Fachhochschule Aachen
Fachbereich Medizintechnik und Technomathematik
Angewandte Mathematik und Informatik B.Sc.

Anforderungsanalyse der iOS-Applikation ASGARD Emergency Manual mit Schwerpunkt auf die Bereitstellung von Daten

eingereicht von: Jana Dill

Martrikelnummer: 3276571

Abgabe: 30.11.2022

1.Prüfer: Prof. Dr. Philipp Rhode

2.Prüfer: M. Eng. Iuri Olari

Inhaltsverzeichnis

1 Einleitung **1**

2 Erwartungen an die Anwendung **2**
2.1 Funktionale Anforderungen 2
2.2 Nicht-Funktionale Anforderungen 3

3 Netzwerkanbindung und Sicherheit **4**
3.1 Lokaler Angriff 4
3.2 Netzwerkangriff 4
3.3 Mobile Applikationen 5

4 Datenbereitstellung **7**
4.1 Datenbank 7
4.1.1 Relationale Datenbank mit Webschnittstelle 8
4.1.2 Core Data 10
4.1.3 SQLite 11
4.1.4 Firebase 12
4.2 Konfigurationsdateien in Auszeichnungssprache 13
4.2.1 iCloud 13
4.2.2 Konfigurationsdatei mit Webserver 15
4.3 Vergleich 17

5 Fazit und Ausblick **19**

1 Einleitung

Das ASGARD Emergency Manual (AEM) soll eine Begleit-Software für das Hauptprodukt ASGARD der Firma Elara Leitstellentechnik GmbH darstellen.
In diesem Zusammenhang ist es notwendig, die Anwendung der Applikation ASGARD in Grundzügen zu verstehen. ASGARD ist ein Kommunikationssystem für Leitstellen der öffentlichen Sicherheit und Industrie und der Behörden und Organisation mit Sicherheitsaufgaben (BOS). Es dient zur zentralen Vermittlung von Notrufen, Funk und der interaktiven Nutzung von Leistellensystemen. Da ASGARD eine zentrale Rolle in dem kritischen Teil der Infrastruktur der öffentlichen Sicherheit ausmacht, bedarf es regelmäßiger Wartung und gegebenenfalls schneller Behebung von auftretenden Fehlerzuständen. Die Behebung von Problemen kann schneller stattfinden, wenn diese ohne Remote-Zugriff eines Supportmitarbeiters durchgeführt werden können. Das bedeutet, dass im Regelfall eine IT-Fachperson der Leitstelle diese Fehlerbehebungen durchführen können soll. Um diesen Vorgang zu vereinfachen, soll das ASGARD Emergency Manual Änleitungenßu verschiedenen Fehlerfällen nach vorheriger Analyse geben.
Es soll eine Liste der grundlegenden Fehler zur Auswahl geben, die durch ein Frage-Antwort-System weiter eingegrenzt werden können. Da verschiedene Leitstellen unterschiedlichste Konfiguration des Aufbaus haben, müssen auch die wiedergegebenen Daten auf die einzelnen Standorte angepasst sein. Daraus schließt sich, dass es anpassbare Konfigurationsdateien für jeden einzelnen Kunden geben muss. Die Applikation erstellt außerdem ein Anwendungsprotokoll, dass der Firma automatisiert zur Verfügung gestellt werden muss.
ASGARD Emergency Manual soll als iOS-App bereitgestellt werden. Zur Anwendung wird jede Leitstelle mindestens ein iPad erhalten. Durch die selbsterklärende Nutzeroberfläche kann ASGARD Emergency Manual intuitiv genutzt werden. Es ist daher nicht nötig, Schulungen für die Verwendung der Applikation für das IT-Personal der Leitstellen anzubieten.

2 Erwartungen an die Anwendung

Die Anforderungen einer Software sind die Grundlage jedes Softwareprojektes. Es ist maßgebend, mit den verschiedenen Stakeholdern in Rücksprache zu sein, um den Erwartungen aller Parteien gerecht zu werden. Dafür ist es notwendig, eine Analyse der erwarteten Software zu erstellen und einen Plan zu entwickeln, der den Aufbau der Applikation mit einbezieht. Dieser Vorgang wird Anforderungsanalyse genannt. Es handelt sich um einen der wichtigsten Schritte in der Softwareentwicklung, damit mehrfache Verbesserung bzw. wiederholtes Verwerfen von Arbeit umgangen werden kann.[1,2]
In einer Anforderungsanalyse werden sowohl die Anforderungen der Software selbst - was soll die Software können? - als auch die Anforderungen an die technischen Eigenschaften der Software gestellt. Diese Anforderungen sind als funktionale und nicht-funktionale Anforderungen bekannt.[3]
Die Applikation soll der eigenständigen Abhilfe von Problemen innerhalb einer Leitstelle dienen. Da eine Leitstelle kritischer Infrastruktur angehört, ist es wichtig, dass auftretende technische Defekte schnellstmöglich behoben werden können. Dies beinhaltet auch, dass die Daten in einem Katastrophenfall, wie zum Beispiel Überschwemmung und damit einhergehendem Internetausfall, verfügbar sein müssen. Es gibt dementsprechend Anforderungen, die komplexer umzusetzen sind und gleichzeitig auch einen höheren Stellenwert in der Umsetzung haben. Um diese Anforderungen entsprechend gewichten zu können, wurde sowohl mit Administratoren verschiedener Leitstellen, dem Leiter der Entwicklung und dem Firmenleiter der ELARA Leitstellentechnik GmbH Interviews geführt.

2.1 Funktionale Anforderungen

Die Applikation soll ein erweitertes Benutzerhandbuch darstellen, dass mit Hilfe eines Frage-Antwort-Prinzips ein Problem soweit einschränkt, dass genaue Anordnungen an den ausführenden Administrator gegeben werden können.
Die Anordnungen sollen in einer Liste dargestellt werden und gegebenenfalls durch weitere Fragen unterbrochen werden können. Dementsprechend baut die To-Do-Liste sich mit dem Abhaken potentiell weiter auf. Durch das Abarbeiten der

1. Fadhl Hujainah u. a., „Software Requirements Prioritisation: A Systematic Literature Review on Significance, Stakeholders, Techniques and Challenges“, *IEEE Access* 6 (2018): 71497–71523, ISSN: 2169-3536, besucht am 5. Oktober 2022, https://doi.org/10.1109/ACCESS.2018.2881755, https://ieeexplore.ieee.org/document/8539976/.

2. Senay Tuna Demirel und Resul Das, „Software requirement analysis: Research challenges and technical approaches“, in *2018 6th International Symposium on Digital Forensic and Security (ISDFS)* (Antalya: IEEE, März 2018), 1–6, ISBN: 978-1-5386-3449-3, besucht am 5. Oktober 2022, https://doi.org/10.1109/ISDFS.2018.8355322, https://ieeexplore.ieee.org/document/8355322/.

3. Hujainah u. a., „Software Requirements Prioritisation“.

vorgegebenen Schritte können so geringfügigere Fehler ohne Hilfe eines Support-Mitarbeiters von einem Administrator behoben werden. Daraus folgt, dass im besten Fall ganze Hotline-Anrufe vermieden werden können.
Das Erfüllen von einzelnen To-Dos sowie das Beantworten von Fragen muss protokolliert werden, damit ein Support-Mitarbeiter der Firma ELARA bei einem resultierendem Hotline-Anruf nach einem nicht erfolgreichen durchgeführten Vorgang nachvollziehen kann, welche Behebungsversuche bereits unternommen wurden. Das Anwendungsprotokoll soll auf einem sowohl vom IT-Administrator auf Kundenseite als auch von Support-Mitarbeitern auf ELARA-Firmenseite erreichbaren Webserver abgelegt werden.

