# Schlussbericht

**flocker**

# Inhalt

# 1 Informationen

## 1.1 Dokumentinformationen

| | |
|---|---|
| **Autor** | Isenschmid Noah |
| **Datum** | 11.06.2024 |
| **Version** | 1.0 |
| **Projekt** | Flocker |
| **Verteiler** | Isenschmid Noah, Reymond David, Lalicata Paulo, Fleury Lucien, Steinlin Timo |

| Version | Datum | Bearbeiter | Arbeiten |
|---|---|---|---|
| V1.0 | 11.06.2024 | Isenschmid Noah | Entwurf und Konzipierung Dokument |
| V1.1 | 11.06.2024 | Fleury Lucien | Aufbereitung Texte |
| V1.2 | 11.06.2024 | Lalicata Paulo<br>Isenschmid Noah | Formulierung Texte |
| V1.3 | 11.06.2024 | Reymond David | Formulierung Schlussbericht |

# 2 Projektinitialisierungsantrag

## 2.1 Ausgangslage

Im Modul 159 besuchten wir den "Techday" an der TFB. Dort wurde uns das Modell "Design Thinking" vorgestellt. An diesem Tag haben wir in der Gruppe unsere Projektidee ausgearbeitet und vorgestellt - flocker. Eine skalierbare und auf Docker basierende Click-to-Run Webapplikation, welche es ermöglicht, einzelne virtuelle Maschinen anhand von div. Präferenzen und Templates (Betriebssystem, Dienste/Applikationen und Netzwerk) zu erstellen.
Vor einer Realisierung eines solchen Projekts mussten wir uns über die zum Teil komplexen Herausforderungen im Klaren sein, welche uns bevorstehen könnten. Diese bezogen sich auf die Integration verschiedener Technologien, die Sicherstellung der Skalierbarkeit und die Anpassung an Benutzerpräferenzen durch vielfältige Templates für Betriebssysteme, Dienste/Applikationen und Netzwerkeinstellungen.
Zusätzlich musste die Einhaltung von Datenschutzgesetzen gewährleistet werden, was die Berücksichtigung von Sicherheits- und Datenschutzanforderungen in der Architektur und Funktionalität der Applikation erforderlich machte. Erforderliche Teamfähigkeiten umfassten technisches Know-how, besonders in Bezug auf die Thematik Docker und Virtualisierungstechnologien, ein tiefes Verständnis für Datenschutzprinzipien, sowie effektive Kommunikation und Koordination, um eine sichere, benutzerfreundliche und rechtskonforme Lösung zu entwickeln.
Da wir alle schon mit Docker (spezifisch Portainer) gearbeitet haben, haben wir bereits die wichtigsten Grundkenntnisse. Auch in der Webseiten Entwicklung haben einige von uns bereits kleine Erfahrungen gesammelt. Diese werden wir in den nächsten Wochen mit Ehrgeiz und Lernfreude gerne vertiefen. Um auch die Integration, Kompatibilität und Virtualisierung dieses Projekts auf die Beine zu stellen, werden wir uns auch intensive mit Virtualisierungstechnologien beschäftigen.

## 2.2 Ausgangslage

In der Initialisierungsphase unseres Projekts möchten wir klar verstehen, welche Schnittstellen und Dienste wir brauchen, um unsere Pläne umzusetzen. Wir zielen darauf ab, genau zu wissen, wie diese Teile zusammenarbeiten, um unsere Ziele effizient zu erreichen. Gleichzeitig ist es uns wichtig, eine einfache und klare Projektmanagementstruktur aufzubauen. Diese Struktur soll festlegen, wer was macht und wie wir miteinander kommunizieren, um alles reibungslos zu organisieren. Ein weiterer wichtiger Punkt ist die Kommunikation mit unseren Kunden. Wir planen, regelmässige Updates und Feedbackschleifen einzuführen, um sicherzustellen, dass die Kundenbedürfnisse und -erwartungen kontinuierlich erfasst und in die Projektentwicklung integriert werden. Diese offene und transparente Kommunikationsweise soll das Vertrauen stärken und sicherstellen, dass unser Projekt den Kundenwünschen entspricht und erfolgreich umgesetzt wird. So stellen wir sicher, dass unser Projekt pünktlich und im Rahmen unseres Budgets erfolgreich ist. Wir hoffen, mit diesen Schritten eine starke Grundlage für unser Vorhaben zu legen, damit wir selbstsicher und mit klarem Plan vorankommen können.

## 2.3 Rahmenbedingungen, Allgemeines

### 2.3.1 Projektmethode

Als Projektmethode verwenden wir HERMES 5.1. Dies wurde vom Projektleiter bestimmt und vom Team einstimmig genehmigt.

HERMES 5.1 ist eine Weiterentwicklung der HERMES-Projektmanagementmethode, die speziell für IT-Projekte in der Schweizer Bundesverwaltung entwickelt wurde, aber auch in anderen Organisationen und Projekten Anwendung findet. Sie bietet einen strukturierten Rahmen für die Planung, Steuerung und Kontrolle von Projekten und legt besonderen Wert auf eine klare Definition von Rollen, Prozessen und Dokumenten.

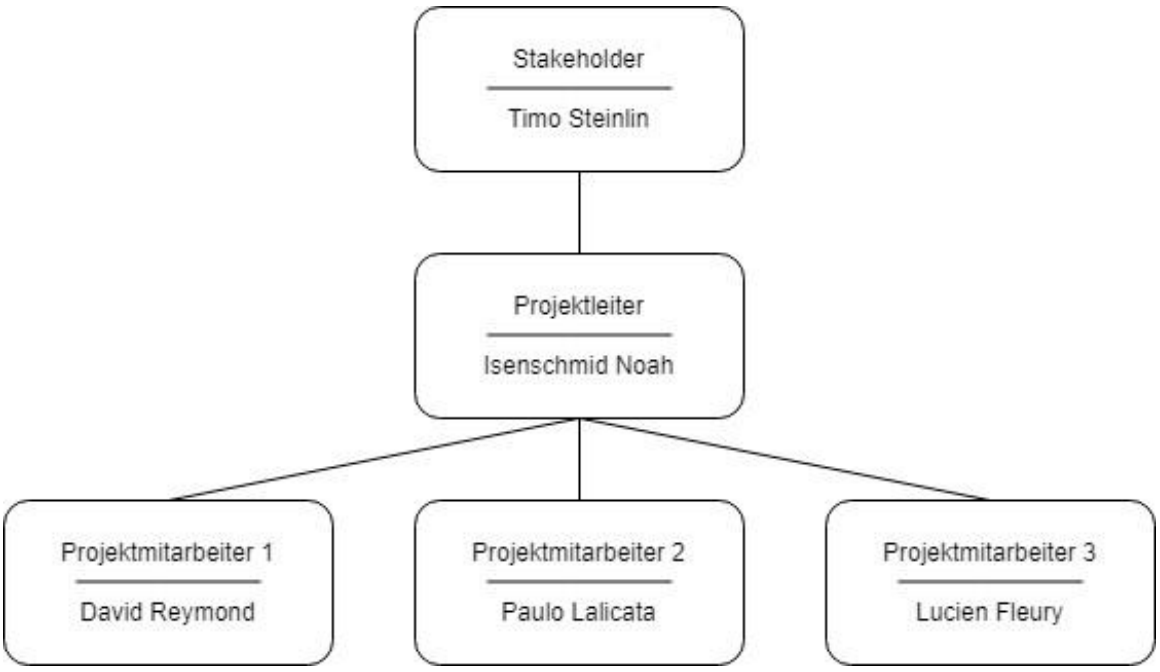
HERMES 5.1 erleichtert die effektive Durchführung von Projekten durch standardisierte Abläufe und Vorlagen, die auf die Bedürfnisse von IT-Projekten zugeschnitten sind. Die Methode umfasst Phasen von der Initialisierung über die Konzeption, Realisierung bis hin zum Abschluss des Projekts und beinhaltet Steuerungselemente wie Qualitäts-, Risiko- und Änderungsmanagement.

Vorteile:

- Strukturierte Vorgehensweise: Fördert die Transparenz und Nachvollziehbarkeit von Projektabläufen.
- Flexibilität: Lässt sich auf unterschiedliche Projektgrößen und -typen anwenden und unterstützt agile sowie traditionelle Projektmanagementansätze.
- Effizienz: Standardisierte Vorlagen und Prozesse sparen Zeit und Ressourcen bei der Projektumsetzung.
- Risikominimierung: Durch das integrierte Risiko- und Qualitätsmanagement werden potenzielle Probleme frühzeitig erkannt und minimiert.
- Förderung der Zusammenarbeit: Klare Rollen- und Verantwortlichkeitsstrukturen verbessern die Kommunikation und Zusammenarbeit im Projektteam und mit Stakeholdern.

## 2.3.2 Organigramm / Projektstruktur

Dieses Organigramm bzw. diese Projektstruktur zeigt die verschiedenen Parteien in Ihrer Struktur auf.



## 2.3.3 Administratives

Um die Sicherheit während des Projekts zu gewähren, werden nur die berechtigten Personen Zugriff auf die nötigen Daten haben. Untenstehend ist eine kleine Matrix dazu aufgeführt. Hier ist ersichtlich, wie alles gestaltet und berechtigt ist.

| | Systemdokumentation | Hardware Infrasturktur | Software Infrastruktur | Backup Strategie | Datenschutz, Datensicherheit | Testing | Weekly Reports | Projektierung, Offerten, Kostenkonzipierung |
|---|---|---|---|---|---|---|---|---|
| Steinlin Timo | | | | | | | r | r |
| Isenschmid Noah | x | r | r | r | r | r | x | x |
| Reymond David | x | x | x | r | r | r | v | |
| Lalicata Paulo | v | r | x | r | x | x | v | |
| Fleury Lucien | v | r | x | x | r | r | v | |

Legende:

X: Vollzugriff, Bearbeiten
V: Vorschläge erteilen, aber ohne bearbeiten
R: Leserechte
*Leer*: Keine Berechtigungen

## 2.3.4 Kontakt

Da dies ein Projekt ist, mit grossem Umfang und welches Agilität und Flexibilität an den Tag legt, sind untenstehend die nötigen Kontakte für alle wichtigen Beteiligten aufgeführt.

| **Rolle** | **Name** | **Kontakt** |
|---|---|---|
| Stakeholder | Steinlin Timo | timo.steinlin@gibb.ch |
| Projektleitung | Isenschmid Noah | nis133927@stud.gibb.ch |
| Auskunft Security | Reymond David | dre112220@stud.gibb.ch |
| Auskunft Datenschutz | Lalicata Paulo | pla135948@stud.gibb.ch |
| Auskunft Backup | Fleury Lucien | lfl135186@stud.gibb.ch |

## 2.3.5 Aufgabenbereiche / Zuständigkeiten

### 2.3.5.1 Isenschmid Noah

Noah Isenschmid fungiert in diesem Projekt als Projektleitung. Zu seinen Aufgaben und Zuständigkeiten gehören folgende Punkte.

- Kommunikation und Koordination mit Stakeholder
- Kundenwünsche an Team vermitteln
- Zeitliches Management und Meilensteinsetzung mit Stakeholder
- Koordination und Aufgabenzuteilung Team
- Budgetierung
- Gewährleistung der angestrebten Produktqualität
- Planung und Optimierung der Ressourcen
- Dazu werden folgende Fähigkeiten benötigt.
- Verhandlungsgeschick und diplomatisches Geschick
- Fähigkeit, komplexe Sachverhalte klar und verständlich zu kommunizieren
- Aktives Zuhören und Empathie
- Übersetzung von Kundenanforderungen in technische Spezifikationen
- Konfliktlösungsfähigkeiten
- Zeitmanagement und Priorisierung
- Delegationsfähigkeiten
- Teamführung und -motivation
- Finanzplanung und -analyse
- Kostenkontrolle
- Risikomanagement
- Kenntnisse in Qualitätsmanagement und -sicherung
- Detailorientierung
- Fähigkeit, Feedback konstruktiv zu geben und zu empfangen
- Ressourcenmanagement
- Strategische Planungsfähigkeiten

### 2.3.5.2 Reymond David

David Reymond ist zuständig für folgende untenstehende Punkte.

- Hardware-Beschaffungen
- Software-Beschaffungen
- Security Implementierung
- Hosting
- Verantwortlichkeit Technische Dokumentation
- Dies erfordert eine breitgefächerte Knowledge Base, welche anhand der untenstehenden Punkte gut zu erkennen ist.
- Technisches Verständnis der Hardware-Anforderungen und -Spezifikationen
- Fähigkeit zur Bewertung und zum Vergleich von Hardware-Optionen
- Verständnis von Software-Lizenzmodellen und -Verträgen
- Bewertung der Software-Qualität und Kompatibilität
- Kenntnisse über verschiedene Softwareanbieter und -produkte
- Fähigkeiten im Management von Software-Assets
- Tiefgehendes Verständnis von Informationssicherheitsprinzipien und -standards (z.B. ISO/IEC 27001, NIST)
- Erfahrung in der Implementierung von Sicherheitslösungen (Firewalls, IDS/IPS, EDR, etc.)
- Risikoanalyse und -bewertungsfähigkeiten
- Kenntnisse in der Entwicklung und Durchsetzung von Sicherheitsrichtlinien und -verfahren
- Fähigkeiten in der Serververwaltung und -wartung
- Kenntnisse in Netzwerktechnologien und -protokollen
- Erfahrung mit Leistungsüberwachung und -optimierung

#### 2.3.5.3 Lalicata Paulo

Paulo Lalicata ist Projektmitarbeiter Nr. 2 und verantwortlich für folgende Bereiche.

- Datenschutz
- Datensicherheit
- Unterstützung Security Implementierung
- Testing des Technischen Projekts
- Für diese Tätigkeitsbereiche werden folgende Fähigkeiten vorausgesetzt.
- Kenntnisse der Schweizer Datenschutzgesetze (DSG, revDSG), Verordnung zum Bundesgesetz über den Datenschutz und der relevanten internationalen Standards (GDPR).
- Fähigkeit zur Bewertung und Anpassung von Datenschutzpraktiken und -policies, um Compliance sicherzustellen.
- Erfahrung in der Durchführung von Datenschutz-Folgenabschätzungen und Audits.
- Fähigkeit zur Entwicklung und Implementierung von Notfallplänen und Reaktionsstrategien auf Sicherheitsvorfälle.
- Erfahrung in der Identifizierung, Bewertung und Mitigation von Sicherheitsrisiken.
- Kenntnisse in der Anwendung von Verschlüsselungstechniken und anderen Datensicherheitsmechanismen.
- Fähigkeit zur Entwicklung und Implementierung von Notfallplänen und Reaktionsstrategien auf Sicherheitsvorfälle.
- Kenntnisse in der Konzeption und Umsetzung von Sicherheitsarchitekturen und -frameworks.
- Erfahrung mit der Integration von Sicherheitstools und -technologien in neue Systeme.
- Fähigkeiten in der Planung und Durchführung von Testszenarien und -fällen, um Funktionalität, Leistung und Sicherheit von Systemen zu überprüfen.
- Kenntnisse in verschiedenen Testmethoden (z.B. Penetrationstests, Vulnerability Assessments, Funktions- und Leistungstests).
- Erfahrung mit Testautomatisierungstools und -frameworks.
- Fähigkeit zur Analyse und Dokumentation von Testergebnissen sowie zur Entwicklung von Empfehlungen zur Verbesserung.

#### 2.3.5.4 Fleury Lucien

Lucien Fleury vervollständigt unser Team, welches das Projekt flocker in Angriff nimmt und nimmt sich folgenden Zuständigkeiten an.

- Unterstützung Testing
- Konzeption und Implementierung Backupstrategie
- Sicherstellung der technischen Funktionalität
- Monitoring
- Um diese Aufgaben bewältigen zu können müssen folgende Anforderungen erfüllt werden.
- Kenntnisse in Testplanung, -design und -durchführung.
- Fähigkeit zur Identifizierung und Dokumentation von Fehlern sowie zur Zusammenarbeit mit Entwicklern bei der Fehlerbehebung.
- Verständnis der Anforderungen an Datenverfügbarkeit und -wiederherstellung.
- Kenntnisse in der Bewertung von Risiken und der Auswahl geeigneter Backup-Technologien und -medien.
- Erfahrung in der Planung von Backup-Zeitplänen und der Durchführung von Wiederherstellungstests.
- Fähigkeit zur Analyse und Diagnose von Systemproblemen.
- Kompetenzen in der Wartung und Aktualisierung von Systemen zur Gewährleistung optimaler Leistung.
- Kenntnisse in der Implementierung und Überwachung von Performance- und Sicherheitsmaßnahmen.
- Erfahrung mit der Einrichtung und Konfiguration von Monitoring-Tools für Systeme, Netzwerke und Anwendungen.
- Fähigkeit zur Analyse von Monitoring-Daten zur frühzeitigen Erkennung von Problemen.
- Kenntnisse in der Alarmierung und im Incident Management.

# 2.4 Aufwand

## 2.4.1 Fixer Aufwand

| Person | Beschreibung | Total in Std. |
|---|---|---|
| Noah Isenschmid | AVOR, Projektmanagement, Kundenkoordination, Teamkoordination, Dokumentation, Umsetzung | 63 |
| Paulo Lalicata | Datenschutz, Datensicherheit, Umsetzung, Testing | 63 |
| Lucien Fleury | Planung Umsetzung, Umsetzung, Testing, Backupstrategie | 63 |
| David Reymond | Dokumentation, Beschaffung Material, Umsetzung, Hosting, Security, Umsetzung | 63 |
| **Total** | | **252** |

3 Stunden pro Woche und Total 21 Wochen.
Das ergibt einen Aufwand von 63 Stunden pro Person, Total 252 Stunden.

Die Fixen Arbeitszeiten erschliessen sich aus den «Modularbeitszeiten», welche obligatorisch für jeden sind, bis das Projekt abgeschlossen ist.

### 2.4.2 Geschätzter Aufwand

| Person | Beschreibung | Aufwand in Std. |
|---|---|---|
| Noah Isenschmid | Nachkorrektur Dokumentation, Neuplanung | 15 |
| Paulo Lalicata | Fehlersuche, Nacharbeiten | 15 |
| Lucien Fleury | Fehlersuche, Nacharbeiten | 15 |
| David Reymond | Fehlersuche, reparieren Infra., Nacharbeiten | 15 |
| **Total** | | **80** |

Der Zusatzaufwand wird auf mögliche Ereignisse geschätzt, welche eintreten könnten, die zusätzlichen Aufwand verursachen. Damit es nicht zu sportlich wird bei Fehler- und oder Vorfällen ist dieser Aufwand grosszügig geschätzt.

Dieser Zusatzaufwand, werden bei allfälligem Eintreten als Kulanzleistungen angeschaut.

## 2.5 Kosten und Material

### 2.5.1 Einmalige Kosten

| Artikel / Material | Kosten | Anzahl | Kosten Total |
|---|---|---|---|
| Raspberry PI 4 Model B 8GB | 120 CHF | 1 | 120 CHF |
| Windows Lizenzen | 200 CHF | 1 | 200 CHF |

### 2.5.2 Freeware und Material ohne Kosten

| Software / Material | Beschreibung |
|---|---|
| Visual Studio Code | Visual Studio Code ist ein kostenloser Quelltext-Editor von Microsoft. Visual Studio Code ist plattformübergreifend für die Betriebssysteme Windows, macOS und Linux verfügbar. |
| Docker | Docker ist eine freie Software zur Isolierung von Anwendungen mit Hilfe von Container Virtualisierung. Docker vereinfacht die Bereitstellung von Anwendungen, weil sich Container, die alle nötigen Pakete enthalten, leicht als Dateien transportieren und installieren lassen. |
| Portainer | Portainer ist eine vielseitige Container Management Software. |
| SQL-Server 2022 | SQL-Server 2022 (16. x) ist eine Weiterentwicklung der SQL Server-Plattform, die auf früheren Releases aufbaut. Sie stellt eine große Auswahl an Entwicklungssprachen, Datentypen und Betriebssystemen zur Verfügung und ermöglicht die Arbeit in lokalen und Cloudumgebungen. |

## 2.5.3 Wiederkehrende Kosten

| Artikel / Material | Kosten | Anzahl | Kosten Total |
|---|---|---|---|
| Strom | 29.90 Rappen/kWh | 52.56 | 15.70 |
| DNS Registrierung (flocker.cloud) | 29.00 CHF/mtl. | 1 | 29.00 CHF/mtl. |
| Supportvertrag (Alle Support- & Wartungsarbeiten werden dadurch als Kulanz verrechnet) | 300 CHF/mtl. | 1 | 300 CHF/mtl. |

## 2.5.4 Dienstleistungen

| Personal | Arbeitszeit in Std. | Kosten / Std. | Kosten Total |
|---|---|---|---|
| Noah Isenschmid | 63 | 125 CHF | 7'875 CHF |
| Lucien Fleury | 63 | 100 CHF | 6'300 CHF |
| David Reymond | 63 | 100 CHF | 6'300 CHF |
| Paulo Lalicata | 63 | 100 CHF | 6'300 CHF |
| | | | **Total**: 26'775 CHF |

## 2.5.5 Totale Projektkosten

### 2.5.5.1 Totale einmalige Kosten

| Artikel / Material | Kosten | Anzahl | Kosten Total |
|---|---|---|---|
| Raspberry Pi 4 Model B 8 GB | CHF 120.-- | 2 | CHF 240.-- |
| Windows Lizenzen | CHF 200.-- | 1 | CHF 200.-- |
| AVOR, Projektmanagement, Kundenkoordination, Teamkoordination, Dokumentation, Umsetzung | CHF 125.-- | 63 | CHF 7’875.-- |
| Datenschutz, Datensicherheit, Umsetzung, Testing | CHF 100.-- | 63 | CHF 6’300.-- |
| Planung Umsetzung, Umsetzung, Testing, Backupstrategie | CHF 100.-- | 63 | CHF 6’300.-- |
| Dokumentation, Beschaffung Material, Umsetzung, Hosting, Security, Umsetzung | CHF 100.-- | 63 | CHF 6’300.-- |
| **Totale einmalige Kosten** | | | **CHF 27'215.--** |

#### 2.5.5.2 Totale wiederkehrende Kosten (jährlich)

| Artikel / Material | Kosten | Anzahl | Kosten Total |
|---|---|---|---|
| DNS Registrierung (flocker.cloud) | CHF 29.50.-- | 1 | CHF 29.50.-- |
| Stromkosten | CHF 0.299.-- | 52.56 kWh* | CHF 15.70 |
| Support- & Wartungsvertrag | CHF 300.-- | 12 | CHF 3600.-- |
| **Totale wiederkehrende Kosten** | | | **CHF 3645.20.--** |

* Die Stromkosten können je nach Stromverbrauch und aktuellem Strompreis variieren. Als Referenz wurde die Strompreispublikation von bern.ch verwendet.

#### 2.5.5.3 Totale Kosten Projekt

| Artikel / Material | Anzahl | Kosten |
|---|---|---|
| Totale Kosten einmalig | 1 | CHF 27’215.-- |
| Totale Kosten Jährlich | 5 | CHF 3645.20.-- |
| **Totale Kosten Projekt (Lebensdauer gerechnet auf 5 Jahre)** | | **CHF 45’441.--** |

# 2.6 Termine

**13.02.2024 13:00** Die Vorarbeiten sind erledigt, der Projektinitialisierungsantrag ist in Kontrolle.

**05.03.2024 16:00** Phase Initialisierung ist abgeschlossen, die Studie ist abgeschlossen, der Projektauftrag ist erteilt, die Zeitplanung ist aufgegleist.

Weitere Termine:

Die genauen Daten für den Abschluss der Konzept- und der Realisierungsphase werden noch vom AG übermittelt. Voraussichtlich sind KW12 und resp. KW20 für die Abgabe geplant, welche jedoch noch nicht definitiv feststehen.

KW24 – 26 sind für Präsentationen der Arbeiten geplant.

# 2.7 Ressourcen

Als Ressourcen für unser Vorhaben werden wir vor allem eigene Geräte verwenden, welche Laptops beinhalten. Zum Teil werden wir die von der GIBB zur Verfügung gestellten Infrastruktur beanspruchen. Über MS Teams werden die Unterlagen zwischengespeichert. Die Kommunikation erfolgt auch über Teams.

Um dieses Projekt möglichst effizient und reibungslos zu erledigen, braucht es natürlich Ressourcen. Ressourcen teilen wir in 3 verschiedene Kategorien auf.

## 2.7.1 Personalressourcen

Personalressourcen beziehen sich auf die Projektmitarbeiter und -beteiligten, einschliesslich der Projektmanager, Teammitglieder, Berater und sonstigen Stakeholder. Diese Ressourcen sind für die Ausführung der Projektarbeit, Entscheidungsfindung und Kommunikation unerlässlich.

### 2.7.2 Materielle Ressourcen

Materielle Ressourcen umfassen in unserem Fall physische Materialien und Ausrüstungen, die für die Durchführung des Projekts benötigt werden. Dazu gehören Büromaterialien, Hardware, Software und andere physische Güter. Hierbei spielt es keine Rolle, ob dies ein Laptop oder ein Raspberry Pi ist. Dies wird nicht differenziert.

### 2.7.3 Finanzielle & Zeitliche Ressourcen

Finanzielle Ressourcen beziehen sich auf das Budget, das für die Durchführung des Projekts zur Verfügung steht. Finanzielle Ressourcen sind notwendig für die Beschaffung von Materialien, die Bezahlung der Mitarbeiter, die Deckung von Betriebskosten und sonstige Ausgaben. Wobei zeitliche Ressourcen sich auf die verfügbare Zeit für die Durchführung des Projekts beziehen. Zeitmanagement ist entscheidend für den erfolgreichen Abschluss von Projektaufgaben innerhalb der festgelegten Fristen. Dies haben wir in eine Kategorie zusammengepackt, da beide nahe beieinanderstehen und beide variabel sind und sich verändern können.

## 2.8 Kommunikation

Die Kommunikation zwischen dem Projektteam und dem Stakeholder ist der Schlüssel zum Erfolg.

So wie die Projektstruktur aufgebaut ist, wird der Stakeholder nur zu den Abschlussgesprächen & -präsentationen mit den Projektmitgliedern Kontakt pflegen. Ansonsten besteht eine klar definierte Hierarchie, welche klar definiert wie die Kommunikationen zwischen AG und Projektteam erfolgt. Untenstehendes zeigt dies klar und verständlich auf.

