anforderungen aller Stakeholder des Systems sowie dessen Funktionalitäten im Konzept miteinzubeziehen [106].

Die in der Abbildung 5.1 dargestellten vier Sichten des 4+1 Modells weisen folgende Charakteristika auf:

- Die **Logische Sicht** befasst sich mit der objektorientierten Zerlegung und der Funktionalität von my.vay für den Endnutzer.
- Die **Entwicklungssicht** beschreibt my.vay aus dem Blickwinkel des Softwaremanagements für den Entwickler und zerteilt das System in einzelne Teilsysteme.
- Die **Ablaufsicht** beschäftigt sich mit der dynamischen Sicht von my.vay und zeigt einzelne Prozesse sowie deren Leistung und Laufzeitverhalten auf.
- Die **Physikalische Sicht** beschreibt my.vay aus der Perspektive der Entwickler und stellt die Verteilung der Softwarekomponenten auf physikalischer Ebene dar.

Die Darstellung 5.1 charakterisiert neben den vier Sichten Szenarien als weitere Komponente, welche zusätzlich alle Perspektiven zusammenfügt und my.vay aus einer weiteren Perspektive, der Gesamtperspektive, beschreibt:

- Die **Szenarien** bilden alle relevanten Hauptanwendungsfälle für my.vay ab. Sie veranschaulichen die Abläufe zwischen den Komponenten und den verschiedenen Nutzern und helfen dabei einzelne Architekturelemente zu überprüfen [106].

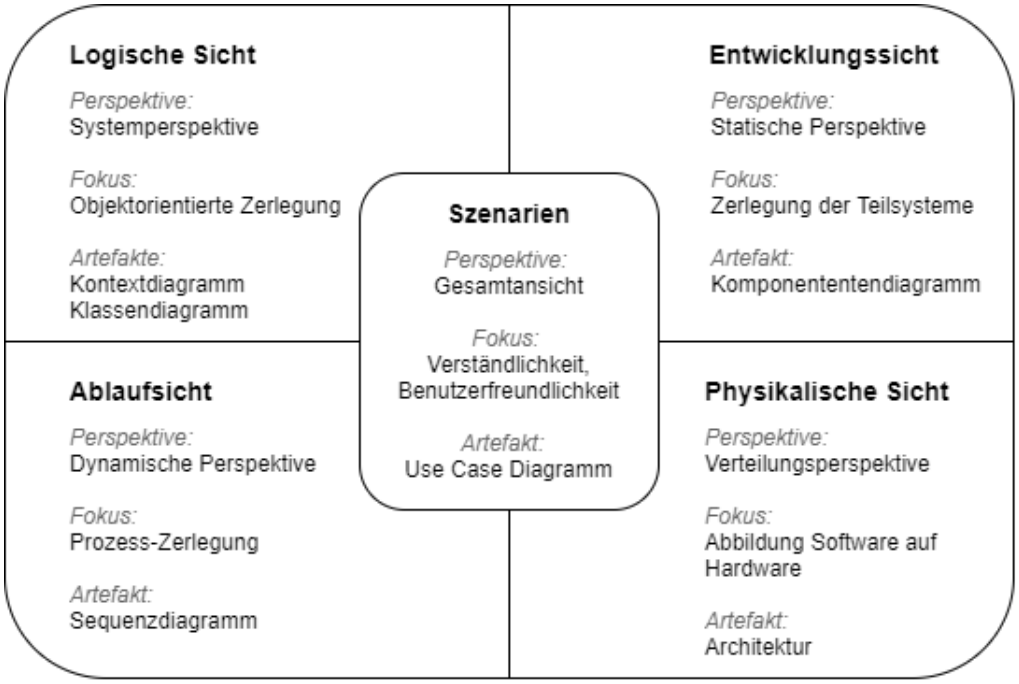

Abbildung 5.1: 4+1 View Model
(Quelle: Darstellung in Anlehnung an [106])

5.2 Szenarien

Die verschiedenen Szenarien erfassen die Anforderungen und spezifizieren die Funktionalität der zu entwickelnden Anwendung. Zur Darstellung dieser wird im Rahmen des Softwareentwicklungsprozesses auf *Use Case*-Diagramme zurückgegriffen. Bei einem *Use Case*-Diagramm handelt es sich um eine grafische Repräsentation eines Anwendungsfalls. Dieser wird aus der Perspektive der verschiedenen Nutzer beschrieben und zeigt die typischen Interaktionen im Rahmen des Systems auf. Ein einzelner Anwendungsfall stellt demnach einen Teil des gesamten Systemverhaltens dar [107]. Im Folgenden werden die Hauptanwendungsfälle, welche aus den im Kapitel 4.2 formulierten Anforderungen resultieren, grafisch abgebildet und beschrieben.

5.2.1 Account management

In dem Anwendungsfall „Account management“ agieren folgende Akteure: Patient und Issuer. Ein Patient kann sich in my.vay registrieren, anmelden und abmelden (siehe Anforderung der Patient Application (ANF-PA) Nummer (Nr.) 1). Des Weiteren kann ein Patient seine hinterlegten Accountinformationen bearbeiten oder gänzlich löschen. Ein Issuer kann sich analog zum Patient ebenfalls mit einem Account registrieren, anmelden und abmelden (siehe Anforderungen der Issuer Application (ANF-IA) Nr. 1), sowie seine Accountdaten bearbeiten und löschen.

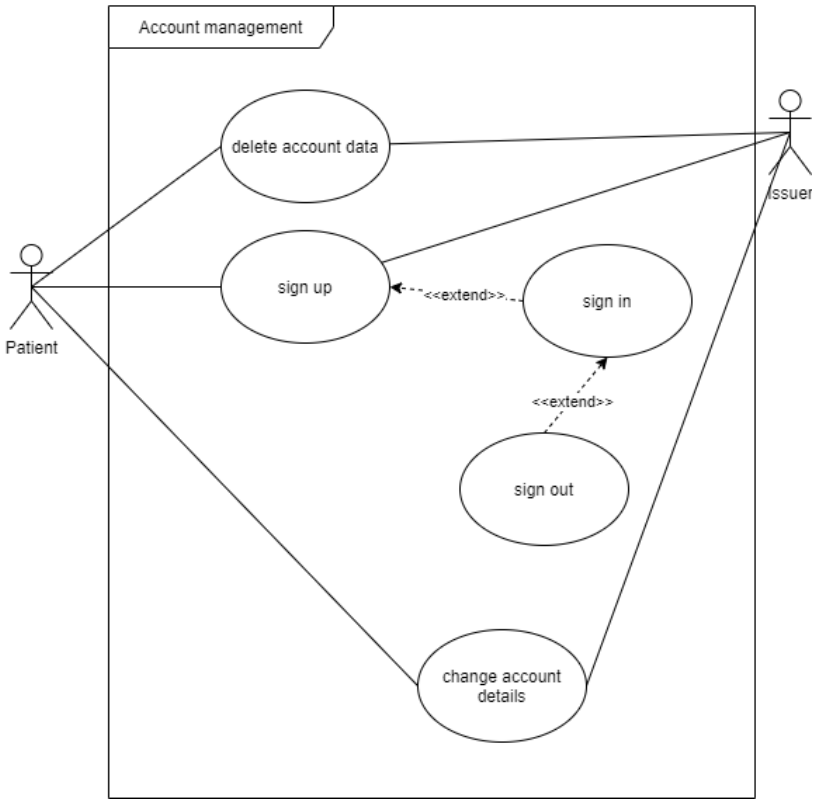

Abbildung 5.2: Use Case Diagram nach UML-Notation 2.5 zu den Interaktionen zwischen Nutzergruppen und *Account management*

5.2.2 Vaccination data management

Der nachfolgende Anwendungsfall „Vaccination data management“ zeigt das Verhalten der folgenden Akteure auf: Patient, Issuer und Researcher. Ein Patient kann seine Impftermine eintragen und an bevorstehende Impftermine erinnert werden, entsprechend ANF-PA Nr.

10. Zusätzlich kann er seine Impfnachweise hinzufügen (siehe ANF-PA Nr. 3), abrufen und selbst erfahrene Nebenwirkungen eintragen, gemäß ANF-PA Nr. 11. Darüber hinaus kann ein Patient seine Impfdaten sowohl mit dem Issuer (siehe ANF-PA Nr. 13), als auch anonymisiert mit dem Researcher teilen (siehe ANF-PA Nr. 12) oder gar löschen. Ein Issuer kann Impfnachweise ausstellen, diese in sein Praxisverwaltungssystem übertragen und die mit ihm geteilten Impfnachweise eines Patients empfangen, gemäß ANF-IA Nr. 2, 4 und 5. Ein Researcher kann mittels der mit ihm geteilten anonymisierten Impfdaten Statistiken erstellen (siehe Anforderungen der Researcher Application (ANF-RA) Nr. 3).

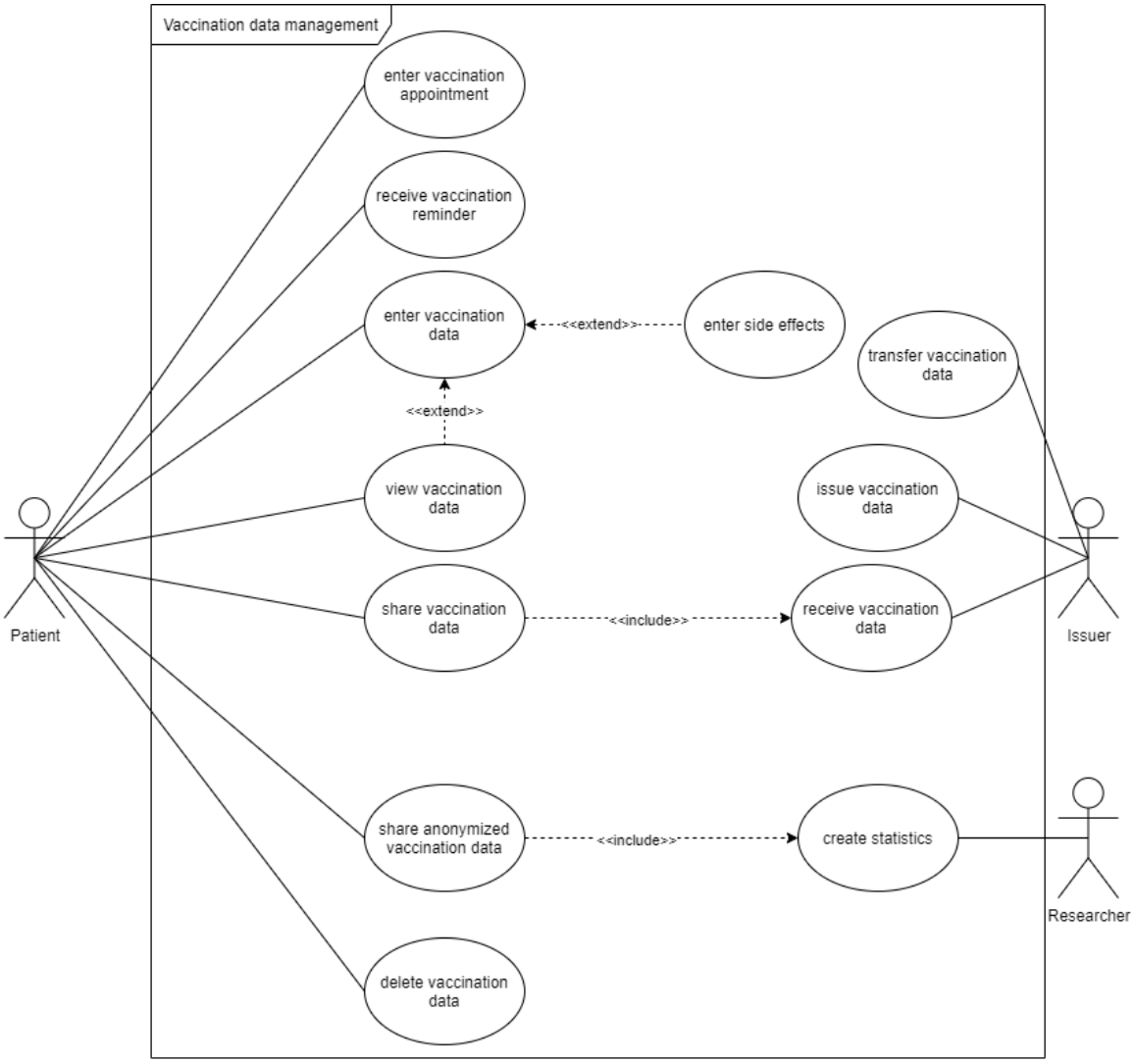

Abbildung 5.3: *Use Case Diagram* nach UML-Notation 2.5 zu den Interaktionen zwischen Nutzergruppen und *Vaccination data management*

5.2.3 Proof management

Im „Proof management"-Anwendungsfall werden die Verhaltensweisen des Patients und Verifiers beleuchtet. Ein Patient kann seinen Impfnachweis anzeigen und ausdrucken, gemäß ANF-PA Nr. 2 und 5. Ein Verifier kann den vom Patienten vorgelegten Impfnachweis überprüfen (siehe Anforderungen der Verifier Application (ANF-VA) Nr. 2).

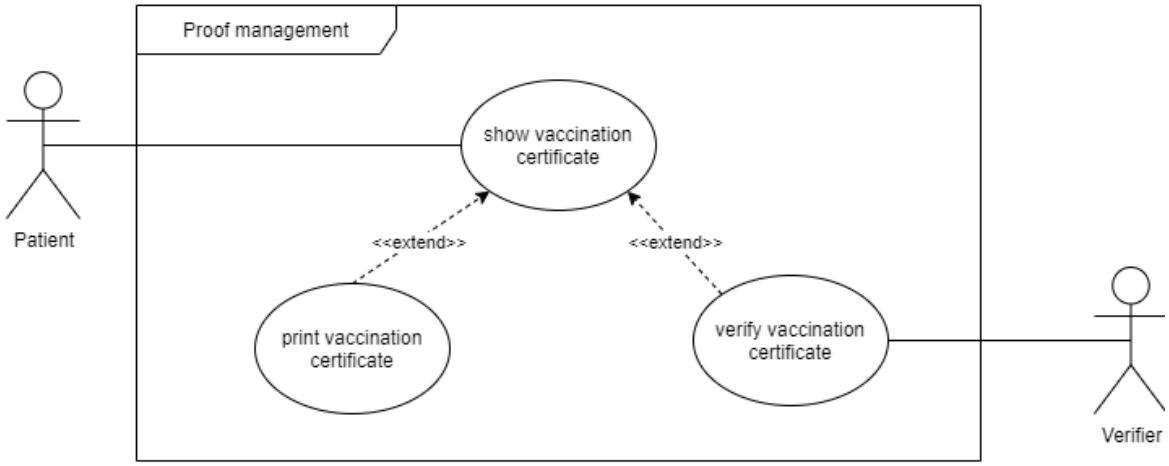

Abbildung 5.4: *Use Case Diagram* nach UML-Notation 2.5 zu den Interaktionen zwischen Nutzergruppen und *Proof management*

5.2.4 Knowledge management

Der Anwendungsfall „Knowledge management" beschreibt, wie die Akteure Patient und Issuer agieren. Ein Patient kann den RKI-Impfkalender und grundsätzliche Impfinformationen einsehen sowie nach bestimmten Impfinformationen suchen, enstprechend ANF-PA Nr. 6, 7 und 8. Ein Issuer kann ebenfalls grundsätzliche Impfinformationen einsehen und nach diesen suchen (siehe ANF-IA Nr. 6).

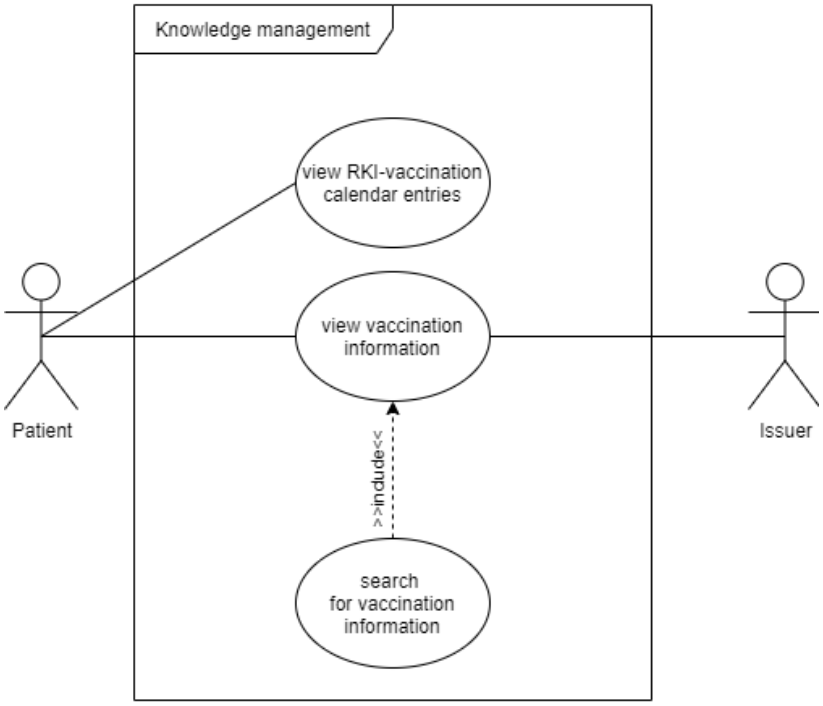

Abbildung 5.5: *Use Case Diagram* nach UML-Notation 2.5 zu den Interaktionen zwischen Nutzergruppen und *Knowledge management*

5.3 Logische Sicht

5.3.1 Kontextdiagramm

Im Folgenden werden die Beziehungen der Plattform my.vay mit dessen Umwelt analysiert und dargestellt. Dadurch kann festgestellt werden, welche Funktionalitäten implementiert werden müssen und welche Dienste von anderen Systemen bezogen werden können. Anhand der Ergebnisse können anschließend Entscheidungen zu der Struktur der Plattform getroffen werden [108].

Das Kontextdiagramm in Abbildung 5.6 veranschaulicht die Systeme in der Umgebung der Plattform my.vay. Damit ein Verifer den QR-Code eines Patients auf Echtheit überprüfen kann, wird ein QR-Scanner benötigt, mit welchem die Plattform zusammenarbeitet (siehe ANF-VA Nr.2). Mittels einer Schnittstelle zu den Praxisverwaltungs- bzw. Krankenhausinformationssystemen soll es Issuern (Ärzten) ermöglicht werden, das eigene System automatisch mit dem digitalen Impfpass des Patients in my.vay zu aktualisieren (siehe

ANF-IA Nr.4). Die Verbindung zu dem DGCG wird benötigt, damit über die Plattform auf QR-Codes der Impfungen zugegriffen und diese auf Richtigkeit überprüft werden können (siehe nicht-funktionale Anforderungen HCERT (NANF-HC) Nr.1).

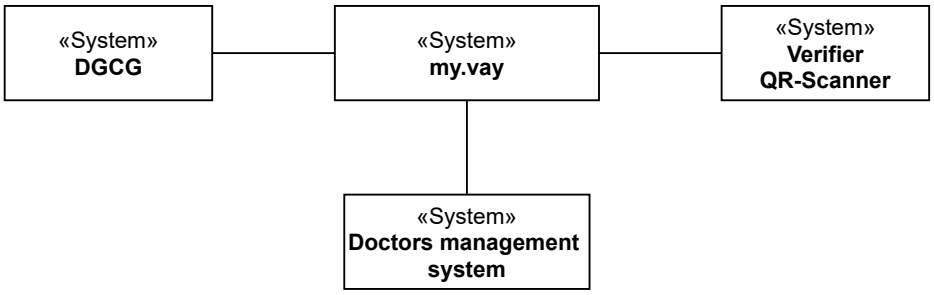

Abbildung 5.6: Kontextdiagramm nach UML-Notation 2.5 von my.vay

5.3.2 Klassendiagramm

Im Nachgang an die Visualisierung der Umgebung von my.vay wird nun das System selbst durch ein Klassendiagramm näher spezifiziert. Im Allgemeinen lassen sich Klassendiagramme in verschiedenen Softwareentwicklungsphasen einsetzen und verschieden detailliert gestalten, sodass unterschiedliche Sachverhalte dargestellt werden können [109]. Nachfolgend wird das Klassendiagramm genutzt, um die wichtigsten Objekte, die das System my.vay beinhalten muss, sowie deren Daten und Assoziationen untereinander zu visualisieren. Als Ausgangspunkt dienen die in Kapitel 4.2 definierten Anforderungen und die theoretische Grundlage zum HCERT in Kapitel 2.2.1.

Das in Abbildung 5.7 dargestellte Klassendigramm beinhaltet neun Klassen, die das System my.vay umfassen soll. Ganz oben im Diagramm befindet sich die Klasse „User“, welche eine Generalisierung der zwei Klassen Issuer und Patient ist. Die Klasse User enthält alle Daten, die von jedem Nutzer von my.vay erfasst werden müssen. Dazu zählt unter anderem ein eindeutiger „identifier“ und ein „password“. Während die Klasse Patient lediglich die Daten des Users besitzt, hat die Klasse Issuer noch eine „adress“ und „signature“.

Die Klasse Patient verfügt neben der Aggregation noch über zwei weitere Assoziationen. Die Erste besteht zur Klasse „Calendar entry“, wobei ein Patient mehrere Calendar entries besitzen kann. Die Klasse Calender entry spiegelt die in Kapitel 4.2 definierte Anforderung wieder, dass ein Patient in my.vay anstehende Impftermine in einem Kalender eintragen und sich an diese erinnern lassen kann. Die zweite Assoziation des Patient besteht zur Klasse „HCERT“. Ein Patient kann hierbei zu mehreren HCERTs zugehörig sein.

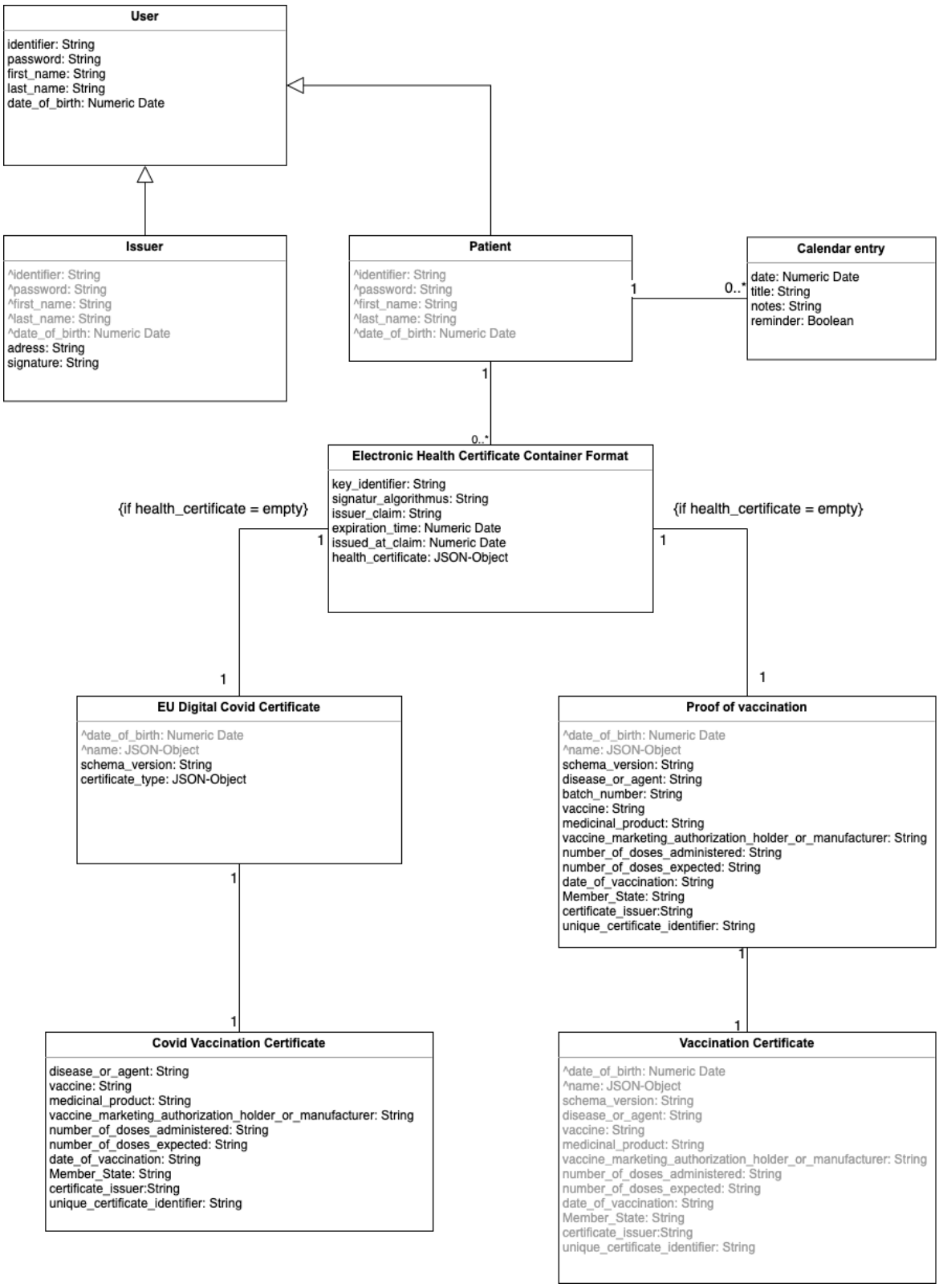

Abbildung 5.7: Klassendiagramm nach UML-Notation 2.5 von my.vay

Die Klassen HCERT, „EU Digital Covid Certificate“ und „Covid Vaccination Certificate“ repräsentieren zusammengenommen einen Impfnachweis für eine COVID-19-Impfung nach den Spezifikationen des digitalen COVID-19-Zertifikates der EU. Demzufolge liegt zwischen HCERT und EU Digital Covid Certificate sowie EU Digital Covid Certificate und Covid Vaccination Certificate jeweils eine Eins-zu-Eins-Assoziation vor. Die Daten innerhalb der einzelnen Klassen leiten sich ebenfalls von den Richtlinien zum digitalen COVID-19-Zertifikat der EU ab.