2.2 Nicht-Funktionale Anforderungen

Die Nutzung der Applikation soll einfach sein und sich dem Kunden von selbst erschließen, um zeitaufwändige und kostspielige Lehrgänge zu vermeiden. Das Design soll dem Kunden helfen, auch in stressigen Situationen die Applikation ohne Probleme verwenden zu können.
Um der Sicherheit der Daten gerecht zu werden, muss eine Authentizifierung erfolgen. Dies könnte über einen Token oder einen Login geschehen.
Die Applikation soll nicht übermäßig lange laden, bis die Benutzeroberfläche nutzbar wird. Das gilt vor allem, wenn durch Änderungen an den Konfigurationen die Daten erneut Heruntergeladen werden müssen. Es sollen solange die alten Daten angezeigt werden, bis die neuen Daten heruntergeladen wurden und die Nutzeroberfläche aktualisiert wird. Im Idealfall werden die Daten nach dem ersten Anmelden heruntergeladen und danach im Hintergrund aktualisiert.
Die Applikation soll einen wichtigen Vorteil in kritischen Situationen bringen. Daher ist es von absoluter Wichtigkeit, dass die Daten nach dem initialen Aufsetzen der Applikation jederzeit erreichbar sind. Dies gilt auch für den Fall, dass kein Internet vorhanden ist. Die Daten müssen daher lokal auf dem Gerät gespeichert werden.
Das Hochladen des Nutzerprotokolls soll asynchron erfolgen. Sollte dies nicht erfolgreich durchgeführt werden können, ist es essentiell, dass die Daten nicht verworfen werden und zu einem späteren Zeitpunkt ein erneuter Upload-Versuch gestartet wird. Dieser Vorgang wiederholt sich, bis die Daten vollständig übertragen werden konnten.

3 Netzwerkanbindung und Sicherheit

Die Anbindung an ein Netzwerk ist unabdingbar, wenn ein Datenaustausch stattfinden soll. Ein Austausch kann sowohl über ein internes Netz, wie zum Beispiel ein Firmennetz, als auch dem Internet stattfinden. Da die Daten der einzelnen Leitstellen zentral geändert werden können und das erhalten eines Protokolls automatisiert sein soll, muss die Anwendung an das Internet angeschlossen sein. Ist dies nicht der Fall, muss ein Support-Mitarbeiter über eine Virtual-Machine auf das Netzwerk des Kunden zugreifen und so die Dateien aktualisieren.
Sobald eine Applikation an das Internet angeschlossen wird, muss viel mehr Wert auf die Sicherheit der Anwendung gelegt werden, da es nun einen neuen Angriffspunkt bietet. Es könnten die lokale Applikation angegriffen werden oder das Netzwerk, das dahinter liegt.
Die Datensicherheit und der Datenschutz sind der Hauptfokus des Produktes. Dies beinhaltet den Schutz davor, dass Daten abgefangen werden und den Schutz vor Verlust von Daten durch Backups.[4]

3.1 Lokaler Angriff

Es ist nicht zu vernachlässigen, dass es zu Komplikationen mit der lokalen Version der Konfigurationsdatei kommt. Das gilt vor allem, wenn Dateien lokal abgelegt werden müssen. In diesem Zusammenhang, ist es notwendig, dass die Daten so abgesichert sind, dass sie nicht durch Dritte verändert werden können.

3.2 Netzwerkangriff

Das Abschotten eines Netzwerkes von der Außenwelt ist ein kompliziertes Unterfangen. Es muss gewährleistet werden, dass ausschließlich die gewünschten Daten gesendet und empfangen werden. Hierfür muss eine Überprüfung von der Applikation sowie der Bereitstellungsschnittstelle durchgeführt werden.
Optimal ist es, wenn es von Nutzerseite aus keinen Weg geben würde, eine fehlerhafte oder gar böswillige Eingabe zu betätigen. Dies kann dadurch verhindert werden, dass der Nutzer keine Freitexteingaben durchführen kann. Die Eingaben der gesendeten Protokolldateien entstehen automatisch und können daher nicht von dem Nutzer verändert werden. Problematisch wird es bei dem Inhalt der Datei. Da die Datei potentiell abgelegt wird, wenn es beispielsweise kein Internet gibt und daher die Datei nicht versendet werden kann, könnte ein Nutzer versuchen, die Datei zu öffnen und zu verändern. Um diese Problematik zu umgehen, ist es wichtig, die ankommende Datei bei Empfang auf dem Endgerät auf ihre Richtigkeit

4. Claudia Eckert, *IT-Sicherheit: Konzepte - Verfahren - Protokolle* [auf ger], 8., aktualisierte und korr. Aufl (München: Oldenbourg, 2013), ISBN: 978-3-486-73587-1 978-3-486-72138-6.

zu überprüfen. Dies gilt auch für die Ablage von Dokumenten auf Serverseite. In diesem Fall dürfen also ausschließlich nicht ausführbare Dateien versendet werden. Jede andere Form von Datei wird verworfen, bevor sie auf dem Server oder dem Gerät gespeichert wird. Hierbei muss besonders auf Man-in-the-middle-Attacken geachtet werden, die den Inhalt der Dateien abfangen und verändern können.[5]
Das Verändern der Konfigurationsdateien kann zu weiteren Problemen führen. Es ist denkbar, dass es Personen gibt, die gewollt oder auch ungewollt, eine Konfigurationsdatei so verändern, dass die Applikation nicht mehr nutzbar ist. Da die Anwendung nur verwendet wird, wenn es zu einem technischen Defekt an dem Hauptprodukt ASGARD kommt, würde eine Manipulation an den Daten erst in einer Notfallsituation auffallen. Es ist also wichtig, die Konfigurationsdateien oder Datenbank bestmöglichst vom Nutzer und Angreifern zu beschützen.
Die Login-Credentials sind Daten, die verwendet werden, um sich gegenüber eines Servers als Nutzer zu identifizieren. Werden diese Daten gestohlen, kann es zu erheblichen Schäden für den Nutzer als auch für den Betreiber kommen. Es gibt verschiedene Angriffe, die durchgeführt werden können, um an Nutzerdaten zu gelangen.
Spyware, die verschiedene Daten aus einem Mobilen Gerät beziehen kann, können in diesem Fall gefährlich werden, da es sich um kritische Inhalte handelt. Daher ist es wichtig die verschiedenen Daten so wenig wie möglich zwischenzuspeichern und nur die notwendigsten Daten für die Anfragen zu versenden. Dies gilt für den Inhalt der Konfigurationsdateien als auch den Login-Credentials.[6]
Die bekannteste Art für Unbefugte an Daten zu gelangen, ist das Phishing. Phishing beschreibt den Vorgang an Daten zu gelangen, indem vorgegeben wird, eine seriöse Anwendung zu sein. Dies ist im Zusammenhang mit einer iOS-Applikation schwierig, da eine weitere Applikation heruntergeladen werden müsste, um sich als das Produkt auszugeben. Einzig das Zurücksetzen eines Passwortes könnte zu solchen Problemen führen. Dies wird in ASGARD Emergency Manual jedoch nicht direkt angeboten und muss mithilfe eines Support-Mitarbeiters geschehen.[7]

3.3 Mobile Applikationen

Mobile Applikationen können Probleme hervorrufen, die auf stationären Systemen nicht auftreten würden. Dieses Phänomen ist durch die veränderte Nutzung eines mobilen Gerätes durch Endnutzer zu beschreiben. Oftmals sind auf den Geräten

5. Eckert, *IT-Sicherheit*.

6. Daojing He, Sammy Chan und Mohsen Guizani, „Mobile application security: malware threats and defenses“, *IEEE Wireless Communications* 22, Nr. 1 (Februar 2015): 138–144, ISSN: 1536-1284, besucht am 27. November 2022, https://doi.org/10.1109/MWC.2015.7054729, http://ieeexplore.ieee.org/document/7054729/.