### 2.8.1 Statusmeldungen

Statusmeldung sind wichtig, da hierdurch der Stakeholder sich auch ein grobes Bild der Realisierung machen kann. Diese erfolgen folgendermassen. Chronologisch aufgeführt von oben nach unten.

| Was? | Wer? | Was (ausführlich)? |
|---|---|---|
| Interne Teamsitzung | Team | Statusmeldungen von Teammitgliedern an Projektleiter. Eventuelle Probleme/Stolpersteine, Allgemeines Feedback und QA |
| Kontaktaufnahme AG | PL | Anhand eines E-Mails oder Meetings wird ein Termin vereinbart und mit den wichtigsten Traktanden (Fortschritt, Budgetauslastung, aktuelle Problem, u.v.m.) |
| Statusmeldung | PL, AG | Ein Meeting zwischen dem PL und dem AG welches Klarheit über das Projekt verschafft. Besprochen werden die Traktanden aus der Traktandenliste und zum Schluss wird noch eine kleine Fragerunde stattfinden, wo sich der AG gegenüber dem PL mit Wünschen o.Ä. äussern kann. |

### 2.8.2 Projektanpassungen seitens AG

Falls Projektanpassungsanfragen seitens Stakeholder gewünscht sind, nimmt dies folgenden Kommunikationsweg.

| Was? | Wer? | Was (ausführlich)? |
|---|---|---|
| Meeting | PL, AG | Bedürfnisaufnahme zwischen PL und AG. Konkrete Besprechung, ob es im Budget und Zeitplan liegt und ob es Realisierbar ist. |
| Interne Teamsitzung | Team | Aufgabenzuteilung durch PL an Projektmitarbeiter. |

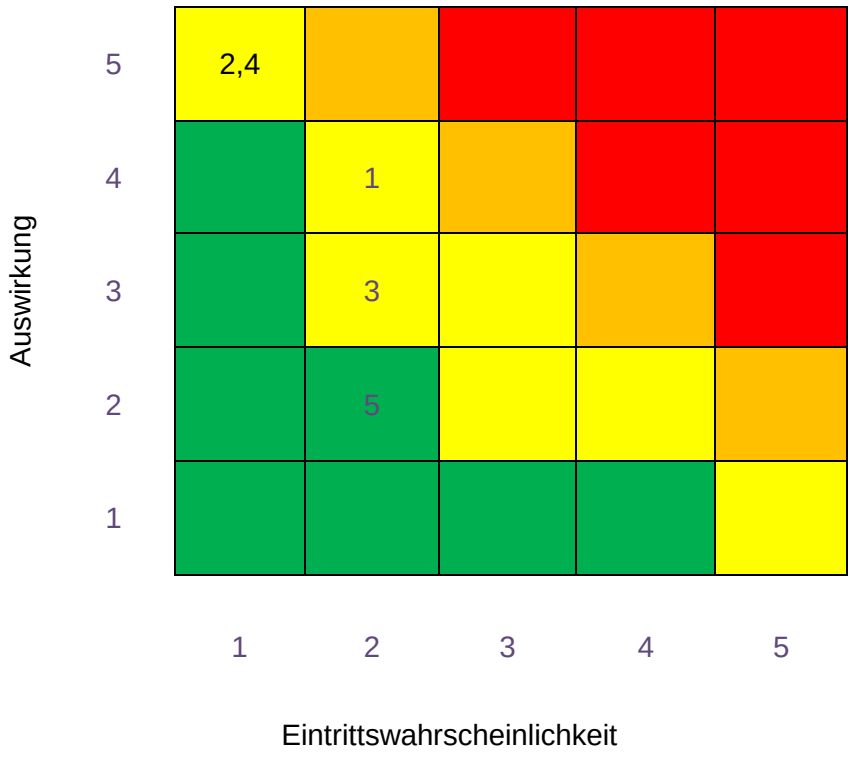
Erledigte Projektanpassungen werden, jeweils an den Statusmeldungen besprochen.

### 2.8.3 Projektanpassungen seitens PL

Falls plötzliche Projektanpassungen seitens Projektteam nötig sind, werden folgende Kommunikationsschritte verwendet.

| Was? | Wer? | Was (ausführlich)? |
|---|---|---|
| Interne Teamsitzung | Team | Meldung von nötiger Anpassung durch Projektmitarbeiter an Projektleiter. Besprechung Folgen und Auswirkungen. Neubudgetierung falls nötig |
| Meeting | PL, AG | Besprechung von nötiger Anpassung. Themen wie Folgen, Änderungen und Auswirkungen werden besprochen, sowie allfällige Anpassungen am Budget und ob die Anpassung als Kulanzleistung gilt oder nicht. |

## 2.9 Risiken




| Nr. | Risiko | Beschreibung | Massnahme | Einschätzung |
|---|---|---|---|---|
| 1 | Funktionsfehler | Wenn der Containerdienst abstürzt, laufen die Virtuellen Maschinen nicht weiter. | Supportbereitschaft | 4,2 |
| 2 | Hardwarefehler | Wenn unsere Hardware (Raspberry Pi) nicht funktioniert. | Ersatzgerät mit Cluster. | 5,1 |
| 3 | VM-Absturz | Wenn die Virtuelle Maschinen abstürzen, laufen die Dienste und Anwendungen nicht weiter. | Snapshots | 3,2 |
| 4 | Stromausfall | Da wir keine Notlösung für Stromausfälle haben laufen unsere Systeme ohne Strom nicht mehr. | USV installieren. | 5,1 |
| 5 | Dienstabsturz | Wenn ein einzelner Dienst abstürzt. | Monitoring einrichten mit automatisiertem Alerting. | 2,2 |

# 3 Studie

## 3.1 Situationsanalyse

### 3.1.1 Ausgangslage

Manchmal wäre es praktisch einfach eine VM nach Belieben schnell und einfach zu erstellen, ohne grosse Installationen, Lizenzen, nervige Netzwerkkonfigurationen, beachten der Virtualisierungsumgebung und das Herunterladen der riesigen ISO-Files. Das ganze über das Web zu bedienen, ohne Software installieren zu müssen.
Uns ist allen AWS bekannt, eine kostenpflichtige, nicht ganz einfach zu navigierende Webapplikation. Zwar bietet AWS viele umfangreiche Funktionen und Möglichkeiten, jedoch kann das für den Durchschnittsverbraucher schnell zu viel und zu komplex werden. Um einmalig eine einfach VM zu erstellen wäre dafür der Aufwand und vor allem die Kosten zu hoch.  Mit flocker wollen wir damit Abhilfe schaffen. Mit der free-to-use Webapplikation ist es möglich per click-to-run einfach und unkompliziert VMs nach Belieben zu erstellen, diese so lange wie gewollt nutzen und bei Nichtgebrauch löschen. Somit würden für den Benutzer die hohen Betriebskosten und der hohe Aufwand wegfallen.

### 3.1.2 Stärken

| Stärken des Projektteams | Stärken des Projektes |
|---|---|
| - STPT1: Gute Kenntnisse in: Docker, Portainer, Datenbanken, Webserver, Netzwerk, Firewall, Domain registering,<br>- STPT2: Arbeitskenntnisse von drei Jahren in folgenden Bereichen: Netzwerk, MS, Scripting, Planung und Management, Coding, Linux und Windows, VM, Firewall, Security, Zertifikate,<br>- STPT3: Effizientes Team-Management<br>- STPT4: Hardware ist schon vorhanden und kann einfach aufgesetzt werden | - STPR1: Kostenlos für den Benutzer<br>- STPR2: Einfache Bedienung<br>- STPR3: Ressourcenschonend<br>- STPR4: Intuitiv und Benutzerfreundlich |

### 3.1.3 Schwächen

| Schwächen des Projekts-teams | Schwächen der vorhandenen Lösungen |
|---|---|
| - SCPT1: Kein breites Wissen in der Applikationsentwicklung<br>- | - SCVL1: Hohe Kosten für den Benutzer<br>- SCVL2: Nicht Benutzerfreundlich<br>- SCVL3: Meistens in Datacentern ausserhalb von der Schweiz (oder Europa) gehostet.<br>- SCVL4: Praktische Features hinter einer Paywall / Subscription |

## 3.2 Ziele

Unsere Ziele sind nach SMART definiert, damit gewährleistet wird, dass die Ziele Spezifisch, messbar, Achieveable, Realistisch und Terminiert sind.

1) Z1: Bis zu der Realisierungsphase des Projektes haben wir uns alle das nötige Wissen für die Applikationsentwicklung, welche es im Rahmen des Projektes benötigt, angeeignet. Bezogen auf die (SCPT1) unsere Schwäche in der Applikationsentwicklung.
2) Z2: Wir halten die Webapplikation für den Benutzer kostenfrei. Bezogen auf die (SCVL1), Lösungen wie AWS haben hohe Kosten für den Benutzer, wir halten unsere Lösung kostenlos. Somit generieren wir keinen Gewinn, aber da wir die Lösung im Betrieb tief halten, ist das für uns kein weiteres Problem.
3) Z3: Die Hardware wird vor der Realisierungsphase aufgesetzt und konfiguriert. Bezogen auf die (STPT4). Damit wir uns auf die Applikationsentwicklung und Anderweitiges konzentrieren können.
4) Z4: Mockups für die Webapplikation sind bis zur Realisierungsphase erstellt. Die Mockups zeigen den genauen Aufbau der Webapplikation, diese sind leicht überschaubar und es sind keine komplexen Menüs und Untermenüs vorhanden. Das Design ist modern und schlicht gehalten. Bezogen auf die Schwäche (SCVL2).
5) Z5: Um eine Datensicherheit zu gewährleisten, haben wir entschieden, jegliche Daten, Dienste und Dazugehörigkeiten in der Schweiz auf unseren eigenen Systemen zu hosten. Wir verzichten somit auf die Unterstützung externen Datacenter-Anbieter. So können wir eine strikte Einhaltung des Datenschutzgesetzes (DSG, DSGVO, revDSG) gewährleisten und können das SCVL3 eliminieren.

### 3.2.1 Rahmenbedingungen

Pro Woche werden uns von der Schule 3 Stunden zur Verfügung gestellt, die wir frei benutzen können, um individuell am Projekt arbeiten zu können. In der Schule (GIBB-IET), können wir in jeglichen Räumen, die frei zur Verfügung stehen, nach Absprache mit der Lehrperson Timo Steinlin, arbeiten. Was uns auch angeboten wird ist ein Gesuch zu stellen, um ausserhalb der Schule zu arbeiten, wie zum Beispiel zu Hause, oder in einem der Betriebe. Dieses Gesuch muss jedoch von dem Lehrbetrieb wie der Lehrperson genehmigt werden. Abgabe/Präsentation des ersten Teils (bis und mit Kapitel 4) der Studie erfolgt per 20.02.2024. Weitere Terminierungen werden im Verlaufe des Projekts kommuniziert und festgelegt. Sobald diese feststehen, werden diese schriftlich festgehalten.

### 3.2.2 Abgrenzungen

Wir setzten Abgrenzungen, um uns bei Vorfällen und oder Unvorhergesehenem zu schützen und tragen somit keine Verantwortung, wenn die Webapplikation bei folgenden Fällen nicht mehr verwendet werden kann:

- Ausfall der Infrastruktur über längere Zeit
- Verlust der Daten bei der Löschung der Systeme / VMs
- Naturkatastrophen.
- Wir setzen diese Abgrenzungen, um klarzustellen, dass wir bei folgenden Punkten keine weiteren Massnahmen bei Anfragen oder ähnlichem ergreifen werden:
- Wir leisten keinen Support für die Verwendung der Webapplikation
- Wir leisten keinen Migrationssupport
- Wir leisten keine Webekampagne für flocker
- Das Projekt wird nach Fertigstellung nicht weiter unterhalten und bei auftretenden Fehlern nicht weiter gelöst
- Keine Updates

## 3.3 Liste der Stakeholder

| Person | Position | Beschreibung | Art |
|---|---|---|---|
| Timo Steinlin | Stakeholder | Kunde / Auftraggeber, wird über den Stand und den Verlauf des Projektes kontinuierlich informiert | Direkt betroffen |
| Noah Isenschmid | Projektmanager | Manager des Projektes, kennt den genauen Stand des Projektes und informiert den Stakeholder über den Stand des Projektes | Direkt betroffen |
| Paulo Lalicata | Projektmitglied | Ist ein Mitglied und aktiver Arbeiter im Projekt. Er weiss über den Stand in seinem Bereich immer bescheid und kann bei Anforderung ein Statusupdate geben | Direkt betroffen |
| Lucien Fleury | Projektmitglied | Ist ein Mitglied und aktiver Arbeiter im Projekt. Er weiss über den Stand in seinem Bereich immer bescheid und kann bei Anforderung ein Statusupdate geben | Direkt betroffen |
| David Reymond | Projektmitglied | Ist ein Mitglied und aktiver Arbeiter im Projekt. Er weiss über den Stand in seinem Bereich immer bescheid und kann bei Anforderung ein Statusupdate geben | Direkt betroffen |
| Swizzonic | Domain Host | Ist in dem Sinne betroffen, da eine Domain bei diesem Anbieter registriert und gekauft wird. Swizzonic muss nicht in dem Sinne informiert werden und oder weiteres zu diesem Projekt wissen / erfahren | Indirekt betroffen |

## 3.4 Anforderungen an das Projekt

| ID | Anforderung | Ziel |
|---|---|---|
| A1 | Unsere Webapplikation muss ohne Subscription oder anderen ähnlichen Gewinn einbringenden Methoden verwendet werden können. | Wir halten die Webapplikation für den Benutzer kostenfrei. Ziel (Z1) |
| A2 | Die Webseite / Webapplikation soll leicht und einfach zu bedienen sein. Es ist benutzerfreundlich gestaltet und es hat nicht zu viele «unnötige» Funktionen. | Mockups sind bis zur Realisierungsphase erstellt. Die Mockups zeigen den genauen Aufbau der Webapplikation, diese sind leicht überschaubar und es sind keine komplexen Menüs und Untermenüs vorhanden. Ziel (Z4) |
| A3 | Alle VMs, Systeme, Daten und so weiter, sind ausschliesslich in der Schweiz und auch nur in der Schweiz verfügbar. Daten VMS usw. dürfen nicht ausserhalb der Schweiz gelagert / vorhanden sein. | Um eine Datensicherheit zu gewährleisten, haben wir entschieden, jegliche Daten, Dienste und Dazugehörigkeiten in der Schweiz auf unseren eigenen Systemen zu hosten. Wir verzichten somit auf die Unterstützung externen Datacenter-Anbieter. So können wir eine strikte Einhaltung des Datenschutzgesetzes (DSG, DSGVO, revDSG) gewährleisten und können das SCVL3 elimieren. Ziel (Z5) |
| A4 | Die Hardware stellt David Reymond und Noah Isenschmid zur verfügung. Das ganze Projekt wird zu Hause gehosted und bereitgestellt. | Die Hardware wird vor der Realisierungsphase aufgesetzt und konfiguriert. Ziel (Z3) |
| A5 | Das Projekt wird mit dem nötigen Wissen und know how erstellt und geleitet. Wir bilden uns in der Applikationsentwicklung weiter, damit das Projekt realisiert werden kann. | Bis zu der Realisierungsphase des Projektes haben wir uns alle das nötige Wissen für die Applikationsentwicklung, welche es im Rahmen des Projektes benötigt, angeeignet. Ziel (Z1) |

# 3.5 Lösungsvarianten

## 3.5.1 Variantenübersicht

### 3.5.1.1 Variante 1

In dieser Variante verwenden wir die smartlearn Umgebung für die Realisierung des Projekts. Wir werden div. VMs und Server brauchen. Zuerst eine VM, welche als Management Host dient. Eine VM welche als Virtualisierungsserver dient. Eine SQL-Datenbank, welche für die Speicherung der Zugangsdaten verwendet wird. Eine VM, die für das Hosting der Website/Webapplikation zuständig ist. Und eine Firewall, welche die Zugänge und das Routing steuert und für die Garantie der Sicherheitsimplementierung.

### 3.5.1.2 Variante 2

Falls wir Variante 2 verwenden, werden wir Gebrauch von unseren Raspberry Pis machen. Wir können hier mit Docker alle nötigen virtuelle Maschinen erstellen und als Virtualisierungshost verwenden wir einen physischen Computer. So können wir Managementhost und Virtualisierungshost klar voneinander trennen und bringen so Struktur und eine erweiterte Sicherheitsebene in unser System.

### 3.5.1.3 Variante 3

Bei der dritten Variante verwenden wir nur einen physischen Computer. Dieser wird als Verwaltungshost und ebenfalls als Virtualisierungshost eingesetzt. Hierbei liegt der Vorteil indem, dass dies eine kostengünstige und zeiteffiziente Lösung ist, jedoch entstehen hier bedenken, bei der Sicherheit, da nichts voneinander abgetrennt ist. Diese Lösung ist absichtlich die dritte Variante, da diese nur als Notlösung verwendet wird.

## 3.5.2 Beschreibung der Varianten

### 3.5.2.1 Variante 1 (smartlearn)

In dieser Option nutzen wir die Umgebung "smartlearn" für unser Projekt. Hierbei setzen wir verschiedene virtuelle Maschinen (VMs) und Server ein.

| | |
|---|---|
| Management-VM | Sie dient als zentrales Nervensystem unseres Projekts. Hier laufen Tools und Software, die für die Überwachung und Steuerung aller anderen Komponenten im System zuständig sind. Diese VM ermöglicht es uns, alle Prozesse effizient zu verwalten. |
| Virtualisierung-VM | Diese VM ist das Herzstück unserer Virtualisierungsumgebung. Sie ermöglicht es uns, mehrere virtuelle Maschinen für unterschiedliche Zwecke zu erstellen und zu verwalten, ohne dass dafür zusätzliche Hardware erforderlich ist. Das spart Kosten und vereinfacht die Skalierung. |
| SQL-Datenbank | Eine robuste Datenbank ist wichtig, um kritische Daten wie Zugangsdaten sicher zu speichern. Die Wahl einer SQL-Datenbank gewährleistet Zuverlässigkeit und eine effiziente Datenverwaltung. |
| Webhosting-VM | Diese VM hat die Aufgabe, unsere Webanwendung oder Website zu hosten. Das bedeutet, sie stellt sicher, dass unsere Anwendung jederzeit online und zugänglich ist. Dies ist besonders wichtig, um eine positive Benutzererfahrung zu gewährleisten. |

| | |
|---|---|
| Firewall | Eine leistungsstarke Firewall schützt unser Netzwerk vor unerwünschten Zugriffen und Angriffen. Sie kontrolliert den Datenverkehr und gewährleistet, dass nur autorisierte Benutzer Zugang zu unseren Systemen haben. Dies ist entscheidend für die Aufrechterhaltung der Sicherheit und Integrität unseres Projekts. |

#### 3.5.2.2 Variante 2 (Einsatz von Raspberry Pis, Docker und Physischen Computer)

Falls Variante zwei gewählt wird, arbeiten wir mit Raspberry Pis als Management-Host und einem Physischen Computer, welcher als Virtualisierungshost fungiert. Hierbei wird mit einer einfachen Struktur gearbeitet. Der Raspberry Pi (Managementhost) arbeitet mit Ubuntu Server und Docker. Auf Docker installieren wir die nötige Software und Schnittstellen wie Cloudflared, Bitwarden, Authelia, Guacamole, Portainer u.v.m. Untenstehend werden die einzelnen Dienste beschrieben und aufgezeigt, für was diese verwendet werden.

| | |
|---|---|
| Managementhost | Der Raspberry Pi wird als Management (kurz mgmt) -host verwendet und wird mit Ubuntu und Docker betrieben. Dieser ist für die Verwaltung, Wartung und Überwachung zuständig. |
| Cloudflared | Dies ist der Dienst, welcher für die Sicherstellung und Funktionalität des Website Traffics zuständig ist. Cloudflare ist ein zuverlässiger und seriöser DNS-Provider. Wir nehmen den Reverse-Proxy und die DNS-Dienste von Cloudflare in Anspruch. |
| Apache Guacamole | Apache Guacamole ist eine Virtualisierungssoftware, in welchen virtuellen Maschinen/Computer virtualisiert bereitgestellt werden können. Dies ist eine gute Wahl, da diese Software OpenSource ist und daher eine grosse Knowledgebase und Community hat. |
| Authelia | Dies ist ein Dienst, welcher verwendet werden kann, um eine Selfhosted Multifaktorauthentifizierung einzurichten. Da wir uns intensiv mit Security auseinandersetzen wollen, haben wir uns dazu entschieden dies zu implementieren. |
| Virtualisierungshost | Der Virtualisierungshost läuft mit Apache Guacamole und ist für die Virtualisierung der virtuellen Maschinen zuständig. |

Diese Variante bietet eine mögliche Lösung, bei welcher man eine gute Balance zwischen Sicherheit und Benutzerfreundlichkeit erreicht. Auch die Verwaltung ist übersichtlich und wir können alles besser verwalten.

#### 3.5.2.3 Variante 3 (All-in-One Lösung)

Variante 3 sieht vor, dass für die Umsetzung des Projekts nur ein einziger physischer Computer verwendet wird. Dieser Computer dient sowohl als Verwaltungshost, als auch als Virtualisierungshost.
Die Entscheidung für Variante 3 bedeutet, dass der physische Computer die gesamte Last der Verwaltung, Virtualisierung und anderen Funktionen übernehmen muss.
Um das Sicherheitsproblem mit dem Nicht-Vorhandensein einer Abtrennung der einzelnen Komponenten, zu lösen, werden folgende Punkte implementiert:

- Netzwerksegmentierung
- Firewall Konfiguration
- Zugriffssteuerung
- Logging
- Überwachung
- Regelmässige Sicherheitsüberprüfungen
- Backup Strategie

Es ist wichtig zu beachten, dass diese Variante Kompromisse in Bezug auf Sicherheit und Skalierbarkeit mit sich bringt und Netzwerktechnisch viel Aufwand betrieben werden muss, um die gewünschte Software sicher auf einem System laufen zu lassen. Das Endprodukt ist jedoch eine All-in-One Lösung, welche somit weniger externe Ressourcen benötigt.

## 3.6 Bewertung der Varianten

| **Anforderung** | | **Variante 1** | | **Variante 2** | | **Variante 3** | |
|---|---|---|---|---|---|---|---|
| **Muss** | | | | | | | |
| Einfaches Design | | Ja | | Ja | | Ja | |
| Kostenlos für den Benutzer | | Ja | | Ja | | Ja | |
| Sicherheit der Systeme | | Ja | | Ja | | Ja | |
| Volle Kontrolle | | Nein | | Ja | | Ja | |
| Sicherstellung der Stabilität | | Nein | | Ja | | Ja | |
| Kostengünstiger Unterhalt | | Ja | | Ja | | Nein | |
| **Kann** | **Gewichtung** | **Punkte** | **Total** | **Punkte** | **Total** | **Punkte** | **Total** |
| Erweiterbar sein | 20 | 2 | 40 | 5 | 100 | 3 | 60 |
| Redundant | 10 | 1 | 10 | 3 | 30 | 2 | 20 |
| Verwaltung | 30 | 3 | 90 | 5 | 150 | 4 | 120 |
| Langzeit Support | 10 | 1 | 10 | 3 | 30 | 5 | 50 |
| Cluster | 30 | 1 | 30 | 5 | 80 | 2 | 60 |
| **Total** | 100 | | 180 | | 390 | | 310 |

**1**: nicht erfüllt **2**: knapp erfüllt **3**: ausreichend **4**: gut **5**: sehr gut

Wie man anhand der obenstehenden Tabelle sieht, ist die Variante 2 in allen Punkten und Bewertungen überlegen. Somit haben wir uns für die zweite Variante entschieden.

## 3.7 Lösungsbeschreibung

Wir wählen wie oben aufgeführt die zweite Variante. Da diese uns die meiste Flexibilität und Sicherheit bietet. Um dieses Projekt umsetzen zu können müssen wir dies gut aufteilen können. Wir unterteilen unser Lösungsbeschrieb in … Teile. «Benötigte Komponenten (Hard- & Software)», «Beschrieb und Verwendung Komponenten» und «Netzwerkstruktur».

### 3.7.1 Benötigte Komponenten

- Raspberry Pi 4 Model B
    - Cortex-A72
    - 8 GB RAM
    - 512 GB SSD
- Physical Computer
    - i9-9900K
    - 32 GB RAM
    - GTX1660Ti
    - 2048 GB NVMe SSD
- Docker
    - OpenSource
    - Containerisierungssoftware
- Portainer
    - Docker Verwaltungssoftware
    - Community Edition
- Apache Guacamole
    - Virtualisierungssoftware
    - OpenSource
- Cloudflared
    - Reverse Proxy
    - DNS Server
- CMS
    - Vakant

## 3.7.2 Beschrieb und Verwendung Komponenten

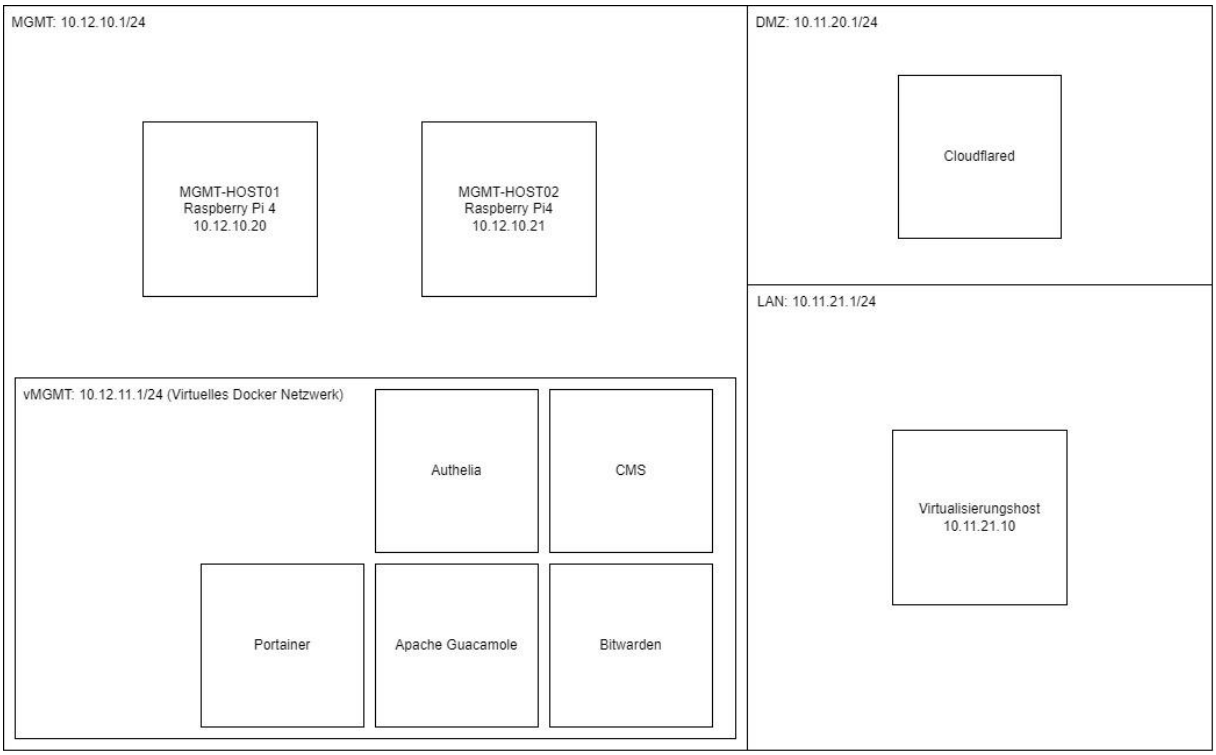
Als Managementhost verwenden wir einen Raspberry Pi. Dieser ist kostengünstig und leistungsstark. Damit können wir unsere komplette Infrastruktur verwalten, monitoren und überwachen. Wir haben uns für einen Managementhost entschieden, da dieser eine grosse Flexibilität bietet. So können wir von einem zentralen Punkt auf alle notwendigen Systeme zugreifen. Auch im Bereich Updates ist ein Managementhost von Vorteil, da man damit einen zentralen Verteiler bereitstellen kann.
Als Virtualisierungshost verwenden wir einen Physischen Computer mit folgenden Spezifikationen.

| **Komponente** | **Artikel** |
|---|---|
| Prozessor (CPU) | i9-9900K |
| Arbeitsspeicher (RAM) | 32 GB DDR4 |
| Grafikkarte (GPU) | Nvidia GeForce GTX 1660 Ti |
| Speicher | 2048 GB NVMe M.2 SSD |
| Netzwerkkarte | 2x 2.5 Gbit RJ45 |

Diese Konfiguration ist ausreichend, um zu allen Zeiten eine performante und stabile Verbindung für uns relevante Zwecke zu bieten.
Wir verwenden Docker als Schnittstelle zwischen unseren Plattformen. Die Verwendung von Docker als Schnittstelle zwischen unseren Plattformen bietet uns viele Vorteilen, die sowohl die Entwicklung als auch den Betrieb von Anwendungen vereinfachen und verbessern. Docker ermöglicht die Containerisierung von Anwendungen und deren Abhängigkeiten in Container, die dann leicht zwischen verschiedenen Umgebungen transportiert und ausgeführt werden können.
Da Docker auf unserem Raspberry Pi Server auf einem Ubuntu Server betrieben wird und wir dadurch kein GUI haben, haben wir entschieden, dass wir eine Verwaltungssoftware verwenden werden – Portainer. Portainer ist eine Software, mit welcher es möglich wird, diverse Prozesse in der Konzeption und Umsetzung in einem einfach zu bedienendem GUI, zu vereinfachen.
Apache Guacamole ist unsere Virtualisierungssoftware. Diese haben wir gewählt, da Sie OpenSource ist, dadurch eine grosse Knowledge Base und Community hat, einfach und benutzerfreundlich zu betreiben, Möglichkeiten für eine erweiterte Sicherheitsimplementierung und eine hohe Skalierbarkeit bietet.
Um unser Vorhaben über eine Webapp übers Internet erreichbar zu machen, müssen wir einen Zugang ins Internet haben. Wir haben uns hier für Cloudflare entschieden. Cloudflare ist ein zuverlässiger Provider für DNS-Hostings und vielem mehr. Wir werden über Cloudflare den Reverse Proxy und den DNS-Dienst verwenden, um ein Routing von unserem Raspberry Pi ins Internet zu machen.
Da wir uns das Wissen in der Webentwicklung noch aneignen müssen, werden wir das CMS, welches wir für die Webapp verwenden werden zu einem späteren Zeitpunkt bestimmen. Zurzeit stehen folgende Optionen zur Debatte.