Die Klasse HCERT besitzt neben der Assoziation zum EU Digital Covid Certificate noch eine weitere zum „Proof of vaccination“. Die beiden zuvor angesprochen Assoziationen unterliegen einer speziellen Bedingung und zwar kann nur eine der beiden Assoziation gleichzeitig existieren. Unabhängig davon orientiert sich die Klasse Proof of vaccination zur Gewährleistung der Interoperabilität an den Richtlinien der EU zum digitalen COVID-19 Zertifikat. Deshalb enthält die Klasse nahezu die gleichen Daten wie eine Kombination aus den Klassen EU Digital Covid Certificate und Covid Vaccination Certificate. Lediglich die „batch number“ kommt zusätzlich noch hinzu, da diese im Impfpass der WHO verpflichtend ist. Zweck des Proof of vaccination ist es also den digitalen Nachweis der COVID-19-Impfung auf alle Impfungen auszuweiten.

Zusammenfassend bietet das Klassendiagramm eine erste Übersicht darüber, welche Daten für das System my.vay zwingend notwendig sind. Ferner wird Aufschluss darüber gegeben, zu welchem Objekt die Daten zugehörig sind und wie die Objekte untereinander zusammengehören.

5.4 Entwicklungssicht

Das Komponentendiagramm in Abbildung 5.8 stellt den Aufbau innerhalb der Plattform my.vay und somit die Entwicklungssicht dar. Die Plattform wird in folgende sechs Subsysteme zerteilt:

- „Patient account management“
- „Issuer account managament“
- „Resarcher account managament“
- „Knowledge management“

- „Vaccination data management“
- „Proof management“

Das Patient account management ist für die Funktionalität und Verwaltung des Patient-*Accounts* verantwortlich. Das Subsystem besteht aus den drei Komponenten „Account Manager“, „Authentication“ und „QR Scanner“. In der Account Manager-Komponente gibt es einen „Application Manager“ sowie einen „Database Manager“. Die beiden Unterkomponenten sind mittels der Schnittstelle „transfer account data“ verbunden. Mithilfe dieser Schnittstelle werden die Accountdaten eines Patients aus dem Database Manager an den Application Manager übermittelt. Wie in ANF-PA Nr. 1 definiert, besteht die Authentication-Komponente aus den Unterkomponenten „Registration“ und „Login“, durch welche sich ein Patient registrieren beziehungsweise einloggen kann. Durch die Schnittstelle „authenticate account“ kann die Komponente Authentication die Daten, die ein Patient beim Registrieren beziehungsweise Anmelden angibt, an die Komponente Account Manager übergeben. Diese überprüft wiederum die Daten auf Richtigkeit, bevor ein Patient eingeloggt wird. Mithilfe der Komponente QR Scanner wird es dem Patient ermöglicht, eine getätigte Impfung mittels einem QR Code in seinen digitalen Impfpass in my.vay einzuscannen. Dies greift ANF-PA Nr. 3 auf.

Das Subsystem Issuer account manager ist ähnlich wie das eben beschriebene Subsystem aufgebaut. Der Unterschied zwischen den beiden Subsystemen liegt darin, dass dieses Subsystem statt der Komponente QR Scanner die Komponente „Vaccination Creator“ enthält, wodurch ein Issuer einen Impfeintrag erstellen kann (siehe ANF-IA Nr.2). Die Komponente ist mit dem Account Manager verbunden, damit die Daten des Issuers, wie bspw. Namen, Arzt-ID oder Signatur, bei der Erstellung eines Impfeintrags aufgenommen werden kann.

Einen ähnlichen Aufbau besitzt das Subsystem Researcher account management. Dieses besteht ebenfalls aus den Komponenten Account Manager und Authentication (siehe ANF-RA Nr.1). Die Funktionalität der Komponenten ist die gleiche wie bei den beiden vorigen Account Management-Subsystemen.

Ein weiteres Subsystem ist das Knowledge management. Durch dieses Subsystem sollen Patients und Issuer, wie in ANF-PA Nr.6, 7 und 8 sowie ANF-IA Nr.6 definiert, auf sämtliche Informationen von Impfungen zugreifen können. Dies ist durch die beiden Schnittstellen „transfer information“ dargestellt. Das Subsystem besteht lediglich aus der Komponente Database Manager, in welcher alle Daten gespeichert sind.

Das Subsystem Vaccination data management besteht aus den Komponenten „Digital vaccination record“, „Sideeffect Query“ und „Vaccination reminder“. Die Digital vaccination record-Komponente stellt dabei den digitalen Impfpass des Patients dar (siehe ANF-PA Nr.2). Dieser besteht aus vielen digitalen Einträgen, welche durch die Unterkomponente „Digital Vaccination“ dargestellt sind. Die einzelnen Einträge werden mithilfe des „QR Scanners“ des „Patient account management“-Subsystems eingetragen, welches ANF-PA Nr.3 aufgreift. Diese Verbindung ist mittels der Schnittstelle „transfer vaccination data“ dargestellt. Damit der digitale Impfpass vollständig ist, erhält das Subsystem durch die Schnittstelle „transfer data“ die Daten des Patients, wie bspw. Namen, Adresse und Geburtsdatum. Die Datenübergabe dieser Schnittstelle ist auch in die gegensätzliche Richtung implementiert, sodass die Daten eines neuen Eintrags gespeichert werden. Die Komponente „Sideeffect Query“ ermöglicht das Erfassen und Speichern von Nebenwirkungen einer durchgeführten Impfung, nach dem Ablauf eines bestimmten Zeitrahmens (siehe ANF-PA Nr.11). Diese Daten werden mithilfe der Schnittstelle „transfer sideeffect data“ in anonymisierter Form ebenfalls in der Datenbank des Researchers für Forschungszwecke gespeichert, entsprechend ANF-RA Nr.2. Die Komponente „vaccination reminder“ ist dafür zuständig den Patient rechtzeitig an eine anstehende Impfung zu erinnern, damit dieser einen Termin bei seinem Arzt ausmachen kann. Damit er weiß in welchen Abständen eine Auffrischung der Impfung erfolgen soll, erhält er die Daten vom Subsystem „Knowledge management“ über die Schnittstelle „transfer vaccination calendar data“ (siehe ANF-PA Nr.9). Des Weiteren stellt diese Komponente dem Patient einen visuellen Kalender zur Verfügung, in welchen er seine Impftermine eintragen kann, entsprechend ANF-PA Nr.10.

Das Subsystem Proof management enthält die Komponente „QR Generator“. Dieser erstellt einen QR-Code für einen bestimmten Impfeintrag, damit der Patient diesen dem Verifier zeigen kann und damit die Echtheit der Impfung nachweisen kann (siehe ANF-PA Nr.4). Hierfür wird der Status einer Impfung durch die Schnittstelle „transfer vaccination status“ vom „Vaccination data management“ zum Subsystem Proof management übermittelt.

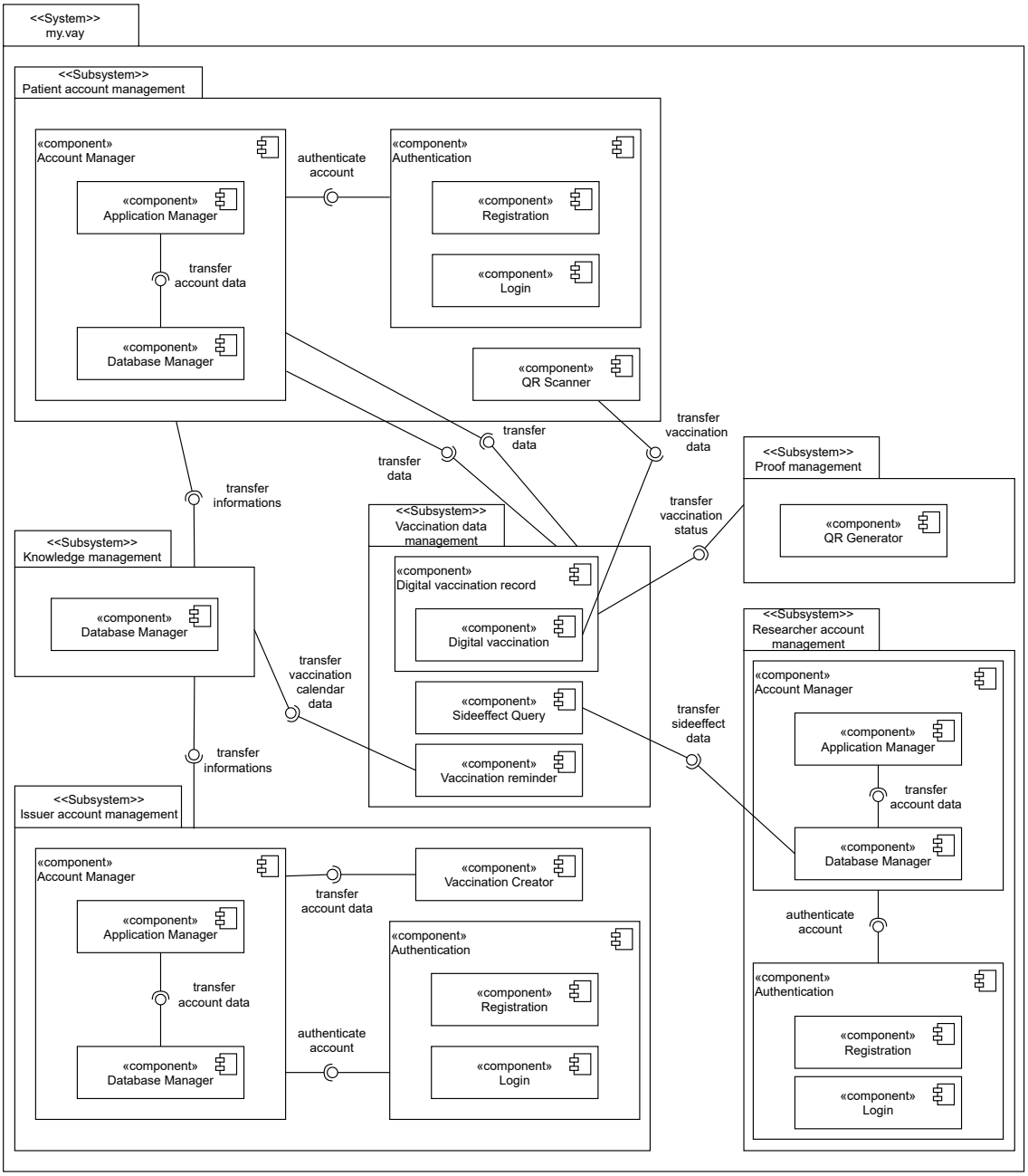

Abbildung 5.8: Komponentendiagramm nach UML-Notation 2.5 zu my.vay

5.5 Ablaufsicht

Die Ablaufsicht nach dem 4+1 View Model wird für dieses Konzept durch Sequenzdiagramme dargestellt. Sequenzdiagramme stellen wechselseitige Interaktionen des Systems

in einem zeitlichen Zusammenhang grafisch dar. Sie beschreiben den Austausch von Nachrichten zwischen einzelnen Objekten beziehungsweise gleichzeitig existierenden Prozessen zur Ausführung einer bestimmten Funktion innerhalb eines Systems [110]. Daher wird nachfolgend der für die Kernfunktionen des my.vay-Systems nötige Nachrichtenaustausch durch einzelne Sequenzdiagramme aufgezeigt und erläutert. Die Kernfunktionen leiten sich aus den zuvor definierten Anforderungen in Kapitel 4.2 ab.

5.5.1 Impfnachweis ausstellen

Für das Ausstellen eines Impfnachweises nutzt ein Issuer sein dafür vorgesehenes *Frontend*, das von den restlichen Nutzergruppen in my.vay nicht erreicht und eingesehen werden kann. Hierbei übergibt ein Issuer über ein Formular die Patienten- und Impfdaten, sodass ein QR-Code aus den Informationen der Eingabefelder generiert wird (siehe Anforderung ANF-IA Nr.2). Das Sequenzdiagramm in Abbildung 5.9 visualisiert diesen Prozess, welcher nachstehend näher ausgeführt wird.

1. Ein Issuer gibt in seinem *Frontend* die Daten der Impfung sowie des geimpften Patient ein.
2. Die Impfdaten werden nach der Eingabe an das zentrale „Vaccination Platform Backend“ übermittelt.
3. Für eine spätere Zwei-Faktor-Authentifizierung wird auf Basis der übermittelten Impf- und Patientendaten eine individuelle Transaktionsnummer (TAN) erstellt.
4. Die TAN sowie die in den Anforderungen definierte dazugehörige Gültigkeit (siehe NANF-HC Nr. 1 und 2) wird an das Objekt „Vaccination Datatabase Cluster“ übermittelt und gespeichert.
5. Nach erfolgreicher Speicherung wird dem „Issuer Frontend“ eine Meldung zurückgegeben, dass die TAN sowie die dazugehörige Gültigkeitsdauer erfolgreich gespeichert wurden.
6. Aus den im ersten Schritt übergebenen Impf- und Patientendaten wird im Issuer Frontend eine CBOR-Datenstruktur erzeugt.
7. Für das CBOR-Objekt wird ein Hashwert kalkuliert.
8. Über das Vaccination Platform Backend sowie das Vaccination Database Cluster wird die persönliche elektronische Signatur des Issuers beim „Sign Service“ angefordert.

9. Ist die Signatur erstellt worden, so wird das nötige Signaturzertifikat zunächst an das Vaccination Platform Backend zurückgegeben.
10. Die im Vaccination Platform Backend gespeicherte TAN wird nach der Erstellung eines gültigen Signaturzertifikats anschließend an den Patient zurückgegeben.
11. Das erstellte Signaturzertifikat wird anschließend an das Issuer Frontend übergeben, sodass aus dem CBOR-Objekt, dem generierten Hashwert und dem Signaturzertifikat ein QR-Code als Impfnachweis erstellt werden kann.
12. Der erstellte QR-Code wird zunächst dem Issuer angezeigt, sodass dieser dem Patient diesen übergeben kann.

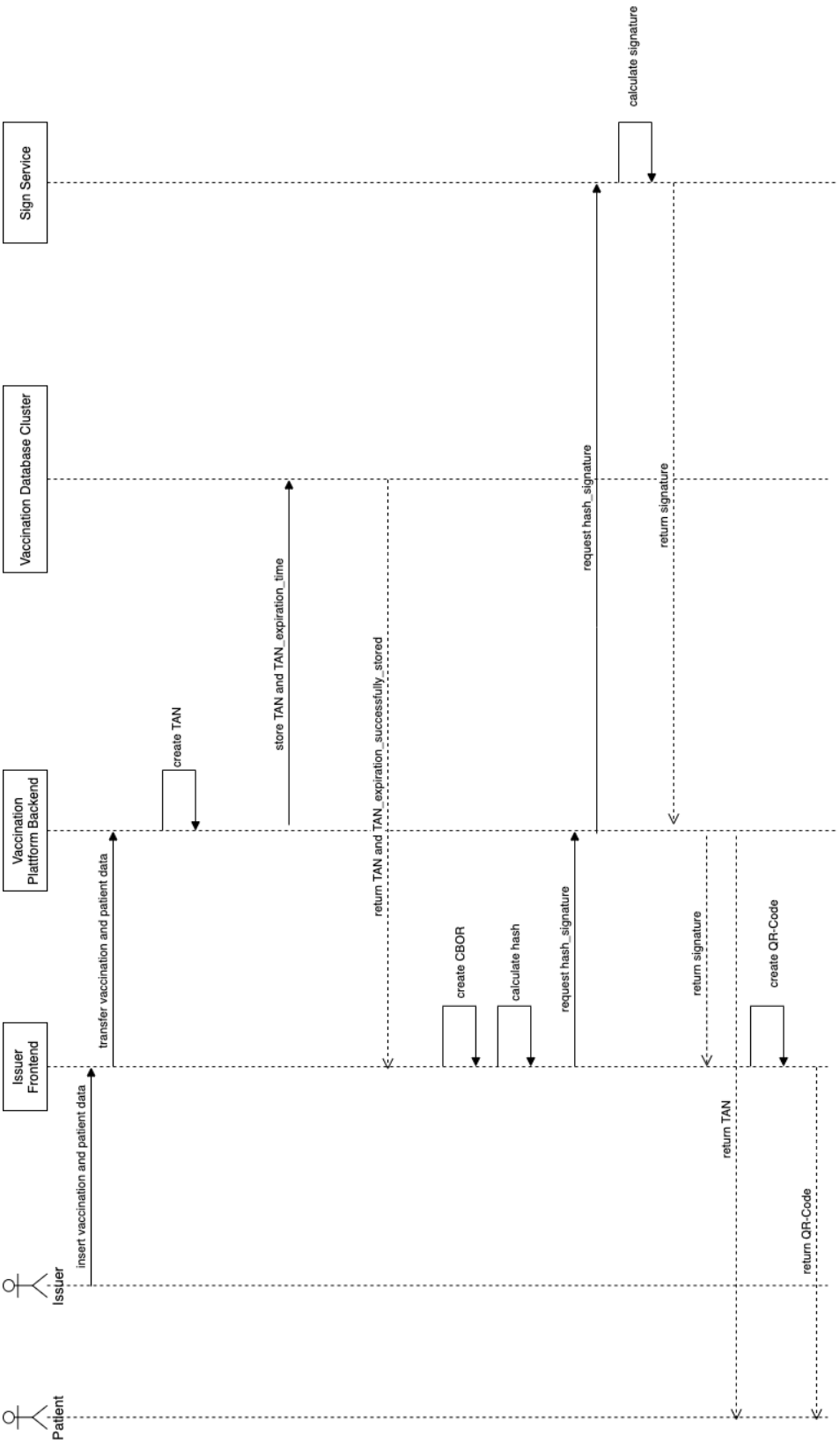

Abbildung 5.9: Sequenzdigramm nach UML-Notation 2.5 zur Erstellung des Impfnachweises

5.5.2 Daten des QR-Codes auslesen

Mit der Funktion des Auslesens des QR-Codes kann ein Patient den durch einen Issuer erstellten QR-Code auslesen. Ziel dahinter ist es, dass die in einem QR-Code enthaltenen Impfdaten in das persönliche Profil beziehungsweise in den digitalen Impfpass des Patients übertragen werden (siehe Anforderung ANF-PA Nr. 3 und 4). Der Prozess sowie der dazugehörige Daten- und Nachrichtenaustausch ist Abbildung 5.10 zu entnehmen und wird im Folgenden beschrieben.

1. Der Patient scannt den erhaltenen QR-Code und übergibt dadurch die darin enthaltenen Patient- sowie Impfdaten dem „Patient Frontend“.
2. Im Patient Frontend werden die Daten anschließend ausgelesen.
3. Für die Validierung des Signaturzertifikats des Issuers wird nach dem Auslesen der Daten die Signatur an den Sign Service übergeben.
4. Der Sign Service prüft daraufhin die Signatur beziehungsweise das Signaturzertifikat auf Richtigkeit.
    a) Ist das Signaturzertifikat ungültig, so wird dies an das Patient Frontend zurückgegeben. Der Patient bekommt dies anschließend in seiner Anwendung angezeigt.
    b) Ist das Signaturzertifikat gültig, so wird dies an das Patient Frontend zurückgegeben. Der Patient bekommt dies anschließend in seiner Anwendung angezeigt.
5. Bekommt ein Patient angezeigt, dass die Signatur und damit der QR-Code gültig ist, so übergibt er die erhaltene TAN an das Patient Frontend.
6. Die einmalige und eindeutige Kennung des QR-Codes sowie die eingegebene TAN werden an das Vaccination Database Cluster übergeben.
7. Im Vaccination Database Cluster wird zunächst die Gültigkeit der TAN durch den Abgleich der Gültigkeitsdauer überprüft.
    a) Ist die Gültigkeitsdauer überschritten, so wird dies an das Patient Frontend gesendet und ein Patient bekommt angezeigt, dass die TAN aufgrund überschrittener Zeit nicht mehr gültig ist.

   b) Ist die Gültigkeitsdauer unterschritten, so wird im Vaccination Database Cluster überprüft, ob die TAN-Eingabe des Patients mit der tatsächlich durch den Issuer erstellte TAN und der zugehörigen Identifikationskennung übereinstimmen.

8. Stimmt die TAN-Eingabe mit der tatsächlich erstellen TAN und deren Hashwert überein, so wird dies an das Patient Frontend weitergeleitet und ein Patient bekommt angezeigt, dass die TAN gültig ist.
9. Anschließend werden die noch im Patient Frontend gespeicherten gehashten Daten des QR-Codes an das Vaccination Database Cluster übermittelt.
10. Im Vaccination Database Cluster werden die transferierten Daten gehasht gespeichert.
11. Die gespeicherten Daten werden anschließend an das Patient Frontend zurückgegeben und dort entschlüsselt.
12. Abschließend werden die entschlüsselten Daten dem Patient in seinem digitalen Impfpass in my.vay angezeigt.

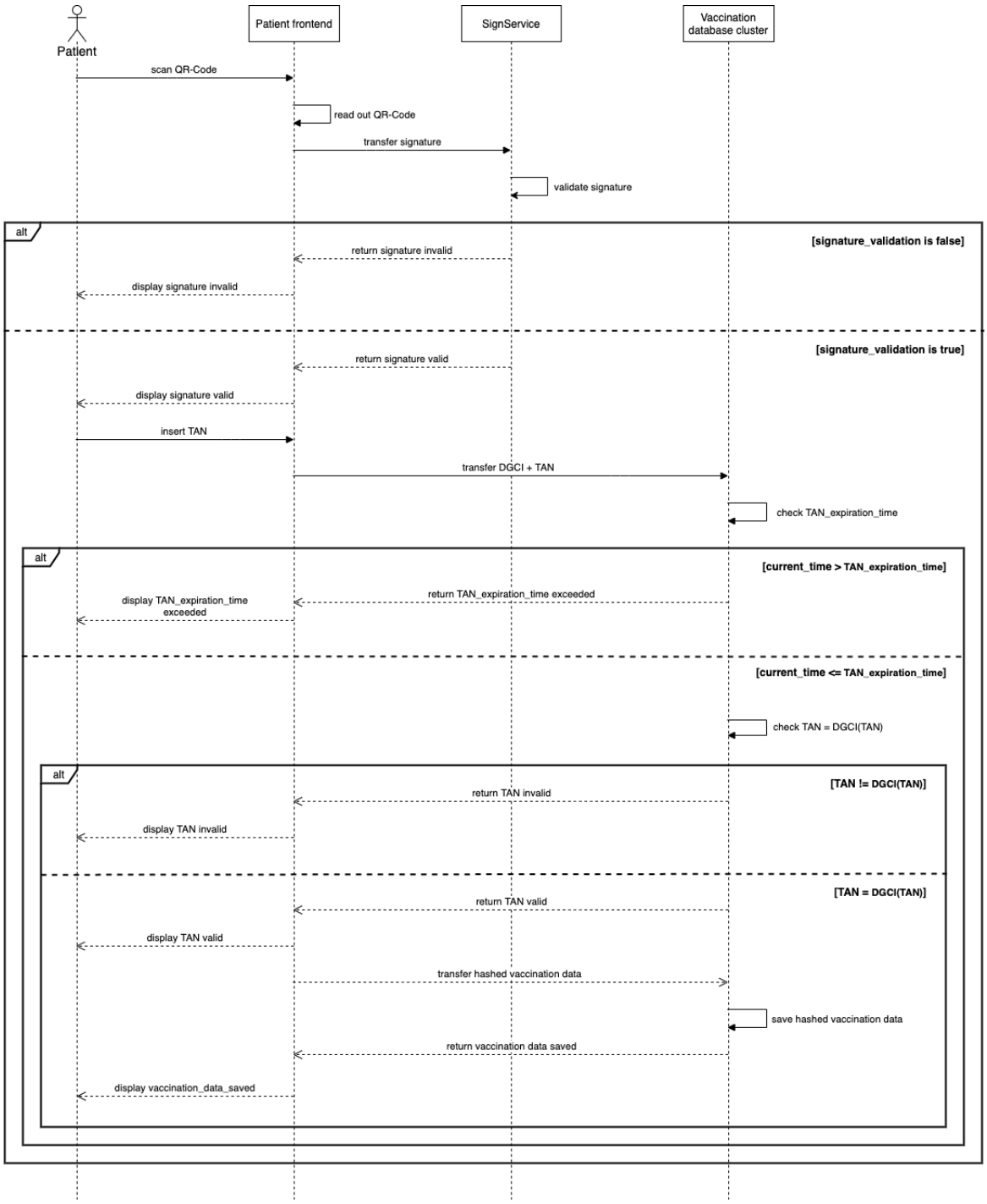

Abbildung 5.10: Sequenzdiagramm nach UML-Notation 2.5 zum Einlesen der Daten eines QR-Codes

5.5.3 Impfdaten anonymisiert zur Verfügung stellen

Wie bereits in den Anforderungen in Kapitel 4.2 aufgelistet, soll ein Patient die Möglichkeit bekommen, seine im digitalen Imfpass gespeicherten Impfdaten anonymisiert für Forschungszwecke zur Verfügung zu stellen. Diese Funktionalität wird insbesondere durch die ANF-RA Nr. 2 und ANF-PA Nr. 12 bestimmt. Hierzu nutzt ein Patient das Patient Frontend, wie in der nachstehenden Abbildung 5.11 ersichtlich.