7. He, Chan und Guizani.

persönliche Daten zu finden, die durch einen Angriff gestohlen werden könnten. Die Sicherheit dieser Daten ist in vielen Fällen durch die Unwissenheit des Nutzers nicht gewährleistet. Es kommt zu fehlenden Updates des Betriebssystems oder dem Zwischenspeichern von Logindaten.[8,9]
Das Sichern von Daten kann so zu einem kritischen Unterfangen werden. Es muss geklärt werden, mit welchen Rechten eine Applikation auf den Speicher des Gerätes zu greifen darf. Um weitere Sicherheitslücken zu schließen, ist es empfehlenswert keine sensiblen Daten, wie beispielsweise Logindaten, zu speichern. Das gilt auch für Zugangsdaten zu Servern und Datenbanken. Es ist notwendig, diese Daten nicht in den Source-Code zu integrieren und die Übertragung über andere Wege zu sichern.[10]
Viele mobile Applikationen nutzen weiterhin HTTP anstelle des sicheren und neueren HTTPS. Dadurch fehlt es an Transport-Layer-Security. Die Daten werden also als unverschlüsselter Text versendet. Dementsprechend ist es notwendig, dass die Applikation HTTPS und gültige Zertifikate verwendet. Das Umgehen solcher Schutzmaßnahmen führt zu erheblichen Sicherheitsproblemen.[11]
Sicherheitsabfragen, die durch den Entwickler oder Nutzer eingestellt werden, können durch Unwissen zu unsicheren Umgang mit Daten führen. Malware könnte explizit diese Einstellungen verändern und somit eine Sicherheitslücke erzeugen.[12]
Es ist nicht undenkbar, dass ein Nutzer ein Endgerät aus einem sicheren Netz entfernt und in ein unsicheres öffentliches Netz bringt. Ist dies der Fall, können Angriffe durch den Internetanschluss stattfinden.

8. Eckert, *IT-Sicherheit*.
9. He, Chan und Guizani, „Mobile application security“.
10. Anurag Kumar Jain und Devendra Shanbhag, „Addressing Security and Privacy Risks in Mobile Applications“, *IT Professional* 14, Nr. 5 (September 2012): 28–33, ISSN: 1520-9202, besucht am 30. November 2022, https://doi.org/10.1109/MITP.2012.72, http://ieeexplore.ieee.org/document/6243128/.
11. Jain und Shanbhag.
12. Jain und Shanbhag.

4 Datenbereitstellung

Die Datenbereitstellung kann über viele verschiedene Wege stattfinden. Um eine optimale Lösung zu finden, müssen die Anforderungen an die Anwendung klar definiert werden.
Die Informationen innerhalb der Applikation sind von den jeweiligen Standorten abhängig. Das bedeutet, dass die jeweiligen Leitstellen unterschiedliche Fragen und Antworten erhalten, die von den Geräten und einhergehenden Systemen abhängig sind. Die Daten müssen jederzeit, also auch in Notsituationen, erreichbar sein. Daher ist eine Persistierung der Daten auf dem Gerät unabdingbar. So kann sichergestellt werden, dass es aufgrund von Applikationsabstürzen oder Gerätneustart zu dem Verlust der im Arbeitsspeicher abgelegten Daten kommt. Zur Übertragung und Sicherung der Daten können Cloud-Services, lokale Datenbanken, ein Server mit Datenbankanbindung oder Konfigurationsdateien verwendet werden.

4.1 Datenbank

Eine Datenbank beschreibt eine Ansammlung von Daten. Diese Daten müssen im logischen Zusammenhang stehen. In dem Anwendungsfall des ASGARD Emergency Manuals soll der Schwerpunkt auf Datenbank Management Systemen (DBMS) liegen. Ein solches System wird dafür verwendet, Datenbanken anzulegen und zu warten.[13]
Die wichtigsten Anforderungen an ein Datenbank Management System ist bekannt als ACID. Die Abkürzung steht für atomicity (Atomarität), consistency (Konsistenz), isolation (Isolation) und durability (Dauerhaftigkeit). Atomarität bedeutet, dass ausschließlich korrekte und ausführbare Anfragen in der richtigen Reihenfolge durchgeführt werden. Kann dies nicht sichergestellt werden, werden alle Änderungen aus dieser Anfrage verworfen und der vorherige Zustand wiederhergestellt. Konsistenz bedeutet, dass die Datenbank an sich konsistent ist. Das heißt, es darf durch Anfragen nicht zu einer instabilen Datenbank kommen, die zum Beispiel bei der selben Anfrage zwei unterschiedliche Ergebnisse liefert. Isolation versichert, dass einzelne Anfragen isoliert voneinander behandelt werden und sich nicht gegenseitig unterbrechen und so fehlerhafte Daten erstellen. Die Dauerhaftigkeit ist besonders essentiell. Sie besagt, dass die Daten innerhalb eines Datenbank Management Systems gespeichert werden müssen. Das gilt besonders bei Fehlerfällen und Systemausfällen. Wurde eine Transaktion durchgeführt muss

13. Ramez Elmasri und Sham Navathe, *Fundamentals of database systems*, Seventh edition, OCLC: ocn913842106 (Hoboken, NJ: Pearson, 2016), ISBN: 978-0-13-397077-7.

diese gespeichert werden.[14]
Eine relationale Datenbank ist dadurch gekennzeichnet, dass die Daten miteinander in Relation stehen. Die einzelnen Daten sind miteinander verknüpft. Das Erstellen von relationalen Datenbanken erfordert Planung und eine gute Übersicht. Die Datenbank besteht aus vielen verschiedenen Tabellen, die entweder Inhalt enthalten als Base Tables oder die als Virtual Tables einzig die Relation enthalten. Es existiert keine Sortierung innerhalb der Tabellen. Dies führt dazu, dass es trotz der starken Abhängigkeit der verschiedenen Tabellen keine Hierarchie gibt.[15]
Um eine relationale Datenbank zu erstellen, muss ein Schema erstellt werden. Dieses gibt an, wie die Datenbank aufgebaut ist. Es beschreibt die einzelne Tabelle und deren Inhalt. Für die einzelnen Spalten und Reihen der Tabellen gibt es Vorgaben, an die sich gehalten werden muss. Eine Reihe entspricht einem Datenobjekt. Daraus folgt, dass eine Spalte einem Attribut des Objektes entspricht. Ein Spaltenname in einer Tabelle muss immer einzigartig sein, um genau zugeordnet werden zu können. Sie können dagegen in einer anderen Tabelle wieder aufgegriffen werden. Die meisten Virtual Tables nutzen die Spaltennamen der Base Tables auf die sie sich beziehen. Eine Reihe in eine Base Table muss immer einen Primary-Key enthalten. Ein Primary-Key ist ein Attribut einer Tabelle, das eindeutig zu zu ordnen ist. Kein weiterer Primary-Key-Eintrag dieser Tabelle darf diesen Wert besitzen. In den meisten Fällen wird für den Primary-Key eine ID als Integer verwendet. Verwendet eine Tabelle Primary-Keys aus einer anderen Tabelle, handelt es sich um Foreign-Keys. Um an die Daten zu gelangen, müssen die Foreign-Keys mit den Primary-Keys aus der in Relation stehenden Tabelle miteinander abgeglichen werden. Die fehlenden Daten werden so zu einer Antwort zusammengefasst.[16]
Um Anfragen an das Datenbank Management System zu stellen, wird überwiegend Structured Query Language (SQL) verwendet. Es ist eine menschenlesbare Sprache. Um eine Anfrage zu stellen, muss die Tabelle angegeben werden. Alle anderen Einschränkungen sind optional und können durch verschiedene Befehle durchgeführt werden.[17]

4.1.1 Relationale Datenbank mit Webschnittstelle

Die Elara Leistellentechnik GmbH verwendet meist MariaDB als Database Management System.