- WordPress
- Drupal
- Contao
- Joomla

### 3.7.3 Netzwerkstruktur




Dies ist eine grobe Netzwerkstruktur wie Sie in der effektiven Umgebung dann aussehen könnte. Es soll aufzeigen welche Komponente verwendet werden. Hier sieht man die einzelnen Dienste, Clients und Server.

# 4 Konzept

## 4.1 Zusammenfassung

Dieses Dokument dient als umfassender Konzeptbericht für unser Projekt. Sein primärer Zweck ist es, einen detaillierten Überblick über die verschiedenen Aspekte des Projekts zu geben, um allen Beteiligten – von Projektmitarbeitern bis hin zu Stakeholdern – eine klare Vorstellung von den Zielen, der Struktur und dem geplanten Vorgehen zu vermitteln. Dies beginnt mit einer detaillierten Definition der Anforderungen, die das Projekt erfüllen sollen. Anschliessend wir die Thematik der Systemarchitektur beschrieben, wie diese aufgebaut wird und mit welchen Diensten und Anwendungen die Umsetzung bzw. Realisierung stattfinden wird. Ausserdem definieren wir hier auch schon das Testkonzept, also welche Tests wir nach der Realisierung durchführen wollen, um die Integrität und Funktionalität unseres Projekts zu testen. Der Schluss dieses Dokument widmet sich der Weiterführung der Projektplanung. Hier gehen wir auf die Planung der nächsten Phase, sowie auf momentane Risiken und mögliche Gefährdungen und wie wir diese bewältigen werden/konnten ein.

## 4.2 Systemanforderung

### 4.2.1 Anforderung an die Funktionalität

#### 4.2.1.1 Kopie der Anforderungen aus der Studie

| ID | Anforderung | Ziel |
|---|---|---|
| A1 | Unsere Webapplikation muss ohne Subscription oder anderen ähnlichen Gewinn einbringenden Methoden verwendet werden können. | Wir halten die Webapplikation für den Benutzer kostenfrei. Ziel (Z1) |
| A2 | Die Webseite / Webapplikation soll leicht und einfach zu bedienen sein. Es ist benutzerfreundlich gestaltet und es hat nicht zu viele «unnötige» Funktionen. | Mockups sind bis zur Realisierungsphase erstellt. Die Mockups zeigen den genauen Aufbau der Webapplikation, diese sind leicht überschaubar und es sind keine komplexen Menüs und Untermenüs vorhanden. Ziel (Z4) |
| A3 | Alle VMs, Systeme, Daten und so weiter, sind ausschliesslich in der Schweiz und auch nur in der Schweiz verfügbar. Daten VMS usw. dürfen nicht ausserhalb der Schweiz gelagert / vorhanden sein. | Um eine Datensicherheit zu gewährleisten, haben wir entschieden, jegliche Daten, Dienste und Dazugehörigkeiten in der Schweiz auf unseren eigenen Systemen zu hosten. Wir verzichten somit auf die Unterstützung externen Datacenter-Anbieter. So können wir eine strikte Einhaltung des Datenschutzgesetzes (DSG, DSGVO, revDSG) gewährleisten und müssen somit das SCVL3 nicht beachten. Ziel (Z5) |
| A4 | Die Hardware stellt David Reymond und Noah Isenschmid zur Verfügung. Das ganze Projekt wird zu Hause gehostet und bereitgestellt. | Die Hardware wird vor der Realisierungsphase aufgesetzt und konfiguriert. Ziel (Z3) |
| A5 | Das Projekt wird mit dem nötigen Wissen und Knowhow erstellt und geleitet. Wir bilden uns in der Applikationsentwicklung weiter, damit das Projekt realisiert werden kann. | Bis zu der Realisierungsphase des Projektes haben wir uns alle das nötige Wissen für die Applikationsentwicklung, welche es im Rahmen des Projektes benötigt, angeeignet. Ziel (Z1) |

#### 4.2.1.2 Verfeinerte Anforderungen

| Anforderungs-ID | Anforderung |
|---|---|
| PROJ_A.1 | Unsere Webapplikation muss ohne gewinn einbringende Methode wie eine Subscription oder anderen Methoden benutzt werden können. Somit kann der Benutzer eine möglichst Kostenfreie Lösung von Flocker verwenden und wir machen keinen Gewinn durch diese Lösung.<br><br>Dies soll zu guten der Kunden kommen, welche das Produkt ohne problematischen Lizenzierungsprobleme oder anderen Umstände verwenden können. |
| PROJ_A.2 | Die Webseite von Flocker soll benutzerfreundlich aufgebaut sein. Ohne unnötige Funktionen oder ablenkendem Design. Die Webseite soll minimalistisch und modern gestaltet werden damit die Übersichtlichkeit der Webseite einfach zu gewinnen ist. |
| PROJ_A.3 | Alle VMs, Systeme, Daten und so weiter, sind ausschliesslich in der Schweiz und auch nur in der Schweiz verfügbar. Damit können wir die Datenschutz Regelungen von der Schweiz erfüllen und den Nutzer eine möglichst sichere und gute Lösung bieten. |
| PROJ_A.4 | Der Raspberry Pi wird von David Reymond zu Verfügung gestellt und der Computer wird von Noah Isenschmid zu Verfügung gestellt. Das ganze Projekt wird von einer Räumlichkeit von jemandem zuhause gehostet und bereitgestellt. Somit können wir nebst den Schulzeiten am Projekt weiterarbeiten und wir müssen die Netzwerkeinstellungen nicht ändern. |
| PROJ_A.5 | Wir bilden uns im Knowhow in der Applikationsentwicklung weiter damit wir alle das Projekt gut realisieren können. Dieses Wissen hilft uns auch bei Anpassung des Projektes oder zukünftige Projekte. |

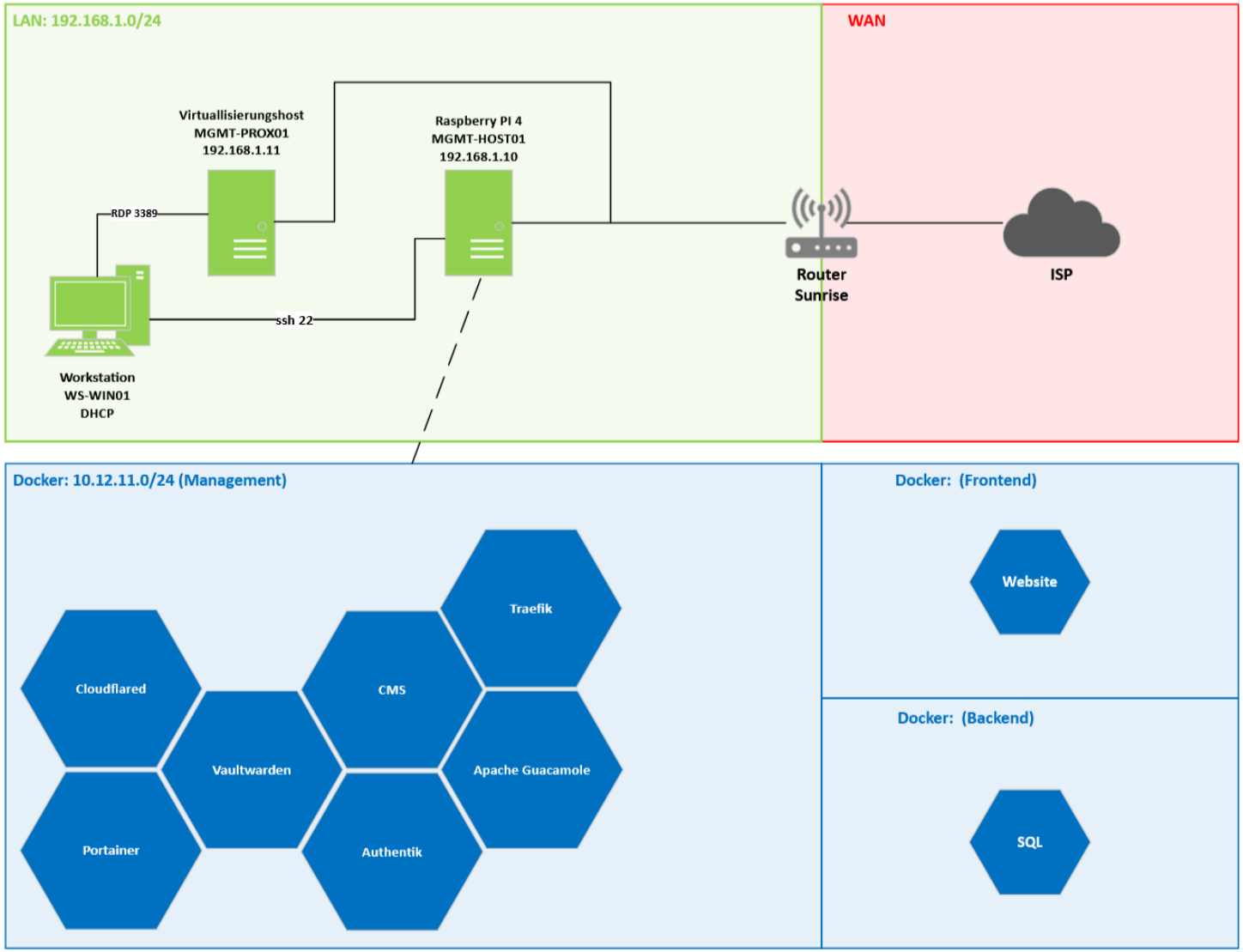
### 4.2.2 Anforderungen an die Informationssicherheit und den Datenschutz

Um die Anforderungen and die Informationssicherheit und den Datenschutz zu gewähren, müssen wir zuerst festlegen, welche Anforderungen abgedeckt werden müssen. Nachfolgend sind die Anforderungen, welche relevant bzw. essenziell für eine saubere Umsetzung sind.

| Anforderungs-ID | Anforderung |
|---|---|
| ISDS_A.1 | Eine Authentifizierungsstufe, um Zugriff auf das System zu erhalten. Folgende Optionen stehen zur Verfügung.<br><br>- Authentifizierung via Benutzername und Passwort<br>- Authentifizierung via OTP oder 2FA |
| ISDS_A.2 | Kein direkter Zugriff von aussen auf den Fileserver und die Datenbank mit den Zugangsdaten. Dies soll so verwirklicht werden, indem dass wir keinerlei Verbindungen auf unserem Reverse Proxy machen und unsere Datenbank so weit wie möglich vom Rest des Systems abschotten und nur die nötigen Ports geöffnet haben. |
| ISDS_A.3 | Administrationsrechte für die Zugriffs- und Sicherheitsinfrastruktur nicht an alle Projektmitglieder, sondern nur den Sicherheitsverantwortlichen. Es soll auch eine interne Benutzerberechtigungsmatrix bestehen. |
| ISDS_A.4 | Zugangsdaten von Benutzern für unsere Cloud einer sicheren Datenbank abgelegt. Dies bedeutet, dass die Datenbank nur für den Datenbankverantwortlichen zur Verfügung gestellt wird. Backups der |

| | |
|---|---|
| | Datenbank werden von den anderen Backups getrennt, auf einem separaten Server gespeichert. |
| ISDS_A.5 | Systemdokumentation und sonstige Dokumente enthalten keine Passwörter. Diese werden in einem eigengehosteten Passwortmanager sicher gespeichert. |
| ISDS_A.6 | Interne Passwortrichtlinien, welche definieren, wie die Passwörter konzipiert werden müssen. Hier ist eine grobe Vorstellung wie diese aussehen könnte.<br>- Mind. 16 Zeichen<br>- Mind. 2 Sonderzeichen<br>- Mind. 2 numerische Zeichen<br>- Gross- & Kleinschreibung |

# 4.3 Systemarchitektur




Version: 0.2
Datum: 25.03.2024
Ersteller: Reymond David
Projekt: flocker

### 4.3.1 Gliederung der Lösung in Module

| Modul | Name | Komponente |
|---|---|---|
| module1 | Physisch | Raspberry Pi 4, Virtualisierungsmaschine, Zugriffscomputer WIN11, Home-Router |
| module2 | Virtualisierung | Proxmox, Apache Guacamole |
| module3 | Security & Access | Bitwarden, Cloudflared, Traefik, Authentik |
| module4 | Management | Docker, Portainer |
| module5 | Web | CMS, Frontend (Website), Backend (SQL) |

Unser Projekt wird in 5 übersichtlichen Modules eingeteilt. Diese sind obenstehend definiert. Wir haben uns absichtlich auf eine sehr spezifische Struktur entschieden, so dass wir alles in einzelne Abschnitte unterteilen können und die einzelnen Komponenten besser verstehen und umsetzen können.
Untenstehend werden diese grafisch mit einem Netzwerkplan dargestellt.

### 4.3.2 Schnittstellen

| Bez. | Schnittstelle | Beschreibung |
|---|---|---|
| E-SS1 | WAN – flocker | Dies ist die Hauptschnittstelle zwischen dem Internet und unserem Projekt – flocker. Diese Schnittstelle macht gebrauch vom Modul 3. Eine Anfrage läuft folgendermassen ab. Ein Benutzer ruft im Internet die Domain https://flocker.cloud auf. Anschliessend wird er von Cloudflared zu unserem Raspberry Pi geschickt und kommt zum Traefik Reverse Proxy. Dieser leitet dann die Anfrage an Authentik weiter, welcher die 2FA Abfrage macht. Sobald diese Abfrage erfolgreich war, wird der Internetbenutzer an den eigentlichen Dienst weitergeleitet.<br><br>Datenfluss: Zugriff auf Internetdienste, Remote-Zugriff, API-Aufrufe<br>Konfiguration: Cloudflared Zero-Trust Tunnel, Traefik, Authentik |
| I-SS1 | PT – module2 | Dieser Zugriff ist das Herzstück der Verwaltung. Dies zeigt die Schnittstelle zwischen dem Projektteam und dem module2, also dem Virtualisierungsmodul auf. Mit dieser Verbindung stellen wir die Integrität, Funktionalität und Gegebenheit für Wartung und Verwaltung sicher. Diese Schnittstelle funktioniert anhand von RDP via TCP-Port 3389. Wir greifen von unseren Arbeitsgeräten aus dem internen Netz darauf zu.<br><br>Datenfluss: Konfigurationen, Updates, Remote Desktop Zugriffe<br>Konfiguration : RDP via TCP-Port 3389 |
| I-SS2 | PT – module3 | Um auf die Zugriffs- und Sicherheitsstruktur zugreifen zu können verwenden wir das module4. Auf dem module4 laufen Docker und Portainer. Portainer wird verwendet, um die Dienste im internen Netz anhand von gewünschten Ports verfügbar zu machen. Auch wird die Docker Verwaltung mit Portainer definitiv vereinfacht und kann so viele Arbeitsschritte einsparen und die Effizienz steigern.<br><br>Datenfluss: Konfigurationen, Updates, Verwaltungsbefehle, API-Aufrufe, Testing, Überwachung und Monitoring<br>Konfiguration: Zugriff via Portainer. Von Portainer und Traefik via Portfreigaben auf spezifischen Dienst per Webzugriff. |
| I-SS3 | module2 – module4 | Diese Schnittstelle dient dazu, die Aktualität von unserer Virtualisierungssoftware auf Schritt zu halten. So kann sich Proxmox mit Image-Repository-Zugriffen immer die neusten Updates, welche durch uns freigegeben werden, holen und so auf einem neuen, validierten und getesteten Stand sein.<br><br>Datenfluss: Image-Repository-Zugriffe, Updates, Managementbefehle, Zugrifftraffic |

| | | |
|---|---|---|
| | | Konfiguration: Docker-Netzwerkbridge, Portfreigabe durch Portainer/Traefik |
| I-SS4 | module4 – module5 | Sodass die Website einfach und effizient bearbeitet, gewartet und angepasst werden kann. Werden wir die Verwaltung und die Runtime des CMS auf unserer Docker Instanz betreiben. Das CMS stellt die Website für flocker bereit. Somit ist dies das Aushängeschild für unser Produkt und sollte unter allen Umständen eine möglichst kleine, wenn nicht sogar keine Ausfallzeit haben. Hierbei können wir mit Docker entgegenwirken, da falls einmal ein Problem bestehen sollte, können wir ganz einfach den Container neu starten oder den Container aus einem Backup wiederherstellen.<br><br>Datenfluss: Backend-Datenbankzugriffe, Frontend-Anfragen, CMS-Daten (Inhalt der Website)<br>Konfiguration: Docker-Compose für Service-Orchestrierung, Umgebungsvariablen für Datenbankverbindungen, Portfreigaben für Webzugriffe |
| I-SS5 | module3 | Das module3 ist eine der Mutterschnitstellen zwischen den Diensten, da dies jede Anfrage verarbeitet, validiert, authentifiziert und autorisiert. Falls eine bestimmte Anfrage von aussen oder innen erfolgt, wird diese zwangsläufig durch Traefik laufen und bei einem Zugriff auf einen bestimmten Dienst mit einer Abfrage von Authentik gekoppelt.<br><br>Datenfluss: Authentifizierungs- & Autorisierungsanfragen, Sicherheits-Token, Zugriffsberechtigungen<br>Konfiguration: API-Gateways über Traefik, OAuth2.0/OIDC-Integration über Authentik, Zugriffskonfigurationen, Secure Zero-Trust Tunnels über Cloudflared |
| I-SS6 | module4 | Das module4 ist in unserem System die sogenannte Grossmutterschnittstelle, da dies die Schnittstelle ist, auf welcher alle Dienste von module3 laufen. Dies ist eine komplexe Konfiguration und darf daher nur von den zuständigen Personen verändert oder angepasst werden. Dies ist die Schnittstelle zwischen dem Projektteam und dem module3. Alle Konfigurationsanpassungen finden hier statt.<br><br>Datenfluss: Container-Monitoring/Status, Service-Konfigurationen, Update-Management.<br>Konfiguration: Portainer-Webzugriff, API-Integrationen für Docker, Sicherheitsrichtlinien für Management-Zugriffe |

## 4.4 Testkonzept

| Nr. | Testfall | Testbeschreibung | Testschritte | Erwartetes Ergebnis |
|---|---|---|---|---|
| 1 | Funktionalität ohne kostenpflichtige Methoden | Überprüfen der Verfügbarkeit und Funktionalität der Flocker Webapplikation ohne die Notwendigkeit kostenpflichtiger Abonnements oder ähnlicher Zahlungsmethoden. | 1. Öffnen der Flocker Webapplikation.<br>2. Versuch, auf alle Hauptfunktionen zuzugreifen, ohne Zahlungsinformationen einzugeben.<br>3. Durchführung typischer Aktionen wie Dateiupload, Datenbearbeitung und Datenspeicherung.<br>4. Überprüfung, ob sämtliche Funktionen ohne die Notwendigkeit von Zahlungsinformationen zugänglich sind.<br>5. Abschluss einer typischen Sitzung ohne Eingabe von Zahlungsinformationen. | Alle Hauptfunktionen der Webapplikation sind ohne die Eingabe von Zahlungsinformationen zugänglich und voll funktionsfähig. |
| 2 | Benutzerfreundlichkeit der Webseite | Überprüfen der Benutzerfreundlichkeit und des Designs der Flocker Webseite. | 1. Bewertung der Startseite und des Navigationsmenüs auf Einfachheit und Klarheit.<br>2. Überprüfung der Anordnung von Schaltflächen und Funktionen auf Benutzerfreundlichkeit.<br>3. Testen der Suchfunktion, um zu sehen, wie leicht Benutzer relevante Informationen finden können.<br>4. Durchführung von typischen Aktionen gemäß den Mockups, um sicherzustellen, dass die tatsächliche Nutzung dem geplanten Design entspricht. | Die Webseite ist leicht navigierbar, weist ein minimalistisches und modernes Design auf und bietet eine intuitive Benutzererfahrung. |

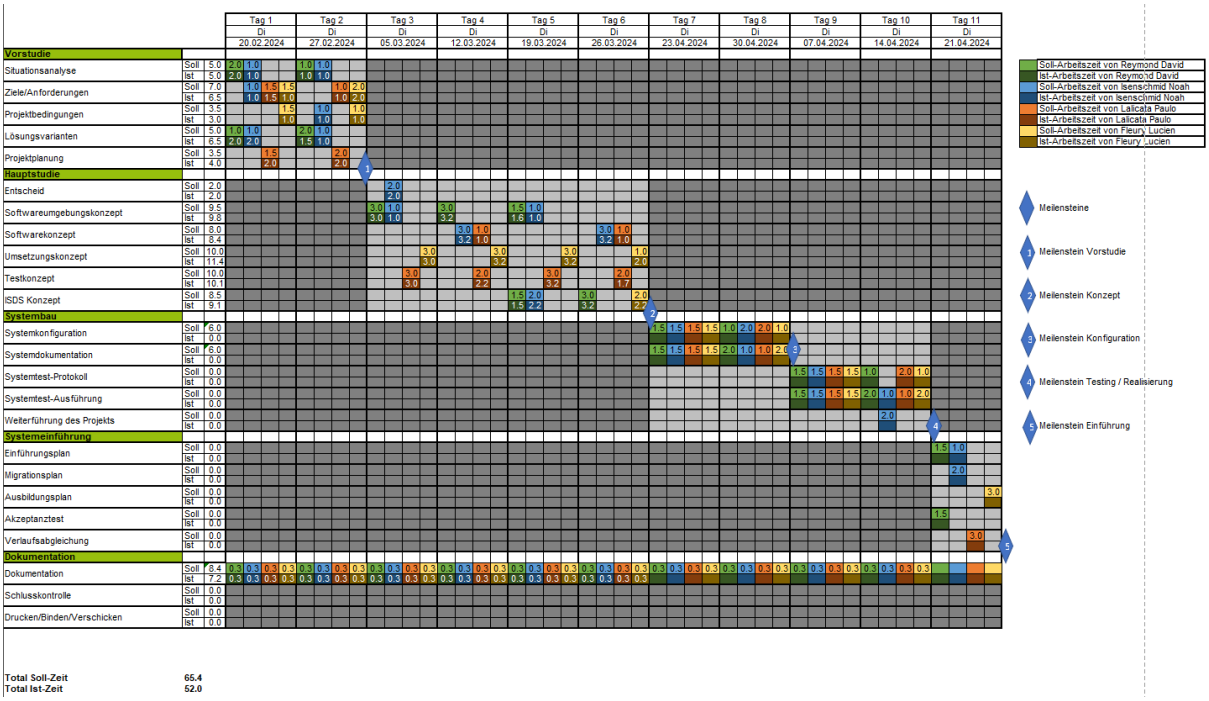
| | | | | |
|---|---|---|---|---|
| 3 | Sicherheit und Lokalität der Daten | Überprüfen, ob sämtliche Daten und Systeme der Flocker Webapplikation ausschließlich in der Schweiz gehostet werden und die Datenschutzbestimmungen eingehalten werden. | 1. Überprüfung der Standortangaben der Server und Datenbanken.<br>2. Verfolgung der Datenflüsse und Überwachung der Netzwerkkommunikation.<br>3. Sicherstellung, dass keine Daten außerhalb der Schweiz gespeichert oder übertragen werden.<br>4. Überprüfung der Datenschutzrichtlinien, um den Standort der Datenspeicherung zu bestätigen. | Alle Daten und Systeme sind ausschließlich in der Schweiz gehostet, um den Datenschutzanforderungen zu genügen. |
| 4 | Leistung und Verfügbarkeit der Hardware | Überprüfen der Leistung und Verfügbarkeit der bereitgestellten Hardware für die Flocker Webapplikation. | 1. Überprüfung der Hardware-Spezifikationen auf Konformität mit den Anforderungen.<br>2. Testen der Server- und Netzwerkverfügbarkeit nach der Installation.<br>3. Durchführung von Leistungstests, um sicherzustellen, dass die Hardware den Anforderungen der Webapplikation entspricht.<br>4. Überprüfung der Sicherheitsvorkehrungen und Zugangsbeschränkungen für die gehostete Hardware. | Die bereitgestellte Hardware ist korrekt konfiguriert, verfügbar und erfüllt die Leistungsanforderungen der Webapplikation. |
| 5 | Wissensstand und Kompetenz des Teams | Überprüfen, ob das Entwicklungsteam über das erforderliche Wissen und Know-how für die Entwicklung, Anpassung und Wartung der Flocker Webapplikation verfügt. | 1. Bewertung der Qualifikationen und Erfahrungen der Teammitglieder im Bereich der Webentwicklung.<br>2. Durchführung von Code-Reviews und Überprüfung der Entwicklungsumgebung. | Das Entwicklungsteam verfügt über das erforderliche Wissen und Know-how für die Entwicklung, Anpassung und Wartung der Webapplikation. |

| | | | | |
|---|---|---|---|---|
| | | | 3. Testen der Teamkommunikation und Zusammenarbeit bei der Lösung von Problemen.<br>4. Überprüfung der Dokumentation und Schulungsunterlagen zur Sicherstellung einer kontinuierlichen Wissensvermittlung. | |
| 6 | Integration der Benutzerschnittstelle (UI) mit Backend-Services | Überprüfen der nahtlosen Integration der Benutzerschnittstelle mit den Backend-Services. | 1. Durchführung von typischen Aktionen über die Benutzeroberfläche, z. B. Dateiupload oder Datenbearbeitung.<br>2. Überwachung der Netzwerkkommunikation zwischen der Benutzerschnittstelle und den Backend-Services.<br>3. Überprüfung der Datenkonsistenz und -integrität nach jeder Aktion.<br>4. Testen von Fehlerszenarien wie Netzwerkausfällen oder unerwarteten Serverantworten. | Die Benutzerschnittstelle integriert sich nahtlos mit den Backend-Services, und Daten werden korrekt zwischen den Systemen ausgetauscht. |
| **7** | API-Funktionalität und -Sicherheit | Überprüfen der Funktionalität und Sicherheit der APIs, die von der Flocker Webapplikation bereitgestellt werden. | 1. Durchführung von API-Anfragen gemäß der Spezifikation.<br>2. Überprüfung der Antwortzeiten und Datenintegrität.<br>3. Testen von Authentifizierungs- und Autorisierungsmethoden, z. B. API-Schlüssel oder OAuth.<br>4. Durchführung von Sicherheitstests wie Injection-Angriffen oder Brute-Force-Angriffen. | Die APIs funktionieren wie erwartet, sind sicher und bieten angemessene Authentifizierungs- und Autorisierungsmethoden. |

| | | | | |
|---|---|---|---|---|
| **8** | Datenbankintegration und -zugriff | Überprüfen der Integration und des Zugriffs auf die Datenbanken der Flocker Webapplikation. | 1. Durchführung von Datenzugriffsoperationen über die Benutzeroberfläche und die API.<br>2. Überprüfung der Datenkonsistenz zwischen Frontend und backend.<br>3. Testen von Datenbankfunktionen wie Transaktionen und Indizierungen.<br>4. Durchführung von Skalierungstests, um die Leistung der Datenbank unter Last zu überprüfen. | Die Datenbankintegration ist stabil, und Daten werden korrekt zwischen der Benutzerschnittstelle und dem Backend ausgetauscht. |