1. Die übergebenen Impfdaten des digitalen Impfasses beziehungsweise einer einzelnen Impfung werden im Patient Frontend zunächst anonymisiert.
2. Die anonymisierten Daten werden danach zuerst an das Vaccination Platform Backend und anschließend an das Vaccination Database Cluster übergeben.
3. Im Vaccination Database Cluster werden die anonymisierten Daten gespeichert.
4. Nach erfolgreicher Speicherung der Daten bekommt ein Patient die Meldung in seinem Patient Frontend, dass die Daten erfolgreich gespeichert werden konnten.

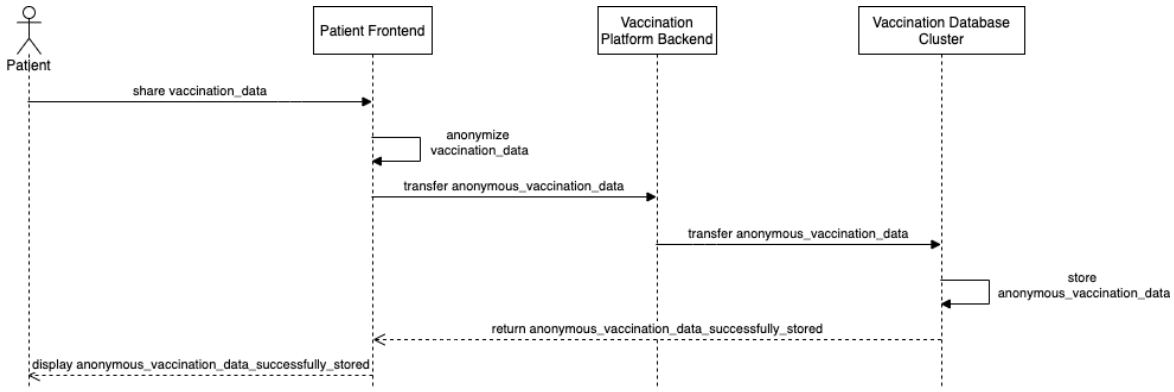

Abbildung 5.11: Sequenzdiagramm nach UML-Notation 2.5 zum anonymisierten Teilen von Impfdaten

5.5.4 Abrufen von Forschungsdaten zu Impfungen

Für das Abrufen von Forschungsdaten bezüglich getätigter Impfungen ist vorgesehen, dass ein Researcher einen Suchbegriff eingeben kann, der zu diesem passende Informationen zurückgibt. Diese können bei Bedarf heruntergeladen werden. Der Prozess dieser Funktion ist in Abbildung 5.12 dargestellt und leitet sich aus den Anforderungen ANF-RA Nr. 1 und 2 ab.

1. Im „Researcher Frontend“ wird ein Suchbegriff in einem dafür vorgesehenen Eingabefeld eingegeben.
2. Nach diesem Suchbegriff wird im gesamten Vaccination Database Cluster gesucht.
3. Informationen zu dem Suchbegriff werden an das Vaccination Platform Backend zurückgegeben und dort sortiert.
4. Die sortierten Suchinformationen werden an das Researcher Frontend zurückgegeben und dort einem Researcher angezeigt.
5. Ist ein Researcher im Begriff Informationen herunterzuladen, so wählt er dies sowie die zu herunterladenen Informationen in seinem Frontend aus.
6. Es wird eine Anfrage nach den Informationen an das Vaccination Platform Backend weitergegeben, die dort weiterhin sortiert gespeichert sind.
7. Die passenden Informationen werden von dem Vaccination Platform Backend an das Researcher Frontend zurückgegeben, in welchem sie durch den Researcher heruntergeladen werden können.

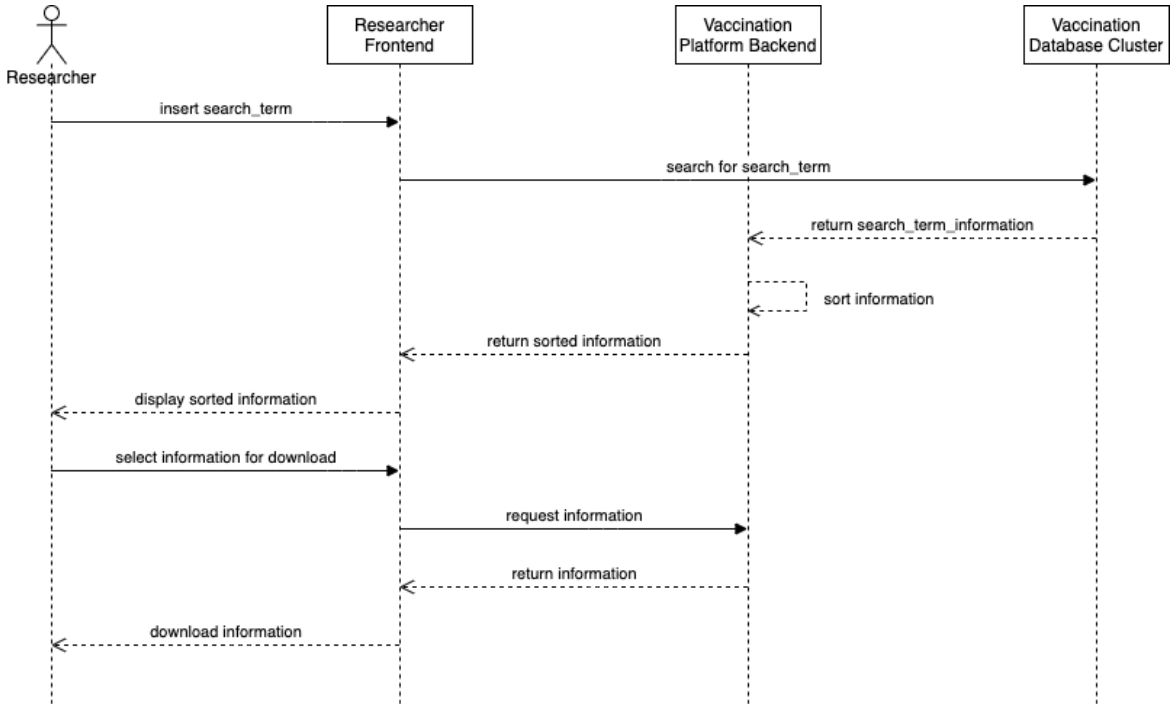

Abbildung 5.12: Sequenzdiagramm nach UML-Notation 2.5 zum Abrufen von Forschungsdaten zu Impfungen

5.5.5 Wissensmanagement des Patients und Issuers

Sowohl Patients als auch Issuer können in einer integrierten Wissensdatenbank von my.wy Impfinformationen abrufen (siehe Anforderung ANF-PA Nr. 6 und ANF-IA Nr. 6). Wie in Abbildung 5.13 und 5.14 zu sehen ist, unterscheiden sich beide Sequenzdiagramme der Nutzergruppen nur in dem jeweiligen *Frontend*.

1. Ein Patient oder Issuer übergibt dem jeweiligen *Frontend* in einem dafür vorgesehenen Eingabefeld einen Suchbegriff.
2. Dieser Suchbegriff wird anschließend an die Wissensdatenbank „Knowledge Database“ weitergegeben.
3. Mit einer Datenbankabfrage wird in dem Objekt „Knowledge Database“ nach dem Suchbegriff gesucht.
4. Nach erfolgreicher Datenbankabfrage werden die zu dem Suchbegriff passenden Informationen dem Patient/Issuer Frontend zurückgegeben, sodass diese einem Patient/Issuer angezeigt werden können.

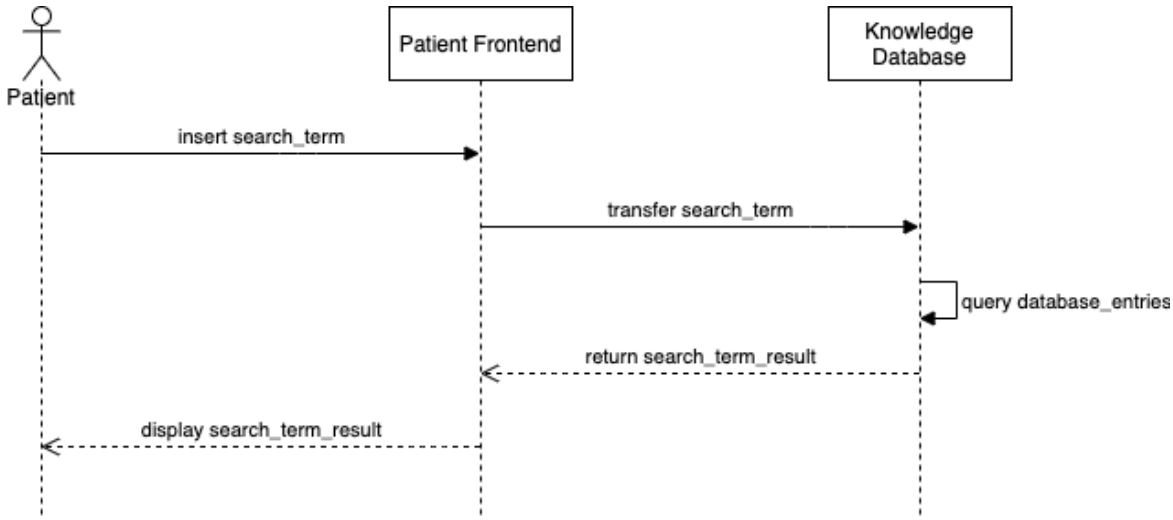

Abbildung 5.13: Sequenzdiagramm nach UML-Notation 2.5 zur Suchfunktion des Patient im Knowledge management

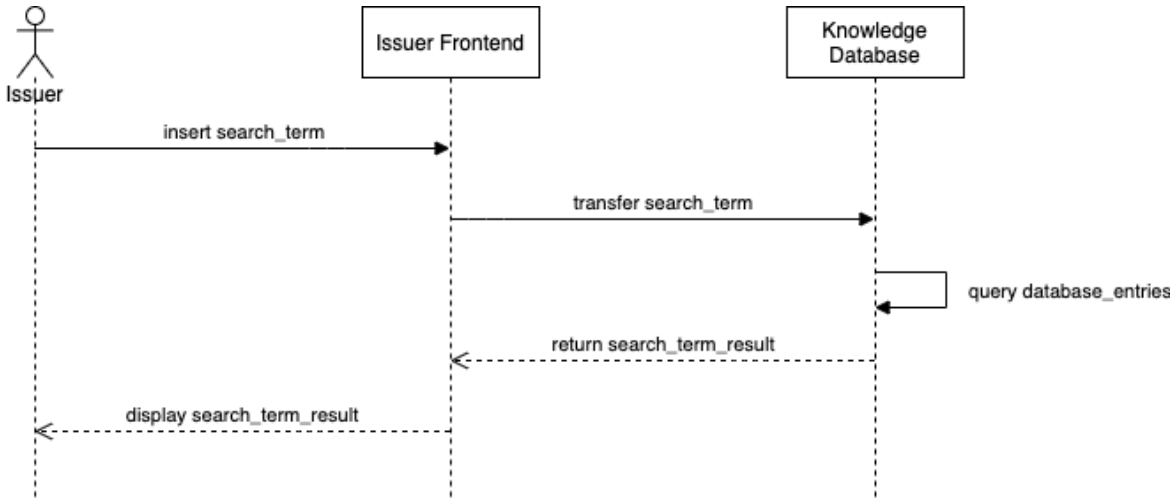

Abbildung 5.14: Sequenzdiagramm nach UML-Notation 2.5 zur Suchfunktion des Issuer im Knowledge management

5.6 Physikalische Sicht

Die Architektur der my.vay-Plattform soll mithilfe eines Schichtenmodells dargestellt werden. Dieses verdeutlicht die Verteilung der Plattform in Bezug auf die benötigten Hardwarekomponenten und deren Vernetzung. Damit die größtmögliche Datensicherheit gewährleistet werden kann, sollen die Hardwarekomponenten der Plattform dezentral aufgebaut sein (siehe Kapitel 2.3.1). Wie in Abbildung 5.15 zu sehen ist, besteht die Architektur aus drei miteinander verknüpften Schichten.

In der obersten Schicht sind die verschiedenen *User* dargestellt, welche auf my.vay zugreifen. Zu diesen zählen der Patient, der Researcher, der Verifier und der Issuer. Jede Nutzergruppe kann über eine individuelle Schnittstelle auf die my.vay-Plattform zugreifen. In der zweiten Schicht my.vay ist für jeden *User* daher ein spezifisches *Frontend* und *Backend* dargestellt, über welches das eigentliche Plattform genutzt und die erforderlichen Daten geladen werden können. Auf der zweiten Schicht sind außerdem zwei Programmierschnittstellen (hier: *APIs*) zu anderen Systemen dargestellt. Diese sind, wie auch im Kontextdiagramm in Kapitel 5.3.1 aufgewiesen, das DGCG und das Doctors Management System (DMS) beziehungsweise PVS.

In der untersten Schicht sind schließlich die verteilten „Data Sources“ modelliert. Diese Schicht zeigt, wie die Daten der Patients und der Issuer physikalisch verteilt gespeichert werden sollen, um sicher, verfügbar, zuverlässig und skalierbar zu sein. Sowohl für den Patient als auch für den Issuer gibt es einen spezifischen „Head Master“, welcher jeweils mit

weiteren *Mastern* verbunden ist. Die Head Master sollen für den entsprechenden Nutzer die benötigten Daten anfordern und gebündelt an das *Backend* in my.vay weiterleiten können. Die Head Master selbst sollen somit keine Daten speichern, sondern lediglich zusammenführen. Der „Patient Master" soll dabei nur auf Daten des Patients zugreifen können und der „Issuer Master" nur auf Daten des Issuers. Unter den beiden Head Mastern sind jeweils weitere „Position-Master" dargestellt, welche mit den letztendlichen „Data Nodes" verbunden sind. Die Position-Master sind mit den Zahlen eins, zwei und drei betitelt, welche stellvertretend für drei verschiedene Standorte stehen sollen. Die Anzahl der Standorte kann bei der Realisierung der Architektur variieren und ist nicht auf die Menge drei standardisiert. Durch die Speicherung der Daten an verschiedenen Standorten soll die Verfügbarkeit und Zuverlässigkeit der Plattform gewährleistet werden können. Sowohl die Head Master als auch die Position-Master sollen außerdem jeweils mit einem „Recovery Master" verbunden sein. Dieser Recovery Master soll über eine „Heartbeat"-Messung ständig die Verfügbarkeit und Funktion des eigentlichen *Masters* überprüfen und bei einem Ausfall dafür sorgen, dass sowohl die Plattform als auch die Daten schnellstmöglich wieder verfügbar sind.

Auf der untersten Ebene in der Data Source-Schicht sind die Data Nodes dargestellt, welche die eigentlichen Daten speichern. Unterhalb des Patient Masters sind die Data Nodes erneut mit den Zahlen eins bis drei nummeriert. Dies soll abbilden, dass die Patient Daten beim Speichern in beispielsweise drei Teile aufgeteilt werden. „Patient Data Node 1" enthält somit andere Daten als „Data Node 2" und „Data Node 3". Um alle Daten des Patients einsehen zu können muss der Patient Master somit die Daten aus allen drei Data Nodes jeweils anfordern. Jeder Position-Master soll außerdem nur zwei der drei Data Node-Teile verwalten. Auch hier kann je nach Implementierung die Anzahl der Data Nodes variieren. Diese Aufteilung der Standorte und verschiedenen Data Nodes soll für eine maximale Sicherheit der Patient Daten sorgen. Sollte ein Position-Master gehackt werden, sind die Patient Daten nicht verwendbar, da jeweils ein drittel der Daten fehlen würde. Erst über das Zusammensetzen im Head Master werden die Daten wieder nutzbar. Jede Patient Data Node existiert außerdem zweimal an verschiedenen Standorten, welches erneut die Ausfallsicherheit garantiert.

Das *Backend* des Patients verfügt zusätzlich über eine Verbindung zum „Researcher Master". Diese Verbindung dient dazu, dass vom Patient eingetragene Impfnebenwirkungen direkt anonymisiert in der „Sideeffect Data Node" abgespeichert werden kann, entsprechend ANF-PA Nr.12. Durch die Anonymisierung soll die Identifizierung des Patients anhand seiner Einträge verhindert werden. Auch der Researcher kann auf den Researcher Master

zugreifen und erhält so für Forschungszwecke Zugriff auf die gespeicherten Nebenwirkungen (siehe ANF-RA Nr.3). Der Researcher Master verfügt wie auch die anderen „Master“ über verschiedene Position Master, um einen Serverausfall zu vermeiden. Innerhalb der Position Master ist neben der jeweiligen Sideeffect Data Node eine „Researcher Data Node“ untergebracht, welche die Profildaten des Researchers beinhaltet und nicht von einem Patient ausgelesen werden darf.

Auch die Daten des Issuers werden exemplarisch an zwei verschiedenen Standorten gespeichert. Anders als die Patient Daten werden die Daten hier jedoch nicht aufgesplittet, da für den Issuer weniger sensible Daten gespeichert werden. Aus diesem Grund werden für den Issuer für das beispielhafte Aufzeigen auch nur zwei Position-Master benötigt. Eine Besonderheit stellt der „Master Position 3“ mit dem „TAN Data Node“ da. Auf diesen *Master* greift sowohl der „Issuer Head Master“ als auch der Patient Head Master zu. Auf dieser Data Node sollen zeitbefristet alle TAN-*Codes* gespeichert werden, die von einem Issuer beim Ausstellen eines Impfzertifikats generiert werden und vom Patient beim Einscannen des QR-Codes benötigt werden (siehe nicht-funktionale Anforderungen Datenschutz und Sicherheit (NANF-DS) Nr.2).

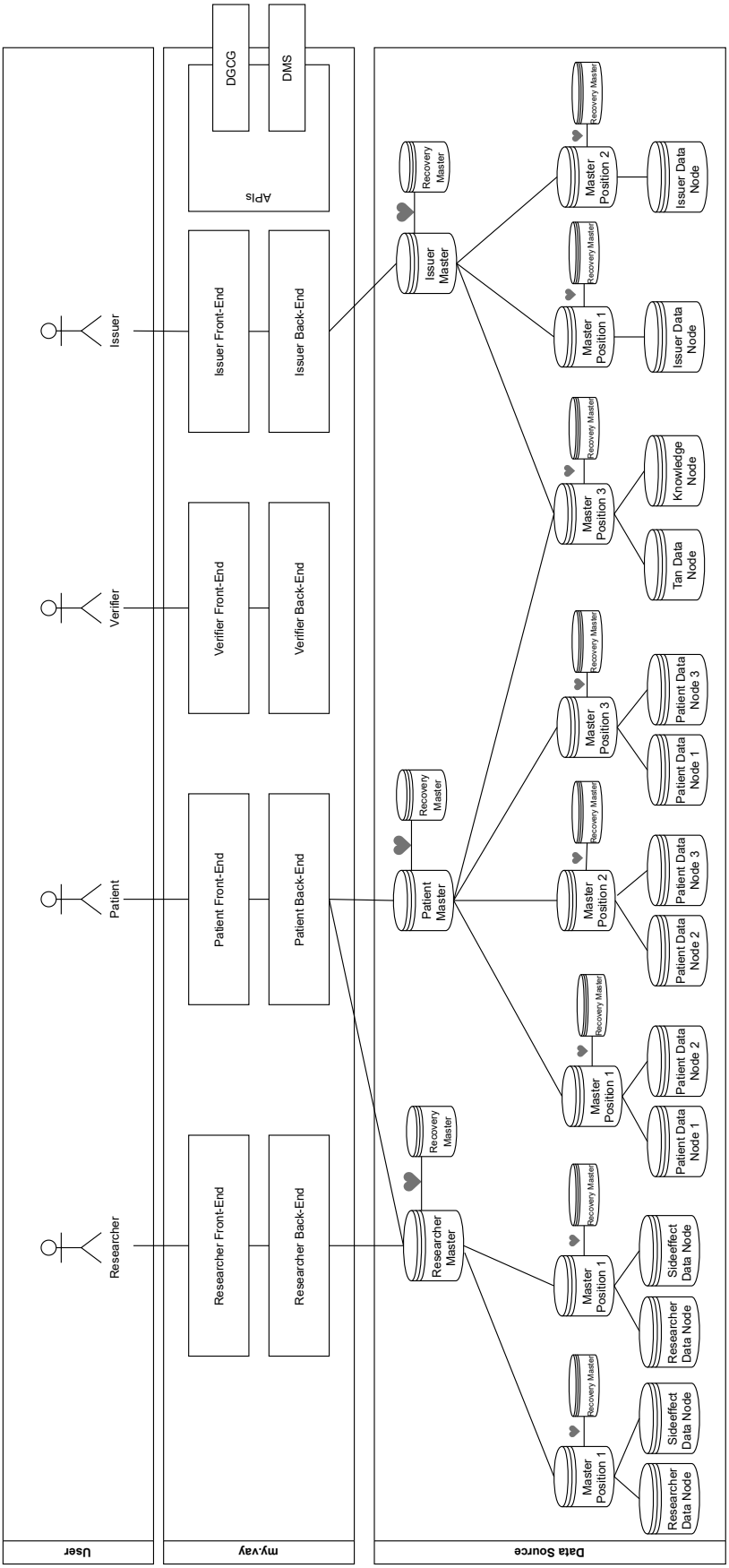

Abbildung 5.15: Architekturdiagramm von my.vay

6 Prototypische Realisierung

Dieses Kapitel beschreibt auszugsweise die prototypische Implementierung von my.vay, unter Verwendung des in Kapitel 5 erarbeiteten Entwurfs. Zunächst wird hierzu eine Abgrenzung zu den definierten Anforderungen des Systems getätigt und beschrieben, welche Funktionalitäten im Prototyp umgesetzt worden sind. Die darauf folgende Implementierungsgrundlage beschreibt die Basis des Prototypen. Hierzu zählen unter anderem die genutzte Programmiersprache sowie genutzte *Frameworks*. Dies bildet die Grundlage,um im Anschluss daran die Umsetzung von my.way vorzustellen. Dabei wird separat auf die einzelnen implementierten Funktionalitäten der Nutzergruppen eingegangen. Der Quellcode des Prototypen kann unter `https://github.com/ValentinFFM/my.vay` eingesehen werden.

6.1 Abgrenzung

Aus Gründen der Einfachheit sowie der begrenzten Zeit wurde sich ausschließlich auf die Funktionalitäten von my.way fokussiert, die für eine prinzipielle Durchführbarkeit des Konzeptes benötigt werden. Dabei wurde unter anderem die ANF-PA Nr. 7 und 8, die sich auf das Wissensmanagement beziehen, außen vorgelassen. Mithilfe des Wissensmanagements sollen dem Patient und dem Issuer sämtliche Informationen zu allen existierenden Impfungen bereitgestellt werden. Für die Bereitstellung des Wissensmanagements würde eine entsprechende Datenbank benötigt werden, welche vielfältige Impfinformationen enthält. Da eine solche aktuell jedoch nicht frei verfüglich existiert, wurde auf die Umsetzung vorerst verzichtet. Zusätzlich handelt es sich bei dem Wissensmanagement zwar um eine sinnvolle, aber nicht um eine umsetzungsrelevante Funktion.

Der Impfkalender, in welchem der Patient ausgemachte Impftermine eintragen und empfohlene (Folge-)Impfungen einsehen kann (siehe ANF-PA Nr. 9 und 10), wurde ebenfalls im Prototyp außen vorgelassen, da auch diese Funktion nicht umsetzungrelevant für das ausgearbeitete Konzept ist, in Zukunft jedoch für ein besseres Nutzererlebnis sorgen kann.