14. *Datenbankmanagementsystem (DBMS) erklärt* [auf Deutsch], Blog, November 2022, https://www.ionos.de/digitalguide/hosting/hosting-technik/datenbankmanagementsystem-dbms-erklaert/.
15. Jan L. Harrington, *Relational database design and implementation* [auf eng], Fourth edition, OCLC: 947597075 (Amsterdam: Morgan Kaufmann/Elsevier, 2016), ISBN: 978-0-12-849902-3.
16. Harrington.
17. Akeel I. Din, *Structured Query Language (SQL): a practical introduction* (Oxford ; Cambridge, Mass: NCC Blackwell, 1994), ISBN: 978-1-85554-357-7.

Das Erstellen einer MariaDB ist ein einmaliger Aufwand, der von einem Systemadministrator durchgeführt werden muss. Danach kommt es nur zu einzelnen Wartungen an dem System, die nicht zeitaufwendig sein sollten. Das Planen der Schemata findet bestenfalls auch nur einmalig statt. Die Schemata können dann dauerhaft verwendet werden, solange es nicht zu Datenstrukturänderungen innerhalb der Applikation kommt.
Die Daten müssen innerhalb der Datenbank gepflegt werden. Eine relationale Datenbank hat den Vorteil, dass viele Einträge wiederverwendet werden könnten. Es kann eine Tabelle mit den grundlegenden Daten erzeugt werden und für jede Leitstelle innerhalb einer weiteren Tabelle die Verknüpfungen hergestellt werden. So können individuelle Konfigurationen erstellt werden, ohne viele Wiederholungen zu erzeugen. Durch die Menge der Einträge kann es jedoch schwierig werden, den Überblick zu behalten. Es kann zu Problemen bei der Erstellung der einzelnen Kundentabellen kommen.
Da ein Datenbank Management System kritische Daten enthält, die nicht an die nicht an die Öffentlichkeit gelangen sollen, muss die Datenbank geschützt werden. Es ist daher wichtig, dass die Datenbank für den Nutzer nicht direkt erreichbar ist. Um das Datenbank Management System an die Anwendung anzubinden, wird also ein Webserver erstellt, der als Schnittstelle zwischen der Anwendung und der Datenbank dient. Es ist nicht zu vernachlässigen, dass der Server vollständig selber geschrieben wird. Das bedeutet, dass alle Sicherheitsaspekte selber abgedeckt werden müssen. Dies beinhaltet Zertifikate für den Server und den Schutz gegen verschiedenste Cyberangriffe. Es ist außerdem notwendig ein Authentifizierungsverfahren zu verwenden, um sicherzugehen, dass nur befugte Personen Zugriff zu den Daten haben. Denkbar wäre Lightweight Directory Access Protocol (LDAP), da dies bei ELARA in anderen Bereichen schon verwendet wird.
Außerdem kann die Datenbank zum Beispiel durch SQL-Injections angegriffen werden. In diesen wird versucht, durch gezielte Anfragen mit bestimmten Schlagwörtern alle Daten aus einer Datenbank zu manipulieren. Es können beispielsweise die Daten gestohlen oder gelöscht werden. Dies kann durch den verantwortungsbewussten Umgang mit den Daten und der Verwendung von Prepared Statements umgangen werden.[18]
Dieser Lösungsansatz der Datenbereitstellung ist davon abhängig, dass der Webserver weiterhin gewartet wird. Der Server muss gehostet werden. Es darf nicht zum Absturz des Servers kommen, ohne das dies bemerkt und behoben wird. Zuletzt löst die Anbindung an eine Datenbank nicht das Problem der ständigen Verfügbarkeit der Daten. Sollte das Endgerät keine Internetverbindung haben,

18. Lwin Khin Shar und Hee Beng Kuan Tan, „Defeating SQL Injection“, *Computer* 46, Nr. 3 (März 2013): 69–77, ISSN: 0018-9162, besucht am 28. November 2022, https://doi.org/10.1109/MC.2012.283, http://ieeexplore.ieee.org/document/6265060/.

können keine Daten bezogen werden. Das bedeutet, dass diese Lösung mit einem weiteren Ansatz kombiniert werden muss, um den Anforderungen gerecht zu werden. Es ist möglich, die Daten aus der Datenbank in eine Datei zu laden und diese Datei als Konfigurationsdatei zu verwenden. Optional kann auf dem Gerät eine lokale Datenbank erstellt werden, die sich bestenfalls automatisch mit der externen Datenbank abgleicht.

4.1.2 Core Data

Core Data ist ein von Apple angebotenes Framework, dass eine Art lokale Datenbank innerhalb einer Applikation erzeugt. Die Daten können zur Persistierung entweder über eine Datenbank (s. 4.1.1) oder über eine Konfigurationsdatei (s. 4.2.2) zur Anwendung gelangen. Core Data dient danach ausschließlich zur Sicherung der Daten auf dem Endgerät.
Core Data ist ein abstraktes System um eine lokale Datenbank innerhalb einer Applikation zu erzeugen, ohne explizit eine Datenbank aufzusetzen. Hierfür verwendet Apple ein System, dass Key-Words Werten zuordnet. Um Core Data verwenden zu können, muss zu Beginn bei Erstellung des Projektes angegeben werden, dass Core Data verwendet werden soll. Diese Angabe ändert den Projektaufbau und muss daher vor der Implementierung erfolgen.[19]
Core Data verwendet das Prinzip des Datenmodels. Das bedeutet, dass die Struktur von Daten festgelegt wird und dann innerhalb der Anwendung während der Implementierung in dieser Form verwendet wird. Core Data speichert also die Daten als Objekte. Diese Objekte werden Entitäten genannt. Eine Entität enthält somit alle Informationen, die auch ein Datenobjekt in der Applikation enthält. Dementsprechend hat eine Entität verschiedenste Properties, die das Objekt ausmachen. Diese sind mit den Spalten einer Datenbank zu vergleichen. Das Model erlaubt es nun, Attribute dieser Properties vorzudefinieren. Es kann beispielsweise einen Default-Wert geben oder ein Minimum oder Maximum angegeben werden. Eine Entität kann Beziehungen zu anderen Entitäten haben. Nachstellbar sind 1-1-/1-n- oder n-m-Beziehung, die aus anderen Datenmodellen schon bekannt sind. Das Datenmodell kann innerhalb von Xcode auch graphisch als Entity-Relationship-Diagramm dargestellt werden. Grundsätzlich soll in Core Data auch immer eine Inverse Beziehung bestehen. Entitäten sollen also immer von beiden Seiten verknüpft werden, um die Integrität der Daten zu sichern. Aufgrund dieses Datenmodels ist es empfehlenswert ein Model-View-Controller Ansatz zu verfolgen. Das Model steht hier für das Datenmodell und den konkreten Daten. Der View steht für das User-Interface und der Controller verbindet den View mit dem Model. Die Core Data API verwendet schon vorgefertigte Klassen, die durch das Datenmodell

19. *Core Data* [auf Englisch], Dokumentation, November 2022, https://developer.apple.com/documentation/coredata.

entstehen und direkt anwendbar sind.[20]
Diese Klassen sind in einem Persistent-Container namens `NSPersistendContainer` anzufinden. Das `NSManagedObjectModel` beschreibt das Datenmodel. Über `NSManagedObjectContext` werden die Änderungen an den Objekten beobachtet. Dieser wird wiederum durch `NSPersistentStoreCoordinator` verändert, sobald dieser auf die Core Data zugreift.[21]
Core Data unterstützt außerdem das automatische Aktualisieren des Datenmodels, sollten an diesem Anpassungen vorgenommen werden. Es kann also jederzeit das Datenmodell geändert werden, ohne Probleme der Datensicherung zu erzeugen. Außerdem speichert das Framework Änderungen zwischen, sodass es kurzzeitig möglich ist, diese zu widerrufen. Ein weiterer Vorteil ist das automatisch integrierte Lazy-Loading. Die Daten werden also erst dann geladen, wenn sie angefragt werden. Dies spart Energie und Zeit und verbessert so die Perfomance der Anwendung.[22]
Zusammenfassend beeinflusst die Verwendung von Core Data den gesamten Implementationsablauf und ist daher sehr aufwendig und bedarf viel Recherche und angeignetes Wissen.