# 4.5 Weiterführung der Projektplanung

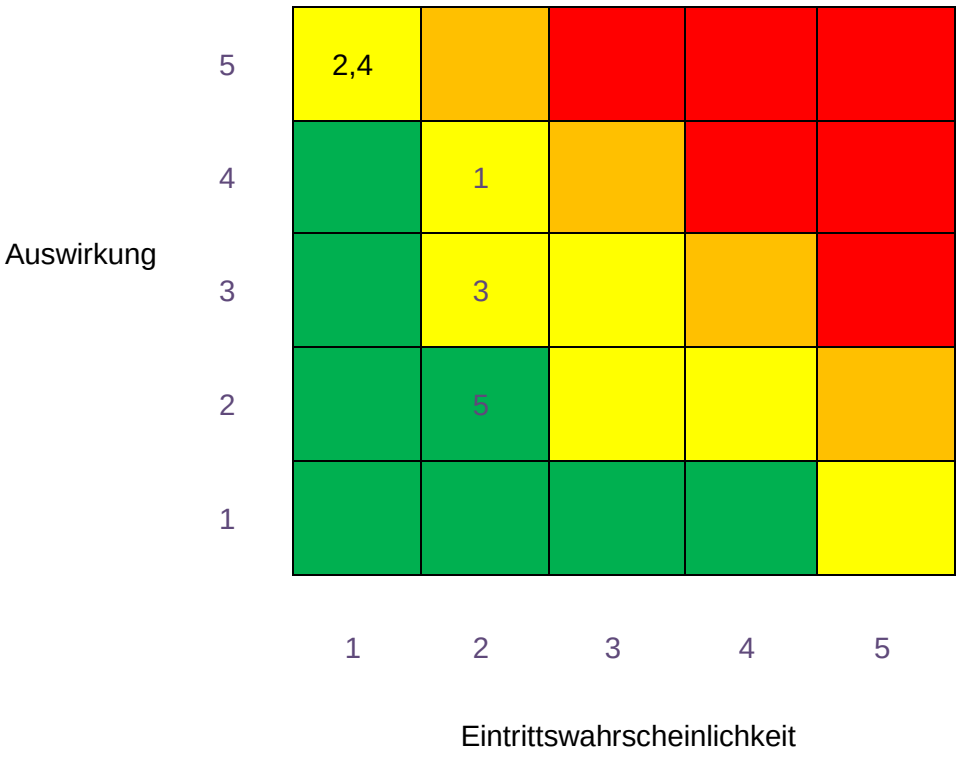
## 4.5.1 Abgleich von Planung und Verlauf des Konzepts




Die Terminplanung der Konzeptphase verlief termingerecht und wir hatten keine grossen Abweichungen. Die Planung für die Realisierungsphase ist mit der Annahme, dass wir 4 Wochen Zeit haben, gestaltet worden. Die Idee ist es, dass das ganze Projektteam bei der Systemkonfiguration mitmacht. Das kann etwas unübersichtlich werden, jedoch sehen wir keine bessere Alternative, da gewisse Aspekte der Konfiguration recht aufwendig werden. Werden wir für die Realisierungsphase weniger Zeit zur Verfügung haben, kann sich die Planung noch ändern.

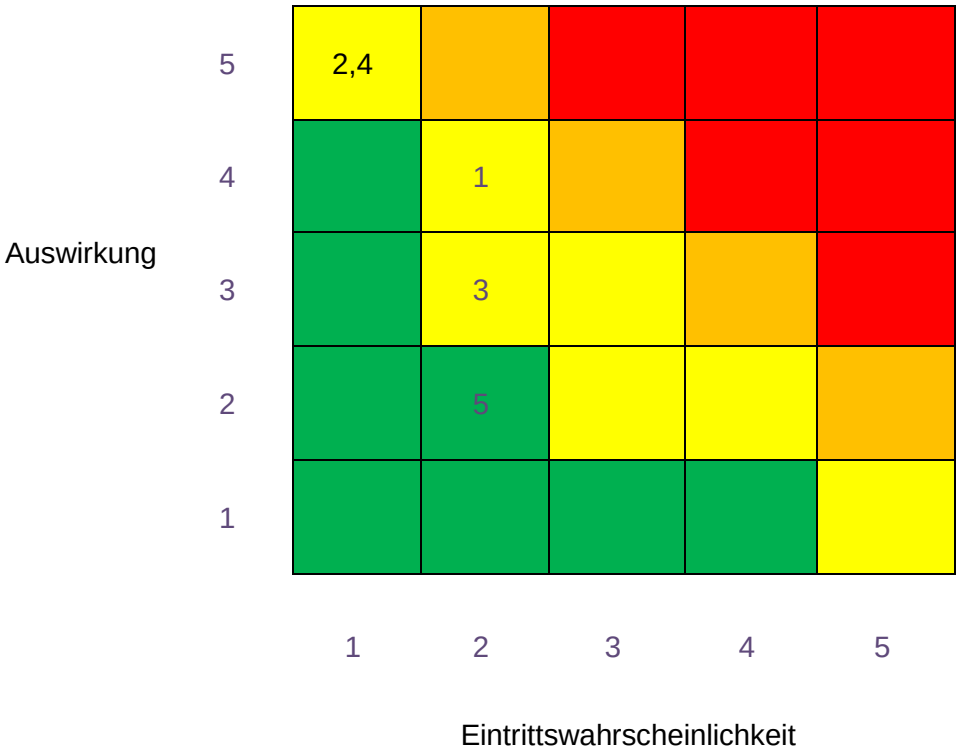
## 4.5.2 Aktualisierung der Risikosituation

### 4.5.2.1 Vorher




| Nr. | Risiko | Beschreibung | Massnahme | Einschätzung |
|---|---|---|---|---|
| 1 | Funktionsfehler | Wenn der Containerdienst abstürzt, laufen die Virtuellen Maschinen nicht weiter. | Supportbereitschaft | 4,2 |
| 2 | Hardwarefehler | Wenn unsere Hardware (Raspberry Pi) nicht funktioniert. | Ersatzgerät mit Cluster. | 5,1 |
| 3 | VM-Absturz | Wenn die Virtuelle Maschinen abstürzen, laufen die Dienste und Anwendungen nicht weiter. | Snapshots | 3,2 |
| 4 | Stromausfall | Da wir keine Notlösung für Stromausfälle haben laufen unsere Systeme ohne Strom nicht mehr. | USV installieren. | 5,1 |
| 5 | Dienstabsturz | Wenn ein einzelner Dienst abstürzt. | Monitoring einrichten mit automatisiertem Alerting. | 2,2 |

#### 4.5.2.2 Aktuell




| Nr. | Risiko | Beschreibung | Massnahme | Einschätzung |
|---|---|---|---|---|
| 1 | Funktionsfehler | Wenn der Containerdienst abstürzt, laufen die Virtuellen Maschinen nicht weiter. | Supportbereitschaft | 4,2 |
| 2 | Hardwarefehler | Wenn unsere Hardware (Raspberry Pi) nicht funktioniert. | Ersatzgerät mit Cluster. | 5,1 |
| 3 | VM-Absturz | Wenn die Virtuelle Maschinen abstürzen, laufen die Dienste und Anwendungen nicht weiter. | Snapshots | 3,2 |
| 4 | Stromausfall | Da wir keine Notlösung für Stromausfälle haben laufen unsere Systeme ohne Strom nicht mehr. | USV installieren. | 5,1 |
| 5 | Dienstabsturz | Wenn ein einzelner Dienst abstürzt. | Monitoring einrichten mit automatisiertem Alerting. | 2,2 |

#### 4.5.2.3 Auswertung

Wie oben im Vergleich erkennbar ist, gibt es bei uns keine Änderungen im Risikomanagement. Alle Risiken, welche momentan bestehen sind jedoch nicht verherrlich, da wir für fast jeden Fall eine optimale Lösung bereitstellen können. Leider können wir dies aus Budget Problemen nicht mit einer sauberen Lösung (USV) umsetzen. Wir sehen jedoch einen Stromausfall als sehr unwahrscheinlich an, da wir in einer sehr sicheren Umgebung unseren Strom beziehen

## 4.5.3 DNS

```
$ORIGIN flocker.cloud.
$TTL 86400
@               IN SOA   (
                    flocker.cloud                ; MNAME
                    hostmaster@flocker.cloud ; RNAME
                    2024031301                   ; SERIAL
                    10800                        ; REFRESH
                    3600                         ; RETRY
                    1209600                      ; EXPIRE
                    10800                        ; MINIMUM
                    )

; NS RECORDS
@                   IN NS    maisie.ns.cloudflare.com.
@                   IN NS    yadiel.ns.cloudflare.com.

; A RECORDS
@                   IN A     212.103.88.223

; CNAME RECORDS
ftp                 IN CNAME flocker.cloud.
www                         IN CNAME flocker.cloud.

; MX RECORDS
@                   3600 IN MX    0 mailgate.computech.ch.

; TXT RECORDS
@                   3600 IN TXT   "v=spf1 include:spf.computech.ch -all"
_dmarc              IN TXT   "v=DMARC1; p=quarantine; adkim=s; aspf=s"
_domainconnect      IN TXT  (
                     "domainconnect.plesk.com/host/web103.computech-rz.ch/port/8443"
                    )
```

#### 4.5.3.1 Beschreibung DNS Records

Obenstehend sind alle DNS Records aufgeführt. Um diese verstehen zu können haben wir ein untenstehend eine Übersicht angefertigt, welche alle DNS Records besteht.

| Kl. | Eintrag | Beschreibung |
|---|---|---|
| SOA | @ | Der DNS-SOA-Eintrag (Start of authority) speichert wichtige Informationen über eine Domain oder Zone, z. B. die E-Mail-Adresse des Administrators, wann die Domain zuletzt aktualisiert wurde und wie lange der Server zwischen den Aktualisierungen warten sollte. |
| NS | ns.cloudflare.com | NS steht für „Nameserver“ und der Nameserver-Eintrag gibt an, welcher DNS-Server der autoritative Server für die betreffende Domain ist (also welcher Server die eigentlichen DNS-Einträge enthält). Im Grunde zeigen NS-Einträge dem Internet, wo die IP-Adresse einer Domain zu finden ist. In unserem Fall sind wir bei Cloudflare gehostet, deswegen zeigen diese Einträge auf Cloudflare. |
| A | 212.103.88.223 | Dies ist die Public-IP-Adresse, wo die aktuelle Website läuft. Da dies noch die Default-Webpage ist, läuft hier das Plesk von CompuTech. Dies wird sich im Verlauf der Realisierung ändern. |
| CNAME | www | Dies ist ein CNAME Eintrag, sodass unsere Website auch per www.flocker.cloud erreicht werden kann. Diese Subdomain signalisiert auch, dass unserer Website vom Internet erreichbar ist. |
| | ftp | Dies ist die ftp Verbindung, welche wir auf unsere Infrastruktur haben. Diese ermöglicht es uns diverse Anpassungen im Hosting Panel zu machen. |
| MX | mailgate.computech.ch | MX steht für «Mail Exchange» dies sind die Einträge, welche bestimmen von welchem Mailserver die Domain Mails versenden kann. Da ein Eigenhosting der Mailserver den Rahmen dieses Projekts sprengen würde, haben wir uns dazu entschieden, die ganze Mailing Geschichte auf den CompuTech Mailservern zu machen. |
| TXT | spf | Ein SPF-Eintrag (Sender Policy Framework) ist eine Art DNS-TXT-Eintrag, der alle Server auflistet, die berechtigt sind, E-Mails von einer bestimmten Domäne zu versenden. Mit einem DNS-TXT-Eintrag („Text“) kann ein Domain-Administrator einen beliebigen Text in das Domain Name System (DNS) eingeben. Da nur die CompuTech berechtigt ist, Mails zu versenden, sind auch nur sie aufgeführt. |
| | _dmarc | Mit einer DMARC-Richtlinie wird Empfängerservern mitgeteilt, welche Aktionen sie bei nicht authentifizierten Nachrichten ausführen sollen, die sie von Ihrer Domain erhalten. Dies wird im Tag «P» angezeigt. In unserem Fall wollen wir, dass die Mails, welche nicht vertrauenswürdig sind in die Quarantäne gestellt werden. |

| | | |
|---|---|---|
| | _domainconnect | Dies ist der Domainconnector von Plesk. Auf dem Plesk von CompuTech ist das Mailing. Dies ist ein Erkenner, dass Plesk weiss, dass dies eine vertrauenswürdige Domain ist und sich damit verbinden darf. |

Wichtig zu beachten ist, dass unsere Website Zugriffe über einen Cloudflare Zero-Trust Argo Tunnel funktionieren.

### 4.5.4 Erklärung Infrastruktur Komponente

Benötigte Hard- und Software

- Raspberry Pi 4 Model B
  - Cortex-A72
  - 8 GB RAM
  - 512 GB SSD
- Physical Computer
  - i9-9900K
  - 32 GB RAM
  - GTX1660Ti
  - 2048 GB NVMe SSD
- Docker
  - OpenSource
  - Containerisierungssoftware
- Portainer
  - Docker Verwaltungssoftware
  - Community Edition
- Apache Guacamole
  - Virtualisierungssoftware
  - OpenSource
- Cloudflared
  - Reverse Proxy
  - DNS Server
- CMS
  - Wordpress

#### 4.5.4.1 Beschrieb und Verwendung Komponente

Als Managementhost verwenden wir einen Raspberry Pi. Dieser ist kostengünstig und leistungsstark. Damit können wir unsere komplette Infrastruktur verwalten, monitoren und überwachen. Wir haben uns für einen Managementhost entschieden, da dieser eine grosse Flexibilität bietet. So können wir von einem zentralen Punkt auf alle notwendigen Systeme zugreifen. Auch im Bereich Updates ist ein Managementhost von Vorteil, da man damit einen zentralen Verteiler bereitstellen kann.
Als Virtualisierungshost verwenden wir einen Physischen Computer mit folgenden Spezifikationen.

| Komponente | Artikel |
|---|---|
| Prozessor (CPU) | i9-9900K |
| Arbeitsspeicher (RAM) | 32 GB DDR4 |
| Grafikkarte (GPU) | Nvidia GeForce GTX 1660 Ti |
| Speicher | 2048 GB NVMe M.2 SSD |
| Netzwerkkarte | 2x 2.5 Gbit RJ45 |

*Tabelle 1 Verwendete Komponente*

Diese Konfiguration ist ausreichend, um zu allen Zeiten eine performante und stabile Verbindung für uns relevante Zwecke zu bieten.
Wir verwenden Docker als Schnittstelle zwischen unseren Plattformen. Die Verwendung von Docker als Schnittstelle zwischen unseren Plattformen bietet uns viele Vorteilen, die sowohl die Entwicklung als auch den Betrieb von Anwendungen vereinfachen und verbessern. Docker ermöglicht die Containerisierung von Anwendungen und deren Abhängigkeiten in Container, die dann leicht zwischen verschiedenen Umgebungen transportiert und ausgeführt werden können.
Da Docker auf unserem Raspberry Pi Server auf einem Ubuntu Server betrieben wird und wir dadurch kein GUI haben, haben wir entschieden, dass wir eine Verwaltungssoftware verwenden werden – Portainer. Portainer ist eine Software, mit welcher es möglich wird, diverse Prozesse in der Konzeption und Umsetzung in einem einfach zu bedienendem GUI, zu vereinfachen.
Apache Guacamole ist unsere Virtualisierungssoftware. Diese haben wir gewählt, da Sie OpenSource ist, dadurch eine grosse Knowledge Base und Community hat, einfach und benutzerfreundlich zu betreiben, Möglichkeiten für eine erweiterte Sicherheitsimplementierung und eine hohe Skalierbarkeit bietet.
Um unser Vorhaben über eine Webapp übers Internet erreichbar zu machen, müssen wir einen Zugang ins Internet haben. Wir haben uns hier für Cloudflare entschieden. Cloudflare ist ein zuverlässiger Provider für DNS-Hostings und vielem mehr. Wir werden über Cloudflare den Reverse Proxy und den DNS-Dienst verwenden, um ein Routing von unserem Raspberry Pi ins Internet zu machen.
Wir haben uns nun für das CMS Wordpress entschieden. Es ist einfach so eine Website bereitzustellen und diese je nach Belieben zu bearbeiten. Vorteil davon ist, dass wir uns nicht allzufest auf die Webentwicklung fokusieren müssen, sondern unseren Fokus auf das Backend und die Funktionen der Webapplikation wenden können.

#### 4.5.4.2 Anforderung an die Infrastruktur

Ein grosser Punkt ist, dass die Infrastruktur sich in der Schweiz befindet, dass erfüllt sie auch, denn sie wird On-Prem bei uns zuhause aufgesetzt.
Die Infrastruktur ist sicher von verschiedenen Angriffen und wird kontinuierlich getestet. Auf die Infrastruktur kann die ganze Zeit zugegriffen werden im Verlauf des Projektes. Es darf zu keinen Ausfällen während dem Projekt kommen.
Der ganze Unterhalt der Infrastruktur muss kostengünstig bleiben im Unterhalt, da wir durch flocker keine Einnahmen generieren.

# 4.6 ISDS-Konzept

## 4.6.1 Einleitung

#### 4.6.1.1 Zweck des Dokuments

Das ISDS-Konzept legt die nötigen Angaben zur Erhaltung und Verbesserung der Informationssicherheit und des Datenschutzes fest. Es fasst die Aspekte der Informationssicherheit und des Datenschutzes im Projekt zusammen.

#### 4.6.1.2 Handhabung des Dokuments

Dieses Dokument ist vertraulich zu behandeln. Es ist untersagt an Dritte und oder Unberechtigte weiterzugeben. Dieses Dokument muss an einem Ort abgelegt werden, auf welchen nur berechtigte Personen Zugriff haben.

#### 4.6.1.3 Allgemeines

Wir verfolgen eine strikte und transparente Informationspolitik und arbeiten ausschliesslich mit Schweizer Unternehmen und Dienstleistern zusammen. Die gesamte Entwicklung, Betrieb, Support, Hosting und Backups des flocker Projektes befinden sich in der Schweiz und werden ausschliesslich aus der Schweiz betreut.

## 4.6.2 Verzeichnis der sicherheitsrelevanten Dokumente

| Dokumententyp | Titel |
|---|---|
| **Gesetze** | https://www.fedlex.admin.ch/eli/cc/1993/1945_1945_1945/de<br>https://www.fedlex.admin.ch/eli/cc/1993/1962_1962_1962/de<br>https://www.fedlex.admin.ch/eli/fga/2020/2696/de |
| **Übergeordnete Sicherheitskonzepte** | Sicherheitskonzepte oder Datenschutzbedingungen des Projektes flocker. |

## 4.6.3 Schützenswerte Daten

Die Daten, welche bei flocker schützenswert sind, sind folgende:

- Benutzerdaten (Benutzername, E-Mail, Passwort) von den End Usern
- Administrator Benutzerdaten (Benutzername, Passwort)
- Authenticator Hashes

#### 4.6.3.1 Expositionsanalyse

Die Daten, welche bei flocker schützenswert sind, sind folgende:

- Benutzerdaten (Benutzername, E-Mail, Passwort) von den End Usern
- Administrator Benutzerdaten (Benutzername, Passwort)
- Authenticator Hashes

#### 4.6.3.2 Sicherheitsmassnahmen

| Information / Asset | Sicherheitsmassnahmen |
|---|---|
| Benutzerdaten User | Passwort Validation, Passwort Hashing, Passwortrichtlinien setzen, Benutzer Validation. |
| Benutzerdaten Administratoren | Passwörter sicher aufbewahren, |
| Benutzerdaten User Speicherung | Die Benutzerdaten werden in einer SQL-Datenbank gespeichert. Damit die Integrität, Availability und Confidentiality gewährleistet ist werden folgende Massnahmen getroffen:<br>- Parametrisierte Datenbankabfragen<br>- Monitor und Audit DB Aktivitäten<br>- DB-Software Up to Date halten,<br>- Encrypt Data an REST und Transit |

### 4.6.4 Netzwerksicherheit gewährleisten

#### 4.6.4.1 Zugriff auf Infrastruktur beschränken

| Beschränkungstyp | Beschreibung |
|---|---|
| **Physischer Zugang** | Da die Hardware in einem Gebäude ist kann keine Fremde Person ohne Schlüssel auf die Hardware zugreifen. |
| **Digitaler Zugang** | Unsere Server sind entweder gar nicht online verfügbar und wenn, dann nur mit einem Login |

#### 4.6.4.2 Erreichbarkeit übers Internet

| Dienste | Beschreibung |
|---|---|
| **Webserver** | Damit unsere Webseite im Internet sichtbar ist muss unser Webserver vom Internet erreichbar sein. |

### 4.6.5 Zertifikatsmanagement für Webserver SSL/TLS

#### 4.6.5.1 Ausstellung der Zertifikate

| Dienst | Link |
|---|---|
| **LetsEncrypt** | https://letsencrypt.org/ |

### 4.6.6 Potenzielle Risiken und Massnahmen

| Gefahr | Massnahme |
|---|---|
| **XSS Cross-site Scripting** | Filter Input bei Ankunft<br>Encode Data beim Output<br>Passende Response Header verwenden |
| **DDOS** | Real-time, adaptive threat monitoring, caching, Rate limiting. |
| **Unbekannte Schwachstelle in der Infrastruktur** | Gutes Testkonzept, kontinuierliche Tests während dem Aufbau der Infrastruktur |

### 4.6.7 Speicherung der Daten

Wir speichern die Daten der Benutzer, welche sich auf der Webapplikation anmelden. Wir implementieren ein Ablaufskonzept für die Benutzer und Passwörter. Wenn ein Benutzer nach längerer Zeit nicht benutzt wird, wird der Benutzer informiert, dass sein Account deaktiviert und gelöscht wird. Das Passwort kann auch ablaufen, hierbei wird der Benutzer auch informiert, dass er sein Passwort ändern / aktualisieren muss. Somit gewährleisten wir, dass sich keine Daten anhäufen und keine alten Daten vorhanden sind. Somit verringern wir das Risiko und sind effizienter mit dem Datenhandling.

# 5 Realisierung

## 5.1 Zusammenfassung

Wir arbeiten mit Raspberry Pis als Management-Host und einem Physischen Computer, welcher als Virtualisierungshost fungiert. Hierbei wird mit einer einfachen Struktur gearbeitet. Der Raspberry Pi (Managementhost) arbeitet mit Ubuntu Server und Docker. Auf Docker installieren wir die nötige Software und Schnittstellen wie Cloudflared, Bitwarden, Authentika, Guacamole, Portainer u.v.m. Untenstehend werden die einzelnen Dienste beschrieben und aufgezeigt, für was diese verwendet werden.