Ein weiterer Aspekt, der in dem vorliegendem Prototyp nicht umgesetzt wurde, ist das Teilen des Impfpasses zur ärztlichen Einsicht (siehe ANF-PA Nr. 13 sowie ANF-IA Nr. 4

und 5). Hier wurde bisher weder spezifiziert, wie das Teilen genau funktionieren soll, noch wie die Funktion in der Applikation des Issuers aufgebaut sein soll.

Des Weiteren können Patients in dem aktuell umgesetzten Prototyp noch nicht nach ANF-PA Nr. 14 und 15 die Impfpässe von Angehörigen in der eigenen Applikation verwalten. Auch dieser Punkt ist nicht für das Funktionieren der Applikation notwendig und kann deshalb zunächst unberücksichtigt bleiben. Als Zwischenlösung ist es für die Patients möglich, ein Konto und somit eine eigene Applikation für den Angehörigen zu erstellen.

In dem Prototyp können Issuer und Patients bei dem Erstellen eines Impfeintrags aktuell lediglich zwischen drei verschiedenen Impfkategorien wählen. Die Auswahl der Kategorien wird derzeitig durch die „Vaccination"-Datenbanktabelle begrenzt, da für jede Impfung eine eigene Logik für das Mindestalter und die Folgeimpfungen erstellt werden muss. Die vorgegebenen Impfung dienen lediglich als Beispiel für die Umsetzung der Impferinnerung und erheben keinen Anspruch auf Vollständigkeit.

Für einen Patient ist im Konzept vorgesehen, dass dieser Impfnebenwirkungen anonymisiert speichern können soll (siehe ANF-PA Nr. 11), um diese Researchern zu Forschungszwecken zur Verfügung zu stellen. Zwar soll ein Patient im Prototyp an das Eintragen von Impfnebenwirkungen erinnert werden und diese in der Datenbank speichern können, jedoch soll vorerst auf die gesammelten Daten kein Zugriff gewährt werden. Die bedeutet zum einen, dass ein Patient selbst die Daten nicht mehr nachlesen kann. Zum anderen wird auf die Umsetzung der gesamten Applikation für Researcher (siehe ANF-RA Nr. 1, 2 und 3) verzichtet, da diese nicht umsetzungsrelevant ist. Mithilfe der gespeicherten Nebenwirkungen in der Datenbank soll jedoch die Grundlage für die spätere Researcher-Applikation gelegt werden, um zu zeigen wie die Impfnebenwirkungen von den Patients gesammelt werden könnten.

Die Applikation eines Issuers wird ebenfalls lediglich abgespeckt umgesetzt. Dabei soll im Prototyp zunächst die Hauptfunktionalität, also das Erstellen von Impfnachweisen mithilfe eines QR-Codes (siehe ANF-IA Nr. 2), der Issuer Application implementiert werden.

Ein weiterer wichtiger Punkt, der erwähnt werden sollte, ist, dass der Prototyp ausschließlich webbasiert umgesetzt wird, auch wenn es sich zukünftig bei my.vay um eine App handeln soll. Dies ist aufgrund der mangelnden Erfahrung von App-Entwicklungen zu begründen. Aus Zeitgründen wurde zusätzlich kein vollständiges *responsive Design* umgesetzt. Der Prototyp ist vorerst auf die Benutzung eines durchschnittlichen *Laptops* ausgelegt.

Aus ähnlichen Gründen wurde statt der im Konzept erwähnten dezentralen Speicherung eine zentrale Speicherung umgesetzt. Die Umsetzung einer dezentralen Speicherung ist wesentlich komplexer und zeitintensiver, zusätzlich fehlt es den Beteiligten hierbei an Erfahrung. Da der Prototyp vorerst keinen hohen Sicherheitsanspruch verfolgt, kann auf die dezentrale Speicherung zur Umsetzung der Hauptfunktionalitäten verzichtet werden. Des Weiteren läuft die Anwendung bislang nur lokal und nicht auf einem externem Server.

Darüber hinaus werden jegliche nicht-funktionale Anforderungen, die in Kapitel 4.2.2 definiert werden, im Prototyp nicht weiter berücksichtigt. Dies liegt daran, dass diese Anforderungen zunächst nicht relevant für das Funktionieren des Prototypen sind und deshalb bei der Implementierung nicht im Fokus standen. Des Weiteren können die Anforderungen der „Benutzerfreundlichkeit“ und „Zuverlässigkeit“ erst mit einem fertig implementierten Prototyp getestet werden.

6.2 Implementierungsgrundlage

Die Implementierung des my.vay-Prototpyen erfolgte unter Einsatz der Entwicklungsumgebung „Visual Studio Code“ [111]. Die zugrundeliegende Programmiersprache ist Python. Für die Entwicklung der Webanwendung my.vay wurde das *Webframework* Flask genutzt. Es wurde sich für dieses *Framework* entschieden, da mit Flask auf eine Vielzahl von Online-Ressourcen zurückgegriffen werden kann, die für die Funktionsfähigkeit des Prototypen benötigt werden. Die erstellten Web-Formulare wurden auf Basis der Bibliothek WTForms implementiert. Mit WTForms können Formulare validiert als auch einheitlich dargestellt werden. Für die *Frontend*-Entwicklung wurde das *Webframework* „Bootstrap“ [112] herangezogen. Bootstrap enthält Hypertextmarkuplanguage (HTML)- und Cascading Style Sheets (CSS)-basierte Designvorlagen. Dies ist von besonderer Relevanz für die implementierten Formulare, Navigationen und Schaltflächen.

Da die Umsetzung der Datenhaltung mit mehreren verteilten Data Nodes, wie es im Konzept von my.vay vorgesehen ist, den Rahmen dieser Arbeit überschreiten würde, entschied man sich bei der Implementierung des Prototypen für eine zentrale Datenbank. Die Datenbank wurde mit dem Datenbankmanagementsystem PostgreSQL umgesetzt. Zur Verknüpfung zwischen der Datenbank und dem Flask-*Framework* wurde die Python-Bibliothek „Flask-SQLAlchemy“ eingesetzt, welche die Nutzung des *ORM-Framework* SQLAlchemy vereinfacht.

6.3 Datenbank von my.vay

Die Funktionalitäten des my.vay-Prototypen, die in Kapitel 6.4 näher beschrieben sind, machen eine gemeinsame Datenspeicherung erforderlich. Diesbezüglich wurde sich, wie bereits zuvor erläutert für eine zentrale Datenbank entschieden. Diese soll an dieser Stelle mit ihren Entitäten und Relationen näher beschrieben werden. Zum besseren Verständnis befindet sich außerdem in Abbildung 6.1 ein Entity-Relationship-Modell in der Chen-Notation, welches die Entitäten, Relationen und dazugehörigen Kardinalitäten darstellt.

Die Datenbank besitzt insgesamt fünf Entitäten, wobei zunächst Issuer und Patient betrachtet werden. Beiden Entitäten dienen zur Speicherung von Nutzerdaten und ermöglichen dadurch unter anderem die Anmeldung und Registrierung als Nutzer im my.vay-Prototyp. Die Attribute der Entitäten sind bis auf den jeweiligen Primärschlüssel identisch und lauten: „password“ für das Nutzerpasswort, „f_name“ für den Vornamen, „l_name“ für den Nachnamen und „date_of_birth“ für das Geburtsdatum. Der Primärschlüssel des Issuers heißt „unique_issuer_identifier“ und der des Patients „unique_patient_identifier“.

Die beiden Entitäten stehen jeweils in Relation zur Entität „Proof_of_vaccination“. Im Falle des Patients handelt es sich um eine 1:N-Relation, welche aus Datensicht den Sachverhalt darstellt, dass ein Patient mehrere Impfnachweise besitzen kann, aber ein Impfnachweis nur zu einem Patient gehört. Die Entität Issuer hat ebenfalls eine 1:N-Relation mit der Entität Proof_of_vaccination, allerdings wird damit ein anderer Sachverhalt modeliert. Der Issuer besitzt nämlich kein Impfnachweis, sondern kann n-viele Impfnachweise ausstellen, während ein Impfnachweis immer nur von einem Issuer ausgestellt worden sein kann.

Die Entität Proof_of_vaccination realisiert folglich die Speicherung der Impfnachweise und ist damit ein wichtiger Bestandteil des my.vay-Prototypen. Der Primärschlüssel der Entität ist ein einzigartiger Identifikator mit der Bezeichnung „unique_certificate_identifier“. Des Weiteren umfasst die Entität die Attribute „date_of_vaccination“ für das Datum der Impfung, „vaccine“ für den Namen des Impfstoffes, „issued_at“ für die Ausstellungszeit des Impfnachweises, „batch_number“ für die Chargennummer des Impfstoffes und „vaccine_marketing_authorization_holder“ für den Hersteller des Impfstoffes. Die Attribute der Entität orientieren sich grundsätzlich an dem HCERT und Proof of Vaccination, die im Konzept in Kapitel 5.3.2 beschrieben wurden. Allerdings wurden bei der prototypischen Implementierung einerseits ein Teil der im Konzept definierten Attribute ausgelassen und anderseits sind gewisse Attribute in anderen Entitäten gespeichert. Durch die Relationen zwischen den Entitäten kann auf diese Attribute nichtsdestotrotz zugegriffen werden. Aus

diesem Grund wird in der Entität Proof_of_vaccination beispielsweise nicht gespeichert, für welche Krankheit der Impfnachweis ist.

Dieses Attribut befindet sich in der Entität „Vaccination“, welche alle Informationen zur Impfungen beinhaltet, die unabhängig von dem zu impfenden Patient und dessen Impfnachweis feststehen. Dazu zählen die Attribute „disease“ für die Krankheit gegen die die Impfung ist, „vaccine_category“ für die Art der Impfung (erste Grundimmunisierung, Auffrischimpfung et cetera), „beginn_age“ für das empfohlene Alter, in dem geimpft werden sollte, „end_age“ für das Alter, in dem spätestens geimpft werden sollte und „distance_to_pre_vaccination“ für den Zeitraum, der zu einer vorherigen Impfung vergangen sein muss. Des Weiteren besitzt die Entität noch eine rekursive Relation, welche die Reihenfolge von Impfungen definiert. Da diese Daten sich nur wenig verändern und nicht in jedem Impfnachweis erneut gespeichert werden sollen, sind sie in der Entität Vaccination ausgelagert. Durch die 1:N-Relation zwischen den beiden Entitäten Proof_of_vaccination und Vaccination, enthält jeder Impfnachweis die Daten zur Impfung und die Daten zur Impfungen können in n-vielen Impfnachweisen stehen.

Die letzte Entität der Datenbank heißt Sideeffects und dient zur Dokumentation von Nebenwirkungen, die nach einer Impfung wohlmöglich auftreten können. Die Entität besitzt dazu für die Art an Nebenwirkung die Attribute „headache“ für Kopfschmerzen, „arm_hurts“ für Armschmerzen, „rash“ für Ausschlag, „tummyache“ für Bauchschmerzen, „fever“ für Fieber und „sideeffects“ für alle anderen Nebenwirkungen. Der Primärschlüssel der Entiät ist der „unique_entry_identifier“ und auch diese Entität besitzt eine 1:1-Relation zur Entität Proof_of_vaccination. Dadurch kann für jeden Impfnachweis einmal angegeben werden, welche Nebenwirkungen im Nachgang an die Impfung aufgetreten sind.

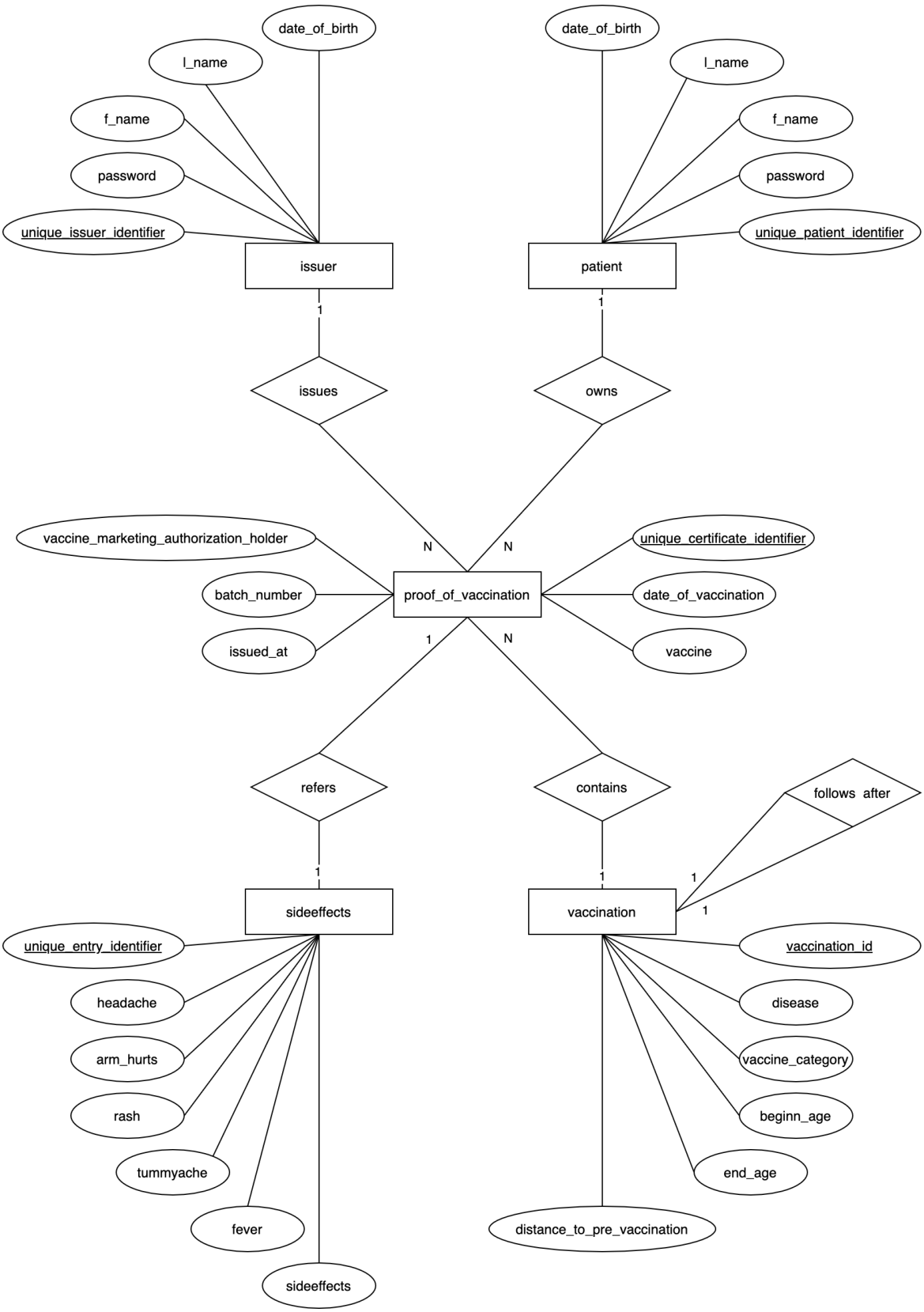

Abbildung 6.1: Entity-Relationship-Modell der my.vay-Datenbank nach Chen-Notation

6.4 Applikation von my.vay

Für die Realisierung des Prototypen wurde, wie in der Abgrenzung bereits beschrieben, ein Fokus auf die drei Nutzergruppen Patient, Issuer und Verifier gelegt. Öffnet man den Prototypen von my.vay, so gelangt man auf die Startseite der Plattform. Diese ist in der Abbildung 6.2 visualisiert.

Abbildung 6.2: Bildschirmfoto der Startseite von my.way

Es ist ersichtlich, dass sowohl ein Issuer als auch ein Patient die Funktion bereitgestellt bekommen, sich über einen *Login* anzumelden oder zu registrieren, entsprechend ANF-PA Nr. 1 und ANF-IA Nr. 1.

Für die Registrierung eines Patients wird die Funktion „patient_registration()“ genutzt. Innerhalb dieser Funktion wird die *form* „PatientRegistrationForm()“ aufgerufen, welche die benötigten *Input*-Felder zum Eintragen von Name, Geburtstdatum, Passwort und der gewünschten Nutzer-ID bereitgestellt. Hat ein Patient die Felder ausgefüllt und klickt auf den *Button* „Registrieren“ werden die Daten an die Patient-Datenbanktabelle übergeben und gespeichert. Ist ein Patient bereits registriert und möchte sich einloggen wird die Funktion „patient_login()“ aufgerufen. Hierbei wird zunächst überprüft, ob der Patient bereits als gültiger Nutzer eingeloggt ist, um direkt auf die *Landing Page* des Patients weitergeleitet zu werden. Ist der Patient noch nicht eingeloggt werden die *Input*-Felder der *form* „PatientLoginForm()“ aufgerufen. Anhand des eingegebenen unique_patient_identifier-Werts wird darauf in der Datenbank überprüft, ob das eingegebene Passwort mit dem in der

Datenbank gespeicherten übereinstimmt. Ist der Login erfolgreich wird die Nutzer-ID in den Cookies gespeichert, um den unique_patient_identifier-Wert an die späteren Funktionen übergeben zu können und den Login des Patent zu bestätigen. Beide Funktionen des Patients sind in Anhang A.4 dargestellt.

Der *Login* und die Registrierung sind beim Issuer äquivalent zum Patient aufgebaut. Für den Issuer werden entsprechend die Funktionen „issuer_registration()“ und „issuer_login()“ verwendet, welche den gleichen Ablauf wie beim Patient befolgen. Der Unterschied besteht lediglich darin, dass der erfolgreiche *Login* mithilfe der Issuer-Datenbanktabelle überprüft und anschließend die *Landing Page* des Issuers angezeigt wird.

Mithilfe des Flask *View Decorators* „@login_required“ können die Seiten des Issuers und des Patients nur nach einem erfolgreichen *Login* aufgrufen werden. Ein Verfier kann den Prototyp von my.way gänzlich ohne Anmeldung und Registrierung nutzen (siehe ANF-VA Nr.1). Es wird lediglich vorausgesetzt, dass das Gerät des Verifiers über eine integrierte Kamera verfügt.

6.4.1 Funktionalitäten des Issuers

Wie in der vorherigen Abgrenzung in Abschnitt 6.1 beschrieben, ist die prototypisch implementierte Kern- und Hauptfunktionalität eines Issuers das Erstellen von Impfnachweisen. Zwar wurde das Wissensmanagement eines Issuers, wie Kapitel 5.1 nicht umgesetzt, jedoch findet sich dieses provisorisch als mögliche Ansicht in der Navigationsleiste des Issuer Frontend wieder. Ebenso das Profil, in welchem ein Issuer seine persönlichen Daten ändern kann. Auf die Erstellung des Impfnachweises als QR-Code wird nachfolgend näher eingegangen.

Erstellung eines Impfnachweises

Für die Erstellung eines Impfnachweises nutzt der Issuer das Plus-*Icon* in der Navigationsleiste des Issuer Frontend. Die angezeigten Eingabefelder, entsprechend der Anforderungen zur Impfdokumentation in Kapitel 4, werden durch die Klasse *ImpfnachweisForm(Form)*, die dem Anhang A.1 beigefügt ist, definiert. Abbildung 6.3 visualisiert diese Komponente der grafischen Benutzeroberfläche. Nach erfolgreicher Eingabe der Daten im vordefinierten Format wird durch Klick des Buttons „Impfnachweis erstellen“ die Funktion „issuer_create_qr()“ in der „routes.py“-Datei aufgerufen, die zunächst ein „Python-dictionary“

Abbildung 6.3: Bildschirmfoto der Eingabefelder für die Erstellung eines Impfnachweises im Issuer Frontend

„proof_of_vaccination“ der eingegebenen Daten erstellt. Dieses Format erleichtert das binäre Kodieren und Auslesen des im nachfolgenden Schritt generierten zweidimensionalen QR-Codes.

Die Funktion zur Erstellung des QR-Codes, ersichtlich im aufgeführten Quelltext in Anhang A.2 nutzt die Python-Bibliothek „qrcode“. Für die anschauliche Darstellung werden die drei Parameter *version*, *box_size* und *border* übergeben, in welchen unter anderem die Größe des QR-Codes sowie die Größe der einzelnen Boxen definiert wird. Mit der Methode „qr.add_data(proof_of_vaccination)“ wird das aus den Eingaben eines Issuers erstellte *Python-dictionary* übergeben. Der Befehl „qr.make()“ in Zeile 10 erstellt schließlich den QR-Code. Dabei wird ein *PilImage*-Objekt als *Bytestream* erstellt, dass durch „qr.make_image“ und „img.save(file_object, 'PNG')“ verarbeitet und temporär im Puffer als Portable Network Graphics (PNG) gespeichert wird. Sodass der QR-Code einem Issuer angezeigt werden kann, muss der *Button* „Impfnachweis als QR-Code anzeigen“ geklickt werden. Hinter diesem *Button* ist in der dazugehörigen HTML-Datei ein *Trigger* implemeniert, sodass sich ein *Pop-Up*-Fenster öffnet, das den erstellten QR-Code enthält. Dies ist in der nachstehenden Abbildung 6.4 visualisiert.

Das Einlesen des QR-Codes ist prinzipiell mit jedem Endgerät möglich, in welchem eine

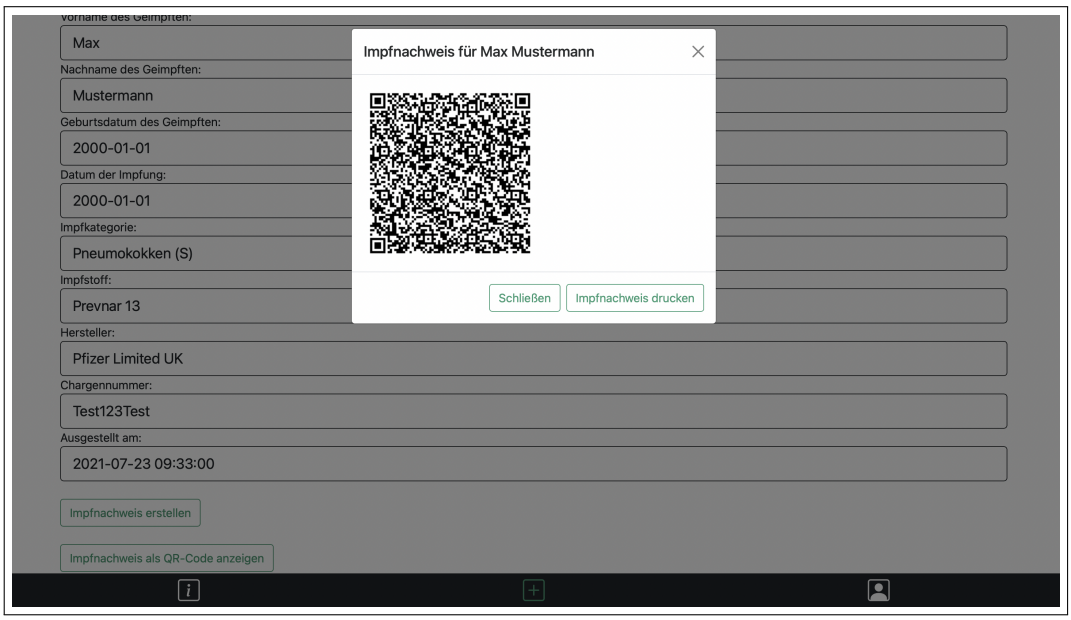

Abbildung 6.4: Bildschirmfoto der Anzeige des erstellten QR-Codes im Issuer Frontend

Kamera integriert ist. Damit der Code beispielsweise durch eine Handykamera ausgelesen werden kann, wird der *Bytestream* in dem *Return*-Befehl in Zeile 14 des Quelltextes A.2 mit „Base64“ kodiert und dekodiert. Bei Base64 handelt es sich um ein Codierungsschema, mit welchem inhaltsbasierte Nachrichten über das Internet übertragen werden können.

6.4.2 Funktionalitäten des Patients

Nach einer erfolgreichen Registrierung beziehungsweise nach einem erfolgreichen Login als Patient, öffnet sich die personalisierte *Landing Page*, auf welcher der digitale Impfpass des Patients zu sehen ist (siehe Abbildung 6.5). Im unteren Bereich der Abbildung ist die schwarze *Navigationbar* zu sehen, welche von rechts nach links auf die QR-Scanner-, die Wissensmanagement-, die Impfpass-, die Impfkalender- und die Profil-Seite verlinkt.

Die Impfpass-Seite stellt die Hauptfunktionalitäten für einen Patient zur Verfügung. Diese beinhalten vor allem das Anzeigen der bisher dokumentierten digitalen Impfungen. Aus Übersichtsgründen werden maximal fünf Impfungen auf einer Unterseite angezeigt (siehe ANF-PA Nr.2). Ein Patient kann zudem nach bestimmten Impfkategorien filtern oder nach einem spezifischen Impfstoffnamen in seinem Impfpass suchen. Für zertifizierte Impfung kann ein QR-Code als Impfnachweis erstellt werden, entsprechend ANF-PA Nr.4. Des Weiteren werden auf dieser Seite Impferinnerungen zur Auffrischung angezeigt und die Mög-

lichkeit geboten eventuelle Impfnebenwirkungen für Forschungszwecke zu dokumentieren (siehe ANF-PA Nr.9 und 10).