4.1.3 SQLite

SQLite ist eine lokale Datenbank innerhalb einer Anwendung. Es speichert die Daten also direkt auf dem Endgerät und verwendet keinen Server. Wie auch schon bei Core Data (s. 4.1.2) müssen die Daten also über eine Datenbank (s. 4.1.1) oder über eine Konfigurationsdatei (s. 4.2.2) zu der Anwendung gelangen und können dann erst mithilfe der Library auf dem Gerät gespeichert werden.
Es handelt sich um ein Open Source Projekt, dass durch die rege Nutzung auch voraussichtlich in einiger Zeit noch gewartet wird. SQLite benötigt weder eine Installation noch eine Konfiguration. Nach dem einbinden der Library kann es einfach verwendet werden. Dadurch kann viel Zeit und Arbeitskraft für das Aufsetzen und Warten des Systems gespart werden. Da SQLite reihenweise seine Daten ausliest und die Datenabschnitte so kleiner sind, ist es signifikant schneller als die selben Daten aus einer Datei einzulesen. Dies führt außerdem dazu, dass keine unnötigen Inhalte geladen werden, da es zu expliziten Anfragen an die Datenbank kommt. Zur Bedienung wird SQL verwendet. Die Library lässt sich also genau wie ein herkömmliches Datenbank Management System verwenden. Für das System gelten auch die ACID Regeln, die die geregelte Nutzung einer Datenbank

20. B. M. Harwani, *Core Data iOS Essentials* [auf eng], OCLC: 726972216 (Birmingham, UK: Packt, 2011), ISBN: 978-1-84969-095-9.
21. *Core Data*.
22. Harwani, *Core Data iOS Essentials*.

gewährleisten.[23]
SQLite kann somit zum Beispiel eine MariaDB lokal auf dem Endgerät exakt wieder darstellen.

4.1.4 Firebase

Firebase ist ein von Google angebotener Cloud-Service. Es nicht notwendig, in irgendeiner Art und Weise einen Server zu hosten, da dies von Google ausgeführt wird. Sämtliche Kosten sowie Aufwand für die Wartung und Erstellung eines Webservers fallen nicht an.
Um Firebase nutzen zu können, muss zuerst das Projekt bei Google registriert werden. Dies geschieht über einen Google-Developer Account. Dieser Account ist erst einmal kostenlos. Da Apple strikt mit dem Einbinden von externen SDKs umgeht, muss die SDK per Hand eingebunden werden. Ist dies geschehen, kann durch Firebase ein Großteil der allgemein bekannten Google-Funktionen verwendet werden. Darunter fallen die Authentifizierung über ein Google-Konto, Sofortnachrichten und die Datenbankanbindung. Das ganze Cloud-System unterstützt außerdem Verschlüsselung über SSL und deckt somit einen wichtigen Aspekt der Sicherheit ab.[24]
Wie bei einem Cloud-Service zu erwarten, kann Firebase Daten über mehrere Endgeräte hinweg synchronisieren. Dabei werden die ACID-Vorgaben eingehalten. Das bedeutet, die Datenbank ist persistent und verliert auch keine Daten, sollte die Applikation abstürzen.[25]
Firebase bietet außerdem einen Offline-Modus an. Dieser muss explizit konfiguriert werden. Ist diese Option eingestellt, wird eine lokale Kopie der Datenbank erstellt und verwendet, sollte es keinen Internetzugriff geben. Sobald die Applikation wieder Netzzugriff hat, aktualisieren sich die Daten. Dies passiert automatisch und muss nicht erneut angestoßen werden. Die Abfrage der Daten funktioniert immer gleich. Das heißt, es müssen bei der Implementierung keine Sonderfälle beachtet werden.[26]
Der Service ist nicht kostenlos. Da in diesem Anwendungsfall aber nur sehr kleine

23. *SQLite Documentation* [auf Englisch], Dokumentation, November 2022, https://www.sqlite.org/docs.html.

24. Wu-Jeng Li u. a., „JustIoT Internet of Things based on the Firebase real-time database“, in *2018 IEEE International Conference on Smart Manufacturing, Industrial & Logistics Engineering (SMILE)* (Hsinchu: IEEE, Februar 2018), 43–47, ISBN: 978-1-5386-3183-6, besucht am 28. November 2022, https://doi.org/10.1109/SMILE.2018.8353979, https://ieeexplore.ieee.org/document/8353979/.

25. Chunnu Khawas und Pritam Shah, „Application of firebase in android app development-a study“, *International Journal of Computer Applications* 179, Nr. 46 (2018): 49–53.

26. *Offline auf Daten zugreifen* [auf Deutsch], Dokumentation, November 2022, https://firebase.google.com/docs/firestore/manage-data/enable-offline.

Zugriffszahlen pro Datei erreicht werden, wären die Ausgaben für den Service nicht hoch. Die Kosten kommen durch die jeweiligen Aufrufe zustande und betragen wenige Cent pro Abfrage nach einem kostenlos nutzbaren Kontingent für einen Tag von 50.000 Einheiten.[27]

4.2 Konfigurationsdateien in Auszeichnungssprache

Eine simple Lösung, unterschiedliche Inhalte innerhalb einer Anwendung wiederzugeben, ist das Verwenden von Konfigurationsdateien. Diese Dateien bestehen aus den verschiedenen Textabschnitten des Inhalts. Die Dateien werden zur einfachen Erstellung in Auszeichnungssprachen geschrieben. Diese sind definiert als maschinenlesbare Sprachen. Es sind also Dateien, die sowohl vom Menschen als auch vom Computer gelesen werden können.
Es wurde vorausgesetzt, dass für etwaige Konfigurationsdateien als Auszeichnungssprache JavaScript Object Notation (JSON) verwendet wird.[28]
JSON ist besonders, da es sowohl als Auszeichnungssprache, als auch als Austauschformat im Internet verwendet wird. Der Aufbau der Daten findet daher nicht als einfache Texte, sondern anhand einer Objektnotation statt. Die einzelnen Objekte sind gekennzeichnet durch geschweifte Klammern und geben einzelne Paramater anhand von Attributen weiter. Da dieser Aufbau sehr spezifisch ist, ist es notwendig, dass während der Entwicklung bekannt ist, welche Attribute benötigt werden. Die Programmierung und der Aufbau der Konfigurationsdatei sind somit essentiell miteinander verbunden und können nur mit Absprache erstellt werden.[29]
Die Konfigurationsdatei muss an das Endgerät gelangen. Hierzu kann ein Cloud-Service oder ein Webserver verwendet werden.