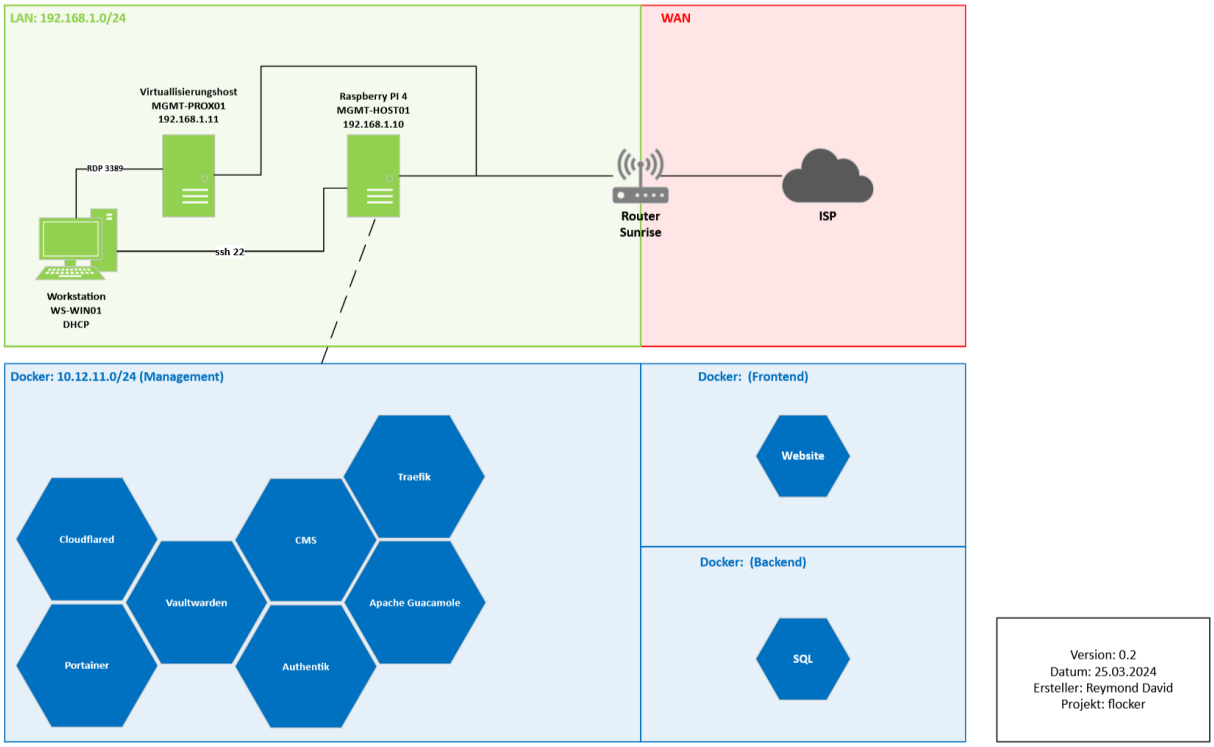
| | |
|---|---|
| Managementhost | Der Raspberry Pi wird als Management (kurz mgmt) -host verwendet und wird mit Ubuntu und Docker betrieben. Dieser ist für die Verwaltung, Wartung und Überwachung zuständig. |
| Cloudflared | Dies ist der Dienst, welcher für die Sicherstellung und Funktionalität des Website Traffics zuständig ist. Cloudflare ist ein zuverlässiger und seriöser DNS-Provider. Wir nehmen den Reverse-Proxy und die DNS-Dienste von Cloudflare in Anspruch. |
| Apache Guacamole | Apache Guacamole ist eine Virtualisierungssoftware, in welcher virtuelle Maschinen/Computer virtualisiert bereitgestellt werden können. Dies ist eine gute Wahl, da diese Software OpenSource ist und daher eine grosse Knowledgebase und Community hat. |
| Authentik | Dies ist ein Dienst, welcher verwendet werden kann, um eine Selfhosted Multifaktorauthentifizierung einzurichten. Da wir uns intensiv mit Security auseinandersetzen wollen, haben wir uns dazu entschieden dies zu implementieren. |

Als Managementhost verwenden wir einen Raspberry Pi. Dieser ist kostengünstig und leistungsstark. Damit können wir unsere komplette Infrastruktur verwalten, monitoren und überwachen. Wir haben uns für einen Managementhost entschieden, da dieser eine grosse Flexibilität bietet. So können wir von einem zentralen Punkt auf alle notwendigen Systeme zugreifen. Auch im Bereich Updates ist ein Managementhost von Vorteil, da man damit einen zentralen Verteiler bereitstellen kann.
Diese Konfiguration ist ausreichend, um zu allen Zeiten eine performante und stabile Verbindung für uns relevante Zwecke zu bieten.
Wir verwenden Docker als Schnittstelle zwischen unseren Plattformen. Die Verwendung von Docker als Schnittstelle zwischen unseren Plattformen bietet uns viele Vorteilen, die sowohl die Entwicklung als auch den Betrieb von Anwendungen vereinfachen und verbessern. Docker ermöglicht die Containerisierung von Anwendungen und deren Abhängigkeiten in Container, die dann leicht zwischen verschiedenen Umgebungen transportiert und ausgeführt werden können.
Da Docker auf unserem Raspberry Pi Server auf einem Ubuntu Server betrieben wird und wir dadurch kein GUI haben, haben wir entschieden, dass wir eine Verwaltungssoftware verwenden werden – Portainer. Portainer ist eine Software, mit welcher es möglich wird, diverse Prozesse in der Konzeption und Umsetzung in einem einfach zu bedienendem GUI, zu vereinfachen.
Apache Guacamole ist unsere Virtualisierungssoftware. Diese haben wir gewählt, da Sie OpenSource ist, dadurch eine grosse Knowledge Base und Community hat, einfach und benutzerfreundlich zu betreiben, Möglichkeiten für eine erweiterte Sicherheitsimplementierung und eine hohe Skalierbarkeit bietet

## 5.2 Technische Detailspezifikation

## 5.2.1 Systemdesign

### 5.2.1.1 Struktur




### 5.2.1.2 Systemspezifikation

### 5.2.1.3 Management Server (Raspberry Pi)

| | |
|---|---|
| **CPU** | Broadcom BCM2711, quad-core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz |
| **Memory** | 8GB LPDDR4 |
| **Connection** | 2.4 GHz and 5.0 GHz IEEE 802.11b/g/n/ac wireless LAN, Bluetooth 5.0, BLE<br>Gigabit Ethernet<br>2 × USB 3.0 ports<br>2 × USB 2.0 ports |
| **GPIO** | Standard 40-pin GPIO header |
| **Video & Sound** | 2 × micro HDMI ports (up to 4Kp60 supported)<br>2-lane MIPI DSI display port<br>2-lane MIPI CSI camera port<br>4-pole stereo audio and composite video port |
| **SD Card Support** | Micro SD card slot for loading operating system and data storage |
| **Input power** | 5V DC via USB-C connector minimum 3A 1<br>5V DC via GPIO header minimum 3A1<br>Power over Ethernet (PoE)–enabled |

| | |
|---|---|
| | (requires separate PoE HAT) |
| **OS** | Linux Ubuntu Server 24.04 LTS |

#### 5.2.1.4 Virtualisierungsserver

| | |
|---|---|
| **CPU** | i9-9900K |
| **Memory** | 32GB |
| **Speicher** | 2048 GB NVMe SSD |
| **Grafikkarte** | Nvidia Geforce GTX 1660Ti |

Den Virtualisierungsserver haben wir nie erhalten, da die Person, welche uns den Server (Virtualisierung Computer) zur Verfügung hätte stellen sollen, in die Ferien gegangen ist, ohne uns Bescheid zu sagen. Somit mussten wir den Proxmox auf eine VM installiere

### 5.2.2 Schnittstellendefinition

| Bez. | Schnittstelle | Beschreibung |
|---|---|---|
| E-SS1 | WAN – flocker | Dies ist die Hauptschnittstelle zwischen dem Internet und unserem Projekt – flocker. Diese Schnittstelle macht gebrauch vom Modul 3. Eine Anfrage läuft folgendermassen ab. Ein Benutzer ruft im Internet die Domain https://flocker.cloud auf. Anschliessend wird er von Cloudflared zu unserem Raspberry Pi geschickt und kommt zum Traefik Reverse Proxy. Dieser leitet dann die Anfrage an Authentik weiter, welcher die 2FA Abfrage macht. Sobald diese Abfrage erfolgreich war, wird der Internetbenutzer an den eigentlichen Dienst weitergeleitet.<br><br>Datenfluss: Zugriff auf Internetdienste, Remote-Zugriff, API-Aufrufe<br>Konfiguration: Cloudflared Zero-Trust Tunnel, Traefik, Authentik |
| I-SS1 | PT – module2 | Dieser Zugriff ist das Herzstück der Verwaltung. Dies zeigt die Schnittstelle zwischen dem Projektteam und dem module2, also dem Virtualisierungsmodul auf. Mit dieser Verbindung stellen wir die Integrität, Funktionalität und Gegebenheit für Wartung und Verwaltung sicher. Diese Schnittstelle funktioniert anhand von RDP via TCP-Port 3389. Wir greifen von unseren Arbeitsgeräten aus dem internen Netz darauf zu.<br><br>Datenfluss: Konfigurationen, Updates, Remote Desktop Zugriffe<br>Konfiguration : RDP via TCP-Port 3389 |
| I-SS2 | PT – module3 | Um auf die Zugriffs- und Sicherheitsstruktur zugreifen zu können verwenden wir das module4. Auf dem module4 laufen Docker und Portainer. Portainer wird verwendet, um die Dienste im internen Netz anhand von gewünschten Ports verfügbar zu machen. Auch wird die Docker Verwaltung mit Portainer definitiv vereinfacht und kann so viele Arbeitsschritte einsparen und die Effizienz steigern.<br><br>Datenfluss: Konfigurationen, Updates, Verwaltungsbefehle, API-Aufrufe, Testing, Überwachung und Monitoring<br>Konfiguration: Zugriff via Portainer. Von Portainer und Traefik via Portfreigaben auf spezifischen Dienst per Webzugriff. |
| I-SS3 | module2 – module4 | Diese Schnittstelle dient dazu, die Aktualität von unserer Virtualisierungssoftware auf Schritt zu halten. So kann sich Proxmox mit Image-Repository-Zugriffen immer die neusten Updates, welche durch uns freigegeben werden, holen und so auf einem neuen, validierten und getesteten Stand sein. |

| | | |
|---|---|---|
| | | Datenfluss: Image-Repository-Zugriffe, Updates, Managementbefehle, Zugrifftraffic<br>Konfiguration: Docker-Netzwerkbridge, Portfreigabe durch Portainer/Traefik |
| I-SS4 | module4 – module5 | Sodass die Website einfach und effizient bearbeitet, gewartet und angepasst werden kann. Werden wir die Verwaltung und die Runtime des CMS auf unserer Docker Instanz betreiben. Das CMS stellt die Website für flocker bereit. Somit ist dies das Aushängeschild für unser Produkt und sollte unter allen Umständen eine möglichst kleine, wenn nicht sogar keine Ausfallzeit haben. Hierbei können wir mit Docker entgegenwirken, da falls einmal ein Problem bestehen sollte, können wir ganz einfach den Container neu starten oder den Container aus einem Backup wiederherstellen.<br><br>Datenfluss: Backend-Datenbankzugriffe, Frontend-Anfragen, CMS-Daten (Inhalt der Website)<br>Konfiguration: Docker-Compose für Service-Orchestrierung, Umgebungsvariablen für Datenbankverbindungen, Portfreigaben für Webzugriffe |
| I-SS5 | module3 | Das module3 ist eine der Mutterschnitstellen zwischen den Diensten, da dies jede Anfrage verarbeitet, validiert, authentifiziert und autorisiert. Falls eine bestimmte Anfrage von aussen oder innen erfolgt, wird diese zwangsläufig durch Traefik laufen und bei einem Zugriff auf einen bestimmten Dienst mit einer Abfrage von Authentik gekoppelt.<br><br>Datenfluss: Authentifizierungs- & Autorisierungsanfragen, Sicherheits-Token, Zugriffsberechtigungen<br>Konfiguration: API-Gateways über Traefik, OAuth2.0/OIDC-Integration über Authentik, Zugriffskonfigurationen, Secure Zero-Trust Tunnels über Cloudflared |
| I-SS6 | module4 | Das module4 ist in unserem System die sogenannte Grossmutterschnittstelle, da dies die Schnittstelle ist, auf welcher alle Dienste von module3 laufen. Dies ist eine komplexe Konfiguration und darf daher nur von den zuständigen Personen verändert oder angepasst werden. Dies ist die Schnittstelle zwischen dem Projektteam und dem module3. Alle Konfigurationsanpassungen finden hier statt.<br><br>Datenfluss: Container-Monitoring/Status, Service-Konfigurationen, Update-Management.<br>Konfiguration: Portainer-Webzugriff, API-Integrationen für Docker, Sicherheitsrichtlinien für Management-Zugriffe |

## 5.2.3 Sicherheit (ISDS)

### 5.2.3.1 Einleitung

Folgend wird das ISDS Konzept beschrieben.

### 5.2.3.2 Zweck des Dokuments

Das ISDS-Konzept legt die nötigen Angaben zur Erhaltung und Verbesserung der Informationssicherheit und des Datenschutzes fest. Es fasst die Aspekte der Informationssicherheit und des Datenschutzes im Projekt zusammen.

### 5.2.3.3 Handhabung des Dokuments

Dieses Dokument ist vertraulich zu behandeln. Es ist untersagt an Dritte und oder Unberechtigte weiterzugeben. Dieses Dokument muss an einem Ort abgelegt werden, auf welchen nur berechtigte Personen Zugriff haben.

### 5.2.3.4 Allgemeines

Wir verfolgen eine strikte und transparente Informationspolitik und arbeiten ausschliesslich mit Schweizer Unternehmen und Dienstleistern zusammen. Die gesamte Entwicklung, Betrieb, Support, Hosting und Backups des flocker Projektes befinden sich in der Schweiz und werden ausschliesslich aus der Schweiz betreut.

### 5.2.4 Anfoderungszuordnung

| Anforderungs-ID | Anforderung |
|---|---|
| PROJ_A.1 | Unsere Webapplikation muss ohne gewinn einbringende Methode wie eine Subscription oder anderen Methoden benutzt werden können. Somit kann der Benutzer eine möglichst Kostenfreie Lösung von Flocker verwenden und wir machen keinen Gewinn durch diese Lösung.<br><br>Dies soll zu guten der Kunden kommen, welche das Produkt ohne problematischen Lizenzierungsprobleme oder anderen Umstände verwenden können. |
| PROJ_A.2 | Die Webseite von Flocker soll benutzerfreundlich aufgebaut sein. Ohne unnötige Funktionen oder ablenkendem Design. Die Webseite soll minimalistisch und modern gestaltet werden damit die Übersichtlichkeit der Webseite einfach zu gewinnen ist. |
| PROJ_A.3 | Alle VMs, Systeme, Daten und so weiter, sind ausschliesslich in der Schweiz und auch nur in der Schweiz verfügbar. Damit können wir die Datenschutz Regelungen von der Schweiz erfüllen und den Nutzer eine möglichst sichere und gute Lösung bieten. |
| PROJ_A.4 | Der Raspberry Pi wird von David Reymond zu Verfügung gestellt und der Computer wird von Noah Isenschmid zu Verfügung gestellt. Das ganze Projekt wird von einer Räumlichkeit von jemandem zuhause gehostet und bereitgestellt. Somit können wir nebst den Schulzeiten am Projekt weiterarbeiten und wir müssen die Netzwerkeinstellungen nicht ändern. |
| PROJ_A.5 | Wir bilden uns im Knowhow in der Applikationsentwicklung weiter damit wir alle das Projekt gut realisieren können. Dieses Wissen hilft uns auch bei Anpassung des Projektes oder zukünftige Projekte. |

| ID | Anforderung | Ziel |
|---|---|---|
| A1 | Unsere Webapplikation muss ohne Subscription oder anderen ähnlichen Gewinn einbringenden Methoden verwendet werden können. | Wir halten die Webapplikation für den Benutzer kostenfrei. Ziel (Z1) |
| A2 | Die Webseite / Webapplikation soll leicht und einfach zu bedienen sein. Es ist benutzerfreundlich gestaltet und es hat nicht zu viele «unnötige» Funktionen. | Mockups sind bis zur Realisierungsphase erstellt. Die Mockups zeigen den genauen Aufbau der Webapplikation, diese sind leicht überschaubar und es sind keine komplexen Menüs und Untermenüs vorhanden. Ziel (Z4) |
| A3 | Alle VMs, Systeme, Daten und so weiter, sind ausschliesslich in der Schweiz und auch nur in der Schweiz verfügbar. Daten VMS usw. dürfen nicht ausserhalb der Schweiz gelagert / vorhanden sein. | Um eine Datensicherheit zu gewährleisten, haben wir entschieden, jegliche Daten, Dienste und Dazugehörigkeiten in der Schweiz auf unseren eigenen Systemen zu hosten. Wir verzichten somit auf die Unterstützung externen Datacenter-Anbieter. So können wir eine strikte Einhaltung des Datenschutzgesetzes (DSG, DSGVO, revDSG) gewährleisten und müssen somit das SCVL3 nicht beachten. Ziel (Z5) |

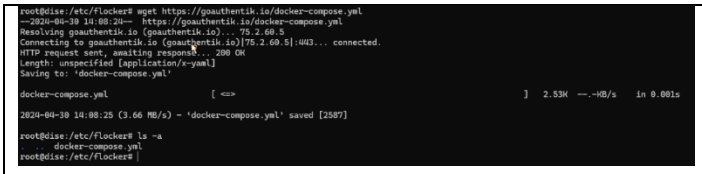
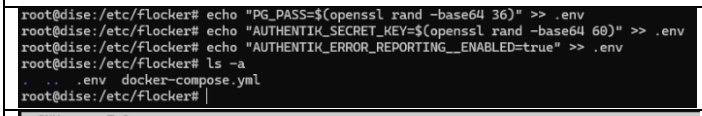
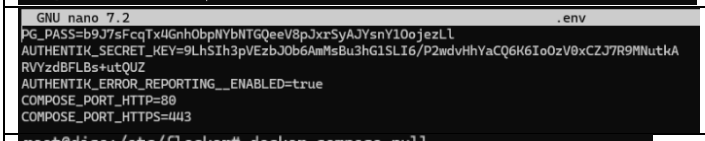
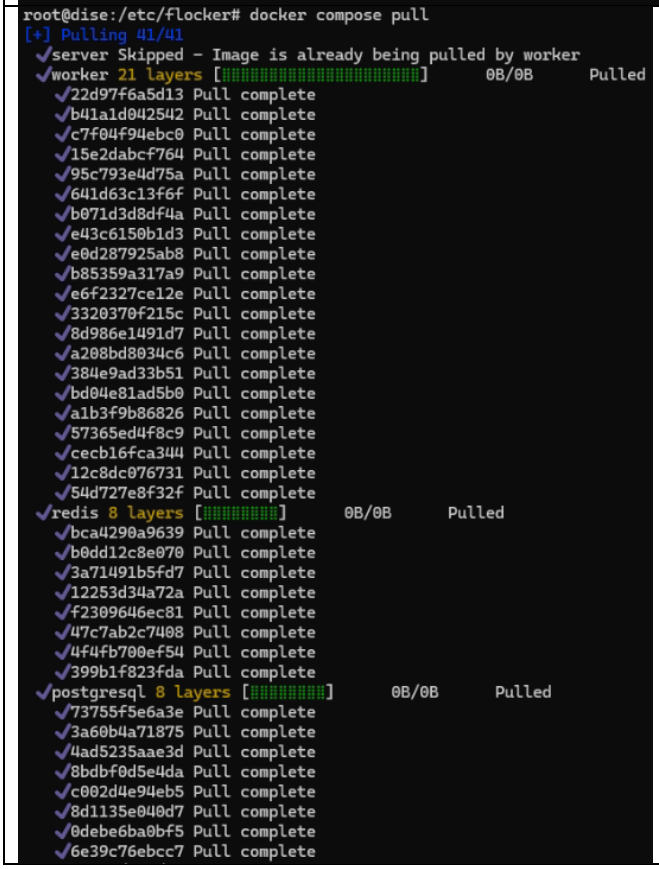
| | | |
|---|---|---|
| A4 | Die Hardware stellt David Reymond und Noah Isenschmid zur Verfügung. Das ganze Projekt wird zu Hause gehostet und bereitgestellt. | Die Hardware wird vor der Realisierungsphase aufgesetzt und konfiguriert. Ziel (Z3) |
| A5 | Das Projekt wird mit dem nötigen Wissen und Knowhow erstellt und geleitet. Wir bilden uns in der Applikationsentwicklung weiter, damit das Projekt realisiert werden kann. | Bis zu der Realisierungsphase des Projektes haben wir uns alle das nötige Wissen für die Applikationsentwicklung, welche es im Rahmen des Projektes benötigt, angeeignet. Ziel (Z1) |

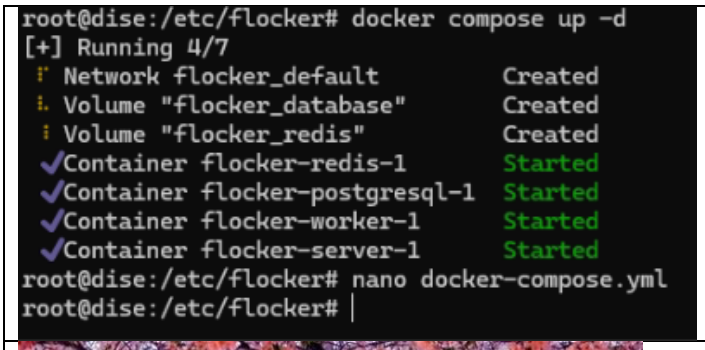
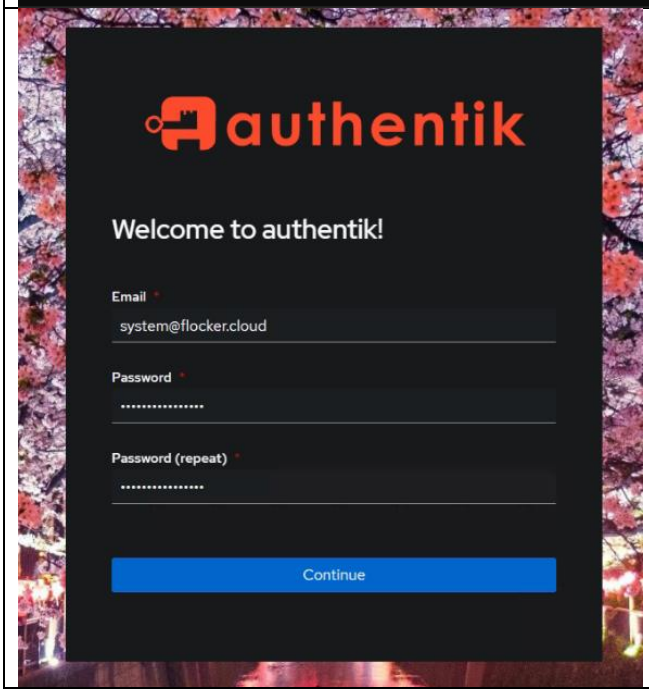
# 5.3 Systemdokumentation

## 5.3.1 Konfigurations-Dokumentation

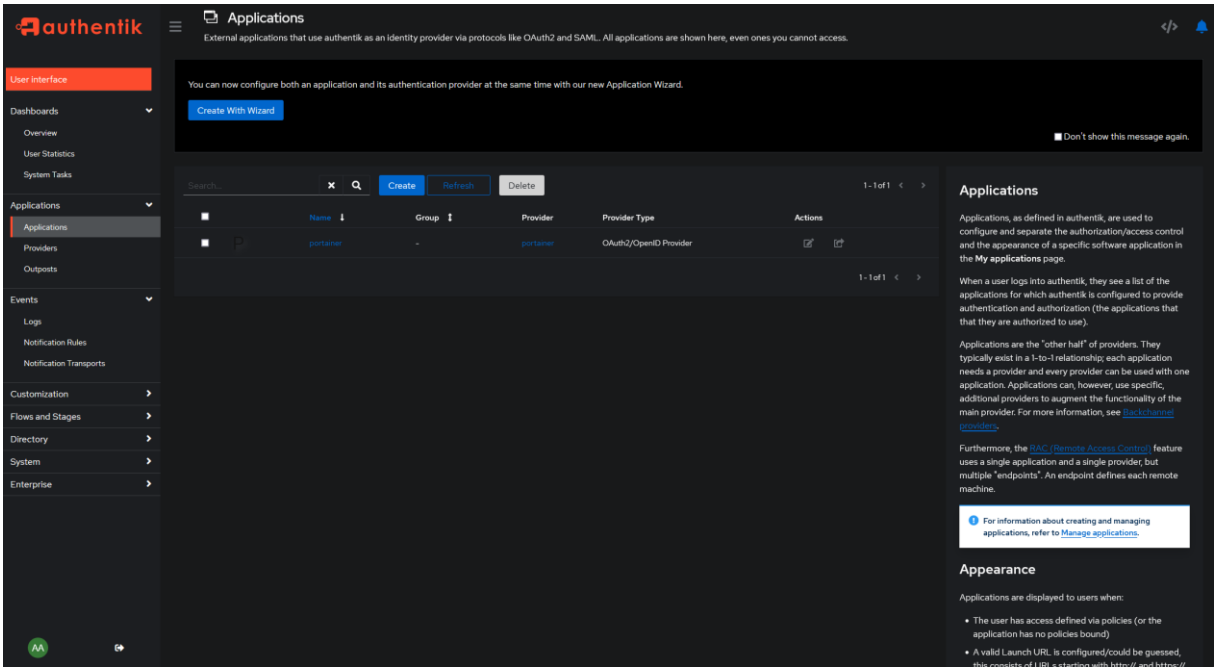
#### 5.3.1.1 Installation und Konfiguration Authentik

| Bild | Beschreibung |
|---|---|

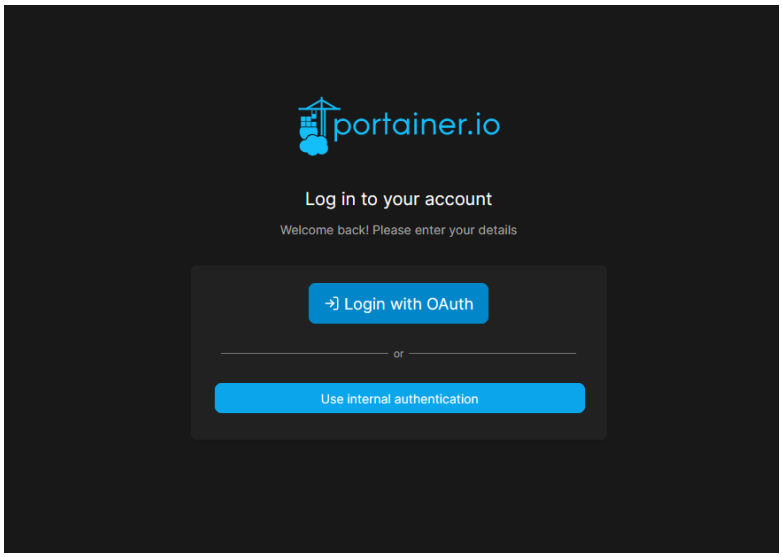
| | |
|---|---|
|  | Das offizielle docker compose File wird heruntergeladen in den Ordner “flocker” abgelegt. |
|  | Hier ist das docker compose File zu sehen, welches ausgeführt wird um Authenik zu erstellen. |
|  | Hier ist das .env File zu sehen. |
|  | Hier wird der Container erstellt. |

| | |
|---|---|
|  | Hier sieht man die Container, welche erstellt und gestartet wurden. |
|  | Nun kann man auf die Authentik Seite zugreifen und sich einloggen. |

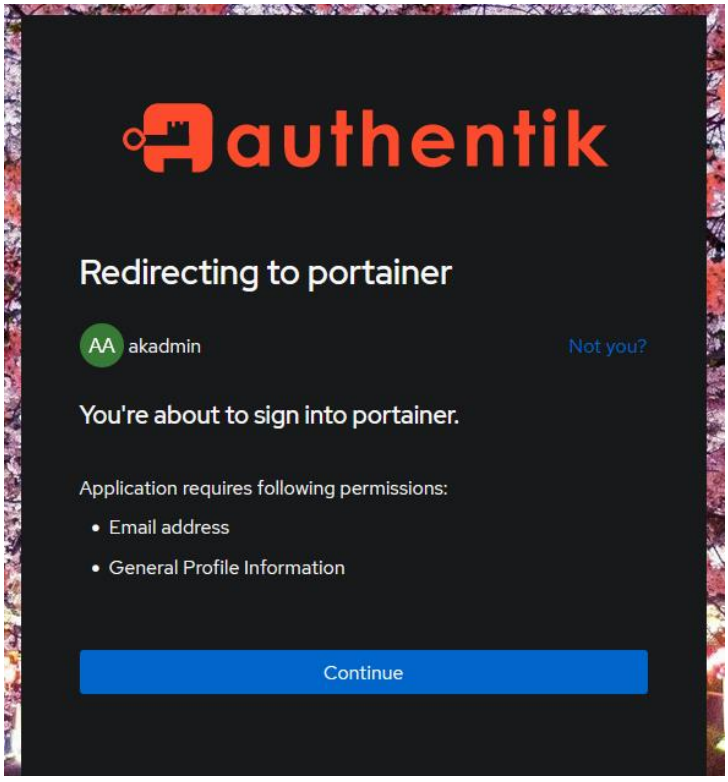
### 5.3.1.2 Authentik Administrations Portal




Da wir auf unseren Applikationen MFA anbieten wollen, benutzen wir Authenik. Hier ist ein Beispiel mit Portainer. Portainer ist hier als Applikation hinterlegt. Authentik stellt nun einen MFA für diese Webseite / Applikation zur verfügung.