Über das Plus-*Icon* in der rechten oberen Ecke kann ein Patient, wie in ANF-PA Nr.3 definiert, eine neue digitale Impfung über einen manuellen Eintrag oder über das Scannen eines QR-Codes hinzufügen. Ein weiterer Weg eine zertifizierte Impfung einzutragen, ist direkt über die QR-Scanner-Seite in der *Navigationbar*. Auf der Profil-Seite kann ein Patient seine Profildaten ändern und sich ausloggen.

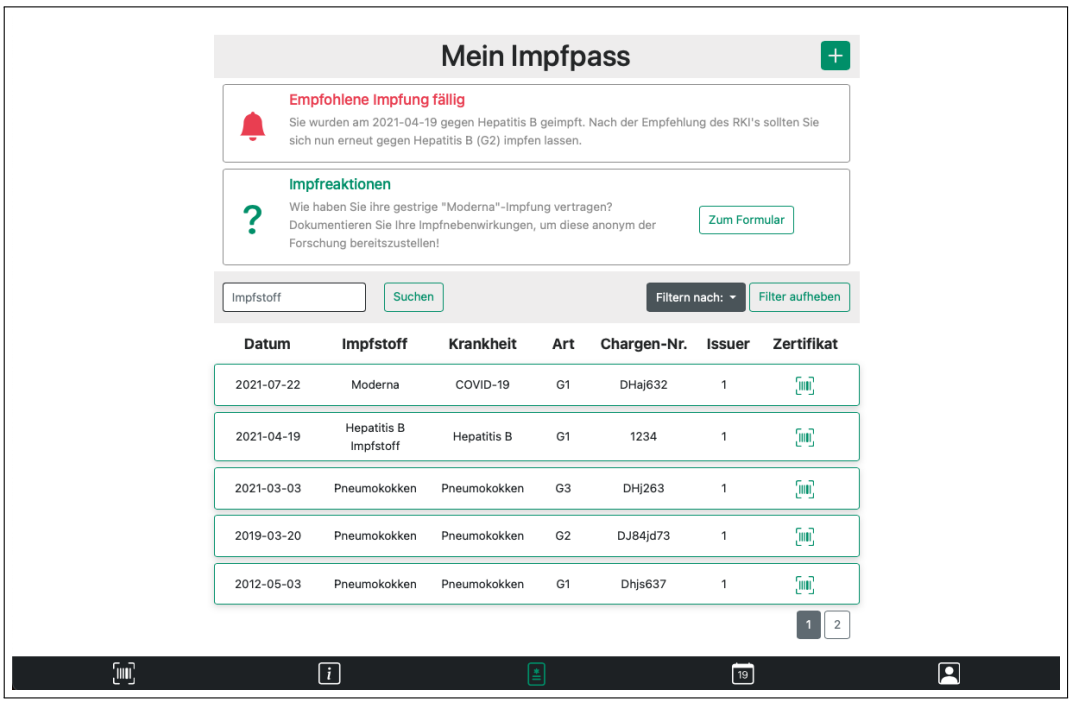

Abbildung 6.5: Bildschirmfoto der *Landing Page* im Patient Frontend

Anzeigen der digitalen Impfungen

Beim Aufruf der Impfass-Seite über die *Navigationbar* wird die Funktion „patient_home()“ aufgerufen (siehe Anhang A.5). Der Funktion werden die Variablen „sort“ und „search“ übergeben. Als *default*-Wert wird dem sort der Wert „date“ übermittelt. Falls nicht nach einem spezifischen Impfnamen gesucht wird, werden durch diesen Wert alle Impfeinträge des Patients absteigend nach Datum sortiert angezeigt. Mithilfe der Funktion patient_login() und der Variablen „current_user.unique_patient_identifier“ werden dem Patient nur die eigenen persönlichen Impfeinträge angezeigt. Wird in der *form* „SearchVaccine()“ jedoch der Name eines Impfstoffs im Suchfeld eingegeben und gesendet, wird die

Datenbank nur nach Einträgen mit diesem Impfstoff durchsucht und diese auf der Plattform entsprechend angezeigt. Alternativ zur Suche kann mittels einem *Drop-Down*-Menü ebenfalls nach den Krankheiten und den entsprechenden Impfkategorien wie beispielsweise COVID-19, Hepatitis B und Pneumokokken gefiltert werden. Hierzu wird die Variable sort jeweils mit dem ausgewähltem Filterwert überschrieben, sodass unterschiedliche *if*-Bedingungen erfüllt werden. In der Funktion patient_home() werden außerdem die Funktionen „check_for_vac_notifications()“ (siehe Anhang A.7) und „show_sideeffect()“ (siehe Anhang A.8) aufgerufen. Diese werden für die Funktionalitäten „Impferinnerung“ und „Impfnebenwirkungen“ benötigt. Die Funktionalitäten werden im weiteren Verlauf dieses Kapitels erläutert.

Anzeigen des Impfnachweises

Über die *Landing Page* hat ein Patient außerdem die Möglichkeit den Impfnachweis einer spezifischen Impfung in Form eines QR-Codes anzeigen zu lassen. Der QR-Code einer Impfung kann in der Zertifikat-Spalte über das jeweilige QR-Code-*Icon* aufgerufen werden. Durch den Klick auf das *Icon* wird die Funktion „open_QR()“ aufgerufen, welche in Anhang A.6 dargestellt ist. Der Funktion wird dabei der unique_certificate_identifier-Wert des ausgewählten Impfeintrags übergeben. Anhand dieses Werts kann in der Datenbank nach dem entsprechenden Impfeintrag gesucht werden. Alle Werte des Impfeintrags werden daraufhin unter ihrem Spaltennamen in einem *dictionary* abgespeichert und in einen zweidimensionalen QR-Code umgewandelt. Der QR-Code und die Werte des Impfeintrags werden der HTML-Datei „patient_show_QR“ übergeben. In der HTML-Datei wird vor dem Anzeigen überprüft, ob es sich bei dem ausgewählten Impfeintrag um eine verifizierte Impfung handelt. Ist die Impfung mittels eines manuellen Eintrags vom Patient selbst erstellt worden, soll für diese kein gültiger QR-Code angezeigt werden (siehe ANF-VA Nr.2). Die Überprüfung findet über den issued_at-Wert statt, da dieser bei einem manuellen Impfeintrag auf den *default*-Wert „1900-01-01 00:00:00“ gesetzt wird, um nicht verifizierte Impfungen erkennen zu können. Ist die Impfung von einem Issuer ausgestellt worden und der issued_at-Wert somit nicht mit dem *default*-Wert befüllt, wird ein gültiger QR-Code angezeigt, welcher als Impfnachweis verwendet werden kann.

Impferinnerung

Die Funktionalität Impferinnerung läuft im Hintergrund der *Landing Page* des Patients. Hierfür wird, wie bereits erwähnt, in der Funktion patient_home() die Funktion check_for_vac_notifications() aufgerufen, welche in Anhang A.7 dargestellt ist. In dieser Funktion wird überprüft, ob der Patient an eine Auffrischungsimpfung erinnert werden muss. Ist dies der Fall, wird dem Patient eine Erinnerungshinweis angezeigt. Um dies umzusetzen, werden in der Funktion zunächst alle Impfeinträge des angemeldeten Patients ausgelesen und in einer Liste gespeichert. Des Weiteren wird ein *default-dictionary* „notification_dict“ erstellt, in welchem die *keys* „next_vaccination_possible“, „next_vaccination_not_exisisting“, „beginn_age_reached“, „end_age_not_reached“ und „dist_between_current_and_next_correct“ zunächst auf *false* gesetzt werden. Mithilfe einer *for*-Schleife werden für jeden Impfeintrag des Patients die zugehörigen Werte in der Datenbanktabelle vaccination ausgelesen. Jeder Impfeintrag wird mit mehreren verschachtelten *if*-Bedingungen auf verschiedene Voraussetzungen überprüft. Zunächst wird kontrolliert, ob es einen folgenden Impfeintrag des Impfstoffs geben muss. Ist dies der Fall wird überprüft, ob dieser Folgende bereits existiert. Gibt es noch keinen wird der Parameter next_vaccination_possible im *defaut-dictionary* für diesen Eintrag auf *true* gesetzt. Als nächstes wird kontrolliert, ob der Patient alt genug, aber nicht zu alt ist und ob ein ausreichender Abstand zu der Vorgänger-Impfung eingehalten werden kann. Sind all diese Bedingungen erfüllt, werden auch die restlichen *keys* des *default-dictionary* auf *true* gesetzt. Haben alle *keys* des *default-dictionary* den Wert *true*, wird das *dictionary* „notification“ mit sämtlichen Informationen der entsprechenden Impfung erstellt. Der Eintrag wird anschließend zu der Liste notifications hinzugefügt und an die Funktion patient_home() und somit an das *Frontend* übergeben. Die Liste wird in der HTML-Seite „patient_vaccination_certificate“ aufgerufen und die entsprechenden Impferinnerungen werden dem Patient auf der *Landing Page* angezeigt.

Impfnebenwirkungen

Ein Eintrag für eine Impfnebenwirkung soll von einem Patient einen Tag nach einer durchgeführten Impfung erstellt werden können. Um erkennen zu können, welche Impfungen einen Tag alt sind, wird die Funktion show_sideeffect() (siehe Anhang A.8) innerhalb der Funktion patient_home() aufgerufen. In der Funktion show_sideeffect() wird dazu die Variable „today“ definiert, welche das aktuelle Tagesdatum um jeweils einen Tag zurückrechnet. Die

Variable today wird daraufhin innerhalb einer *for*-Schleife mit dem date_of_vaccination-Wert aller Impfeinträge des Patients verglichen. Stimmen die beiden Daten überein, wird in einer zweiten *for*-Schleife überprüft, ob für den gefundenen Impfeintrag bereits eine Impfnebenwirkung in der Datenbank gespeichert wurde. Hierzu wird der unique_certificate_identifier-Wert in der Sideeffect-Tabelle mit dem unique_certificate_identifier des Impfeintrags verglichen. Alle Impfeinträge, die das passende Datum und noch keinen Eintrag in der Sideeffect-Tabelle besitzen, werden in der Liste „sideeffects_list" gespeichert. Diese Liste wird an die Funktion patient_home() übergeben, wodurch auf der *Landing Page* für alle Impfeinträge in der Liste jeweils ein grüner Hinweis erstellt werden kann, welcher dazu auffordert eventuelle Impfnebenwirkungen für Forschungszwecke zu speichern. Innerhalb dieses Hinweises ist ein grüner *Button* untergebracht, welcher die Funktion „new_sideeffect()" aufruft (siehe Anhang A.9). Durch den Klick auf den jeweiligen Hinweis-*Button* wird der ausgewählte unique_certificate_identifier-Wert des entsprechenden Impfeintrags mit übergeben. Mithilfe der Funktion werden schließlich die *Input*-Felder der *form* „AddSideeffects()" angezeigt, welche dann vom Patient befüllt werden können. Die Eingaben der Eingabefelder werden zusammen mit dem unique_certificate_identifier-Wert in der Datenbank gespeichert, wodurch der Impfnebenwirkung-Hinweis für die Impfung nicht länger angezeigt wird.

Manueller Impfeintrag

Ein Patient hat zwei Möglichkeiten einen neuen Impfeintrag in seinen Impfpass hinzuzufügen, manuell oder per QR-Code. Das manuelle Hinzufügen eines Impfeintrags ist vor allem bei älteren Impfungen sinnvoll, wenn diese nur aus Gründen der Vollständigkeit benötigt werden und nicht als zertifizierter Impfnachweis. Wird ein Impfeintrag manuell hinzugefügt und nicht von einer zertifizierten Stelle mittels eines QR-Codes ausgestellt, kann dieser nicht als Impfnachweis verwendet werden. Um einen Impfnachweis manuell hinzuzufügen, muss ein Patient zuerst auf das Plus-*Icon* in der rechten oberen Ecke und anschließend auf „Manueller Impfeintrag" klicken. Dadurch wird die Funktion „addVaccination()" (siehe Anhang A.10) aufgerufen, welches die *Input*-Felder der *form* AddVaccination() anzeigt. Über die *Input*-Felder kann ein Patient alle Details zu einer durchgeführten Impfung eintragen. Das *Drop-Down*-Menü des *SelectFields* Impfkategorie wird dabei automatisch mit den Werten aus der Vaccination-Tabelle befüllt, sodass ein Patient vereinfacht die vaccination_id eintragen kann. Dem Patient wird hierzu im *Frontend* die vaccination_category und das disease angezeigt, welche im *Backend* mit der dazugehörigen vaccination_id verknüpft

sind. Der unique_certificate_identifier-Wert des neuen Eintrags wird erneut automatisch hochgezählt. Die Eingaben werden schließlich in der Datenbank gespeichert und dem Patient als neuer Impfeintrag angezeigt.

Impfeintrag über QR-Code

Neben dem zuvor vorgestellten, manuellen Impfeintrag kann ein Patient auch zertifizierte Impfnachweise in seinen digitalen Impfpass einfügen. Dies setzt voraus, dass der Patient einen Impfnachweis als QR-Code erhalten hat, der von einem Issuer, wie in Kapitel 6.4.1 beschrieben, ausgestellt wurde. Um den Impfnachweis hinzuzufügen, kann der Patient entweder auf das Plus-*Icon* in der oberen rechten Ecke und danach auf „Scan“ klicken oder er klickt direkt auf das Scan-*Icon* in der *Navigationbar*. Unabhängig davon, für welche Variante sich der Patient entscheidet, wird er auf eine neue Seite mit der Route „patient_scan()“ weitergeleitet. Auf dieser Seite wird, wie in Abbildung 6.6 zu sehen ist, ein *Button* mit der Aufschrift „QR-Code Überprüfung starten“ und ein Live-Bild von der Kamera angezeigt, die an das vom Patient genutzte Gerät angeschlossen ist.

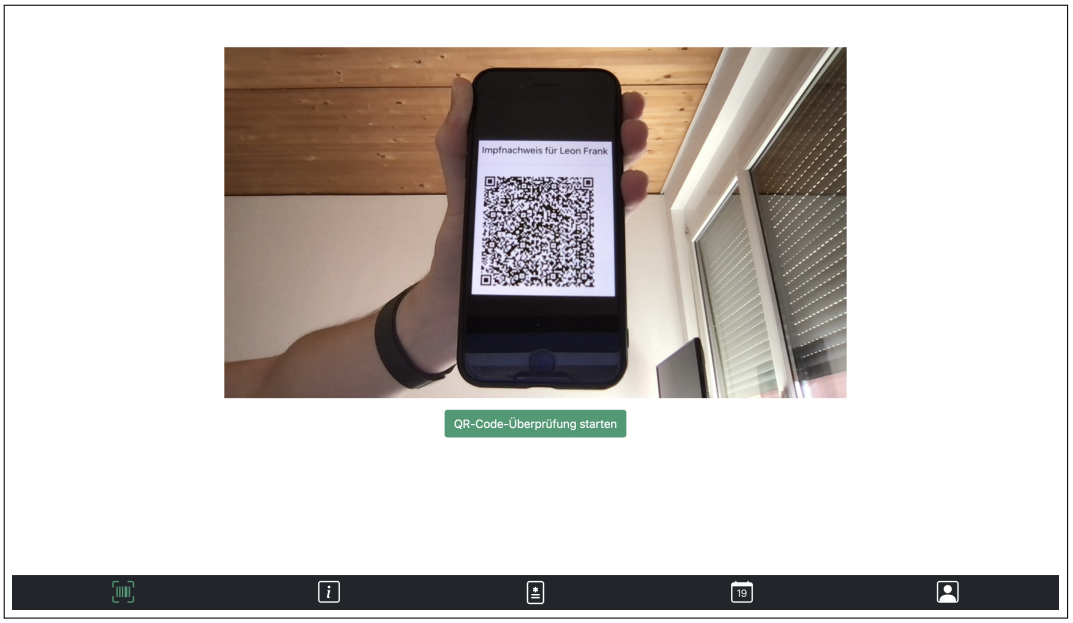

Abbildung 6.6: Bildschirmfoto des Impfeintrags mittels QR-Code im Patient Frontend

Das Live-Bild wird, wie in Quelltext A.3 Zeile 8 dargestellt, mithilfe eines „<img>-tags“ eingebettet, dessen Dateipfad die Route „camera_stream()“ ist. Die Route, welche in

Quelltext A.11 beschrieben ist, gibt ein „Response“-Objekt zurück, das die Funktion „generate_frames()“ aufruft. Diese Funktion wird ebenfalls innerhalb der Route definiert und nutzt zunächst in Zeile 12 die Python-Bibliothek „OpenCV“, um das Objekt „camera“ zu instanzieren. Das Objekt besitzt die Funktion „read()“, die jedes einzelne Bild der Kamera zurückgibt. Dieses wird in Zeile 22 erst in ein „JPG“-Bild und dann in Zeile 23 in ein „byte“-Objekt konvertiert. Anschließend erzeugt der „yield“-Befehl einen Generator, der das byte-Objekt in einem Format zurückgibt, welches vom „<img>-tag“ akzeptiert wird. Die *while*-Schleife in Zeile 14 sorgt dafür, dass der Quelltext der Zeilen 15-24 iterativ ausgeführt wird. Zusammenfassend dient die Route camera_stream() dazu, kontinuierlich das neuste Bild der Kamera auszulesen und zurückgegeben, sodass es innerhalb des „<img>-tags“ angezeigt werden kann.

Neben dem Live-Bild der Kamera gibt es auf der Seite auch noch den bereits angesprochen *Button*. Sobald der Patient auf diesen klickt, wird, wie in Quelltext A.3 Zeile 11 zu sehen ist, die Route „patient_qr_result()“ aufgerufen. Die Route ruft als erstes die Funktion „read_in_frame_for_decoding()“ auf, welche in Quelltext A.13 abgebildet ist. Die Funktion read_in_frame_for_decoding() ist ähnlich aufgebaut, wie die schon beschriebenen Funktion „generate_frames()“. Der Unterschied zwischen den zwei Funktionen besteht darin, dass die Funktion read_in_frame_for_decoding() die einzelnen Bilder der Kamera nicht zurückgibt, sondern mit ihnen die Funktion „decode_QR_code_from_frame()“ aufruft.

Die Funktion decode_QR_code_from_frame(), die in Quelltext A.14 beschrieben ist, bekommt bei ihrem Aufruf ein Bild als Parameter übergeben. Mit diesem Bild ruft sie die Funktion „decode()“ der Bibliothek „pyzbar“ auf. Diese Funktion sucht in dem übergebenen Bild nach einem QR-Code. Falls sich ein QR-Code in dem Bild befindet, so wird dieser dekodiert und seine Daten werden als „binary“-Objekt zurückgegeben. Die decode_QR_code_from_frame()-Funktion liest die relevanten Daten aus dem Objekt aus und konvertiert sie zu einem *dictionary*, welches abschließend zurückgegeben wird.

Sobald ein QR-Code identifiziert und die darin enthaltenden Daten an die Funktion „read_in_frame_for_decoding()“ zurückgegeben wurden, stoppt die *while*-Schleife und die Funktion „decode()“ wird nicht mehr aufgerufen. Anstelle dessen gibt auch die „read_in_frame_for_decoding()“-Funktion die von der „decode_QR_code_from_frame()“- Funktion stammenden Daten an die Route „patient_qr_result“ zurück. In der Route werden darauf hin die folgenden Abgleiche jeweils mit einer *if*-Bedingungen gemacht:

- Impfnachweis wurde noch von keinem anderen Patient eingetragen (siehe Zeile 16 Quelltext A.12)
- Impfnachweis stammt von einem, der Datenbank bekannten Issuer (siehe Zeile 23 Quelltext A.12)
- Vorname, Nachname und Geburtsdatum des Impfnachweises stimmen mit denen des angemeldeten Patients überein (siehe Zeile 26 Quelltext A.12)

Sind alle drei Bedingungen erfüllt, ist der Impfnachweis gültig und kann in die Datenbank eingetragen werden. Dazu wird ein Objekt des Datenbank-Modells Proof_of_vaccination erstellt und dieses in der Datenbank gespeichert. Anschließend bekommt der Patient eine Seite zur Bestätigung des Impfeintrages angezeigt. Sollte eine der zuvor beschrieben Bedingungen allerdings nicht erfüllt sein, wird eine Seite mit einer entsprechenden Fehlermeldung ausgegeben.

Profil verwalten

Beim Aufruf der Profil-Seite wird die Funktion „patient_profil()“ aufgerufen. Mithilfe der *form* „PatientUpdateForm()“ kann der angemeldete Patient seine Daten (z.B. Name oder Passwort) ändern. Um die Seiten eines Patient zu sehen, ist es Voraussetzung, dass man als registrierter Patient angemeldet ist. Hierfür sorgt der *View Decorator* @login_required, welches von Flask bereitgestellt wird. Damit jeder Patient nur seine eigene Daten einsehen kann, wird außerdem mit der Flask-Identifikationsnummer (ID) „current_user“ gearbeitet.

6.4.3 Funktionalitäten des Verifiers

Im Gegensatz zu den Patients und Issuers benötigen Verifiers keine Registrierung, bevor sie eine Funktionalität nutzen wollen (siehe ANF-VA Nr. 1). Gleichzeitig gibt es allerdings auch nur eine Funktionalität, welche von einem Verifier genutzt werden kann und zwar die Überprüfung eines Impfnachweises, welche nachfolgend beschrieben wird. Die Überprüfung ist entsprechend ANF-VA Nr. 2 umgesetzt.

Impfnachweis überprüfen

Nachdem ein Verifier auf der allgemeinen *Landing Page* auf den *Button* „Impfnachweis überprüfen“ geklickt hat, wird er auf die Seite mit der Route „verifier()“ weitergeleitet. Die Seite sieht bis auf die fehlende *Navigationbar* genauso aus, wie die in Abbildung 6.6 dargestellte Seite für den Impfeintrag des Patients. Da bereits in Kapitel 6.4.2 erläutert wurde, mithilfe welcher Funktion das Live-Bild der Kamera angezeigt wird, wird an dieser Stelle auf ein erneute Erklärung verzichtet. Neben dem Live-Bild der Kamera gibt es auf der Seite den *Button* „QR-Code Überprüfung starten“. Wenn der Verifier auf diesen klickt, wird die Route „verifier_qr_result()“ aufgerufen, die in Quelltext A.15 beschrieben ist.

Innerhalb der Route wird zunächst die Funktion „read_in_frame_for_decoding()“ aufgerufen. Da diese bereits in Kapitel 6.4.2 angesprochen wurde, wird von einer wiederholenden Erklärung abgesehen. Sobald die Funktion read_in_frame_for_decoding() ein Impfnachweis als *dictionary* zurückliefert, werden die folgenden zwei Abgleiche mit jeweils einer *if*-Bedingung gemacht:

- Impfnachweis enthält alle verpflichtenden Daten
- Impfnachweis ist in der Datenbank korrekt hinterlegt

Falls die beiden Bedingungen erfüllt sind, wird dem Verifier eine Seite angezeigt, die die Korrektheit des Impfnachweises bestätigt. Sollte dies nicht der Fall sein, dann wird dem Verifier eine Seite mit einer Fehlermeldung angezeigt. Die beiden Seiten sind in Abbildung 6.7 dargestellt.