4.2.1 iCloud

Da es sich bei ASGARD Emergency Manual um eine iOS-Applikation handelt, ist als Cloud-Service iCloud zu verwenden die naheliegendste Lösung. iCloud ist die von Apple entwickelte angebotene Cloud. Um ein Apple-Gerät verwenden zu können, muss ein Apple-Account erstellt werden, der wiederum automatisch einen iCloud Zugriff erhält. Für die Applikation wird ein Geschäftsaccount erstellt, da dieser keine persönlichen Daten enthalten soll, wird er von der Elara Leitstellentechnik GmbH verwaltet. Der Account wird alleinig für die Nutzung des ASGARD

27. *Firestore-Preise* [auf Deutsch], Blog, November 2022, https://cloud.google.com/firestore/pricing.
28. Lindsay Bassett, *Introduction to JavaScript object notation: a to-the-point guide to JSON* [auf eng], First edition, OCLC: 918989291 (Sebastopol, CA: O'Reilly Media, 2015), ISBN: 978-1-4919-2945-2.
29. Bassett.

Emergency Manuals verwendet. Daher können die Konfigurationsdateien einfach in der iCloud abgelegt werden. Dieser Vorgang setzt voraus, dass das Endgerät ausschließlich für diesen Zweck verwendet wird. Die Datenübertragung würde so vollständig durch den Apple Service stattfinden.
Damit iCloud an eine Applikation angebunden werden kann, muss CloudKit verwendet werden. Bei CloudKit handelt es sich um eine API von Apple, die die Verwaltung von Daten mit iCloud ermöglicht. CloudKit teilt die iCloud in Container, die für eine Applikation verwendet werden können. Diese Container sind ein logischer Bereich, also ein physikalisch nicht unterscheidbarer Bereich, in dem alle Daten aus dieser Anwendung zu finden sind. Jeder Container hat wiederum eine öffentliche Datenbank und beinhaltet die private Datenbank eines Nutzers und die geteilte Datenbank des Nutzers. Eine nicht öffentliche Datenbank innerhalb eines Containers hat Zonen, auf die in verschiedenen Weisen zugegriffen werden kann. Somit ist es möglich, selektiv auf die einzelnen Gruppen innerhalb einer Zone zuzugreifen.[30]
CloudKit bietet verschiedenen Entwicklungsumgebungen für die Entwicklung und Produktiv-Betrieb. Die Datenbankschemata, die in der Entwicklung erstellt werden, werden so in die Produktion übertragen. Apple erlaubt nach der Aktualisierung einer Datenbank ausschließlich das Hinzufügen von neuen Parametern, um die rückwirkende Kompatibilität mit der Applikation zu gewährleisten. Dieser Vorgang erlaubt es, die Applikation in jeder Version zur Verfügung zu stellen, führt aber innerhalb der Datenbank zu erheblichen Anhäufung von nicht mehr notwendigen Daten durch immer mehr auftretende Attribute, während die ungenutzten nicht entfernt werden.[31]
Durch die Nutzung von Rollen innerhalb des Frameworks, können in simpler Art Nutzerrollen verwaltet werden. Die Rollen geben den Nutzern verschiedene Rechte innerhalb der unterschiedlichen Zonen. Somit ist es möglich, dass ein Nutzer Schreibrechte in der einen und ausschließlich Leserechte in einer anderen Datenbank besitzt. Die Rolle World beinhaltet alle Nutzer, Authenticate umschließt alle authenthifizierten Nutzer und Creator sind alle Nutzer, die einen Eintrag in einer Datenbank erstellt haben und sich authentifizieren können.
Durch die Cloud-Anbindung können die Daten von jedem Endgerät genutzt werden, auf dem der Nutzer authentifiziert ist. Sollte das Gerät gewechselt werden müssen, kommt es zu keinen Problemen bei der Übertragung der Daten von einem Gerät auf das andere.[32]
Die Nutzung des CloudKit Services bietet sich für groß angelegte Projekte an, die auf die Teilung und Wiederverwendung von Daten angewiesen sind. Es wird ver-

30. *CloudKit* [auf Englisch], Dokumentation, November 2022, https://developer.apple.com/documentation/cloudkit/.
31. *CloudKit*.
32. *CloudKit*.

einfacht Daten miteinander auszutauschen und trotzdem einen privaten Anteil an Daten verwalten zu können. Durch die Nutzung des Frameworks liegt die Verantwortung der Verschlüsselung und der sicheren Übertragung vollkommen bei Apple. Die Firma hat keinen Einfluss auf diesen Aspekt des Datenschutzes.
Gleichzeitig sorgt Apple durch die rückführende Kompatibilität dafür, dass die Applikation in jeder Version genutzt werden kann. Das ist nicht immer wünschenswert und kann so dazu führen, dass ein Kunde einen veralteten Lösungsvorschlag durchführt. Außerdem sorgt es für Sicherungskopien der Daten innerhalb des Containers.
Zusammenfassend bietet CloudKit beziehungsweise iCloud ein solides Standbein zum Austausch von Daten zwischen mehreren Geräten und Nutzern und der Möglichkeit, schnelle und einfache Backups zu erstellen. Die Anbindung des Frameworks ist dagegen komplizierter und erfordert eine Person, die sich mit dem Thema gut auskennt. Das Nutzen der iCloud Container ist über eine Weboberfläche Apples möglich. Dazu muss der Apple-Developer-Account verwendet werden. Nicht jeder Entwickler besitzt die Rechte um diese Anpassungen zu tätigen. So wird die Einteilung des Workloads auf wenige Personen begrenzt.
Die Verwendung von iCloud ist bis zu 5 GB kostenlos, danach bringt es Kosten mit sich. Überschreitet die private iCloud also diesen Speicherplatz, muss der Kunde selber für die Verwendung von mehr Platz zahlen um die Applikation erwartungsgemäß verwenden zu können. Die iCloud wird also sowohl über den Developer-Account der Firma als auch über den privaten Apple-Account der Kunden bereitgestellt.[33]

4.2.2 Konfigurationsdatei mit Webserver

Die Übertragung einer Konfigurationsdatei kann mit einem Webserver getätigt werden. Hierfür muss ein Server aufgesetzt werden und dieser gewartet und verwaltet werden. Anhand von Absprachen zwischen der Entwicklerin und einem Sstemadministrator wurde festgelegt, dass ein Nextcloud Server verwendet werden soll.
Die Verwendung eines eigens aufgesetzten Webservers ermöglicht es, eine Schnittstelle zu erschaffen, die genau auf die Anwendung zugeschnitten ist. Der Server muss dazu in der Lage sein, die Konfigurationsdatei zu speichern und auf Anfrage eines Nutzers zu verschicken. Die Daten müssen hinter einer Authentifizierung gesichert werden. Das bedeutet, dass der Server Login Daten für die Nutzer hinterlegt haben muss. Nachdem der Nutzer sich gegenüber des Servers authentifiziert hat, wird ein Session Token vergeben, der für weitere Anfragen verwendet wird. Der Server muss dahingegen mit den passenden Zertifikaten ausgestattet sein, um

33. *iCloud-Pläne und -Preisgestaltung* [auf Deutsch], Support, November 2022, https://support.apple.com/de-de/HT201238.