Auf der Portainer Seite ist un ein «Login with OAuth» zu sehen.




Clickt man auf den «Login with OAuth» Button, wird man zu der Authtik Seite weitergeleitet. Hier kann man sich nun mit einem Benutzer anmelden.

Das prkatische an Authentik ist auch noch, dass es MFA und SSO in einem ist. Somit sind unsere Applikationen mit MFA und SSO versehen und die Benutzer bekommen die nötige Security.

#### 5.3.1.3 Wordpress Installation und Konfiguration

Docker Compose File für die Wordpress Seite:

```
version : '3.9' #using docker compose version 3.9


services:
 #Defining mysql service
  mysqldb:
    image: mysql
    environment: #feel free to change these variables as needed
      MYSQL_ROOT_PASSWORD: flocker
      MYSQL_DATABASE: wordpressdb #name of database to be created
on start of image startup
      MYSQL_USER: wordpressuser #this user will be created/granted
superuser permissions for the database(wordpressdb)specified above
      MYSQL_PASSWORD: flocker
    restart: always #this ensures when server reboots container
resttarts
    volumes:
      - mysqldb_data:/var/lib/mysql #This will help persist
mysqldb container data generated at '/var/lib/mysql' in named
volume mysqldb_data
    networks:
      - mysitenet


  #Defining wordpress service
  wordpress:
    image: wordpress #use latest wordpress image from docker hub
    restart: always  #this ensures when server reboots container
restarts
    depends_on:
      - mysqldb
    ports:
      - 8080:80    #expose container internal port 80 to
host(raspberry pi) pot 8080
    environment:
      WORDPRESS_DB_HOST: mysqldb #mysql db hostname
      WORDPRESS_DB_NAME: wordpressdb #mysql database to be used
for wordpress database
      WORDPRESS_DB_USER: wordpressuser #should be same as
MYSQL_USER
      WORDPRESS_DB_PASSWORD: flocker #should be same as
MYSQL_PASSWORD
    volumes:
```

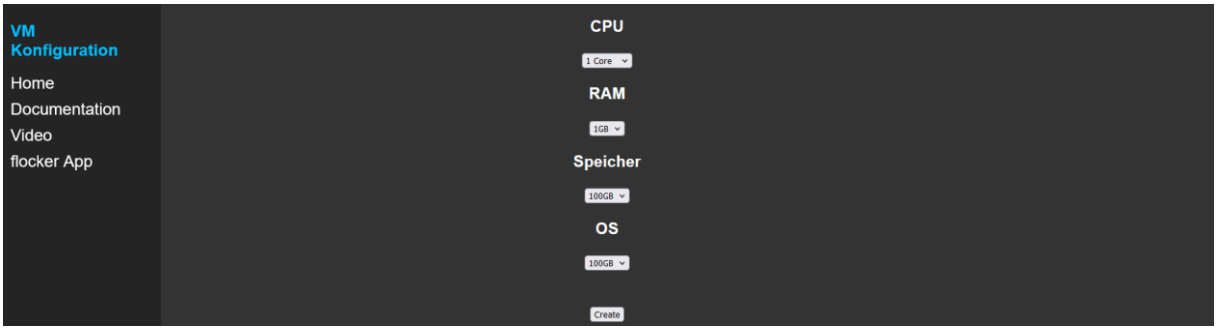
```
      - wordpress_data:/var/www/html #This will help persist
wordpress container data generated at '/var/www/html' in named
volume wordpress_data
    networks:
      - mysitenet

#Define volumes
volumes:
  mysqldb_data:
  wordpress_data:

#Define networks
networks:
  mysitenet:
```

Die Webseite wurde mit diesem Docker Compose File erstellt.

Nach der Installation und Benutzung von WordPress ist uns aufgefallen, dass wir damit nicht weiterarbeiten können, da wir in Word nicht die Java Funktionen und API-Abfragen machen können, die wir benötigen.

Unsere Lösung ist es eine selbst erstellte «HTML» Seite zu machen in welcher wir Java Code integrieren können.

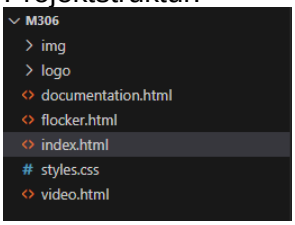
#### 5.3.1.4 Eigene HTML-Seite für die flocker App




So würde die Webapp aussehen. Da wir zu wenige Kenntnisse über das Programmieren in folgenden Bereichen haben ist es uns nicht gelungen das umzusetzen: Springboot backend, Proxmox API und Java. Die Anforderung von einer eigens erstellten Webseite aus auf Proxmox eine VM zu erstellen, überstieg unsere Kompetenz und war im Zeitrahmen nicht möglich. Damit konnten wir die Anforderung bzw. das Ziel nicht erfüllen.

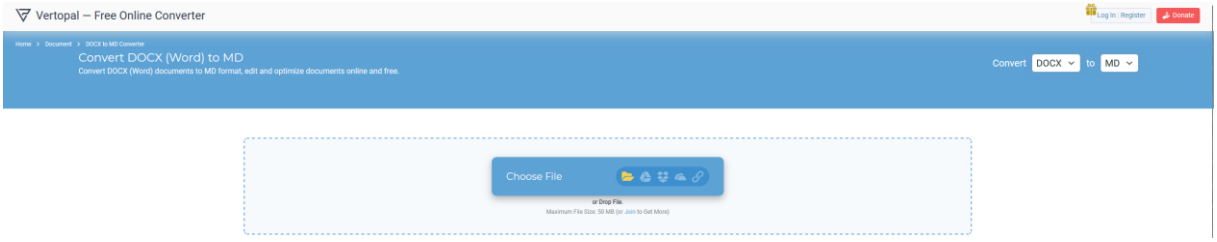
#### 5.3.1.5 HTML-Dokumentation und Video-Seite

Projektstruktur:



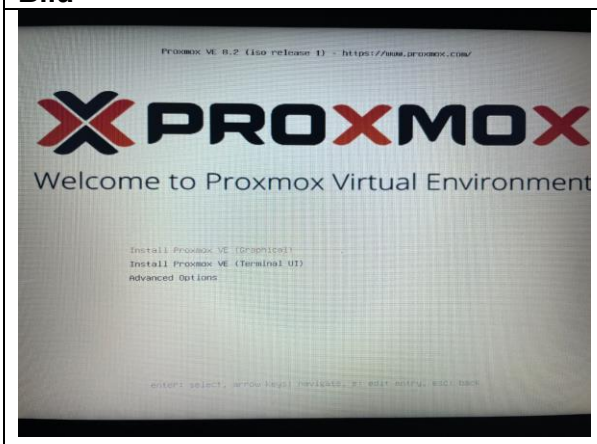
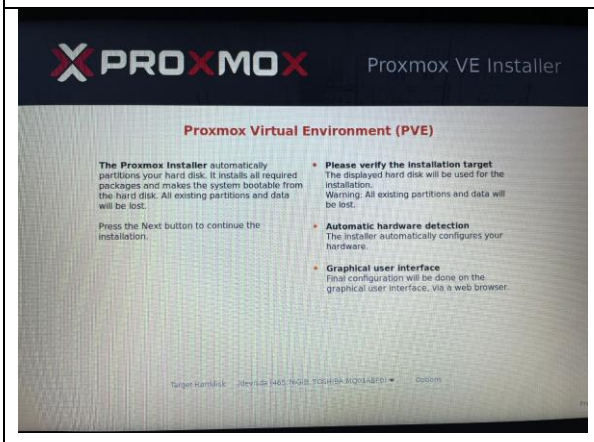
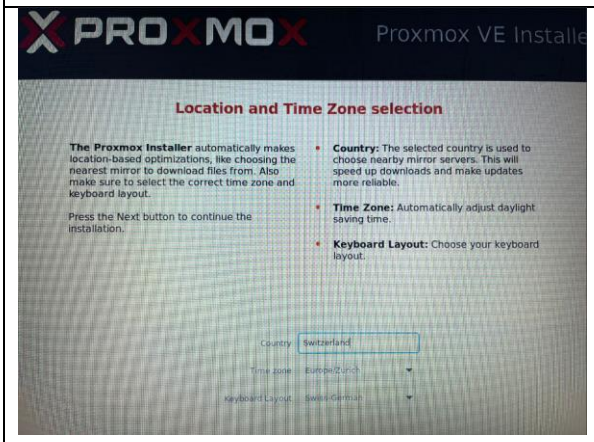
#### 5.3.1.6 Dokumentation Konvertierung

Die Dokumentation wurde mit vertopal.com in Markdown konvertiert.




Von DOCX zu MD konvertiert.

### 5.3.1.7 Proxmox Installation und Konfiguration

| Bild | Beschreibung |
|---|---|
|  | Wir installieren hier Proxmox mit einem GUI für eine einfachere Installation. |
|  | |
|  | Hier wählen wir die Location und Time Zone des Servers. |

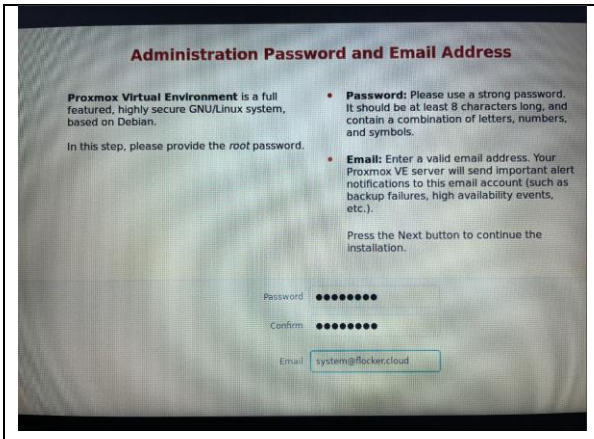
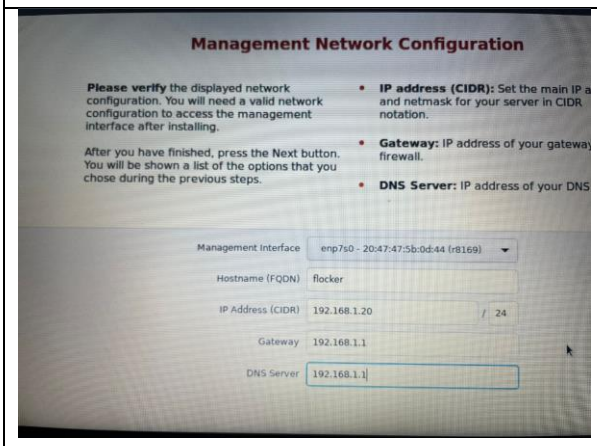
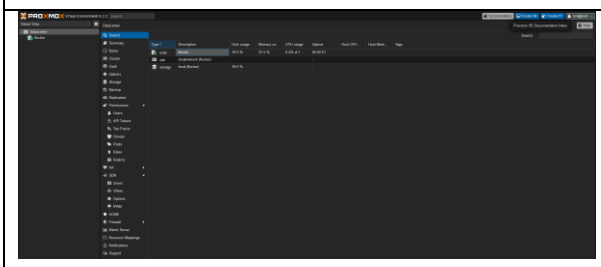
| | |
|---|---|
|  | Wir erstellen ein Admin Passwort und geben die E-Mail an. |
|  | Hier konfigurieren wir das Netzwerk, über welches wir zugreifen wollen.<br><br>Danach wird Proxmox installiert, nach der Installation sollten wir auf die Webbasierte Dienstseite zugreifen können. |
|  | Über den Browser auf https://192.168.1.20:8006 kann nun auf das Proxmox Admininterface zugegriffen werden. |

*Tabelle 2 Proxmox*

Weiteres müssen wir nicht konfigurieren. Proxmox steht nun zur Benutzung bereit.
Darauf laufen später die VMs, welche von der flocker.ch Webseite aus erstellt werden.

Wie schon im beschrieben, ist es uns nicht gelungen VMs von der Webseite aus zu erstellen.

#### 5.3.1.8 Apache Guacamole

**Problembeschreibung:**

Bei Apache Guacamole stellte sich das Problem heraus, dass es keine Möglichkeit gibt es für den Raspberry Pi aufzusetzen. Es gibt kein offizielles Repo / Image, welches man verwenden könnte.

Daher konnten wir das Vorhaben von Apache Guacamole nicht umsetzen.

### 5.3.1.9 Portainer Umgebung

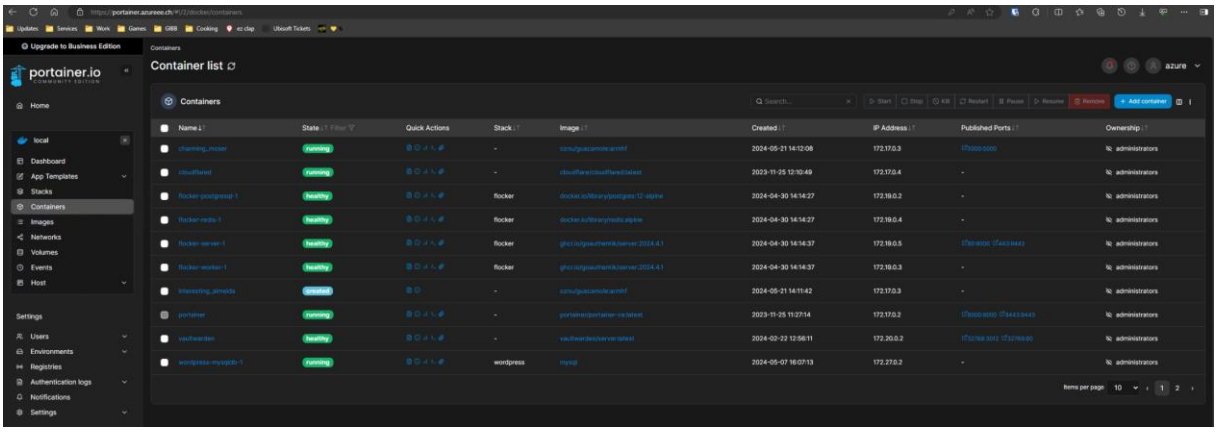



*Abbildung 1 Portainer Oberfläche mit laufende Container*

Hier laufen die verschiedenen Container für das flocker Projekt.

Wir haben einmal Authentik (den MFA), Portainer selbst damit Portainer überhaupt läuft, Cloudflared ist der Cloudflare Tunnel, damit wir in Cloudflare die Applikationen aus dem Web erreichbar machen können. Vaultwarden haben wir aufgesetzt, um unsere Passwörter für das Projekt zu speichern.

Auf der flocker-webseite läuft unsere Applikation (wenn sie funktionieren würde) und dort ist auch unsere ganze Dokumentation + Dokumentationsvideo abgelegt.

### 5.3.1.10 Docker Netzwerke

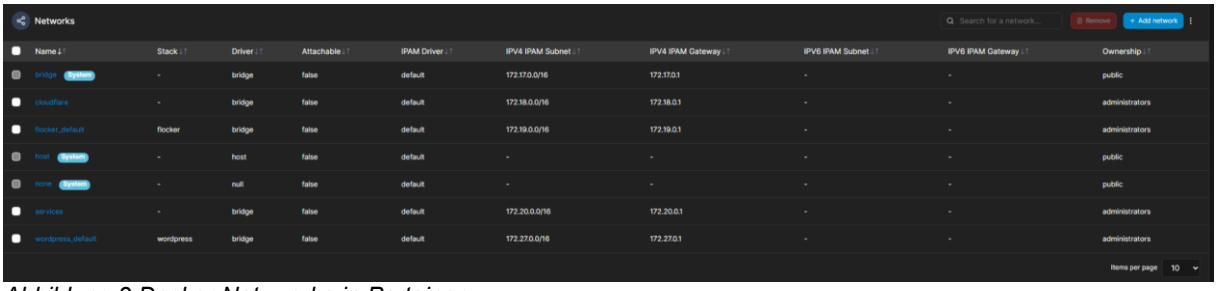
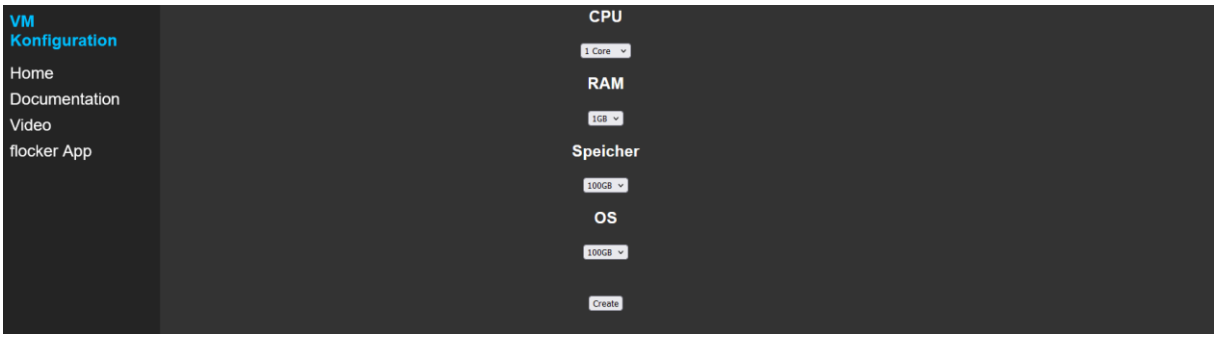



*Abbildung 2 Docker-Netzwerke in Portainer*

Wir benutzen zwei Netzwerke für unser Projekt einmal das Bridge, cloudflare und flocker_default Netzwerk.

#### 5.3.1.11 Anwendungsfunktionalität




Die Anwendung ist sehr simpel. Möchte eine VM erstellt werden, wählt man unter den entsprechenden Menüs die Werte aus welche man haben möchte. Ausgewählt werden kann zwischen den folgenden Parameter:

CPU:
- 1 Core
- 2 Cores
- 4 Cores

RAM
- 1GB
- 2GB
- 4GB

Speicher
- 100GB
- 150GB
- 200GB

OS
- Windows 11
- Linux Ubuntu Server

Hat man die Werte ausgewählt, welche man für die VM möchte, klickt man auf den «Create» Button. Die VM wird auf Proxmox erstellt.

# 5.4 Systemtest

## 5.4.1 Testspezifikationen

#### 5.4.1.1 Testkonzept

| Nr. | Testfall | Testbeschreibung | Testschritte | Erwartetes Ergebnis |
|---|---|---|---|---|
| 1 | Funktionalität ohne kostenpflichtige Methoden | Überprüfen der Verfügbarkeit und Funktionalität der Flocker Webapplikation ohne die Notwendigkeit kostenpflichtiger Abonnements oder ähnlicher Zahlungsmethoden. | 6. Öffnen der Flocker Webapplikation.<br>7. Versuch, auf alle Hauptfunktionen zuzugreifen, ohne Zahlungsinformationen einzugeben.<br>8. Durchführung typischer Aktionen wie Dateiupload, Datenbearbeitung und Datenspeicherung.<br>9. Überprüfung, ob sämtliche Funktionen ohne die Notwendigkeit von Zahlungsinformationen zugänglich sind.<br>10. Abschluss einer typischen Sitzung ohne Eingabe von Zahlungsinformationen. | Alle Hauptfunktionen der Webapplikation sind ohne die Eingabe von Zahlungsinformationen zugänglich und voll funktionsfähig. |
| 2 | Benutzerfreundlichkeit der Webseite | Überprüfen der Benutzerfreundlichkeit und des Designs der Flocker Webseite. | 5. Bewertung der Startseite und des Navigationsmenüs auf Einfachheit und Klarheit.<br>6. Überprüfung der Anordnung von Schaltflächen und Funktionen auf Benutzerfreundlichkeit.<br>7. Testen der Suchfunktion, um zu sehen, wie leicht Benutzer relevante Informationen finden können.<br>8. Durchführung von typischen Aktionen gemäss den Mockups, um sicherzustellen, dass die tatsächliche Nutzung dem geplanten Design entspricht. | Die Webseite ist leicht navigierbar, weist ein minimalistisches und modernes Design auf und bietet eine intuitive Benutzererfahrung. |

| | | | | |
|---|---|---|---|---|
| 3 | Sicherheit und Lokalität der Daten | Überprüfen, ob sämtliche Daten und Systeme der Flocker Webapplikation ausschliesslich in der Schweiz gehostet werden und die Datenschutzbestimmungen eingehalten werden. | 5. Überprüfung der Standortangaben der Server und Datenbanken.<br>6. Verfolgung der Datenflüsse und Überwachung der Netzwerkkommunikation.<br>7. Sicherstellung, dass keine Daten ausserhalb der Schweiz gespeichert oder übertragen werden.<br>8. Überprüfung der Datenschutzrichtlinien, um den Standort der Datenspeicherung zu bestätigen. | Alle Daten und Systeme sind ausschliesslich in der Schweiz gehostet, um den Datenschutzanforderungen zu genügen. |
| 4 | Leistung und Verfügbarkeit der Hardware | Überprüfen der Leistung und Verfügbarkeit der bereitgestellten Hardware für die Flocker Webapplikation. | 5. Überprüfung der Hardware-Spezifikationen auf Konformität mit den Anforderungen.<br>6. Testen der Server- und Netzwerkverfügbarkeit nach der Installation.<br>7. Durchführung von Leistungstests, um sicherzustellen, dass die Hardware den Anforderungen der Webapplikation entspricht.<br>8. Überprüfung der Sicherheitsvorkehrungen und Zugangsbeschränkungen für die gehostete Hardware. | Die bereitgestellte Hardware ist korrekt konfiguriert, verfügbar und erfüllt die Leistungsanforderungen der Webapplikation. |
| 5 | Wissensstand und Kompetenz des Teams | Überprüfen, ob das Entwicklungsteam über das erforderliche Wissen und Know-how für die Entwicklung, Anpassung und Wartung der Flocker Webapplikation verfügt. | 5. Bewertung der Qualifikationen und Erfahrungen der Teammitglieder im Bereich der Webentwicklung.<br>6. Durchführung von Code-Reviews und Überprüfung der Entwicklungsumgebung.<br>7. Testen der Teamkommunikation und Zusammenarbeit bei der Lösung von Problemen.<br>8. Überprüfung der Dokumentation und Schulungsunterlagen zur Sicherstellung einer kontinuierlichen Wissensvermittlung. | Das Entwicklungsteam verfügt über das erforderliche Wissen und Know-how für die Entwicklung, Anpassung und Wartung der Webapplikation. |

| | | | | |
|---|---|---|---|---|
| 6 | Integration der Benutzerschnittstelle (UI) mit Backend-Services | Überprüfen der nahtlosen Integration der Benutzerschnittstelle mit den Backend-Services. | 5. Durchführung von typischen Aktionen über die Benutzeroberfläche, z. B. Dateiupload oder Datenbearbeitung.<br>6. Überwachung der Netzwerkkommunikation zwischen der Benutzerschnittstelle und den Backend-Services.<br>7. Überprüfung der Datenkonsistenz und -integrität nach jeder Aktion.<br>8. Testen von Fehlerszenarien wie Netzwerkausfällen oder unerwarteten Serverantworten. | Die Benutzerschnittstelle integriert sich nahtlos mit den Backend-Services, und Daten werden korrekt zwischen den Systemen ausgetauscht. |
| **7** | API-Funktionalität und -Sicherheit | Überprüfen der Funktionalität und Sicherheit der APIs, die von der Flocker Webapplikation bereitgestellt werden. | 5. Durchführung von API-Anfragen gemäss der Spezifikation.<br>6. Überprüfung der Antwortzeiten und Datenintegrität.<br>7. Testen von Authentifizierungs- und Autorisierungsmethoden, z. B. API-Schlüssel oder OAuth.<br>8. Durchführung von Sicherheitstests wie Injection-Angriffen oder Brute-Force-Angriffen. | Die APIs funktionieren wie erwartet, sind sicher und bieten angemessene Authentifizierungs- und Autorisierungsmethoden. |
| **8** | Datenbankintegration und -zugriff | Überprüfen der Integration und des Zugriffs auf die Datenbanken der Flocker Webapplikation. | 5. Durchführung von Datenzugriffsoperationen über die Benutzeroberfläche und die API.<br>6. Überprüfung der Datenkonsistenz zwischen Frontend und backend.<br>7. Testen von Datenbankfunktionen wie Transaktionen und Indizierungen.<br>8. Durchführung von Skalierungstests, um die Leistung der Datenbank unter Last zu überprüfen. | Die Datenbankintegration ist stabil, und Daten werden korrekt zwischen der Benutzerschnittstelle und dem Backend ausgetauscht. |

| | | | | |
|---|---|---|---|---|
| 9 | Externe Systemintegration | Überprüfen der Integration mit externen Systemen, die von der Flocker Webapplikation genutzt werden. | 1. Durchführung von Aktionen, die die Interaktion mit externen Systemen erfordern, z. B. Authentifizierung über Drittanbieter oder Datenimport/-export.<br>2. Überwachung der Kommunikation mit externen Systemen und Überprüfung der Datenintegrität.<br>3. Testen von Fehlerbehandlungsszenarien wie Ausfällen externer Dienste.<br>4. Überprüfung der Sicherheit der Datenübertragung und der Datenschutzrichtlinien im Zusammenhang mit externen Systemen. | Die Integration mit externen Systemen funktioniert wie erwartet und gewährleistet die Sicherheit und Integrität der übertragenen Daten. |

#### 5.4.1.2 Testresultate

| Nr | Testfall | Erwartetes Ergebnis | Tatsächliches Ergebnis |
|---|---|---|---|
| 1 | Funktionalität ohne kostenpflichtige Methoden | Alle Hauptfunktionen der Webapplikation sind ohne die Eingabe von Zahlungsinformationen zugänglich und voll funktionsfähig. | Die Webapplikation ist komplett kostenlos zu bedienen. |
| 2 | Benutzerfreundlichkeit der Webseite | Die Webseite ist leicht navigierbar, weist ein minimalistisches und modernes Design auf und bietet eine intuitive Benutzererfahrung. | Die Seite ist einfach gehalten und bietet lediglich ein kleines Menü für die Seiten an. |
| 3 | Sicherheit und Lokalität der Daten | Alle Daten und Systeme sind ausschliesslich in der Schweiz gehostet, um den Datenschutzanforderungen zu genügen. | Alle Dienste sind in der Schweiz gehosted. |
| 4 | Leistung und Verfügbarkeit der Hardware | Die bereitgestellte Hardware ist korrekt konfiguriert, verfügbar und erfüllt die Leistungsanforderungen der Webapplikation. | Die Hardware ist fähig, die Applikation auszuführen und zu betreiben. |
| 5 | Wissensstand und Kompetenz des Teams | Das Entwicklungsteam verfügt über das erforderliche Wissen und Know-how für die Entwicklung, Anpassung und Wartung der Webapplikation. | Das Team hat zu wenig Know-How über die API-Schnittstelle. Die Dienste aufzusetzen und zu konfigurieren beherrscht das Team. |
| 6 | Integration der Benutzerschnittstelle (UI) mit Backend-Services | Die Benutzerschnittstelle integriert sich nahtlos mit den Backend-Services, und Daten werden korrekt zwischen den Systemen ausgetauscht. | Die API-Schnittstelle zwischen Backend und Applikation steht nicht. Das Know-How ist nicht vorhanden um diese korrekt einzurichten. |

| | | | |
|---|---|---|---|
| 7 | API-Funktionalität und -Sicherheit | Die APIs funktionieren wie erwartet, sind sicher und bieten angemessene Authentifizierungs- und Autorisierungsmethoden. | Siehe oben. Authentik an sich funktioniert, jedoch ist es nicht möglich, eine VM über das Web-Interface zu erstellen. |
| 8 | Datenbankintegration und -zugriff | Die Datenbankintegration ist stabil, und Daten werden korrekt zwischen der Benutzerschnittstelle und dem Backend ausgetauscht. | Siehe Oben |
| 9 | Externe Systemintegration | Die Integration mit externen Systemen funktioniert wie erwartet und gewährleistet die Sicherheit und Integrität der übertragenen Daten. | Die Daten sind momentan nur auf einer abgetrennten Server-Umgebung gespeichert und ermöglicht nur Zugriff über direkte Verbindungen. Daher sind die Daten sicher. |

#### 5.4.1.3 Testauswertung

Die Tests sind leider nicht alle Erfolgreich. Die Dienste an sich stellen dabei kein Problem dar, jedoch sobald es darum geht, die Web-Applikation mit dem Backend zu verbinden, stösst unser Team auf einen Stolperstein. Die API-Schnittstelle erweist sich als grosses Hindernis, welches wir mit unserem Know-How nicht überwinden können. Es bräuchte ein Vorstudium über APIs um das gewünschte Resultat zu erreichen, da man für unser Vorhaben noch einiges selbst konfigurieren muss.