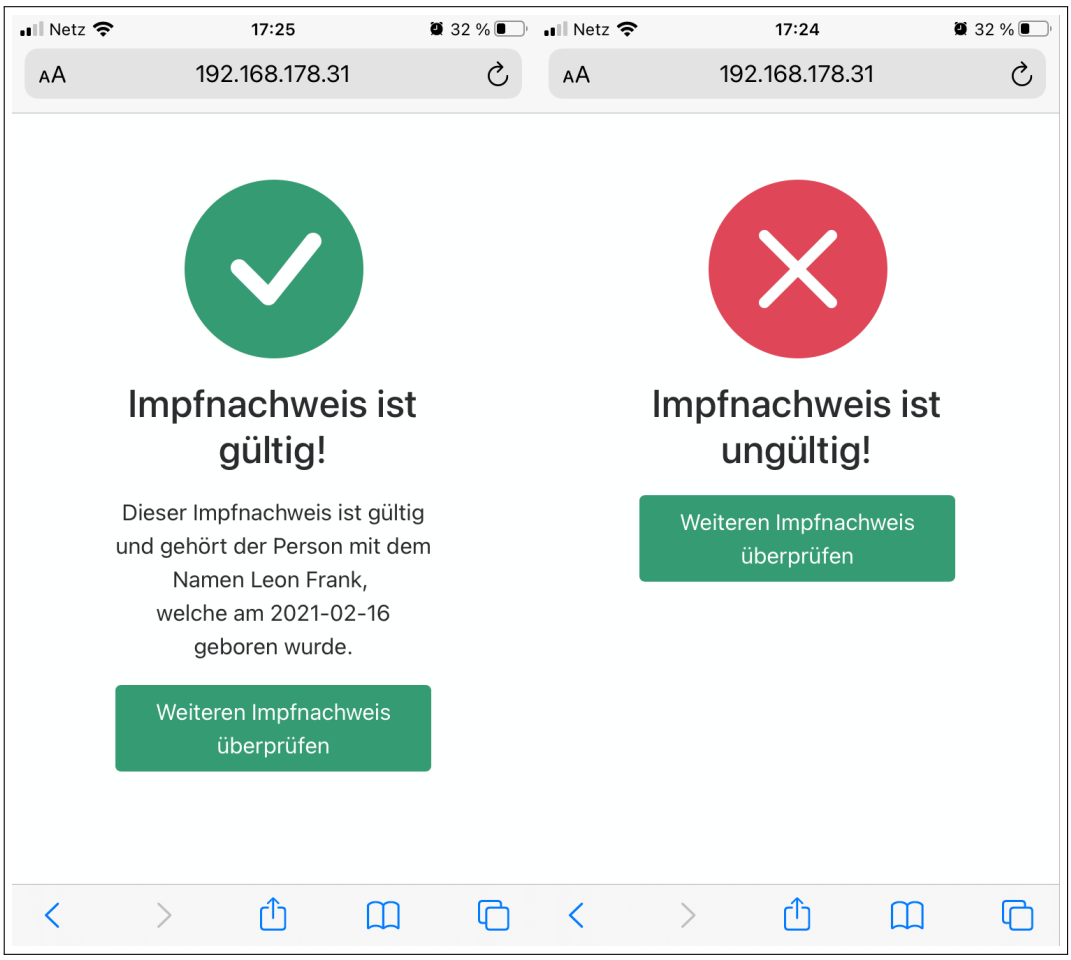

Abbildung 6.7: Zwei Bildschirmfotos der Überprüfung eines Impfnachweises im Verifier Frontend. Die linke Seite zeigt das Ergebnis bei einem korrekten Impfnachweis und die rechte Seite bei einem falschen Impfnachweis.

7 Kritische Würdigung

In diesem Kapitel werden die Ergebnisse der vorliegenden Seminararbeit näher betrachtet und hinterfragt. Diese betreffen zum einen das erarbeitete Konzept von my.way und zum anderen den implementierten Prototyp. Dabei soll insbesondere auf die Umsetzung der Anforderungen sowie die Gewährleistung der Datensicherheit eingegangen werden. Diese ist von besonderer Bedeutung, da sowohl personenbezogene medizinische Daten verarbeitet als auch europäische Richtlinien eingehalten werden müssen.

7.1 Nicht umgesetzte Anforderungen

Die in Kapitel 4.2.1 definierten funktionalen Anforderungen konnten größtenteils in dem erarbeiteten Konzept sowie dem Prototyp umgesetzt werden. Die in Kapitel 4.2.2 definierten nicht-funktionalen Anforderungen blieben, wie bereits in Kapitel 6.1 erläutert, bei der Implementierung des Prototypen unberücksichtigt. Dies fordert besonders datenschutzrechtliche Bedenken im Umgang mit my.way.

Des Weiteren ist festzustellen, dass einige definierte funktionale Anforderungen nicht realisiert wurden. Diese werden nachfolgend bezüglich der betroffenen Nutzergruppen aufgeführt. Das nicht erfüllen dieser funktionalen Anforderung stellt das Konzept und den Prototyp vor gewisse Herausforderungen.

Patient

Bereits für das Konzept wurden die ANF-PA Nr. 14 und 15 nicht beachtet. Bei diesen handelt es sich um das Verwalten der digitalen Impfpässe von Angehörigen in der eigenen Applikation. Dies hat zur Folge, dass beispielsweise Neugeborene für Ihre Grundimmunisierungen bereits einen Zugang für my.vay benötigen. Auch im Prototyp wurden diese Anforderungen außen vor gelassen, sodass ein Patient nach einer Registrierung und einem Login nur seinen eigenen digitalen Impfpass einsehen und verwalten kann.

Den Prototyp betreffend wurde unter anderem die ANF-PA Nr. 5 nicht umgesetzt. Ein Impfnachweis kann dadurch nicht ausgedruckt werden. Dies stellt vor allem Patients ohne

Smartphone-Verfügbarkeit vor Probleme, da kein digitales Nachweisen einer Impfung erfolgen kann. Dies hat zur Folge, dass der analoge Impfausweis trotz digitalem Abbild stets mitgeführt werden muss.

Die ANF-PA Nr. 7 und 8 sind Anforderungen des Knowledge-Managements und wurden bislang ebenfalls lediglich im Konzept, nicht aber im Prototyp umgesetzt. Bei dem Knowledge-Management können sämtliche Informationen über eine Impfung gefunden werden, die Patients und Issuer zur Verfügung gestellt bekommen. Hierzu zählen generelle Informationen, empfohlene Abstände zwischen Auffrischungsimpfungen sowie benötigte Impfungen für Reisen ins Ausland. Es wurde somit das erste der vier Kernziele von my.vay, welche in Kapitel 4.1.1 definiert sind, nicht implementiert. Des Weiteren fehlend im Prototyp sind ANF-PA Nr. 10 und 13, die hierdurch my.vay in seinen über den klassischen digitalen Impfpass hinausgehenden Funktionalitäten einschränken. Impftermine können derzeit nicht in einem Kalender eingetragen und der digitale Impfpass nicht mit einem Arzt geteilt werden.

Teilweise implementiert wurde die ANF-PA Nr. 11. Zwar kann ein Patient seine Nebenwirkungen eintragen, diese jedoch nicht einsehen und teilen.

Issuer

Die Anforderungen an die Funktionalitäten des Issuers wurden vollkommen konzipiert. Im Prototyp hingegen blieben die ANF-IA Nr. 4 und 5 unberücksichtigt. Ein Teilen der Impfdaten eines Patient mit dem Issuer ist somit nicht möglich. Demnach kann der behandelnde Arzt die Daten im Praxisverwaltungssystem nicht abgleichen.

Researcher

Die ANF-RA Nr. 1, 2 und 3 wurden bislang ausschließlich im Konzept umgesetzt, da die Umsetzung der Researcher-Applikation im Prototyp abgegrenzt wurde (siehe Kapitel 6.1). Ein Datenzugriff für Forschungszwecke ist durch my.vay daher aktuell nicht möglich.

7.2 Aspekte der Datensicherheit und des Datenschutzes

In Hinblick der Datensicherheit ist die Umsetzung von my.vay mit der QR-Code-Technologie als kritisch anzusehen. Bei QR-Codes ist zwar ein schnelles und einfaches Auslesen der Daten möglich, jedoch stellt dies bei sensiblen Daten auch ein entsprechendes Sicherheitsrisiko für den Datenschutz dar. Hier sollte deshalb in der Zukunft noch näher betrachtet werden, wie tragbar diese Technologie in Bezug auf die Sicherheit tatsächlich ist. Derzeitig stellt der QR-Code jedoch ein gut umsetzbares Übertragungsmedium dar, welches auch beim digitalen COVID-19 Zertifikat der EU genutzt wird und an welchem sich der Prototyp sowie das Konzept orientieren.

Ein weiterer Punkt in Thema Sicherheit ist, dass die Daten bislang im Prototyp nicht verschlüsselt werden, sondern sie lesbar in der Datenbank liegen. Nach Artikel 32 Absatz 1 DSGVO müssen „Auftragsverarbeiter geeignete und organisatorische Maßnahmen“ treffen, um die „Pseudonymisierung und Verschlüsselung personenbezogener Daten“ gewährleisten zu können. Zudem müssen die Faktoren Vertraulichkeit, Integrität und Verfügbarkeit der Daten sichergestellt werden und die Verschlüsselung der Daten muss nach Artikel 5 Absatz 2 der DSGVO nachweisbar sein. Dadurch verstößt der aktuell implementierte Prototyp gegen die DSGVO und erfüllt dadurch nicht die NANF-DS Nr. 1, weshalb in Zukunft hier überlegt werden sollte, mit welcher Verschlüsselungstechnik die Daten am besten vor fremden Zugriffen geschützt sind. Für die Verschlüsselung der Daten könnte unabhängig des digitalen COVID-19-Zertifikats der EU die europäischen Richtlinien für alle Daten herangezogen werden. Zur Verstärkung der Sicherheit sollte bei der Weiterentwicklung des Prototypen außerdem die dezentrale Speicherung, wie im Konzept beschrieben, umgesetzt werden.

In Bezug auf den bereits erwähnten Datenschutz ist es in dem Prototyp außerdem als problematisch anzusehen, dass die Patients vor dem Versenden der eingetragenen Nebenwirkungen nicht explizit zustimmen müssen, dass die Daten für Forschungszwecke genutzt werden dürfen. Es ist lediglich ein Hinweis zu sehen, dass die Daten für die Forschung anonym genutzt werden. Zusammen mit der Implementierung der Researcher Applikation sollte eine solche Zustimmung des Patients umgesetzt werden.

Ebenfalls kritisch zu betrachten sind die Funktionalitäten des Verifiers in Bezug auf die prototypische Implementierung. So besteht aktuell eine Verbindung zur gesamten my.vay-

Datenbank, sobald ein Verifier eine Impfung verifiziert und den Impfnachweis einscannt, da die Scan-Funktion in my.vay selbst implementiert wurde. Aus datenschutzrechtlichen Gründen empfiehlt es sich an dieser Stelle, die Verifier-Komponente ausgelagert und separiert zu implementieren, sodass ein Zugriff auf die Datenbank nicht ermöglicht werden kann.

7.3 Architektur

Die in Kapitel 5.6 erarbeitete Architektur des Konzepts ist ebenfalls kritisch zu betrachten. Die Umsetzung der Plattform mit dezentraler Speicherung bietet zwar den Vorteil der Datensicherheit, jedoch darf durch die Komplexität der Architektur und den jeweiligen Zugriff auf die Data Nodes die Latenzzeit nicht zu hoch werden. Des Weiteren muss die Replikation der Daten zwischen den Data Nodes an verschiedenen Standorten über ein sicheres Übertragungsmedium stattfinden.

7.4 Fehlende Evaluierung

Gesamtheitlich betrachtet ist anzumerken, dass aus Zeitgründen keine Evaluierung durchgeführt werden konnte, weshalb keine gehaltvollen Aussagen zur Benutzerfreundlichkeit und -akzeptanz formuliert werden können. Zukünftig sollte der Prototyp von my.vay beispielsweise einem Feldversuch unterzogen werden, damit genau diese Aspekte für die künftigen Nutzer analysiert und verbessert werden können. Des Weiteren sollte das Design des Prototypen mithilfe von *Mockups* erstellt werden, sodass eine Applikation mit guter *User Experience* entsteht, da dies nicht im Fokus dieser Seminararbeit lag. Zudem können keine Aussagen zu den in Kapitel 4.1.1 definierten Kernzielen drei und vier getätigt werden, da weder evaluiert werden konnte, ob durch das erarbeitete Konzept tatsächlich eine nachhaltige Verbesserung des Nachweisvorgangs einer durchgeführten Impfung erreicht noch, ob die durchschnittliche Impfquote durch eine Erinnerungsfunktion gesteigert werden kann.

8 Zusammenfassung

Das folgende Kapitel fasst die Inhalte der Seminararbeit zusammen und bietet einen Ausblick auf die künftige weitere Umsetzung.

8.1 Fazit

Die anhaltende COVID-19-Pandemie hat die Wichtigkeit von Impfungen und deren Dokumentation erneut in den Fokus der Menschheit gerückt. Zusätzlich wurde in Pandemiezeiten erkannt, dass vor allem der Impfpass vielfältige Digitalisierungsmöglichkeiten bietet. Das Ziel der Arbeit war es ein Konzept zur Verbesserung des aktuellen Impfpasses zu konzipieren. Um dies umzusetzen, wurden zunächst verschiedene grundlegende Thematiken recherchiert und beschrieben. Hierzu zählen beispielsweise die aktuellen Herausforderungen des analogen Impfpasses und das digitale COVID-19-Zertifikat der EU. Daraufhin wurde ermittelt, welche verwandten Arbeiten in diesem Bereich bereits existieren und wie bisherige Lösungen gestaltet sind. Anschließend wurden die Kernziele für die Umsetzung des möglichen Konzepts sowie dessen Anforderungen definiert. Die funktionalen Anforderungen adressieren dabei die verschiedenen Nutzergruppen und deren jeweilige Applikation. Zu den Nutzergruppen zählen der Patient, der Issuer, der Verifier und der Researcher. In den anschließenden nicht-funktionalen Anforderungen wird unter anderem die Benutzerfreundlichkeit und der Datenschutz thematisiert.

Aus den Anforderungen wurde schließlich in Anlehnung an das 4+1 View Model ein Konzept mithilfe verschiedener UML-Diagramme entwickelt. Wie in den Diagrammen beschrieben, soll die Plattform my.vay entwickelt werden, welche aus sechs Subsystemen entsteht. Diese sollen die Funktionalitäten für die *Applications* der Nutzergruppe der Patients, der Issuer und der Researcher bereitstellen. Die Hauptkomponente der Plattform ist der Impfpass des Patients, welcher alle digitalen Impfeinträge anzeigt, an Impfauffrischungen erinnert und die Eintragungen von Nebenwirkungen ermöglicht. Die Ausstellung von Impfeinträgen findet mittels eines QR-Codes über den Issuer statt. Neben dem Impfpass für den Patient stellt my.vay Patients und Issuern außerdem ein Wissensmanagement zur Verfügung, in

welchem allgemein wichtige Informationen zu Impfungen und deren Nebenwirkungen nachlesbar sind. Des Weiteren kann mithilfe von QR-Codes die Gültigkeit einzelner Impfeinträge nachgewiesen werden. Diese Funktion ist für die Nutzergruppe des Verifiers relevant.

In einem nächsten Schritt wurden die Hauptfunktionalitäten des dargestellten Konzepts in Form eines Prototypen umgesetzt. Der Prototypen stellt eine mit Python implementierte webbasierte Anwendung bereit. Mithilfe des Protoypen konnte der Prozess eines digitalen Impfpasses grundlegend visualisiert werden, sodass ein erster Eindruck darüber entsteht, wie ein künftiger digitaler Impfprozess aussehen kann. Der Prototyp erhebt durch verschieden getroffene Abgrenzungen keinen Anspruch auf Vollständigkeit, ermöglicht es jedoch die Nützlichkeit der Plattform unmittelbar zu testen.

Zusammenfassend ist festzustellen, dass das Konzept von my.vay alle gesetzten Kernziele in ihren Grundzügen aufgreift. Mithilfe des Konzepts und des Prototypen konnte die Forschungsfrage „Welche Funktionalitäten sollte ein digitaler Impfpass für die bestmögliche Unterstützung des Impfprozesses bieten und wie sind diese umzusetzen?“ aufgegriffen und beantwortet werden, indem eine Variante der Umsetzung aufgezeigt wird. Der Impfprozess wird dabei durch verschiedene digitale Funktionalitäten ausgestaltet. Hierzu zählt sowohl der eigentliche Impfpass mit den digitalen Impfeinträgen als auch die Impferinnerung, der Impfkalender, die Speicherung der Impfnebenwirkungen, das Wissensmanagement und der Impfnachweis. Auf diese Weise wird ein schneller, digitaler, nachvollziehbarer und umfassender Impfpass geboten.

8.2 Ausblick

In einer zukünftigen weiteren Umsetzung von my.vay sollten die aufgegriffenen Kernziele des Konzepts auf den tatsächlichen Erfolg mit Hilfe des Prototypen getestet werden. Dieser könnte beispielsweise mittels eines Feldversuchs auf Nutzerfreundlichkeit und Bedienbarkeit überprüft werden. Die Evaluierung ermöglicht es, Auffälligkeiten und Probleme über das Feedback der Nutzergruppen zu erkennen und die Plattform dadurch agil weiterentwickeln zu können. Hierbei sollten zudem Mockups für die Plattform entworfen werden, um das Design des Prototypen zu dokumentieren und eine positive *User Experience* sicherzustellen.

In einem nächsten Schritt sollten außerdem die bisher abgegrenzten Anforderungen von my.vay umgesetzt werden. Hierzu zählt insbesondere der Impfkalender für den Patient und

das Wissensmanagement. Über den Impfkalender kann dann auch die Funktionalität implementiert werden, Impftermine einzutragen. Die Informationen des Wissensmanagements sollten außerdem von einer Fachperson vor der Veröffentlichung auf Richtigkeit überprüft werden. Darüber hinaus sollte auch das Teilen der Impfdaten des Patients mit einer ärztlichen Einrichtung, sowie die Verbindung zum PVS ermöglicht werden. Zudem sollte auch die Applikation des Researchers entwickelt werden, um frühestmöglich die Daten für Forschungszwecke nutzen zu können.

A Quelltext des Prototypen

A.1 ImpfnachweisForm(Form)

```
1
2 from wtforms import TextField, StringField, DateTimeField, BooleanField,
      SubmitField, IntegerField, DateField, PasswordField, SelectField
3 from wtforms.validators import DataRequired, Length
4
5 class ImpfnachweisForm (Form):
6     f_name = TextField("Vorname␣des␣Geimpften:␣")
7     l_name = TextField("Nachname␣des␣Geimpften:␣")
8     date_of_birth = DateField("Geburtsdatum␣des␣Geimpften:␣",validators
          =[DataRequired(), Length(max=30)],render_kw={"placeholder": "YYYY
          -mm-dd"})
9     date_of_vaccination= DateField("Datum␣der␣Impfung:␣", validators=[
          DataRequired(), Length(max=30)],render_kw={"placeholder": "YYYY-
          mm-dd"})
10    vaccination_id = SelectField("Impfkategorie:␣")
11    vaccine = StringField("Impfstoff:␣")
12    vaccine_marketing_authorization_holder = TextField("Hersteller:␣")
13    batch_number = StringField("Chargennummer:␣")
14    issued_at = DateTimeField ("Ausgestellt␣am:␣", render_kw={"
          placeholder": "YYYY-mm-dd␣hh:mm:ss"})
15    generate_certificate =  SubmitField("Impfnachweis␣erstellen")
```

Quelltext A.1: Form „ImpfnachweisForm(Form)“

A.2 Funktion „issuer_create_qr()“

```
1  def issuer_create_qr():
2      form = ImpfnachweisForm()
3      [...]
4      qr = {}
5      img = []
6      file_object = io.BytesIO()
7      [...]
8      qr = QRCode(version=1, box_size=3,border=3)
9      qr.add_data(proof_of_vaccination)
10     qr.make()
11     img = qr.make_image (fill = 'black', back_color = 'white')
12     img.save(file_object,'PNG')
13
14     return render_template("/issuer/issuer_create_qr.html", form=form,
           qr=""+b64encode(file_object.getvalue()).
           decode('ascii'))
```

Quelltext A.2: Ausschnitt der Funktion „issuer_create_qr()“

A.3 *Template* „patient_scan“

```
<!-- Bootstrap container with rows and colums for better page layout -->
<div class="container my-5">
        <div class="row">
                <div class="col-2"> </div>
                <div class="col-8 text-center">

                        <!-- Image, calling with jinja the camera_stream
                             route for the live-view of the camera. -->
                        <img class="mb-3" src="{{ url_for('camera_stream
                            ') }}" width="100%">

                        <!-- Button, which redirects to the
                            patient_qr_result route for verifiying the
                            presented QR-Code -->
                        <a class="btn btn-success" href="{{ url_for('
                            patient_qr_result') }}">QR-Code-Überprüfung
                            starten</a>

                </div>
                <div class="col-2"></div>
        </div>
</div>
```

Quelltext A.3: Ausschnitt des *Templates* „patient_scan“

A.4 Route „patient_login()“ und „patient_registration()“

```
1  from my_vay import app, db
2  from flask import render_template, url_for, redirect, flash, session
3  from my_vay.forms import PatientLoginForm, PatientRegistrationForm
4  from flask_login import login_user, current_user, login_required
5  from my_vay.models import Patient
6
7
8  # Patient - Login route
9  @app.route("/patient/login", methods =["GET", "POST"])
10 def patient_login():
11
12     # Redirects user to the patient landing page, if he is already
           signed in
13     if current_user.is_authenticated:
14         if session['user_type'] == 'patient':
15             return redirect(url_for('patient_home'))
16
17     # Loads the PatientLoginForm from forms.py
18     form = PatientLoginForm()
19
20     # If the form is submitted and validated then...
21     if form.validate_on_submit():
22
23         # Database is queryed based on the unique_patient_identifier
24         patient = Patient.query.filter_by(unique_patient_identifier=form
               .unique_patient_identifier.data).first()
25
26         # If a patient with the entered unique_patient_identifier exists
               and the password in the database is the same as in the form
              then...
27         if patient and patient.password == form.password.data:
28
29             # Patient is written into a cookie and user is logged in
30             session['user_type'] = 'patient'
31             login_user(patient, remember=form.remember.data)
32
33             # Patient is redirected to the patient landing page
34             return redirect(url_for('patient_home'))
```

```
35
36          else:
37              flash('Es existiert kein Patient mit dieser Nutzer ID!', '
                    danger')
38              return redirect(url_for('patient_login'))
39
40      return render_template('patient/patient_login.html', form=form)
41
42  # Patient - Registration route
43  @app.route("/patient/registrierung", methods =["GET", "POST"])
44  def patient_registration():
45      form = PatientRegistrationForm()
46
47      # If the form is submitted and validated then...
48      if form.validate_on_submit():
49
50          # a new patient is added to the database
51          new_patient = Patient(f_name=form.f_name.data, l_name=form.
                l_name.data, date_of_birth=form.date_of_birth.data,
                unique_patient_identifier=form.unique_patient_identifier.data
                , password=form.password.data)
52          db.session.add(new_patient)
53          db.session.commit()
54
55          return redirect(url_for('patient_login'))
56
57      return render_template('patient/patient_registration.html', form=
            form)
```

Quelltext A.4: Route „patient_login()“ und „patient_registration()“

A.5 Route „patient_home()“

```
1
2 from my_vay import app, db
3 from flask import render_template, abort, url_for, redirect, request,
      session, Response
4 from my_vay.forms import SearchVaccine
5 from flask_login import current_user, login_required
6 from my_vay.models import Patient, Proof_of_vaccination, Vaccination
7
8
9 # Patient - Landing page route
10 @app.route("/patient",methods=['POST', 'GET'])
11 @app.route("/patient/<string:sort>", methods=['POST', 'GET'])
12 @app.route("/patient/<string:sort>/<string:search>", methods=['POST', '
      GET'])
13 @login_required
14 def patient_home(sort='date', search=''):
15
16     page = request.args.get('page', 1, type=int)
17     #shows search form
18     form = SearchVaccine()
19     vaccine_search = False
20
21     # Checks if a search term is used. If yes then only vaccinations are
           shown that fulfill the vaccine search statement. Otherwise all
          vaccinations entries for this patient are shown.
22     if search:
23         vaccine_search = True
24         search = '%' + search + '%'
25         branch = Proof_of_vaccination.query.filter(Proof_of_vaccination.
               unique_patient_identifier == current_user.
               unique_patient_identifier).filter(Proof_of_vaccination.
               vaccine.like(search)).paginate(page=page, per_page=5)
26         vaccination = Vaccination.query.filter(Proof_of_vaccination.
               vaccination_id == Vaccination.vaccination_id).all()
27     else:
28         # Depending on an argument in the url, the patients are sorted
               in different ways. The value 'date' is set as the default
               sort value. All vaccination entries are shown in ordered by
               date_of_vaccination.
29         if sort == 'date':
```

```
30              branch = Proof_of_vaccination.query.filter_by(
                    unique_patient_identifier=current_user.
                    unique_patient_identifier).order_by(Proof_of_vaccination.
                    date_of_vaccination.desc()).paginate(page=page, per_page
                    =5)
31              vaccination = Vaccination.query.filter(Proof_of_vaccination.
                    vaccination_id == Vaccination.vaccination_id).all()
32
33          # Different filter values can be selected through the drop-down-
                menu
34          elif sort == 'Pneumokokken':
35              branch = Proof_of_vaccination.query.filter(
                    Proof_of_vaccination.unique_patient_identifier ==
                    current_user.unique_patient_identifier).paginate(page=
                    page, per_page=5)
36              vaccination = Vaccination.query.filter(Vaccination.disease
                    == sort).all()
37
38          elif sort == 'Hepatitis B':
39              branch = Proof_of_vaccination.query.filter(
                    Proof_of_vaccination.unique_patient_identifier ==
                    current_user.unique_patient_identifier).paginate(page=
                    page, per_page=5)
40              vaccination = Vaccination.query.filter(Vaccination.disease
                    == sort).all()
41
42          elif sort == 'COVID-19':
43              branch = Proof_of_vaccination.query.filter(
                    Proof_of_vaccination.unique_patient_identifier ==
                    current_user.unique_patient_identifier).paginate(page=
                    page, per_page=5)
44              vaccination = Vaccination.query.filter(Vaccination.disease
                    == sort).all()
45
46          else:
47              abort(404)
48
49      # runs if a search value was submitted
50      if form.is_submitted():
51          return redirect(url_for('patient_home', sort='date', search=form
                .name.data))
52
```

```
53      #call the functions check_for_vac_notifications() and
            show_sideeffect()
54      vac_notifications = check_for_vac_notifications()
55      vac_sideeffects = show_sideeffect()
56
57      return render_template('patient/patient_vaccination_certificate.html
            ', branch=branch, sort=sort, form=form, search=vaccine_search,
            vac_notifications=vac_notifications, vac_sideeffects=
            vac_sideeffects, vaccination=vaccination)
```