sich gegenüber dem Nutzer zu authentifizieren.
Dies ist besonders in iOS-Applikationen notwendig, da Apple den Umgang mit nicht vertrauenswürdigen Internetseiten immer weiter eingrenzt. Hierfür hat Apple in ihren eigenen Guidelines die App Transport Security (ATS) eingeführt. Apple zwingt somit seine Nutzer, ausschließlich mit Transport Layer Security Protocol geschützte Seiten zu verwenden. Diese verhindert das Nutzen von Internetseiten, die nicht Apples Standards entsprechen. Ein Aufruf dieser Seiten ist daher nicht möglich. So soll die Sicherheit der Nutzer gewährleistet werden.[34]
Der Aufbau des Servers ist an dieser Stelle maßgebend. Jeder Kunde hat einen passwortgeschützden Ordner. Innerhalb dieses Ordners werden die Konfigurationsdateien abgelegt. Diese Dateien müssen bestimmte Schlagwörter enthalten, damit sie in der Applikation richtig zugeordnet werden können. Veraltete Dateien können entweder umbenannt oder überschrieben werden. Eventuell vorhandene Bilder werden in einem separaten Bilder-Ordner abgelegt. Die Bilder werden dann heruntergeladen sobald die Konfigurationsdatei ausgelesen wird. Hierfür muss jeweils eine neue Anfrage an den Server gestellt werden. Je nach Anzahl der Bilder kann dies einige Zeit in Anspruch nehmen. Daher ist es wichtig in der Applikation dafür zu sorgen, dass die Medien im Hintergrund heruntergeladen werden. Sollte es zu Fehlern bei der Beziehung der Bilder kommen, muss außerdem sichergestellt werden, dass ein erneuter Versuch des Herunterladens gestartet wird. Generell sollte die Applikation aber auch ohne die unterstützenden Medien funktionieren.
Das Anwendungsprotokoll soll nach jeder Maßnahme aktualisiert und auf den Server hochgeladen werden. Auf dem Server soll sich für jede durchgeführte Fehlerbehebungsmaßnahme eine Datei befinden, die nach dem Vorgang und dem Erstellungszeitpunkt benannt ist. Die Datei wird solange erweitert bis der Vorgang abgeschlossen oder abgebrochen wird. Dies soll automatisch in einem extra Ordner innerhalb des Kunden-Ordners geschehen. Dieser Vorgang ist nicht ohne weitere Vorbereitung möglich. Um Dateien auf dem Server abzulegen, muss ein Skript geschrieben werden, das bei Aufruf eine Datei auf dem Server ablegt. Auch dieser Vorgang kann nur durch vorherige Authentifizierung stattfinden. Das Skript muss außerdem sicherstellen, dass es sich bei der abzulegenden Datei nicht um Malware handelt. Als generelle Vorgabe wird von der ELARA Leistellentechnik GmbH Python verwendet.
Der Aufwand eines eigenen Webservers übersteigt die Nutzung einer Cloud um weiten. Der Server muss aufgesetzt und gewartet werden und benötigt dafür geschultes Personal. Außerdem müssen die Zertifikate aktualisiert werden. Dafür liegt die Verantwortung für die Datensicherheit und des Datenschutz alleinig auf der Seite des Anbieters. Dies sollte bei richtiger Anwendung dazu führen, dass die Si-

34. *Preventing Insecure Network Connections*, Artikle, November 2022, https://developer.apple.com/documentation/security/preventing_insecure_network_connections.

cherheit gewährleistet ist. Es hat niemand außer den gewollten Parteien Zugriff auf den Server. Das Erstellen eines Skripts für das Hochladen von Dateien ist eine einmalige Investierung von Zeit und ist danach nicht mehr einschränkend für den produktiv Betrieb. Vorausgesetzt, es steht genug Speicherplatz zur Verfügung, entstehen keine weiteren Kosten für den Server.

4.3 Vergleich

Die verschiedenen Anwendungen haben jeweils Vor- und Nachteile. Um eine sinnvolle Entscheidung für die Datenbereitstellung und Persistierung treffen zu können, ist es notwendig die Anforderungen und die einhergehenden Kosten mit der Nutzbarkeit aufzuwiegen.
Wenn über Datenbereitstellung für Anwendungen geredet wird, ist eine Datenbank eine der ersten Ideen. Das Prinzip wird seit Jahrzehnten verwendet und ist immer noch eine der wichtigsten Datenquellen. Eine Datenbank erleichtert das Wiederverwerten von schon genutzten Elementen zwischen verschiedenen Leitstellen. Wird das Datenbank Management System selbst aufgesetzt, so kann von einer hohen Sicherheit auch für kritische Daten ausgegangen werden. Der Schutz der Daten liegt somit alleine in der Hand der Firma. Andererseits bringt dieses Vorgehen Mehraufwand mit sich, da ein Mitarbeiter die Datenbank und Schemata erstellen und danach warten muss.
Durch die Verwendung eines Cloud-Services kann viel Verantwortung abgegeben werden. Dies ist vor allem förderlich, wenn es keinen Experten für Datenschutz und Datensicherheit in der Firma gibt. Die kostenpflichtigen Services sind in den meisten Fällen gut geschützt. Problematisch ist es ausschließlich, wenn es zu größeren Cyberangriffen auf diese Anbieter kommt. Der explizite Schutz der kritischen Daten kann nach außen hin nicht weiter gesteigert werden.
Dahingegen ermöglicht das Hosten eines eigenen Webservers soviel Investition in die Sicherheit, wie gewünscht wird. Dafür ist der gesamte Vorgang signifikant arbeitsintensiver. Es muss viel mehr Zeit in Implementierung, Installation und Wartung des Servers gesteckt werden. Außerdem ist es notwendig, dass sich mindestens ein Mitarbeiter mit adäquaten Sicherheitsvorkehrungen auskennt. Dieser Mitarbeiter muss bei Fehlern zeitnah Abhilfe schaffen, da die Applikation ansonsten nicht nutzbar ist.
Das Erstellen von Konfigurationsdateien per Hand ist eine extrem aufwendige Arbeit, die sehr viel Zeit in Anspruch nimmt. Es ist schwierig bis hin zu unmöglich Daten wieder zu verwenden ohne durch übermäßiges Kopieren und Einfügen Fehler zu begehen. Sollen so Dateien für alle Leitstellen erstellt werden, ist die Aufgabe sehr zeitaufwendig. Dahingegen ist das Erstellen von Konfigurationsdateien aus Datenbankeinträgen durch eine Automatisierung einfach. Umsetzbar ist dies durch Parser, die Objekte in Auszeichnungssprachen übertragen können. Ledeglich das

erneute Laden der lokal abgelegten Daten würde länger dauern, als das Beziehen von einer lokalen Datenbank.
Aus den erschlossenen Informationen, kann abgeleitet werden das die von Apple angebotenen Optionen iCloud und Core Data für große Applikationen mit vielen Nutzern und großen Datenmengen herrvoragend geeignet sind. Beide Möglichkeiten sind aber auch mit viel Recherche und einer aufwendigen Implementierung verbunden. Grundsätzlich ist die Anbindung einer Datenbank für den Leistungsaufwand und das benötigte Fachwissen eine Maßnahme, um einen sinnvollen Mittelweg zu erreichen. Viele Mitarbeiter haben Grundkentnisse einer Datenbankanbindung und konnten schon Erfahrung mit der Umsetzung und Nutzung machen. Die Daten könnten zur Speicherung als Konfigurationsdatei abgelegt werden oder mit Hilfe von SQLite wiederum als Datenbank lokal weiter verwendet werden. Diese Art der Persistierung der Daten scheint besonders sinnvoll, weil die Datenbanken sich einfach gegenseitig aktualisieren können, sobald Internet vorhanden ist.
Firebase kann ein starkes Mittel sein, um eine Datenquelle anzubinden. Hier ist die Dokumentation eher schwierig und lässt erahnen, dass die erstmalige Konfiguration aufwendig ist. Sobald diese abgeschlossen ist, sollte sich die Datenbank optimal verhalten. Ein Nachteil sind hier die nicht hundert prozentig vorhersehbaren Kosten. Außerdem ist die Auslagerung von sensiblen Daten, ohne dass diese gesondert gesichert werden können, ein Problem.