# 5.5 Arbeitsjournale

## 5.5.1 David Reymond

### 5.5.1.1 KW 17

| Datum | Tätigkeiten | Bemerkungen | Nächste Schritte |
|---|---|---|---|
| 23.04.2024 | - Dokumentation Struktur erstellt<br>- Antrag für externe Arbeiten erstellt und dem Projektleiter weitergegeben<br>- Arbeiten aufgeteilt und sich organisiert<br>- Planung und Abmachungen für die Arbeiten zuhause erstellt<br>- Docker Networking Thema eingelesen und die besten Varianten für unser Netzwerk rausgesucht | Praktisch haben wir noch nichts umgesetzt, da die Hardware zuhause und noch nicht aufgesetzt ist. Neues Wissen über Docker Networking, welches gebraucht wird für das Projekt. | - Hardware (Virtualisierungs Server und Management Server) aufsetzen und konfigurieren<br>- Dokumentation weiterführen<br>- Arbeitsjournal schreiben |

Die Realisierungsphase des Projektes hatte heute dem 23.04.2024 angefangen. Wir bekamen von Timo Steinlin eine kurze Einleitung bezüglich des Realisierungsberichtes. Er erzähle und betonte, dass das Arbeitsjournal ein essenzieller Bestandteil der IPA-Arbeit sein wird. Sein Input war sehr hilfreich, da er Insights gegeben hatte, auf was besonders bei einer IPA zu achten ist. Im Projektteam haben wir besprochen, wer was macht und wie wir nun weiterhin vorgehen. Mit dem weiteren Vorgehen ist gemeint, dass wir beschlossen haben, die nächsten drei Wochen, also von KW18 bis und mit KW20 von zuhause aus zu arbeiten. Grund ist folgender, nämlich befindet sich unsere ganze Hardware bei uns zuhause, uns ist es nicht möglich ein VPN aufzubauen, da es unser ISP nicht ermöglicht ein DMZ, statische Routen etc. zu erstellen.

Somit habe ich ein Antrag für schulexterne Arbeiten erstellt und diesen unserem Projektleiter Noah Isenschmid weitergegeben, welchen er dann am Stakeholder Timo Steinlin weiterleiten wird. Timo Steinlin wird dann den Antrag im besten Fall genehmigen, oder bei schlechter Argumentation ablehnen. Der erste Tag bzw. die erste Woche der Realisierung war mehr oder weniger nicht wirklich produktiv für das Projekt selbst, da sich die Hardware zuhause befindet und noch aufgesetzt und konfiguriert werden muss, konnten wir an der Umsetzung selbst noch nicht arbeiten. Was gemacht werden konnte ist jedoch die ganze Dokumentation Struktur, damit müssen wir uns später nicht mehr auseinandersetzten, sondern einfach nur noch in die Kapitel eintragen. Somit können wir uns vollkommen auf die Umsetzung konzentrieren. Ich habe angefangen mit der Recherche von und über Docker Networking.

Da dieses Thema sehr komplex und verwirrend sein kann ist es wichtig, früh im Projekt über diese Networking Art Bescheid zu wissen, damit es ohne grosse Nachrecherche umgesetzt werden kann.

#### 5.5.1.2 KW 18

| Datum | Tätigkeiten | Bemerkungen | Nächste Schritte |
|---|---|---|---|
| 30.04.2024 | - Docker Netzwerke Recherche und Konfiguration | Besseres Wissen über Docker Netzwerke | Aufsetzen der Dienste und Konfiguration |

Heute habe ich mich nur mit den Docker Netzwerken befasst.

In diesem Projekt ist es wichtig, dass die Netzwerke stimmen. In Docker gibt es sehr viele Möglichkeiten Netzwerke zu erstellen und zu verwalten. Da wir hohe Ansprüche an die Security haben ist es wichtig, dass die Netzwerke richtig konfiguriert sind und sich die Applikationen / Dienste in den jeweiligen Netzwerken befinden.

Durch Portainer, einer Grafischen Oberfläche für Docker ist es einfach die Netzwerke zu erstellen. Das ist auch einer der Gründe, warum wir Portainer verwenden, wegen den Netzwerken. Hätten wir Portainer nicht müssten wir die Netzwerke über die Comandline erstellen und verwalten und das wäre ersten zeitintensiv und zweitens unnötig mühsam.

Die ganzen Netzwerke zu erstellen war einfach und die Dienste dazu hinzuzufügen noch einfacher. Im Docker Compose File gibt man das Netzwerk an, in welchem der Dienst laufen soll, und schon wird der Container dem Netzwerk zugeteilt.

Als nächstes würden wir die Webseite und die ganzen Dienste aufsetzen.

#### 5.5.1.3 KW 19

| Datum | Tätigkeiten | Bemerkungen | Nächste Schritte |
|---|---|---|---|
| 07.05.2024 | - WordPress aufgesetzt und konfiguriert<br>- Authentik konfiguriert, Gruppen und Benutzer erstellt<br>- Arbeitsjournal geschrieben und weiteres Vorgehen<br>- Fehlersuche bei WordPress, Tunneling von Cloudflare und Authentik<br>- WordPress Seite konfiguriert<br>- Proxmox aufgesetzt | - Webseite flocker.ch ist nicht erreichbar<br>- Zeitdruck ist immens<br>- Projekt ist zu gross für so wenig Zeit | - Testing<br>- Virtualisierungshost aufsetzen<br>- Dokumentation abschliessen und abgeben |

Heute habe ich mit Noah zusammen WordPress aufgesetzt, die Seite, welche wir benutzen werden, um VMs aufzusetzen. Dabei kam es zu Komplikationen, einmal hatten wir lange bis WordPress an sich lief, da der Error Log sehr irreführend war. Es sagte uns es sein ein Problem mit dem Apache2 Server, da wir aber gar kein Apache2 in Verwendung haben, waren wir sehr verwirrt. Nach einigem herumprobieren fand ich die Lösung im Docker Compose File, welches wir erstellt hatten. Der Fehler war, dass wir im Docker Compose File ein Netzwerk angegeben haben, welches erstellt werden und der Container direkt joinen soll. Das funktionierte so nicht und die Lösung war es manuell dem definierten Docker Netzwerk zu joinen.

Des Weiteren sind aus unerklärlichen Gründen etliche Container abgestürzt, da mussten wir auch noch Fehlersuche betreiben, gerade bei Authentik konnte die Seite nicht mehr erreicht werden. Nach einem Restart der Container funktioniere nun alles wieder. Ich habe in Authentik Gruppen und User erstellt. Einmal eine Admin Gruppe für die Systemadministratoren, damit diese in Authentik alles einstellen und konfigurieren können. Die Installation und Konfiguration von WordPress habe ich in der Flocker Realisierung dokumentiert und beschrieben. Das Arbeitsjournal habe ich weitergeführt und die nächsten Schritte beschrieben. Die Webseite ist noch nicht von aussen aus erreichbar, da uns langsam die Zeit ausgeht. Wir werden voraussichtlich diese Funktion nicht umsetzen können, erstens wegen dem DNS-Eintrag, dieser wird von Cloudflare nicht aktualisiert und der Zeitdruck ist immens. Die nächsten Schritte sind die Dokumentation abzuschliessen, Testing durchzuführen und den Praktischen Teil des Projektes abzuschliessen.

Da der «Lieferant» unseres Virtualisierungshost, ohne Bescheid zu geben in die Ferien ist, haben wir nun ein Problem mit dem Proxmox. Daher setzte ich eine VM auf, auf welcher nun Proxmox läuft. Die Installation war einfach, ich lud eine ISO-Datei herunter und erstellte mit der ISO-Datei eine VM auf welcher nun Proxmox läuft. Somit können wir Proxmox nutzen auch ohne Hardware.

#### 5.5.1.4 KW 20

| Datum | Tätigkeiten | Bemerkungen | Nächste Schritte |
|---|---|---|---|
| 14.05.2024 | - Coding der HTML-Webseite<br>- Coding Java, API Proxmox VE<br>- Dokumentieren<br>- Arbeitsjournal weitergeführt | Kein Erfolg beim Java Code, viel zu komplex. | Dokumentation fertig stellen und erklären, warum das Projekt gescheitert ist. |

Heute habe ich versucht den Code, quasi die App zu erstellen. Die Funktion, um von einer HTML-Seite eine VM in Proxmox zu erstellen. Erstens musste ich da die Dokumentation von der Proxmox VE API lesen und verstehen und ohne Applikationserfahrung (ausser ein bisschen in Python) stellte sich das als sehr herausfordernd heraus. Danach ist mir aufgefallen, dass das Backend am besten aus Springboot bestehen sollte. Daher musste ich mich in Springboot einlesen und das ganze verstehen (was mir nicht geling). Nach paar Stunden wurde mir bewusst, wie extrem schwierig das Ganze ist, wenn man keine Coding Erfahrungen hat, gerade noch mit so wenig Zeit. Gut der Zeitdruck war dann weniger das Problem, das grösste Problem war das Riesenprojekt mit den ganzen Anforderungen für welches wir absolut nicht die Erfahrung und das Knowhow haben.

Was ich noch weiter gemacht habe ich die HTML-Seite zu designen und versucht einen Teil zu erfüllen. Indem wir eine Webseite aufzeigen, mit der potenziell eine VM in Proxmox erstellt werden könnte. Da wir unsere Dokumentation und das flocker Video auf einer selbst gehosteten Webseite «Präsentieren» wollen, band ich die flocker App auf diese Webseite mit ein, dass alles an einem Platzt ist, denn uns ist aufgefallen, dass wir mit WordPress nicht weiter machen hätten können, denn dort war es noch komplizierter ein Backend, API etc. umzusetzen.

Ja und so fing das Projekt immer weiter zu zerfallen an, denn Apache Guacamole hatte auch nicht funktioniert, es gibt kein offizielles Image und oder Repo, um Apache Guacamole auf einem Raspberry Pi aufzusetzen. Also zusammengefasst heisst das, dass wir erstens keine VM erstellen können, zweitens nicht einmal über das Web auf eine Proxmox VM zugreifen können. Somit sind eigentlich die wichtigsten Funktionen / Ziele nicht erfüllt.

Wie es weiter geht, ist mir eigentlich relativ klar, aus meiner Sicht gibt es so wie es jetzt aussieht keine Option ein fertiges Produkt zu erstellen und zu präsentieren. Die Tools funktionieren nicht und wir haben uns einfach absolut überschätzt in unseren Fähigkeiten und den Möglichkeiten.

Da wir noch eine Woche länger Zeit bekamen, werde ich nächste Woche die Dokumentation abschliessen, den Webserver für unsere Webseite aufsetzten und wenigstens den zum Laufen bringen.

#### 5.5.1.5 KW 21

| Datum | Tätigkeiten | Bemerkungen | Nächste Schritte |
|---|---|---|---|
| 21.05.2024 | - Dokumentation abgeschlossen<br>- Letztes Arbeitsjournal geschrieben<br>- Problematik erklärt<br>- Letzte Versuche mit Code und Apache Guacamole | Das Projekt wurde nicht so fertig gestellt, wie wir uns es vorgestellt hatten. | Die nächste Phase, die «Einführung» vorbereiten. |

Heute versuchte ich noch die letzten Versuche Apache Guacamole zum Laufen zu bringen. Auf GitHub hatte ich ein Repository gefunden, um Apache Guacamole auf dem Raspberry Pi zum Laufen zu bringen. Er war der Einzige, der so ein Repo erstellt hatte und dies auch erklärte. Jedoch habe ich es nach einigem Probieren und genauem folgen seiner Anleitung es nicht zum Laufen gebracht. Es hätte grundsätzlich nicht viel geändert, da wir keine VM hatten auf welche zugegriffen werden konnte, da diese Funktion gar nicht existierte. Somit akzeptierte ich es und fokussierte mich auf die Erklärung, warum es gescheitert ist.

Damit wir die Dokumentation auf unserer Webseite anzeigen lassen können, habe ich einen Markdown Converter verwendet, Pandoc. Mit Pandoc ist es möglich Word und oder PDF in eine Markdown Language zu konvertieren und diese in den HTML-Code einzubinden. Das hatte gut funktioniert einige manuelle Anpassungen am Markdown musste ich noch machen, weil die Formatierung nach dem Konvertieren nicht ganz stimmte, das war aber eine kleine Sache.

**Fazit**

Das Projekt war viel zu gross und zu komplex. Das Projekt war mehr ein Appli Projekt als ein Plattformentwickler Projekt.

Fürs nächste Mal würde ich die Tools, welche man potenziell braucht, besser recherchieren und grundsätzlich besser vorbereiten. Somit würde man nicht auf folge-Stolpersteine treffen, welche das Projekt ruinieren bzw. unmöglich machen.

Für das nächste Mal würde ich auch mehr in die Richtung Plattformentwicklung gehen und nicht in die absolute Applikationsentwicklung. Das stellt sich aber immer als schwierig heraus als Plattformentwickler neue und innovative Ideen zu finden.

## 5.5.2 Paulo Lalicata

### 5.5.2.1 KW 17

| Datum | Tätigkeit | Bemerkung | Nächste Schritte |
|---|---|---|---|
| 23.04.2024 | - In Team nächstes Vorgehen besprochen<br><br>- Arbeiten aufgeteilt<br><br>- Zeitplan angepasst<br><br>- Dokumentation vorbereitet | Die Realisierung wird etwas mühsam zum umsetzen sein, da sich die Hardware bei Noah befindet. Die Dokumentation und die Arbeitsteilung sind klar definiert. | - Hardware einrichten<br><br>- Dokumentation schreiben<br><br>- Website einrichten |

Die Realisierungsphase begann am 23.04.2024. Unser Team konnte bis jetzt immer gute Resultate abliefern und ich habe das Gefühl, dass wir auch hier gut arbeiten werden. Die Arbeitsteilung konnten wir ohne grosse Probleme machen. Wir haben als Team beantragt, dass wir die zukünftige Lektion von zu Hause aus machen werden, da sich die benötigte Infrastruktur bei Noah befindet und wir von der GIBB aus nur schlecht darauf zugreifen können.
Wir lesten uns etwas in die Anforderungen der Realisierung hinein und haben dementsprechend unsere Prioritäten festgelegt. Noah wird während dem Projekt den Überblick als Teamleiter behalten und uns als Team so leiten, dass jeder etwas zu tun hat. Nächste Woche werde ich damit beginnen, die Website für unser Projekt zu gestalten, sodass die wichtigsten Informationen vorhanden sind.

### 5.5.2.2 KW 18

| Datum | Tätigkeit | Bemerkung | Nächste Schritte |
|---|---|---|---|
| 30.04.2024 | - Webseite eingerichtet<br><br>- Dokumentation geschrieben | Die Website besteht nur aus HTML und CSS, da ich JS nicht kann. Die Seite beinhaltet die wichtigsten Informationen.<br><br>Im Team die Dokumentation überarbitet | - Webseite fertig gestalten |

*Tabelle 3 KW18 Paulo*

**Reflexion**

Diese Woche arbeitete ich daran, eine Website mit HTML und CSS zu gestalten. Ich verwendete dabei eine Grundlage von Noah, welche er schon bei einer privaten Webseite verwenden hatte. Die Darstellung sollte unser Project flocker gut repräsentieren. Die Webseite beinhaltet Links zur Dokumentation, zum Video und zu einer Zusammenfassung des Projekts.
Die Dokumentation ist auf gutem Weg. Zeitlich gesehen kann es aber etwas knapp werden, daher hoffe ich, dass wir das irgendwie noch schaffen.

#### 5.5.2.3 KW 19

| Datum | Tätigkeit | Bemerkung | Nächste Schritte |
|---|---|---|---|
| 07.05.2024 | - HTML/CSS Weitergestaltet<br><br>- Mit Konfiguration geholfen | Weiter an Webseite gearbeitet<br><br>David und Noah bei Konfiguration geholfen | - Webseite fertig gestalten |

*Tabelle 4 KW19 Paulo*

**Reflexion**

Diese Woche arbeitete ich weiter an der Webseite. Sie sieht mittlerweile etwas besser aus und kann einigermassen präsentiert werden.
Ich half David und Noah bei der Konfiguration. Die Dienste Authentik und Portainer konnten wir zum laufen bringen, jedoch haben wir mühe bei der Schnittstelle von Guacamole. Die Schnittstelle richtig zu konfogurieren verlangt API-Kenntnisse, welche wir uns noch aneignen müssen. Jedoch fällt es uns schwer, etwas mit dem Thema anzufangen, da es sich um Entwicklungfragen handelt, bei dem wir kaum Erfahrung haben.

#### 5.5.2.4 KW 20

| Datum | Tätigkeit | Bemerkung | Nächste Schritte |
|---|---|---|---|
| 14.05.2024 | - Dokumentation weiter geschrieben<br><br>- Testfälle vorbereitet | Diese Woche waren wir nicht im Homeschool. Deshalb konnten wir nicht an unserer Konfiguration weiterarbeiten. | - Dokumentation Fertigstellen |

*Tabelle 5 KW20 Paulo*

**Reflexion**

Diese Woche waren wir wieder in der Schule. Das erschwerte uns die Arbeit an unserer Umgebung, da wir keinen direkten Zugriff darauf haben. Wir verbrachten also diese Zeit damit, die Dokumentation weiter zu überarbeiten. Ich und David machten noch einen Feinschliff für die Webseite. Zudem erarbeiteten wir ein Testprotokoll, welches wir nächste Woche ausfüllen werden.
Die Abgabe der Realisierung konnten wir auf nächste Woche verschieben.

#### 5.5.2.5 KW 21

| Datum | Tätigkeit | Bemerkung | Nächste Schritte |
|---|---|---|---|
| 21.05.2024 | - Abgabe Realisierung<br><br>- Dokument Fertiggestellt<br><br>- Testprotokoll erarbeitet und fertiggestellt | Dokument überarbeitet und bis 23:59 abgeben<br><br><br>Testresultate Dokumentiert | Einführungsphase |

*Tabelle 6 KW21 Paulo*

**Reflexion**

Diese Woche ist die Abgabe der Realisierung und ich arbeitete mit dem Team zusammen an der Fertigstellung des Dokuments. Ich notierte das Testprotokoll, sowie unsere Beobachtungen nach der Ausführung der Tests. Zudem half ich dabei, letzte Konfigurationen zu machen, welche die Anmeldung auf die Applikation ermöglicht. Wir informierten unseren Stakeholder über die aktuelle Lage des Projekts und dass es so, wie wir es im Konzept geplant haben, nicht umsetzbar ist mit unseren Know-How. Wir haben unser Ziel etwas zu hoch gesetzt und dabei unsere Kenntnisse, welche benötigt werden, etwas vernachlässigt. Das nagt auch etwas an unserer Motivation, sodass die Arbeit mühsam wurde, da unser Ziel nicht ohne intensiven Aufwand erreichbar scheint.

## 5.5.3 Noah Isenschmid

### 5.5.3.1 KW 17

| Datum | Tätigkeiten | Zeit in H | Bemerkungen | Nächste Schritte |
|---|---|---|---|---|
| 23.04.2024 | -Dokumentation File erstellt<br>-Grundstruktur der Dokumentation erstellt<br>-Planung für das weitere Vorgehen für zuhause | 3 | Da wir unsere Infrastruktur bei mir zuhause haben konnten wir an diesem Tag keine Praktischen Arbeiten machen | Dokumentation, Umsetzung und Integration des Systems |

*Tabelle 7 KW17 Noah*

Heute ging es mit der Phase Realisierung los. Als Einleitung in die Realisierungsphase erhielten wir einen Input von unserem Stakeholder – Timo Steinlin, bezüglich der Art und Form der Realisierungsdokumentation. Als wir mit der Realisierungsphase unserer Projektarbeit begonnen haben, haben wir uns nicht direkt in die Umsetzung gestürzt, sondern ich als Projektleiter habe mein Team koordiniert und die Aufgaben aufgeteilt. Um das Projekt so gut wie nur möglich zu strukturieren. Ich habe heute vor allem die Dokumentation vorbereitet und die Formatierung, sowie die Gliederung erstellt. Unsere Dokumentation soll informativ, jedoch keinen Überfluss an unnötigen Informationen haben. Um dies zu erreichen, wollen wir zu unseren Bildern und Installationsdokumentationen kurze, prägnante, jedoch immer noch aussagekräftige Beschreibungen schreiben. Für die Arbeiten in der Realisierungsphase müssen wir von zu Hause arbeiten, da die Hardware bei uns zu Hause steht. Leider ist es uns nicht möglich, diese mit in die Schule zu nehmen oder ein VPN aufzubauen, da unser ISP (Sunrise) keine VPN-Verbindung auf dem Router unterstützt. Mit den Arbeiten von zu Hause können wir sicherstellen, dass unsere Produktivität nicht durch vermeidbare Umständlichkeit (z.B. Netzwerkausfall) behindert werden. Heute konnten wir noch nicht direkt mit der Realisierung beginnen, da wir wie gesagt die Hardware nicht vor Ort hatten. Technisch gesehen war es ein sehr unproduktiver Tag. Diesen hätte man im Voraus besser planen müssen. Das ist gewissermassen mein Fehler, da ich nicht damit gerechnet habe, dass wir nur drei Wochen Zeit haben für die Realisierungsphase. Abschliessend kann ich sagen, selbst wenn wir technisch nicht produktiv waren, konnten wir viel organisatorisches von unserer Pendenzenliste streichen. Nächste Woche beginnen wir mit der Realisierung. Der Umsetzung von flocker.

#### 5.5.3.2 KW 18

| Datum | Tätigkeiten | Zeit in H | Bemerkungen | Nächste Schritte |
|---|---|---|---|---|
| 30.04.2024 | - Konfiguration und Implementierung Docker und Portainer. Bereitstellung Bitwarden | 3 | - Docker i.O.<br>- Portainer i.O.<br>- Bitwarden i.O. | Installation und Konfiguration Authentik, Guacamole, Nginx und Proxmox |

*Tabelle 8 KW18 Noah*

Heute habe ich mich auf die Einrichtung und Konfiguration verschiedener Systeme konzentriert. Dazu gehören Docker, Portainer und Bitwarden. Docker und Portainer waren unkompliziert, wodurch eine solide Grundlage für die Containerisierung und einfache Verwaltung unserer Anwendungen geschaffen wurde. Bitwarden ist eine sichere Passwortverwaltungslösung, die für unsere Sicherheitsprotokolle wichtig ist.

Im nächsten Schritt werde ich Authentik, Guacamole, Nginx und Proxmox installieren. Diese Tools sind wichtig, um unsere Infrastruktur effizient und sicher zu machen. Authentik hilft uns, sicher zu sein. Sie hilft uns dabei, unsere Sicherheit besser zu machen. Guacamole ermöglicht es uns, Fernzugriff auf unsere Systeme zu ermöglichen. Nginx ist ein Web-Server. Er macht unsere Last besser und macht unser Netzwerk sicherer. Proxmox ermöglicht es uns, unsere virtuellen Maschinen effizient zu verwalten, was für die Skalierung unserer Ressourcen unerlässlich ist.

Diese Erfahrungen haben meine Fähigkeiten in der Systemadministration verbessert und mir geholfen, moderne IT-Infrastrukturtechnologien zu nutzen. Ich habe gelernt, flexibel zu bleiben und neue Tools einzusetzen, um unsere Systeme sicher und effizient zu machen. Diese Projekte haben mein technisches Wissen verbessert und meine Fähigkeit zur selbstständigen Arbeit verbessert. Es war eine gelungene Woche und wir konnten sauber, wenn auch ein bisschen holprig in die technische Umsetzung starten.

#### 5.5.3.3 KW 19

| Datum | Tätigkeiten | Zeit in H | Bemerkungen | Nächste Schritte |
|---|---|---|---|---|
| 07.05.2024 | - WordPress aufgesetzt und konfiguriert<br>- Authentik konfiguriert, Gruppen und Benutzer erstellt<br>- Arbeitsjournal geschrieben und weiteres Vorgehen<br>- Fehlersuche bei WordPress, Tunneling von Cloudflare und Authentik<br>- WordPress Seite konfiguriert<br>- Proxmox aufgesetzt | 3 | - Webseite flocker.ch ist nicht erreichbar<br>- Zeitdruck ist immens<br>- Projekt ist zu gross für so wenig Zeit | - Testing<br>- Virtualisierungshost aufsetzen<br>- Dokumentation abschliessen und abgeben |

*Tabelle 9 KW19 Noah*

Heute haben David und ich WordPress aufgesetzt, die Seite, auf der wir WordPress installieren werden. Es gab Probleme, weil der Fehler-Log nicht richtig war. Wir haben gesagt, dass es ein Problem mit dem Apache2-Server gibt. Wir benutzen keinen Apache2. Darum waren wir sehr verwirrt. Nach einiger Suche fand ich die Lösung in unserem Docker Compose File. Wir haben ein Netzwerk aus dem Docker Compose File gemacht. Das Netzwerk soll eine Plattform sein. Dann soll der Container direkt gestartet werden. Es ging nicht. Man musste das Docker Netzwerk manuell einrichten.

Einige Container sind aus unbekannten Gründen abgestürzt. Wir haben auch noch Fehler gefunden. Man konnte die Seite nicht mehr erreichen. Die Container funktionieren jetzt wieder gut. Ich habe Gruppen und Gruppen für Leute gemacht. Eine Admin-Gruppe für die Systemadministratoren, um alles in Authentik zu konfigurieren. Ich habe die Installation und Konfiguration von WordPress auf der Flocker-Installation erklärt. Ich habe das Arbeitsblatt weiter gemacht und die nächsten Schritte beschrieben. Die Webseite ist noch nicht von aussen erreichbar. Wir können das leider nicht machen. Cloudflare aktualisiert den DNS-Eintrag nicht und das dauert sehr lange. Die nächsten Schritte sind die Dokumentation zu erstellen, das Testing durchzuführen und den praktischen Teil abzuschliessen.

Es gibt ein Problem mit Proxmox. Deshalb setze ich eine VM auf, die nun Proxmox läuft. Ich lud eine ISO-Datei herunter und erstellte eine VM auf Proxmox. Proxmox funktioniert auch ohne Hardware.