Quelltext A.5: Route „patient_home()“

A.6 Route „open_QR()“

```
1 from my_vay import app, db
2 from flask import render_template
3 from my_vay.forms import AddSideeffects, ImpfnachweisForm,
      PatientLoginForm, AddVaccination, PatientRegistrationForm,
      IssuerRegistrationForm, IssuerLoginForm, IssuerUpdateForm,
      PatientUpdateForm, SearchVaccine
4 from flask_login import current_user
5 from my_vay.models import Patient, Proof_of_vaccination, Vaccination
6 from base64 import b64encode, decode
7 import io
8 from qrcode.main import QRCode
9 import cv2
10 import ast
11 import sys
12 from pyzbar import pyzbar
13 from pyzbar.pyzbar import decode
14
15
16 # Patient - Show QR-Code of proof_of_vaccination
17 @app.route("/patientQR/<int:unique_certificate_identifier>", methods=['
      POST', 'GET'])
18 def open_QR(unique_certificate_identifier):
19
20     # collects the values for the selected vaccination entry so a QR
          code can be generated
21     branch = Proof_of_vaccination.query.filter_by(
          unique_certificate_identifier=unique_certificate_identifier).
          first()
22     vaccination = Vaccination.query.filter(Proof_of_vaccination.
          vaccination_id == Vaccination.vaccination_id).first()
23
24     qr = {}
25     img = []
26     file_object = io.BytesIO()
27
28     # save vaccinations values in dictionary
29     proof_of_vaccination= {}
30     proof_of_vaccination['unique_certificate_identifier']= branch.
          unique_certificate_identifier
31     proof_of_vaccination['date_of_vaccination']= branch.
```

```
        date_of_vaccination.strftime('%Y-%m-%d')
32      proof_of_vaccination['vaccine']= branch.vaccine
33      proof_of_vaccination['vaccination_id'] = branch.vaccination_id
34      proof_of_vaccination['vaccine_marketing_authorization_holder']=
            branch.vaccine_marketing_authorization_holder
35      proof_of_vaccination['batch_number']= branch.batch_number
36      proof_of_vaccination['issued_at']= branch.issued_at.strftime('%Y-%m
            -%d␣%H:%M:%S')
37      proof_of_vaccination['unique_patient_identifier']= branch.
            unique_patient_identifier
38      proof_of_vaccination['unique_issuer_identifier']= branch.
            unique_issuer_identifier
39
40      print(proof_of_vaccination)
41
42      # create QR code with given values
43      qr = QRCode(version=1, box_size=6,border=4)
44      qr.add_data(proof_of_vaccination)
45      qr.make()
46          #qr = qrcode.make(proof_of_vaccination)
47      img = qr.make_image (fill = 'black', back_color = 'white')
48      img.save(file_object,'PNG')
49
50      return render_template('patient/patient_show_QR.html', branch =
            branch, vaccination=vaccination, qr=""+
            b64encode(file_object.getvalue()).decode('ascii'))
```

Quelltext A.6: Route „open_QR()“

A.7 Route „check_for_vac_notifications()“

```
from my_vay import app, db
from flask_login import current_user
from my_vay.models import Proof_of_vaccination, Vaccination
from datetime import date, timedelta
from dateutil import relativedelta


# Function for returning notifications for missing vaccinations
def check_for_vac_notifications():
    # Returns the proof_of_vaccinations of the logged-in-user from the database
    list_of_proof_of_vaccinations = Proof_of_vaccination.query.filter_by(unique_patient_identifier=current_user.unique_patient_identifier).all()
    notifications = []

    # Iterates through the proof_of_vaccinations
    for entry_in_proof_of_vaccinations in list_of_proof_of_vaccinations:

        # Returns the associated vaccination from proof_of_vaccinations
        associated_entry_in_vaccination = Vaccination.query.get(entry_in_proof_of_vaccinations.vaccination_id)
        next_vaccination_id = associated_entry_in_vaccination.next_vaccination_id

        notification_dict = {
            "next_vaccination_possible" : False,
            "next_vaccination_not_exisisting" : False,
            "beginn_age_reached": False,
            "end_age_not_reached": False,
            "dist_between_current_and_next_correct": False
        }

        # If this vaccination has a following vaccination...
        if next_vaccination_id:
            notification_dict["next_vaccination_possible"] = True

            #... then it's checked, if the following vaccination is missing.
            next_vaccination_already_existing = Proof_of_vaccination.
```

```
                query.filter_by(vaccination_id=next_vaccination_id).first
                ()
35
36              if next_vaccination_already_existing is None:
37                  notification_dict["next_vaccination_not_exisisting"] =
                        True
38
39                  # Date of birth from current_user and today's date is
                        detected
40                  user_date_of_birth = current_user.date_of_birth
41                  date_today = date.today()
42
43                  print("user_date_of_birth:␣" + str(type(
                        user_date_of_birth)), file=sys.stderr)
44                  print("date_today:␣" + str(type(date_today)), file=sys.
                        stderr)
45
46                  # Calculate difference in months between date_of_birth
                        and today (Age in months)
47                  age = relativedelta.relativedelta(date_today,
                        user_date_of_birth)
48                  age_in_months = age.months + age.years * 12
49
50                  # Returns the next vaccination from database
51                  next_vaccination = Vaccination.query.get(
                        next_vaccination_id)
52
53                  # Beginn_age, end_age and distance to previous
                        vaccination are read-out
54                  next_vaccination_beginn_age = next_vaccination.
                        beginn_age
55                  next_vaccination_end_age = next_vaccination.end_age
56                  next_vaccination_dist_to_pre_vac = next_vaccination.
                        distance_to_pre_vaccination
57
58                  if next_vaccination_beginn_age:
59                      if age_in_months >= next_vaccination_beginn_age:
60                          notification_dict["beginn_age_reached"] = True
61                      else:
62                          notification_dict["beginn_age_reached"] = False
63                  else:
64                      notification_dict["beginn_age_reached"] = True
```

```
65
66                  if next_vaccination_end_age:
67                      if age_in_months <= next_vaccination_end_age:
68                          notification_dict["end_age_not_reached"] = True
69                      else:
70                          notification_dict["end_age_not_reached"] = False
71                  else:
72                      notification_dict["end_age_not_reached"] = True
73
74                  if next_vaccination_dist_to_pre_vac:
75
76                      date_of_current_vac = entry_in_proof_of_vaccinations
                            .date_of_vaccination
77                      dist_between_current_and_next_vac = relativedelta.
                            relativedelta(date_today, date_of_current_vac)
78                      dist_between_current_and_next_vac_in_months =
                            dist_between_current_and_next_vac.months +
                            dist_between_current_and_next_vac.years * 12
79
80                      if next_vaccination_dist_to_pre_vac <=
                            dist_between_current_and_next_vac_in_months:
81                          notification_dict["
                                dist_between_current_and_next_correct"] =
                                True
82                      else:
83                          notification_dict["
                                dist_between_current_and_next_correct"] =
                                False
84                  else:
85                      notification_dict["
                            dist_between_current_and_next_correct"] = True
86              else:
87                  notification_dict["next_vaccination_not_exisisting"] =
                        False
88          else:
89              notification_dict["next_vaccination_possible"] = False
90
91          if notification_dict["beginn_age_reached"] == True and
                notification_dict["end_age_not_reached"] == True and
                notification_dict["dist_between_current_and_next_correct"] ==
                 True and notification_dict["next_vaccination_not_exisisting"
                ] == True and notification_dict["next_vaccination_possible"]
```

```
            == True:
92
93              notification = {
94                  "disease" : next_vaccination.disease,
95                  "current_unique_certificate_identifier" :
                        entry_in_proof_of_vaccinations.
                        unique_certificate_identifier,
96                  "current_date_of_vaccination":
                        entry_in_proof_of_vaccinations.date_of_vaccination,
97                  "current_vaccine_category" :
                        associated_entry_in_vaccination.vaccine_category,
98                  "next_vaccine_category" : next_vaccination.
                        vaccine_category
99              }
100
101             notifications.append(notification)
102
103     return notifications
```

Quelltext A.7: Route „check_for_vac_notifications()“

A.8 Route „show_sideeffect()“

```
from my_vay import app, db
from flask_login import current_user
from my_vay.models import Proof_of_vaccination, Sideeffects
from datetime import date, timedelta
from dateutil import relativedelta


# Function to deside whether a sideeffect reminder should be shown or not, is called in patient_home
def show_sideeffect():
    from datetime import date
    today = 0
    sideeffects_list=[]
    sf_show = Proof_of_vaccination.query.filter_by(unique_patient_identifier=current_user.unique_patient_identifier).all()
    sideeffect_check = Sideeffects.query.all()
    today = date.today()
    #get the date of yesterday
    for i in range (1):
            today-= timedelta(days=1)
    today = today.strftime('%Y-%m-%d')

    #checks if a vaccination has the same date as the today variable, because then a reminder should be shown
    for entry in sf_show:
        classifier= False
        if entry.date_of_vaccination.strftime('%Y-%m-%d') == today:
            for val in sideeffect_check:
                    if entry.unique_certificate_identifier == val.unique_certificate_identifier:
                        classifier = True
                        break
            if classifier == False:
                #sideeffects can only be entered if an entry for this vaccination does not already exist
                sideeffects_list.append(entry)
    # reuturns list of vaccination entries in proof_of_vaccination that fulfill conditions
```

```
33      return sideeffects_list
```

Quelltext A.8: Route „show_sideeffect()“

A.9 Route „new_sideeffect()“

```
from my_vay import app, db
from flask import render_template, url_for, redirect, session
from my_vay.forms import AddSideeffects
from flask_login import current_user,login_required
from my_vay.models import Proof_of_vaccination Sideeffects


# Patient - Sideeffects route
# is only callable if show_sideeffect() function is satisfied
@app.route("/patient/sideeffects/<int:unique_certificate_identifier>",
    methods=['POST', 'GET'])
@login_required
def new_sideeffect(unique_certificate_identifier):
    branch = Proof_of_vaccination.query.filter_by(
        unique_certificate_identifier=unique_certificate_identifier).
        first()

    #shows the form to create a new sideeffect entry
    form = AddSideeffects()

    # If the form is submitted then...
    if form.is_submitted():
        unique_entry_identifier = 1
        while Sideeffects.query.filter_by(unique_entry_identifier=
            unique_entry_identifier).first() is not None:
            unique_entry_identifier = unique_entry_identifier + 1
        # a new sideeffect is added to the database
        new_sideeffects = Sideeffects(unique_entry_identifier=
            unique_entry_identifier, unique_certificate_identifier=
            unique_certificate_identifier ,headache=form.headache.data,
            arm_hurts=form.arm_hurts.data, rash=form.rash.data, fever=
            form.fever.data, tummyache=form.tummyache.data, sideeffects=
            form.sideeffects.data)
        db.session.add(new_sideeffects)
        db.session.commit()

        return redirect(url_for('patient_home'))

    return render_template('patient/patient_sideeffects.html', branch =
```

```
branch , form=form )
```

Quelltext A.9: Route „new_sideeffect()“

A.10 Route „addVaccination()“

```
1 from my_vay import app, db
2 from flask import render_template, url_for, redirect, session
3 from my_vay.forms import AddVaccination
4 from flask_login import current_user, login_required
5 from my_vay.models import Proof_of_vaccination, Vaccination
6
7
8 # Patient - Proof of vaccination manual entry route
9 @app.route("/patient/impfeintrag/manuell", methods=['POST', 'GET'])
10 @login_required
11 def addVaccination():
12
13     # shows form to create a new vaccination entry - this entry is not
           verified so no QR code can be displayed for it later
14     form = AddVaccination()
15
16     # selects drop-down-choices for the vaccination_id SelectField (
           vaccine_category and disease are shown in frontend instead of
           vaccination_id)
17     form.vaccination_id.choices = [(int(vaccination.vaccination_id),
           vaccination.disease + "␣(" + vaccination.vaccine_category + ")")
           for vaccination in Vaccination.query.all()]
18
19     # if the form is submitted...
20     if form.is_submitted():
21         # the next open unique_certificate_identifier is selected...
22         unique_certificate_identifier = 1
23         while Proof_of_vaccination.query.filter_by(
               unique_certificate_identifier=unique_certificate_identifier).
               first() is not None:
24             unique_certificate_identifier =
                   unique_certificate_identifier + 1
25         # and a new vaccination entry is added to the database, issed_at
               value and vaccine_marketing_authorization_holder get a
               default value, because the vaccination entry is not verified
26         new_vaccination = Proof_of_vaccination(
               unique_certificate_identifier=unique_certificate_identifier,
               unique_patient_identifier= current_user.
               unique_patient_identifier, date_of_vaccination = form.
               date_of_vaccination.data, vaccine = form.vaccine.data,
```

```
            batch_number=form.batch_number.data, vaccination_id=form.
            vaccination_id.data, unique_issuer_identifier=form.
            unique_issuer_identifier.data,
            vaccine_marketing_authorization_holder="/", issued_at="
            1900-01-01␣00:00:00")
27          db.session.add(new_vaccination)
28          db.session.commit()
29
30          return redirect(url_for('patient_home'))
31
32      return render_template('patient/patient_vaccination_manual_entry.
        html', form=form)
```

Quelltext A.10: Route „addVaccination()“

A.11 Route „camera_stream()“

```python
# Necessary imports for the route
from my.vay import app
from flask import Response
import cv2

#  Camera stream route
@app.route("/camera_stream")
def camera_stream():

    def generate_frames():
        # Instantiation of a camera object refering to the camera of the device
        camera = cv2.VideoCapture(0)

        while True:
            # Returning frames from the camera
            success, frame = camera.read()

            if not success:
                break
            else:
                # Coverting frames to a jpg
                ret, buffer = cv2.imencode('.jpg', frame)
                frame = buffer.tobytes()
                yield b'--frame\r\n' b'Content-Type: image/jpeg\r\n\r\n' + frame + b'\r\n'

    # Returning Resonse with calling function generate_frames
    return Response(generate_frames(), mimetype='multipart/x-mixed-replace; boundary=frame')
```

Quelltext A.11: Route „camera_stream()“

A.12 Route „patient_qr_result()“

```
# Necessary imports for the route
from my.vay import app, db
from my.vay.models import Proof_of_vaccination
from flask import render_template
from flask_login import login_required, current_user

# Patient - QR-Code result route
@app.route("/patient/impfeintrag/scan-result")
@login_required
def patient_qr_result():

    # Calls read_in_frame_for_decoding(), which returns a dictionary representing the proof of vaccination
    proof_of_vaccination_qr = read_in_frame_for_decoding()

    # Checks if proof of vaccination already exists in the database. If so, then an error message is displayed.
    if Proof_of_vaccination.query.filter_by(unique_certificate_identifier = proof_of_vaccination_qr["unique_certificate_identifier"]).first():

        error_message ="[...]"
        return render_template("patient/patient_scan_result.html", error_message=error_message)

    else:
        # Checks if the issuer of the certificate is existing in the database
        if Issuer.query.filter_by(unique_issuer_identifier = proof_of_vaccination_qr["unique_issuer_identifier"]).first():

            # Checks if first name, last name and birthdate are the same of proof_of_vaccination and the logged in user.
            if proof_of_vaccination_qr['f_name'] == current_user.f_name and proof_of_vaccination_qr['l_name'] == current_user.l_name and proof_of_vaccination_qr['date_of_birth'] == str(current_user.date_of_birth):

                # If it's the same, then the proof_of_vaccination is added to the database.
```

```
29                  new_vaccination = Proof_of_vaccination(
                        unique_certificate_identifier =
                        proof_of_vaccination_qr["
                        unique_certificate_identifier"],
                        unique_patient_identifier=current_user.
                        unique_patient_identifier, date_of_vaccination=
                        proof_of_vaccination_qr["date_of_vaccination"],
                        vaccine=proof_of_vaccination_qr["vaccine"],
                        batch_number=proof_of_vaccination_qr["batch_number"],
                         vaccination_id=proof_of_vaccination_qr["
                        vaccination_id"], unique_issuer_identifier=
                        proof_of_vaccination_qr["unique_issuer_identifier"],
                        vaccine_marketing_authorization_holder=
                        proof_of_vaccination_qr["
                        vaccine_marketing_authorization_holder"], issued_at=
                        proof_of_vaccination_qr["issued_at"])
30
31                  db.session.add(new_vaccination)
32                  db.session.commit()
33
34                  return render_template("patient/patient_scan_result.html
                        ", proof_of_vaccination_qr=proof_of_vaccination_qr)
35              else:
36                  # If the data is not equal, then an error message is
                        displayed.
37                  error_message = "[...]"
38                  return render_template("patient/patient_scan_result.html
                        ", error_message=error_message)
39          else:
40              # If the issuer is not existing, then an error message is
                    displayed.
41              error_message = "[...]"
42              return render_template("patient/patient_scan_result.html",
                    error_message=error_message)
```

Quelltext A.12: Route „patient_qr_result()“

A.13 Route „read_in_frame_for_decoding()“

```
1 # Necessary imports for the function
2 import cv2
3
4 # Calling decode(frame) for every frame of the camera
5 def read_in_frame_for_decoding():
6     # Instanciation of a camera object
7     camera = cv2.VideoCapture(0)
8
9     # For every returned frame of the camera the function decode() is
          called, until it returns a result
10    while True:
11        success, frame = camera.read()
12
13        if not success:
14            break
15        else:
16            decodedObject = decode_QR_code_from_frame(frame)
17
18            if decodedObject:
19                break
20
21    return decodedObject
```

Quelltext A.13: Funktion „read_in_frame_for_decoding()“

A.14 Funktion „decode_QR_code_from_frame()“

```
1  # Necessary imports for the function
2  from pyzbar import pyzbar
3  from pyzbar.pyzbar import decode
4  from base64 import decode
5  import ast
6
7  # Decoding a QR-Code in a given frame
8  def decode_QR_code_from_frame(frame):
9      # Decode QR-Code
10     decodedObject = pyzbar.decode(frame)
11
12     # If decodedObject exists, then the data is returned as string
13     if decodedObject:
14         # Read out the data of the QR-Code
15         decodedObjectData = decodedObject[0].data
16
17         # Decode the content with UTF-8
18         decodedObjectDataUTF = decodedObjectData.decode('utf-8')
19
20         # Covert String to
21         decodedObjectDict = ast.literal_eval(decodedObjectDataUTF)
22
23         return decodedObjectDict
```

Quelltext A.14: Funktion „decode_QR_code_from_frame()“

A.15 Route „verifier_qr_result()“

```
1 # Verifier - QR-Code result route
2 @app.route("/verifier/scan-result")
3 def verifier_qr_result():
4     proof_of_vaccination_correct = False
5     patient = None
6
7     proof_of_vaccination_qr = read_in_frame_for_decoding()
8
9     # Checks if encoded QR-code contains all relevant data
10    if "unique_certificate_identifier" in proof_of_vaccination_qr and "
          date_of_vaccination" in proof_of_vaccination_qr and "vaccine" in
          proof_of_vaccination_qr and "
          vaccine_marketing_authorization_holder" in
          proof_of_vaccination_qr and "batch_number" in
          proof_of_vaccination_qr and "issued_at" in
          proof_of_vaccination_qr and "unique_patient_identifier" in
          proof_of_vaccination_qr and "unique_issuer_identifier" in
          proof_of_vaccination_qr and "vaccination_id" in
          proof_of_vaccination_qr:
11
12        unique_certificate_identifier = proof_of_vaccination_qr["
              unique_certificate_identifier"],
13        date_of_vaccination = proof_of_vaccination_qr["
              date_of_vaccination"],
14        vaccine = proof_of_vaccination_qr["vaccine"],
15        vaccine_marketing_authorization_holder = proof_of_vaccination_qr
              ["vaccine_marketing_authorization_holder"],
16        batch_number = proof_of_vaccination_qr["batch_number"],
17        issued_at = proof_of_vaccination_qr["issued_at"],
18        unique_patient_identifier = proof_of_vaccination_qr["
              unique_patient_identifier"],
19        unique_issuer_identifier = proof_of_vaccination_qr["
              unique_issuer_identifier"],
20        vaccination_id = proof_of_vaccination_qr["vaccination_id"]
21
22        # Checks if proof_of_vaccination exists in the database
23        if Proof_of_vaccination.query.filter_by(
              unique_certificate_identifier=unique_certificate_identifier,
              date_of_vaccination=date_of_vaccination, vaccine=vaccine,
              vaccine_marketing_authorization_holder=
```

```
            vaccine_marketing_authorization_holder, batch_number=
            batch_number, issued_at=issued_at, unique_patient_identifier=
            unique_patient_identifier, unique_issuer_identifier=
            unique_issuer_identifier, vaccination_id=vaccination_id).
            first():
24
25              patient = Patient.query.filter_by(unique_patient_identifier=
                    unique_patient_identifier).first()
26              proof_of_vaccination_correct = True
27
28      return render_template("verifier/verifier_scan_result.html",
            proof_of_vaccination_correct=proof_of_vaccination_correct,
            patient=patient)
```

Quelltext A.15: Route „verifier_qr_result()“

Literaturverzeichnis

[1] Aerzteblatt. *COVID-19-Krankheitslast in Deutschland im Jahr 2020*. Feb. 2021. URL: https://www.aerzteblatt.de/archiv/217880/COVID-19-Krankheitslast-in-Deutschland-im-Jahr-2020 (besucht am 25. 07. 2021).

[2] VFA Die forschenden Pharmaunternehmen. *Herdenimmunität: Mit Impfungen sich selbst und andere schützen*. Jan. 2021. URL: https://www.vfa.de/de/arzneimittel-forschung/impfen/herdenimmunitaet (besucht am 25. 07. 2021).

[3] Gregg N. Milligan und Alan D. T. Barrett. *Vaccinology*. O'Reilly Media, 2015, S. 313. ISBN: 9781118636282.

[4] Robert-Koch-Institut. *Epidemiologisches Bulletin*. Juli 2021. URL: https://www.rki.de/DE/Content/Infekt/EpidBull/Archiv/2021/Ausgaben/27_21.pdf?__blob=publicationFile (besucht am 25. 07. 2021).

[5] Rober-Koch-Institut. *Epidemiologisches Bulletin*. Aug. 2018. URL: https://www.rki.de/DE/Content/Infekt/EpidBull/Archiv/2018/Ausgaben/32_18.pdf?__blob=publicationFile (besucht am 25. 07. 2021).

[6] Patrick Tarkowski. *E-Health: Digitalisierung im Gesundheitswesen geht zu langsam voran*. Feb. 2019. URL: https://digital-magazin.de/e-health-digitalisierung-im-gesundheitswesen/ (besucht am 25. 07. 2021).

[7] Manuela Dursun und Christian Saathoff. *Gefälschte Impfpässe werden zum Problem*. 11. Mai 2021. URL: https://www.tagesschau.de/investigativ/report-mainz/gefaelschte-impfpaesse-101.html (besucht am 11. 06. 2021).

[8] Bundesministerium für Gesundheit. *Schutzimpfungen*. 9. Juni 2021. URL: https://www.bundesgesundheitsministerium.de/themen/praevention/impfungen/schutzimpfungen.html.

[9] Robert Koch Institut. *Impfen*. 9. Juni 2021. URL: https://www.rki.de/DE/Content/Infekt/Impfen/impfen_node.html.

[10] Robert Koch Institut. *Welche Abstände sind zwischen Impfungen einzuhalten?* 9. Juni 2021. URL: https://www.rki.de/SharedDocs/FAQ/Impfen/AllgFr_Impfschema/FAQ04.html;jsessionid=B98566646ADB814DF414FB38E3607748.internet121?nn=2391120.

[11] Robert Koch Institut. *Kann bei der Impfung vom Impfschema abgewichen werden?* 9. Juni 2021. URL: https://www.rki.de/SharedDocs/FAQ/Impfen/AllgFr_Impfschema/FAQ02.html;jsessionid=B98566646ADB814DF414FB38E3607748.internet121?nn=2391120.

[12] S. Dittmann. *Risiko des Impfens und das noch größere Risiko, nicht geimpft zu sein*. 1. Apr. 2020. URL: https://www.rki.de/DE/Content/Infekt/Impfen/Bedeutung/Downloads/Dittmann_Risiko.pdf?__blob=publicationFile.

[13] Robert Koch Institut. *Antworten des Robert Koch-Instituts und des Paul-Ehrlich-Instituts zu den 20 häufigsten Einwänden gegen das Impfen*. 9. Juni 2021. URL: https://www.rki.de/DE/Content/Infekt/Impfen/Bedeutung/Schutzimpfungen_20_Einwaende.html;jsessionid=B98566646ADB814DF414FB38E3607748.internet121?nn=2391120.