5 Fazit und Ausblick

Durch die komplexen Anforderungen an die Applikation, ist es schwierig, eine eindeutig sich abhebende Lösung zu bestimmen.
Da in einem nächsten Schritt eine Webschnittstelle erstellt werden soll, wäre es sinnvoll, eine Lösung zu wählen, die sowohl in einer iOS-Applikation gut einzubinden ist und sich gut in einer Webpage einbinden lässt. Hierzu bieten sich in erster Linie die Anbindungen über einen Server an. Dieser könnte so erweitert werden, dass er als Backend für die Webanwendung dient. Die Daten könnten somit unverändert weiterverwendet werden. Somit muss die selbe Arbeit nicht zweimal erledigt werden.
Ein weiterer zu bedenkender Punkt ist die Arbeitsaufteilung des Projektes. Da die Anwendung alleinig von der MATSE-Auszubildenden implementiert werden soll und ein Systemadministrator die Erstellung der Daten übernehmen soll, ist es sinnvoll, eine Lösung zu wählen, die weder zu komplex noch zu aufwendig ist. Eine zweckmäßige Lösung wäre somit zum Beispiel die Umsetzung über eine Datenbank und eine damit verbundene Kopie als SQLite Datenbank.
Schlussendlich wurde sich dazu entschieden, die Daten als Konfigurationsdateien über einen Nextcloud-Server zur Verfügung zu stellen. Diese sind in JSON verfasst. Da Webanwendungen als Objekte in den meisten Fällen JSON verwenden, können auch so die Daten einfach ausgelesen werden. Da die Daten sich für die verschiedenen Leitstellen zwar oft ähneln, aber selten gleich sind, kommt es auch nicht zu erheblichen Mehraufwand durch das Schreiben der verschiednen Dateien.

Literaturverzeichnis

Bassett, Lindsay. *Introduction to JavaScript object notation: a to-the-point guide to JSON* [auf eng]. First edition. OCLC: 918989291. Sebastopol, CA: O'Reilly Media, 2015. ISBN: 978-1-4919-2945-2.

Core Data [auf Englisch]. Dokumentation, November 2022. https://developer.apple.com/documentation/coredata.

Datenbankmanagementsystem (DBMS) erklärt [auf Deutsch]. Blog, November 2022. https://www.ionos.de/digitalguide/hosting/hosting-technik/datenbankmanagementsystem-dbms-erklaert/.

Demirel, Senay Tuna, und Resul Das. „Software requirement analysis: Research challenges and technical approaches". In *2018 6th International Symposium on Digital Forensic and Security (ISDFS)*, 1–6. Antalya: IEEE, März 2018. ISBN: 978-1-5386-3449-3, besucht am 5. Oktober 2022. https://doi.org/10.1109/ISDFS.2018.8355322. https://ieeexplore.ieee.org/document/8355322/.

CloudKit [auf Englisch]. Dokumentation, November 2022. https://developer.apple.com/documentation/cloudkit/.

Din, Akeel I. *Structured Query Language (SQL): a practical introduction*. Oxford ; Cambridge, Mass: NCC Blackwell, 1994. ISBN: 978-1-85554-357-7.

SQLite Documentation [auf Englisch]. Dokumentation, November 2022. https://www.sqlite.org/docs.html.

Eckert, Claudia. *IT-Sicherheit: Konzepte - Verfahren - Protokolle* [auf ger]. 8., aktualisierte und korr. Aufl. München: Oldenbourg, 2013. ISBN: 978-3-486-73587-1 978-3-486-72138-6.

Elmasri, Ramez, und Sham Navathe. *Fundamentals of database systems*. Seventh edition. OCLC: ocn913842106. Hoboken, NJ: Pearson, 2016. ISBN: 978-0-13-397077-7.

Firestore-Preise [auf Deutsch]. Blog, November 2022. https://cloud.google.com/firestore/pricing.

Harrington, Jan L. *Relational database design and implementation* [auf eng]. Fourth edition. OCLC: 947597075. Amsterdam: Morgan Kaufmann/Elsevier, 2016. ISBN: 978-0-12-849902-3.

Harwani, B. M. *Core Data iOS Essentials* [auf eng]. OCLC: 726972216. Birmingham, UK: Packt, 2011. ISBN: 978-1-84969-095-9.

He, Daojing, Sammy Chan und Mohsen Guizani. „Mobile application security: malware threats and defenses“. *IEEE Wireless Communications* 22, Nr. 1 (Februar 2015): 138–144. ISSN: 1536-1284, besucht am 27. November 2022. https://doi.org/10.1109/MWC.2015.7054729. http://ieeexplore.ieee.org/document/7054729/.

Hujainah, Fadhl, Rohani Binti Abu Bakar, Mansoor Abdullateef Abdulgabber und Kamal Z. Zamli. „Software Requirements Prioritisation: A Systematic Literature Review on Significance, Stakeholders, Techniques and Challenges“. *IEEE Access* 6 (2018): 71497–71523. ISSN: 2169-3536, besucht am 5. Oktober 2022. https://doi.org/10.1109/ACCESS.2018.2881755. https://ieeexplore.ieee.org/document/8539976/.

iCloud-Pläne und -Preisgestaltung [auf Deutsch]. Support, November 2022. https://support.apple.com/de-de/HT201238.

Jain, Anurag Kumar, und Devendra Shanbhag. „Addressing Security and Privacy Risks in Mobile Applications“. *IT Professional* 14, Nr. 5 (September 2012): 28–33. ISSN: 1520-9202, besucht am 30. November 2022. https://doi.org/10.1109/MITP.2012.72. http://ieeexplore.ieee.org/document/6243128/.

Khawas, Chunnu, und Pritam Shah. „Application of firebase in android app development-a study“. *International Journal of Computer Applications* 179, Nr. 46 (2018): 49–53.

Li, Wu-Jeng, Chiaming Yen, You-Sheng Lin, Shu-Chu Tung und ShihMiao Huang. „JustIoT Internet of Things based on the Firebase real-time database“. In *2018 IEEE International Conference on Smart Manufacturing, Industrial & Logistics Engineering (SMILE)*, 43–47. Hsinchu: IEEE, Februar 2018. ISBN: 978-1-5386-3183-6, besucht am 28. November 2022. https://doi.org/10.1109/SMILE.2018.8353979. https://ieeexplore.ieee.org/document/8353979/.

Offline auf Daten zugreifen [auf Deutsch]. Dokumentation, November 2022. https://firebase.google.com/docs/firestore/manage-data/enable-offline.

Preventing Insecure Network Connections. Artikle, November 2022. https://developer.apple.com/documentation/security/preventing_insecure_network_connections.

Shar, Lwin Khin, und Hee Beng Kuan Tan. „Defeating SQL Injection“. *Computer* 46, Nr. 3 (März 2013): 69–77. ISSN: 0018-9162, besucht am 28. November 2022. https://doi.org/10.1109/MC.2012.283. http://ieeexplore.ieee.org/document/6265060/.

Eidesstattliche Erklärung

Hiermit versichere ich, dass ich die Seminararbeit mit dem Thema

Anforderungsanalyse der iOS-Applikation ASGARD Emergency Manual mit Schwerpunkt auf die Bereitstellung von Daten

---

selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe, alle Ausführungen, die anderen Schriften wörtlich oder sinngemäß entnommen wurden, kenntlich gemacht sind und die Arbeit in gleicher oder ähnlicher Fassung noch nicht Bestandteil einer Studien- oder Prüfungsleistung war.

Ich verpflichte mich, ein Exemplar der Seminararbeit fünf Jahre aufzubewahren und auf Verlangen dem Prüfungsamt des Fachbereiches Medizintechnik und Technomathematik auszuhändigen.

Name: Jana Dill

Aachen, den 30.11.2022

Unterschrift der Studentin / des Studenten