[14] Die Ständig Impfkommission. *Wie Impfempfehlungen zusatande kommen*. 9. Juni 2021. URL: https://www.impfen-info.de/assets/impfen-info.de/STIKO/#.

[15] Bundeszentrale für gesundheitliche Aufklärung. *Das Impfsystem in Deutschland*. 9. Juni 2021. URL: https://www.impfen-info.de/wissenswertes/impfsystem-in-deutschland.html.

[16] Robert Koch Institut. *Aufklärung vor Schutzimpfungen*. 9. Juni 2021. URL: https://www.rki.de/SharedDocs/FAQ/Impfen/Aufklaerung/FAQ-Liste.html;jsessionid=B98566646ADB814DF414FB38E3607748.internet121?nn=2391120.

[17] Robert Koch Institut. *Wie soll bei fehlender Impfdokumentation vorgegangen werden?* 9. Juni 2021. URL: https://www.rki.de/SharedDocs/FAQ/Impfen/AllgFr_AllgemeineFragen/FAQ01.html;jsessionid=B98566646ADB814DF414FB38E3607748.internet121?nn=2391120.

[18] Robert Koch Institut. *Impfkalender der Ständigen Impfkommission*. 9. Juni 2021. URL: https://www.rki.de/DE/Content/Infekt/Impfen/Impfkalender/Impfkalender_node.html;jsessionid=84CC72466AAE79043F2A6B6080BD9371.internet121.

[19] Robert Koch Institut. *Welche Abstände sind zwischen Impfungen einzuhalten?* 9. Juni 2021. URL: https://www.rki.de/SharedDocs/FAQ/Impfen/AllgFr_Impfschema/FAQ04.html;jsessionid=7B842A60C20F2ACB59D044D644EBED3B.internet061?nn=2391120.

[20] Robert Koch Institut. *Impfkalender (Standardimpfungen) für Säuglinge, Kinder, Jugendliche und Erwachsene; 2020/2021*. 11. Juni 2021. URL: https://www.rki.de/DE/Content/Kommissionen/STIKO/Empfehlungen/Aktuelles/Impfkalender.pdf?__blob=publicationFile.

[21] Robert Koch Institut. *Wie soll bei fehlender Impfdokumentation vorgegangen werden?* 9. Juni 2021. URL: https://www.rki.de/SharedDocs/FAQ/Impfen/AllgFr_AllgemeineFragen/FAQ01.html;jsessionid=7B842A60C20F2ACB59D044D644EBED3B.internet061?nn=2391120.

[22] Bundeszentrale für gesundheitliche Aufklärung. *Impfpass*. 9. Juni 2021. URL: https://www.impfen-info.de/mediathek/lexikon.html#index_I.

[23] Wikipedia. *Internationaler Impfausweis (2019)*. 5. Juli 2021. URL: https://de.wikipedia.org/wiki/Impfausweis#/media/Datei:Internationaler_Impfausweis.png.

[24] Die Techniker. *Der Impfausweis*. 9. Juni 2021. URL: https://www.tk.de/techniker/gesundheit-und-medizin/praevention-und-frueherkennung/impfungen-medizinische-hintergruende/impfausweis-2010182?tkcm=ab.

[25] Felix Huesmann. *Gefälschte Impfpässe stellen Bundesregierung vor Herausforderungen*. 6. Mai 2021. URL: https://www.rnd.de/politik/gefaelschte-impfpaesse-stellen-bundesregierung-vor-herausforderungen-TZDB5Z5E3ND3XFRLMH4O4YK6AI.html (besucht am 11. 06. 2021).

[26] Kripo live. *Fälscher bieten Impfausweise mitteldeutscher Impfzentren an*. 11. Mai 2021. URL: https://www.mdr.de/nachrichten/deutschland/panorama/corona-impfpass-impfausweis-faelschung-telegram-100.html (besucht am 14. 06. 2021).

[27] aerzteblatt.de. *Coronaimpfnachweis: Falsche Eintragung und Nutzung falscher Impfpässe strafbar*. 21. Mai 2021. URL: https://www.aerzteblatt.de/nachrichten/123979/Coronaimpfnachweis-Falsche-Eintragung-und-Nutzung-falscher-Impfpaesse-strafbar (besucht am 14. 06. 2021).

[28] Bayrischer Rundfunk. *Fragen zum Impfausweis: Impfpass verloren, was tun?* 7. Juli 2021. URL: https://www.br.de/nachrichten/wissen/fragen-zum-impfausweis-impfpass-verloren-was-tun,SN6dtx3 (besucht am 12. 07. 2021).

[29] General-Anzeiger. *Bonner berichten von fehlenden Nachweisen im Impfpass*. 1. Mai 2021. URL: https://ga.de/bonn/stadt-bonn/corona-bonner-berichten-von-fehlenden-nachweisen-im-impfpass_aid-57658177 (besucht am 12. 07. 2021).

[30] Dominik Schrahe und Thomas Städter. „COVID-19-Impf- und -Testnachweise". In: *Datenschutz und Datensicherheit - DuD* 45.5 (Apr. 2021), S. 315–319. DOI: 10.1007/s11623-021-1441-2.

[31] Robert Koch Institut. *Impfquoten*. 9. Juni 2021. URL: https://www.rki.de/DE/Content/Infekt/Impfen/Impfstatus/impfstatus_node.html;jsessionid=B98566646ADB814DF414FB38E3607748.internet121.

[32] Eberl, Jens. *Nicht ohne meinen Impfpass?* 11. Mai 2021. URL: https://www.tagesschau.de/inland/gesellschaft/impfnachweis-101.html (besucht am 11. 06. 2021).

[33] aerzteblatt.de. *Israels Grüner Pass bringt Erleichterungen für Coronageimpfte*. 22. Feb. 2021. URL: https://www.aerzteblatt.de/nachrichten/121373/Israels-Gruener-Pass-bringt-Erleichterungen-fuer-Coronageimpfte (besucht am 15. 06. 2021).

[34] Patricia Schlagenhauf et al. „Variants, vaccines and vaccination passports: Challenges and chances for travel medicine in 2021". In: *Travel Medicine and Infectious Disease* 40 (März 2021), S. 101996. DOI: 10.1016/j.tmaid.2021.101996.

[35] Piotr Heller. *Freie Fahrt für digitale Bürger*. 30. Mai 2021. URL: https://www.faz.net/aktuell/wissen/computer-mathematik/digitaler-impfausweis-freie-fahrt-fuer-geimpfte-buerger-17362678.html.

[36] Europäische Kommission. *Digitales COVID-Zertifikat der EU*. URL: https://ec.europa.eu/info/live-work-travel-eu/coronavirus-response/safe-covid-19-vaccines-europeans/eu-digital-covid-certificate_de#zeitleiste (besucht am 15. 06. 2021).

[37] Europäische Kommission. *Vorschlag für eine VERORDNUNG DES EUROPÄISCHEN PARLAMENTS UND DES RATES über einen Rahmen für die Ausstellung, Überprüfung und Anerkennung interoperabler Zertifikate zur Bescheinigung von Impfungen, Tests und der Genesung mit der Zielsetzung der Erleichterung der Freizügigkeit während der COVID-19-Pandemie (digitales grünes Zertifikat)*. 17. März 2021. URL: https://eur-lex.europa.eu/resource.html?uri=cellar:38de66f4-8807-11eb-ac4c-01aa75ed71a1.0023.02/DOC_1&format=PDF (besucht am 15. 06. 2021).

[38] Europäische Kommission. *eHealth and COVID-19*. URL: https://ec.europa.eu/health/ehealth/covid-19_en (besucht am 16. 06. 2021).

[39] Dominik Lauck. *So funktioniert der digitale Impfpass*. 14. Juni 2021. URL: https://www.tagesschau.de/inland/gesellschaft/digitaler-impfpass-107.html (besucht am 17. 06. 2021).

[40] eHealth Network. *EU DCC Validation Rules*. Version 1.00. 9. Juni 2021. URL: https://ec.europa.eu/health/sites/default/files/ehealth/docs/eu-dcc_validation-rules_en.pdf (besucht am 17. 06. 2021).

[41] eHealth Network. *Guidelines on COVID-19 citizen recovery interoperable certificates - minimum dataset*. Version 1. 15. März 2021. URL: https://ec.europa.eu/health/sites/default/files/ehealth/docs/citizen_recovery-interoperable-certificates_en.pdf (besucht am 19. 06. 2021).

[42] eHealth Network. *Guidelines on Technical Specifications for Digital Green Certificates Volume 1*. Version 1.0.5. 21. Apr. 2021. URL: https://ec.europa.eu/health/sites/default/files/ehealth/docs/digital-green-certificates_v1_en.pdf (besucht am 17. 06. 2021).

[43] eHealth Network. *Guidelines on Technical Specifications for Digital Green Certificates Volume 2 - European Digital Green Certificate Gateway*. Version 1.3. 21. Apr. 2021. URL: https://ec.europa.eu/health/sites/default/files/ehealth/docs/digital-green-certificates_v2_en.pdf (besucht am 17. 06. 2021).

[44] eHealth Network. *Guidelines on Technical Specifications for Digital Green Certificates Volume 3 - Interoperable 2D Code*. Version 1.3. 21. Apr. 2021. URL: https://ec.europa.eu/health/sites/default/files/ehealth/docs/digital-green-certificates_v3_en.pdf (besucht am 17. 06. 2021).

[45] eHealth Network. *Guidelines on Technical Specifications for Digital Green Certificates Volume 4 - European Digital Green Certificate Applications*. Version 1.3. 21. Apr. 2021. URL: https://ec.europa.eu/health/sites/default/files/ehealth/docs/digital-green-certificates_v4_en.pdf (besucht am 17. 06. 2021).

[46] eHealth Network. *Guidelines on Technical Specifications for Digital Green Certificates Volume 5 - Public Key Certificate Governance*. Version 1.02. 21. Apr. 2021. URL: https://ec.europa.eu/health/sites/default/files/ehealth/docs/digital-green-certificates_v5_en.pdf (besucht am 17. 06. 2021).

[47] eHealth Network. *Guidelines on Technical Specifications for EU Digital COVID Certificates - JSON Schema Specification*. 9. Juni 2021. URL: https://ec.europa.eu/health/sites/default/files/ehealth/docs/covid-certificate_json_specification_en.pdf (besucht am 19. 06. 2021).

[48] eHealth Network. *Guidelines on Value Sets for Digital Green Certificates*. Version 1.1. 12. Mai 2021. URL: https://ec.europa.eu/health/sites/default/files/ehealth/docs/digital-green-value-sets_en.pdf (besucht am 19. 06. 2021).

[49] eHealth Network. *Guidelines on verifiable vaccination certificates - basic interoperability elements*. Version 2. 12. März 2021. URL: https://ec.europa.eu/health/sites/default/files/ehealth/docs/vaccination-proof_interoperability-guidelines_en.pdf (besucht am 19. 06. 2021).

[50] eHealth Network. *OUTLINE Interoperability of health certificates Trust framework*. Version 1. 12. März 2021. URL: https://ec.europa.eu/health/sites/default/files/ehealth/docs/trust-framework_interoperability_certificates_en.pdf (besucht am 19. 06. 2021).

[51] Europäische Kommission: Generaldirektion für Gesundheit und Lebensmittelsicherheit. *EU health preparedness: A common list of COVID-19 rapid antigen tests, including those of which their test results are mutually recognised, and a common standardised set of data to be included in COVID-19 test result certificates*. 16. Juni 2021. URL: https://ec.europa.eu/health/sites/default/files/preparedness_response/docs/covid-19_rat_common-list_en.pdf (besucht am 19. 06. 2021).

[52] Bundesamt für Sicherheit in der Informationstechnik. *Country Signing Certificate Authority*. URL: https://www.bsi.bund.de/DE/Themen/Oeffentliche-Verwaltung/Elektronische-Identitaeten/Public-Key-Infrastrukturen/CSCA/csca.html (besucht am 11. 07. 2021).

[53] Carsten Bormann. *Concise Binary Object Representation (CBOR)*. Dez. 2020. URL: https://datatracker.ietf.org/doc/html/rfc8949 (besucht am 11. 07. 2021).

[54] Charlotte Couvé. *Zentrale oder dezentrale Datenspeicherung – die richtige Strategie finden*. 1. Okt. 2020. URL: https://www.incloud.de/magazin/zentrale-dezentrale-datenspeicherung (besucht am 05. 06. 2021).

[55] Ann-Christin Vahl. *Speicherung von Gesundheitsdaten – zentral oder dezentral?* 1. Apr. 2020. URL: https://journal.lifetime.eu/de/speicherung-von-gesundheitsdaten-zentral-oder-dezentral/ (besucht am 05. 06. 2021).

[56] Python. *About Python*. Feb. 2014. URL: https://www.python.org/about/ (besucht am 25. 07. 2021).

[57] Linux. *Linux.org*. Juli 2021. URL: https://www.linux.org/ (besucht am 25. 07. 2021).

[58] Microsoft. *Windows*. Mai 2021. URL: https://www.microsoft.com/de-de/windows (besucht am 25. 07. 2021).

[59] Apple. *macOS Big Sur*. Juli 2020. URL: https://www.apple.com/de/macos/big-sur/ (besucht am 25. 07. 2021).

[60] Mark Lutz. *Python kurz & gut: Für Python 3. x und 2.7*. O'Reilly Media, 2014, S. 1.

[61] BigData Insider. *Definition: Was ist Flask?* Jan. 2021. URL: https://www.bigdata-insider.de/was-ist-flask-a-994166/ (besucht am 25. 07. 2021).

[62] Pallets Projects. *Flask - web development, one drop at a time*. Juli 2019. URL: https://flask.palletsprojects.com/en/2.0.x/ (besucht am 25. 07. 2021).

[63] Code Study Block. *Flask - WTForms*. Dez. 2019. URL: https://www.codestudyblog.com/cnb11/1124193640.html (besucht am 25. 07. 2021).

[64] Pallets Projects. *Form Validation with WTForms*. Juli 2019. URL: https://flask.palletsprojects.com/en/2.0.x/patterns/wtforms/ (besucht am 25. 07. 2021).

[65] auth0. *SQLAlchemy ORM Tutorial for Python Developers*. Nov. 2017. URL: https://auth0.com/blog/sqlalchemy-orm-tutorial-for-python-developers/ (besucht am 25. 07. 2021).

[66] Ubuntuusers. *SQLAlchemy*. Feb. 2011. URL: https://wiki.ubuntuusers.de/SQLAlchemy/ (besucht am 25. 07. 2021).

[67] SQLAlchemy. *Key Features of SQLAlchemy*. Aug. 2011. URL: https://www.sqlalchemy.org/features.html (besucht am 25. 07. 2021).

[68] IONOS Digitalguide. *PostgreSQL: Das objektrelationale Datenbank-Managementsystem unter der Lupe*. Sep. 2018. URL: https://www.ionos.de/digitalguide/server/knowhow/postgresql/ (besucht am 25. 07. 2021).

[69] Storage Insider. *Was ist PostgreSQL (Postgres)?* Jan. 2021. URL: https://www.storage-insider.de/was-ist-postgresql-postgres-a-981832/ (besucht am 25. 07. 2021).

[70] PostgreSQL. *Was ist PostgreSQL?* Aug. 2003. URL: http://postgresql.de/was-ist-postgresql (besucht am 25. 07. 2021).

[71] John Irwin et al. „Aspect-oriented programming of sparse matrix code“. In: *nternational Conference on Computing in Object-Oriented Parallel Environments* 1343 (2005), S. 249–256. DOI: https://doi.org/10.1007/3-540-63827-X_68.

[72] Iris Uitz und Michael Harnisch. „Der QR-Code. Aktuelle Entwicklungen und Anwendungsbereiche“. In: *Informatik Spektrum* 35 (2012), S. 339–347. DOI: https://doi.org/10.1007/s00287-012-0608-5.

[73] Martin Lehmann. *QR-Codes*. Techn. Ber. Technische Universität Dresden, o.J.

[74] culture4life GmbH. *Gemeinsam das Leben erleben*. URL: https://www.luca-app.de/ (besucht am 30. 07. 2021).

[75] Dominik Schrahe und Thomas Städter. „COVID-19-Impf-und-Testnachweise“. In: *Datenschutz und Datensicherheit-DuD* 45.5 (2021), S. 315–319.

[76] Tagesschau. *So funktioniert der digitale Impfpass*. Juli 2021. URL: https://www.tagesschau.de/inland/gesellschaft/digitaler-impfpass-107.html (besucht am 15. 07. 2021).

[77] Bundesministerium für Gesundheit. *Fragen und Antworten zum digitalen Impfnachweis*. Juli 2021. URL: https://www.bundesgesundheitsministerium.de/coronavirus/faq-covid-19-impfung/faq-digitaler-impfnachweis.html (besucht am 15. 07. 2021).

[78] AOK Plus. *Zahlen und Fakten zur AOK PLUS*. Nov. 2020. URL: https://www.aok.de/pk/plus/inhalt/zahlen-und-fakten-zur-aok-plus/ (besucht am 15. 07. 2021).

[79] AOK Plus. *eImpfpass - Gemeinsam Geschichte schreiben*. Jan. 2021. URL: https://www.aok.de/gp/fileadmin/user_upload/Arzt_Praxis/Aerzte_Psychotherapeuten/Vertraege_Vereinbarungen/Modellvorhaben/Plus/sac_flyer_e_impfpass.pdf (besucht am 14. 06. 2021).

[80] KVS Sachsen. *Ein Jahr eImpfpass der AOK PLUS – ab 2021 Vereinfachungen in der Software*. Dez. 2020. URL: https://www.kvs-sachsen.de/fileadmin/data/kvs/img/Mitglieder/KVS-Mitteilungen/2020-12/kvsm2020-12_Schutzimpfungen.pdf (besucht am 14. 06. 2021).

[81] Digitale Stadt Düsseldorf. *e-Estonia: Estland führt EU-weit ersten Impfpass ein*. März 2021. URL: http://www.digitalestadtduesseldorf.de/e-estonia-estland-fuehrt-eu-weit-ersten-impfpass-ein/ (besucht am 15. 07. 2021).

[82] MDR. *Estland führt ersten digitalen Impfpass Europas ein*. März 2021. URL: https://www.mdr.de/nachrichten/welt/osteuropa/politik/estland-impfpass-digital-100.html (besucht am 15. 07. 2021).

[83] Gesundheit.GV.AT - Öffentliches Gesundheitsportal Österreichs. *Elektronischer Impfpass (e-Impfpass)*. Mai 2021. URL: https://www.gesundheit.gv.at/elga/was-ist-elga/elektronischer-impfpass (besucht am 15. 07. 2021).

[84] ELGA.GV.AT. *Informationen zum e-Impfpass für Bürgerinnen und Bürger*. Jan. 2021. URL: https://www.elga.gv.at/e-impfpass/e-impfpass/ (besucht am 15. 07. 2021).

[85] Gesundheit.GV.AT - Öffentliches Gesundheitsportal Österreichs. *Die Elektronische Gesundheitsakte (ELGA)*. Mai 2021. URL: https://www.gesundheit.gv.at/elga/inhalt (besucht am 15. 07. 2021).

[86] World Health Organization (WHO). *My COVID Pass*. 2021. URL: https://innov.afro.who.int/emerging-technological-innovations/my-covid-pass-3287 (besucht am 11. 06. 2021).

[87] Africa CDC. *Trusted Travel*. 2021. URL: https://africacdc.org/trusted-travel/ (besucht am 11. 06. 2021).

[88] Luke Daniel. *African airlines are testing their own Covid-19 passport*. 18. März 2021. URL: https://www.businessinsider.co.za/african-airlines-are-testing-their-own-covid-19-passport-2021-3 (besucht am 11. 06. 2021).

[89] Birgit Eger. *China stellt digitalen Reise-Impfpass vor*. 9. März 2021. URL: https://www.tagesschau.de/ausland/asien/corona-china-impfausweis-101.html (besucht am 11. 06. 2021).

[90] Roxanne Liu and Tony Munroe. *China launches COVID-19 vaccination certificates for cross-border travel*. 9. März 2021. URL: https://www.reuters.com/article/us-health-coronavirus-china-certificates-idUSKBN2B10LO (besucht am 11. 06. 2021).

[91] IATA. *IATA Travel Pass for Travelers*. 2021. URL: https://www.iata.org/en/youandiata/travelers/iata-travel-pass-for-travelers/ (besucht am 11. 06. 2021).

[92] Aerzteblatt. *App in Großbritannien soll Coronaimpfstatus nachweisen*. Mai 2021. URL: https://www.aerzteblatt.de/nachrichten/123878/App-in-Grossbritannien-soll-Coronaimpfstatus-nachweisen (besucht am 15. 07. 2021).

[93] Heise Online. *Ist Englands neuer Impfpass ein Vorbild für andere Länder?* Mai 2021. URL: https://www.heise.de/ratgeber/Ist-Englands-neuer-Impfpass-ein-Vorbild-fuer-andere-Laender-6052359.html (besucht am 15. 07. 2021).

[94] aerzteblatt.de. *Israels Grüner Pass bringt Erleichterungen für Coronageimpfte*. 22. Feb. 2021. URL: https://www.aerzteblatt.de/nachrichten/121373/Israels-Gruener-Pass-bringt-Erleichterungen-fuer-Coronageimpfte (besucht am 11. 06. 2021).

[95] Pierre Heumann. *Rückkehr in die Normalität: Wie der Grüne Pass in Israel funktioniert*. 4. März 2021. URL: https://www.handelsblatt.com/politik/international/corona-impfung-rueckkehr-in-die-normalitaet-wie-der-gruene-pass-in-israel-funktioniert/26960028.html?ticket=ST-14151378-1sfJoftXpKCjZYSeFUxd-ap5 (besucht am 11. 06. 2021).

[96] Martin Glinz. „Requirements Engineering I". In: *Nicht funktionale Anforderungen. Universität Zürich, Institut für Informatik, Zürich* (2006).

[97] Ostermann, Klaus. *Software Engineering Anforderungsanalyse*. URL: http://ps.informatik.uni-tuebingen.de/teaching/ss15/se/2_Anforderungsanalyse.pdf (besucht am 11. 07. 2021).

[98] Schramm, Prof. Dr. Wolfgang. *ANFORDERUNGSANALYSE UND -SPEZIFIKATION*. URL: https://services.informatik.hs-mannheim.de/~schramm/see/files/Kapitel03_01.pdf (besucht am 11. 07. 2021).

[99] JuraForum.de. *Soll-Vorschrift - Definition, juristische Bedeutung & Ermessen bei Rechtsnorm*. URL: https://www.juraforum.de/lexikon/soll-vorschrift (besucht am 11. 07. 2021).

[100] Apple Inc. *iOS 14. Völlig neu. Und ganz vertraut.* URL: https://www.apple.com/de/ios/ios-14/ (besucht am 11. 07. 2021).

[101] Google Ireland Limited. *Jetzt neu: Android 11*. URL: https://www.android.com/intl/de_de/ (besucht am 11. 07. 2021).

[102] Apple Inc. *Safari Unglaublich schnell. Extrem sicher.* URL: https://www.apple.com/de/safari/ (besucht am 11. 07. 2021).

[103] Google LLC. *Mehr Möglichkeiten mit dem neuen Chrome*. URL: https://www.google.de/chrome/?brand=FHFK&gclsrc=aw.ds&gclid=Cj0KCQjwiqWHBhD2ARIsAPCDzakowS6zDTxLKFhmRaFx5JVHAJWeqbkeqD_1XwhIF6BksfeWYfjB1hUaAktvEALw_wcB (besucht am 11. 07. 2021).

[104] Mozilla Corporation. *Aus Liebe zum Web*. URL: https://www.mozilla.org/de/ (besucht am 11. 07. 2021).

[105] Microsoft Corporation. *Microsoft Edge ist der beste Browser zum Einkaufen*. URL: https://www.microsoft.com/de-de/edge (besucht am 11. 07. 2021).

[106] Philippe B Kruchten. „The 4+ 1 view model of architecture“. In: *IEEE software* 12.6 (1995), S. 42–50.

[107] Jochen Seemann und Jürgen Wolff von Gudenberg. „Das Use-Case-Diagramm“. In: *Software-Entwurf mit UML*. Springer, 2000, S. 15–25.

[108] Ian Sommerville. *Software Engineering*. Pearson Education Deutschland GmbH, 1982, S.226 f. ISBN: 9783863268350. URL: https://ebookcentral.proquest.com/lib/dhbw-mannheim/detail.action?docID=5764007#.

[109] Ian Sommerville. *Software Engineering*. Pearson Studium, 1. Okt. 2018, S. 173–177. 896 S. ISBN: 3868943447. URL: https://www.ebook.de/de/product/33792940/ian_sommerville_software_engineering.html.

[110] Lucidchart. *UML Sequenzdiagramm*. 2021. URL: https://www.lucidchart.com/pages/de/uml-sequenzdiagramme (besucht am 03. 07. 2021).

[111] Microsoft Corporation. *Code editing. Redefined.* URL: https://code.visualstudio.com/ (besucht am 27. 07. 2021).

[112] Twitter Inc. *Build fast, responsive sites with Bootstrap*. URL: https://getbootstrap.com/ (besucht am 27. 07. 2021).