#### 5.5.3.4 KW 20

| Datum | Tätigkeiten | Zeit in H | Bemerkungen | Nächste Schritte |
|---|---|---|---|---|
| 14.05.2024 | - Unterstützung Coding der HTML-Webseite<br>- Coding Java, API Proxmox VE<br>- Dokumentieren<br>- Arbeitsjournal weitergeführt | 3 | Kein Erfolg beim Java Code, viel zu komplex. | Dokumentation fertig stellen und erklären, warum das Projekt gescheitert ist. |

*Tabelle 10 KW20 Noah*

Heute habe ich David unterstützt, bei der Erstellung des Codes. Eine HTML-Seite in Proxmox erstellen. Wir mussten zuerst die Dokumentation von Proxmox VE API lesen und verstehen. Ohne Erfahrung in Python war das sehr schwierig. Wir haben bemerkt, dass das Backend am besten aus einem Springboot besteht. Deshalb mussten wir uns in Springboot einlesen und alles verstehen, was uns nicht gelang. Nach einigen Stunden haben wir gemerkt: Es ist schwer, mit Coding zu arbeiten. Gerade noch mit so wenig Zeit. Das grösste Problem war das grosse Projekt, bei dem wir nicht genug Erfahrung und Wissen haben.

Ich David ebenfalls unterstützt, die HTML-Seite zu designen und einen Teil davon umzusetzen. Wir zeigen eine Webseite. Mit dieser Webseite kann man eine VM in Proxmox machen. Da wir unsere Dokumentation und das flocker Video auf einer selbst gehosteten Webseite präsentieren wollen, banden wir die flocker App auf diese Webseite ein, denn dort war es noch komplizierter, ein Backend, API etc. umzusetzen.

Das Projekt wurde immer schwieriger, weil Apache Guacamole nicht funktioniert hat. Es gibt kein offizielles Image und Repo, um Apache Guacamole auf einem Raspberry Pi zu installieren. Also: Wir können keine VM machen. Und wir können nicht über das Internet auf eine Proxmox VM zugreifen. Somit sind die wichtigsten Funktionen / Ziele nicht erfüllt.

Als Projektleiter bereitet mir dieses Projekt leider mittlerweile ein bisschen kummer, da wir nicht mehr weiterwissen. Wir haben zurzeit diverse Probleme. Cloudflare macht Probleme bei der Domain-Validierung. Die Lieferschwierigkeiten mit dem Virtualisierungsserver, die API Programmierung. Ich denke, es gibt keine Möglichkeit, ein fertiges Produkt zu erstellen und zu präsentieren. Wir haben uns überschätzt in unseren Fähigkeiten und Möglichkeiten.

Ich werde nächste Woche die Dokumentation abschliessen und und den Webserver für unsere Webseite einrichten.

#### 5.5.3.5 KW 21

| Datum | Tätigkeiten | Zeit in H | Bemerkungen | Nächste Schritte |
|---|---|---|---|---|
| 21.05.2024 | - Dokumentation abgeschlossen<br>- Grammatik und Orthographieprüfung<br>- Beschrieb Problematik | 3 | Das Projekt konnte nicht zu unserer Gunsten fertiggestellt werden | Einführung |

*Tabelle 11 KW21 Noah*

Heute ist der letzte Tag der Realisierungsphase. Wir mussten feststellen, dass unser Projekt resultierend einfach nicht umsetzbar ist. Dieses Problem resultiert aus verschiedenen Faktoren, wie die Verzögerung des Liefertermins, der Fehlerquelle Cloudflare, viel zu wenig ausgeprägte Knowledge im API-Bereich. Ich muss ehrlich gestehen, dass ich ein bisschen enttäuscht bin, da wir dieses Projekt nicht gemäss unserem Konzept fertigstellen konnten.

Aus technischer Sicht es im Bereich API gescheitert. Wir hätten uns nicht zu sehr auf die API-Programmierung verlassen sollen. Grundsätzlich denke ich, dass das Projekt umsetzbar gewesen wäre, jedoch hätten wir jemand mit Knowledge in APIs gebraucht und dazu auch schlicht und einfach mehr Zeit.

Damit wir die Dokumentation auf unserer Webseite anzeigen lassen können, habe ich David dabei unterstützt für die Einrichtung einens Markdown Converter, Pandoc. Mit Pandoc ist es möglich Word und oder PDF in eine Markdown Language zu konvertieren und diese in den HTML-Code einzubinden.

## 5.5.4 Lucien Fleury

### 5.5.4.1 KW 17

| Datum | Tätigkeiten | Zeit in H | Bemerkungen | Nächste Schritte |
|---|---|---|---|---|
| 23.04.2024 | -Dokumentation File erstellt<br>-Grundstruktur der Dokumentation erstellt<br>-Planung für das weitere Vorgehen für zuhause | 3 | Da wir unsere Infrastruktur bei Noah zuhause haben konnten wir an diesem Tag keine Praktischen Arbeiten machen | -Anfangen mit dem Praktischen Teil<br>-Dokumentation weiterschreiben<br>-Arbeitsjournal weiterschreiben |

*Tabelle 12 KW17 Lucien*

Heute haben wir uns in der Schule mit der Realisierungsdokumentation beschäftigt. Wir haben angefangen, die Vorlage in das Dokument zu kopieren und den Text zu schreiben. Aber es gab ein Problem mit dem Layout. Die Überschriften funktionierten nicht so wie wir es wollten. Das war ein bisschen schwierig, aber wir haben versucht, es zu lösen. Wir haben uns alle darüber gesprochen, wie wir das Layout besser machen können. Ein paar Ideen hatten wir schon, aber es war schwer, sie umzusetzen. Zum Glück haben wir dann und wir haben einige Tipps bekommen, wie wir das Problem lösen können. Es hat eine Weile gedauert, aber am Ende haben wir eine Lösung gefunden, die uns allen gefallen hat.
Danach habe ich noch einen Antrag geschrieben. Darin habe ich darum gebeten, von zu Hause aus arbeiten zu dürfen, weil es für uns in der Gruppe viel einfacher geht, weil wir die ganze Hardware zuhause haben.
Insgesamt war der Nachmittag gut, obwohl wir die kleinen Probleme hatten. Aber am Ende haben wir eine Lösung gefunden und ich habe meinen Antrag geschrieben.

### 5.5.4.2 KW 18

| Datum | Tätigkeiten | Zeit in H | Bemerkungen | Nächste Schritte |
|---|---|---|---|---|
| 30.04.2024 | -Angefangen mit der Webseite<br>-Dokumentation weitergeschrieben<br>-Arbeitsjournal geschrieben | 3 | Noah hatte den Server noch nicht bei sich zuhause. Deshalb konnten wir das richtige Projekt noch nicht am richtigen Ort starten | -Weitermachen im Praktischen<br>-Weiter in der Dokumentation<br>-Arbeitsjournal weiterführen |

*Tabelle 13 KW18 Lucien*

Heute haben wir zum ersten Mal von zu Hause ausgearbeitet und uns direkt an die Aufgabe gemacht, den Server aufzusetzen. Leider hatten wir ein kleines Hindernis: Noah hatte den echten Server noch nicht zur Verfügung. Deshalb haben wir uns zunächst auf die Erstellung der Webseite konzentriert und nur einen kleinen Teil des eigentlichen Servers aufgesetzt. Aus diesem Grund konnten wir in dieser Woche mit dem Server nicht so weit kommen, wie wir es uns vorgenommen hatten. Dennoch war es ein guter Start und wir haben das Beste aus der Situation gemacht. Das Arbeiten von zu Hause aus hatte auch seine Vorteile – es war eine interessante Erfahrung und eine neue Art der Zusammenarbeit für uns.

#### 5.5.4.3 KW 19

| Datum | Tätigkeiten | Zeit in H | Bemerkungen | Nächste Schritte |
|---|---|---|---|---|
| 07.05.2024 | -Dokumentation geschrieben<br>-Webseite verfeinert<br>-Praxis im Projekt weitergemacht | 3 | Das wir Noah server immernochnicht zur Verfügung hatten konnten wir noch nicht mit diesem Beginnen | - Weitermachen im Praktischen<br>-Weiter in der Dokumentation<br>-Arbeitsjournal weiterführen |

*Tabelle 14 KW19 Lucien*

In dieser Woche habe ich mich weiterhin auf die Dokumentation konzentriert, während meine Teamkollegen an der Webseite und der allgemeinen Infrastruktur arbeiteten. Ich hatte keine grösseren Probleme, jedoch bereitete uns die Webseite gelegentlich kleinere Schwierigkeiten. Das Hauptproblem bestand jedoch darin, dass Noahs Server immer noch nicht verfügbar war.
Daher mussten wir vorläufig den Server auf unseren privaten Geräten installieren, was suboptimal war, aber uns dennoch erlaubte, einige Fortschritte zu machen. Trotz dieser Hindernisse war die Teamarbeit effektiv, und jeder trug seinen Teil dazu bei, das Projekt voranzutreiben. Am Ende der Woche hatten wir wichtige Teile der Dokumentation fertiggestellt und die Grundlage für die nächsten Schritte gelegt.

#### 5.5.4.4 KW 20

| Datum | Tätigkeiten | Zeit in H | Bemerkungen | Nächste Schritte |
|---|---|---|---|---|
| 14.05.2024 | Dokumentation weitergeführt | 3 | -Noah konnte den Server noch immer nicht organisieren | Infrastruktur |

*Tabelle 15 KW20 Lucien*

Da wir diese Woche nicht von zu Hause ausarbeiten konnten, haben wir uns ausschliesslich auf die Dokumentation konzentriert. Dabei konnten wir viele Punkte nachschreiben, die wir bereits umgesetzt, aber noch nicht in die Dokumentation aufgenommen hatten.
Es war eine produktive Zeit, in der wir wichtige Informationen strukturiert und festgehalten haben, um das Gesamtprojekt transparenter und besser dokumentiert zu machen.

#### 5.5.4.5 KW 21

| Datum | Tätigkeiten | Zeit in H | Bemerkungen | Nächste Schritte |
|---|---|---|---|---|
| 21.05.2024 | -Dokumentation fertiggestellt<br>-Praxisarbeiten fertiggestellt<br>-Arbeitsjournal fertiggestellt | 3 | Wir konnten trotz all diesen Fehler unser Projekt abschliessen. | - |

*Tabelle 16 KW21 Lucien*

In dieser Woche waren wir voll im Endspurt für das Projekt. Während meine Kollegen sich um die Technik kümmerten, habe ich mich darauf konzentriert, die Dokumentation zu verbessern. Es gab noch ein paar wichtige Punkte, die fehlten oder überarbeitet werden mussten, um das Projekt vollständig darzustellen.
Trotz einiger Probleme mit bestimmten Systemen haben wir hart gearbeitet, um alles zu vervollständigen. Dank unserer guten Zusammenarbeit konnten wir die fehlenden Teile erfolgreich ergänzen.
Obwohl es einige Schwierigkeiten gab, sind wir stolz darauf, dass wir am Ende ein gutes Ergebnis erzielen konnten. Es war eine spannende Fahrt, aber wir haben das Projekt erfolgreich abgeschlossen.

# 6 Einführungsbericht

## 6.1 Zusammenfassung

Die Einführung beschreibt die von uns aufgebaute Umgebung. Da das Projekt nicht vollumfänglich ungesetzt werden konnte, durch verschiedene Faktoren, werden hier die Elemente / Komponente beschrieben, welche funktionstüchtig sind.

Das Dokument bietet sowohl einen Überblick über das Projekt wie auch einen kleinen Einblick in das Technische.

Neben dem Einführungsplan enthält dieses Dokument einen Ausbildungsplan für die Benutzer und Administratoren, einen Akzeptanztest, in welchem die Abnahme und die Testresultate dokumentiert sind und eine Zusammenfassung der ganzen Projektplanung.

## 6.2 Pläne

### 6.2.1 Einführungsplan

Unsere Virtualisierungslösung löst kein Altsystem ab und ist ein reines Projekt für das Modul 306. Eine Migration von bestehenden Daten ist daher nicht notwendig und wird auch nicht auf einem Produktiven System eingesetzt.

Unsere Lösung dient als Alternative zum lokalen Server-Hosting und wird Online zugänglich sein.

Um die Nutzung der Virtualisierungsplattform zu gewährleisten, sind folgende Schritte notwendig:

1. Anmeldung/Registrierung bei der von uns bereitgestellten Onlinelösung.
2. Aufsetzen der gewünschten Virtuellen Maschinen
3. Die gewünschten Dienste direkt in den virtuellen Maschinen installieren oder ein vorbereitets Image bereitstellen.

### 6.2.2 Migrationsplan

Wir haben keine vorhandenen Daten, welche wir migrieren müssen. Es ist also kein Migrationsplan nötig.

### 6.2.3 Ausbildungsplan

Unser Ausbildungsplan ist in zwei Bereiche gegliedert. Der Anwender- und der Administratorschulung.

## 6.3 Schulungen

### 6.3.1 Anwenderschulung

Als Anwender sehen wir alle User, welche sich auf die fertigen virtuellen Maschinen verbinden werden. Eine Schulung ist von unserer Seite nicht notwendig. Lediglich soll der Anwender wissen, wie man auf gewünschte Services zugreift, welche auf den Maschinen gehostet werden.

### 6.3.2 Administrationsschulung

Als Administratoren sehen wir alle User, welche unsere Lösung direkt verwenden und virtuelle Maschinen in Betrieb nehmen.
Die Dokumentationen dienen als Basis für alle Funktionen.
Ist wenig bis kein Wissen über virtuelle Maschinen vorhanden, soll ein Kurs in die Wege geleitet werden, welche die Thematik genauer erklärt. Wichtig dabei ist das Verständnis über die Schnittstellen der virtuellen Maschinen, damit Services zugänglich gemacht werden können.

### 6.3.3 Akzeptanztest

Der Akzeptanztest des Kunden wird anhand der Testspezifikationen (Realisierungsbericht 2.1 Testspezifikationen) durchgeführt. Diese ist jedoch noch ausstehend.

## 6.4 Testprotokolle

| Nr. | Testfall | Testbeschreibung | Testschritte | Erwartetes Ergebnis |
|---|---|---|---|---|
| 1 | Funktionalität ohne kostenpflichtige Methoden | Überprüfen der Verfügbarkeit und Funktionalität der Flocker Webapplikation ohne die Notwendigkeit kostenpflichtiger Abonnements oder ähnlicher Zahlungsmethoden. | 1. Öffnen der Flocker Webapplikation.<br>2. Versuch, auf alle Hauptfunktionen zuzugreifen, ohne Zahlungsinformationen einzugeben.<br>3. Durchführung typischer Aktionen wie Dateiupload, Datenbearbeitung und Datenspeicherung.<br>4. Überprüfung, ob sämtliche Funktionen ohne die Notwendigkeit von Zahlungsinformationen zugänglich sind.<br>5. Abschluss einer typischen Sitzung ohne Eingabe von Zahlungsinformationen. | Alle Hauptfunktionen der Webapplikation sind ohne die Eingabe von Zahlungsinformationen zugänglich und voll funktionsfähig. |
| 2 | Benutzerfreundlichkeit der Webseite | Überprüfen der Benutzerfreundlichkeit und des Designs der Flocker Webseite. | 1. Bewertung der Startseite und des Navigationsmenüs auf Einfachheit und Klarheit.<br>2. Überprüfung der Anordnung von Schaltflächen und Funktionen auf Benutzerfreundlichkeit.<br>3. Testen der Suchfunktion, um zu sehen, wie leicht Benutzer relevante Informationen finden können.<br>4. Durchführung von typischen Aktionen gemäss den Mockups, um sicherzustellen, dass die tatsächliche Nutzung dem geplanten Design entspricht. | Die Webseite ist leicht navigierbar, weist ein minimalistisches und modernes Design auf und bietet eine intuitive Benutzererfahrung. |

| | | | | |
|---|---|---|---|---|
| 3 | Sicherheit und Lokalität der Daten | Überprüfen, ob sämtliche Daten und Systeme der Flocker Webapplikation ausschliesslich in der Schweiz gehostet werden und die Datenschutzbestimmungen eingehalten werden. | 1. Überprüfung der Standortangaben der Server und Datenbanken.<br>2. Verfolgung der Datenflüsse und Überwachung der Netzwerkkommunikation.<br>3. Sicherstellung, dass keine Daten ausserhalb der Schweiz gespeichert oder übertragen werden.<br>4. Überprüfung der Datenschutzrichtlinien, um den Standort der Datenspeicherung zu bestätigen. | Alle Daten und Systeme sind ausschliesslich in der Schweiz gehostet, um den Datenschutzanforderungen zu genügen. |
| 4 | Leistung und Verfügbarkeit der Hardware | Überprüfen der Leistung und Verfügbarkeit der bereitgestellten Hardware für die Flocker Webapplikation. | 1. Überprüfung der Hardware-Spezifikationen auf Konformität mit den Anforderungen.<br>2. Testen der Server- und Netzwerkverfügbarkeit nach der Installation.<br>3. Durchführung von Leistungstests, um sicherzustellen, dass die Hardware den Anforderungen der Webapplikation entspricht.<br>4. Überprüfung der Sicherheitsvorkehrungen und Zugangsbeschränkungen für die gehostete Hardware. | Die bereitgestellte Hardware ist korrekt konfiguriert, verfügbar und erfüllt die Leistungsanforderungen der Webapplikation. |
| 5 | Wissensstand und Kompetenz des Teams | Überprüfen, ob das Entwicklungsteam über das erforderliche Wissen und Know-how für die Entwicklung, Anpassung und Wartung der Flocker Webapplikation verfügt. | 1. Bewertung der Qualifikationen und Erfahrungen der Teammitglieder im Bereich der Webentwicklung.<br>2. Durchführung von Code-Reviews und Überprüfung der Entwicklungsumgebung.<br>3. Testen der Teamkommunikation und Zusammenarbeit bei der Lösung von Problemen.<br>4. Überprüfung der Dokumentation und Schulungsunterlagen zur Sicherstellung einer kontinuierlichen Wissensvermittlung. | Das Entwicklungsteam verfügt über das erforderliche Wissen und Know-how für die Entwicklung, Anpassung und Wartung der Webapplikation. |

| | | | | |
|---|---|---|---|---|
| 9 | Externe Systemintegration | Überprüfen der Integration mit externen Systemen, die von der Flocker Webapplikation genutzt werden. | 1. Durchführung von Aktionen, die die Interaktion mit externen Systemen erfordern, z. B. Authentifizierung über Drittanbieter oder Datenimport/-export.<br>2. Überwachung der Kommunikation mit externen Systemen und Überprüfung der Datenintegrität.<br>3. Testen von Fehlerbehandlungsszenarien wie Ausfällen externer Dienste.<br>4. Überprüfung der Sicherheit der Datenübertragung und der Datenschutzrichtlinien im Zusammenhang mit externen Systemen. | Die Integration mit externen Systemen funktioniert wie erwartet und gewährleistet die Sicherheit und Integrität der übertragenen Daten. |

## 6.5 Testresultate

| Nr | Testfall | Erwartetes Ergebnis | Tatsächliches Ergebnis |
|---|---|---|---|
| 1 | Funktionalität ohne kostenpflichtige Methoden | Alle Hauptfunktionen der Webapplikation sind ohne die Eingabe von Zahlungsinformationen zugänglich und voll funktionsfähig. | Die Webapplikation ist komplett kostenlos zu bedienen. |
| 2 | Benutzerfreundlichkeit der Webseite | Die Webseite ist leicht navigierbar, weist ein minimalistisches und modernes Design auf und bietet eine intuitive Benutzererfahrung. | Die Seite ist einfach gehalten und bietet lediglich ein kleines Menü für die Seiten an. |
| 3 | Sicherheit und Lokalität der Daten | Alle Daten und Systeme sind ausschliesslich in der Schweiz gehostet, um den Datenschutzanforderungen zu genügen. | Alle Dienste sind in der Schweiz gehostet. |
| 4 | Leistung und Verfügbarkeit der Hardware | Die bereitgestellte Hardware ist korrekt konfiguriert, verfügbar und erfüllt die Leistungsanforderungen der Webapplikation. | Die Hardware ist fähig, die Applikation auszuführen und zu betreiben. |
| 5 | Wissensstand und Kompetenz des Teams | Das Entwicklungsteam verfügt über das erforderliche Wissen und Know-how für die Entwicklung, Anpassung und Wartung der Webapplikation. | Das Team hat zu wenig Knowhow über die API-Schnittstelle. Die Dienste aufzusetzen und zu konfigurieren, beherrscht das Team. |
| 6 | Integration der Benutzerschnittstelle (UI) mit Backend-Services | Die Benutzerschnittstelle integriert sich nahtlos mit den Backend-Services, und Daten werden korrekt zwischen den Systemen ausgetauscht. | Die API-Schnittstelle zwischen Backend und Applikation steht nicht. Das Knowhow ist nicht vorhanden, um diese korrekt einzurichten. |
| 7 | API-Funktionalität und -Sicherheit | Die APIs funktionieren wie erwartet, sind sicher und bieten angemessene Authentifizierungs- und Autorisierungsmethoden. | Siehe oben. Authentik an sich funktioniert, jedoch ist es nicht möglich, eine VM über das Web-Interface zu erstellen. |
| 8 | Datenbankintegration und -zugriff | Die Datenbankintegration ist stabil, und Daten werden korrekt zwischen der Benutzerschnittstelle und dem Backend ausgetauscht. | Siehe oben |
| 9 | Externe Systemintegration | Die Integration mit externen Systemen funktioniert wie erwartet und gewährleistet die Sicherheit und Integrität der übertragenen Daten. | Die Daten sind momentan nur auf einer abgetrennten Server-Umgebung gespeichert und ermöglicht nur Zugriff über direkte Verbindungen. Daher sind die Daten sicher. |

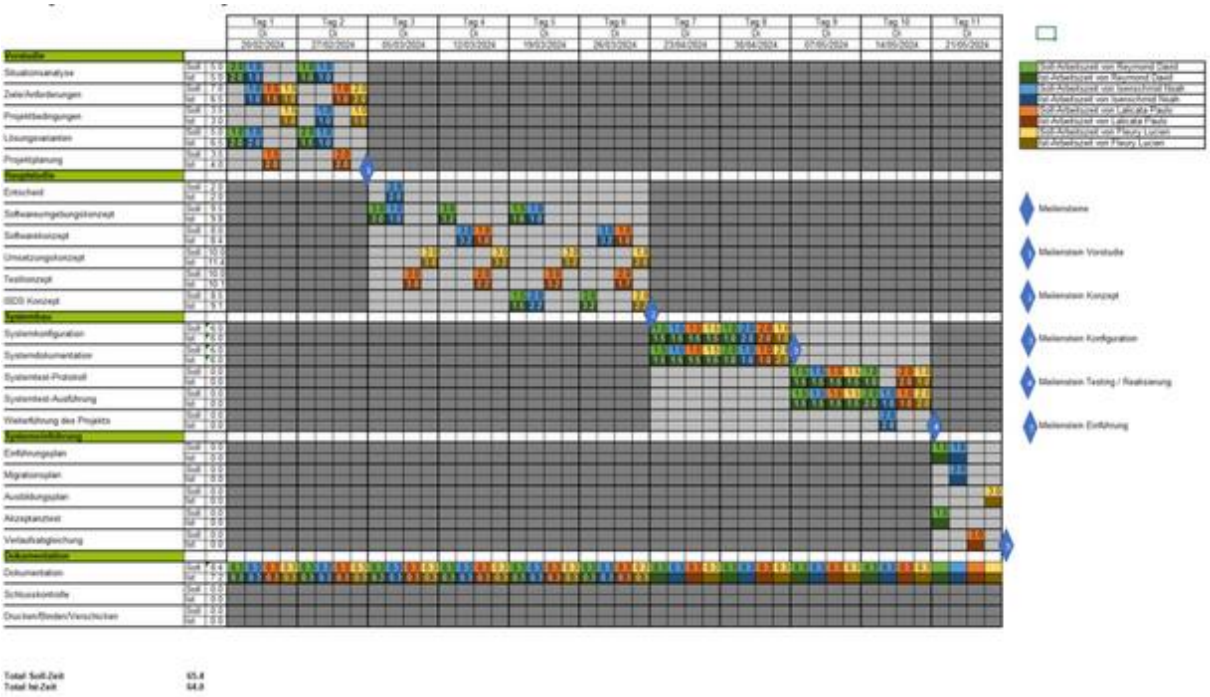
# 6.6 Zusammenfassung der Projektplanung

## 6.6.1 Termine

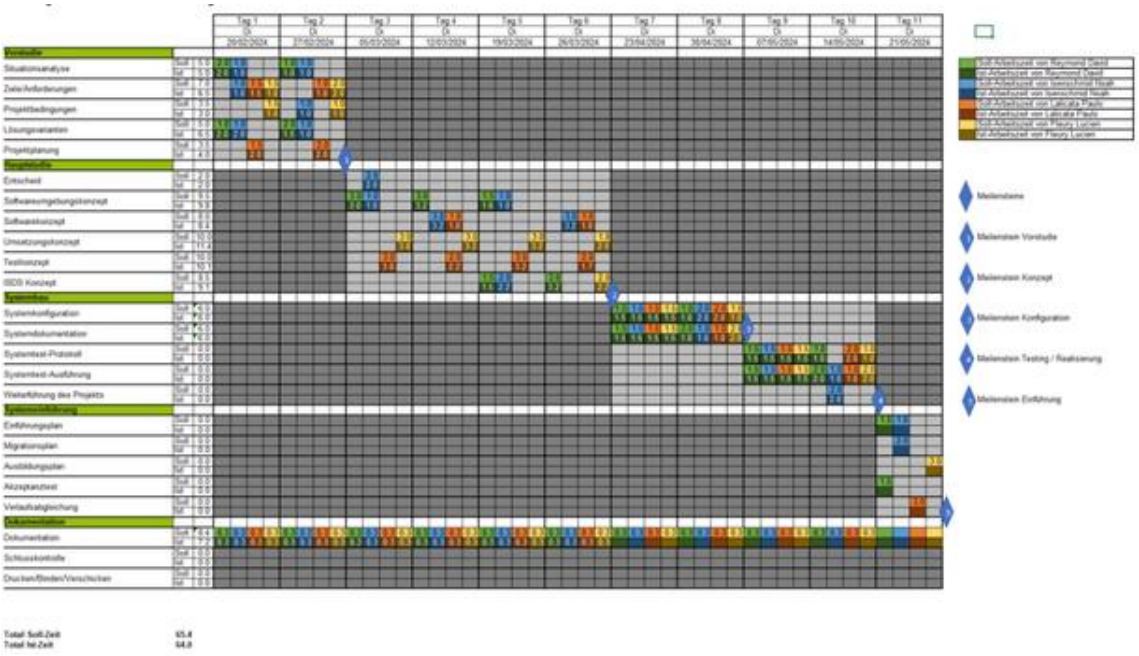
Die Abgabetermine wurden durch das ganze Projekt eingehalten.

Jedoch konnten die geplanten Zeiten in der Realisierungsphase nicht ganz eingehalten werden, wir brauchten die «Pufferwoche», da es für das Projekt allgemein noch mehr Zeit brauchte.

## 6.6.2 Geplanter Zeitplan



## 6.6.3 Effektiver Zeitplan



## 6.6.4 Ergebnisse

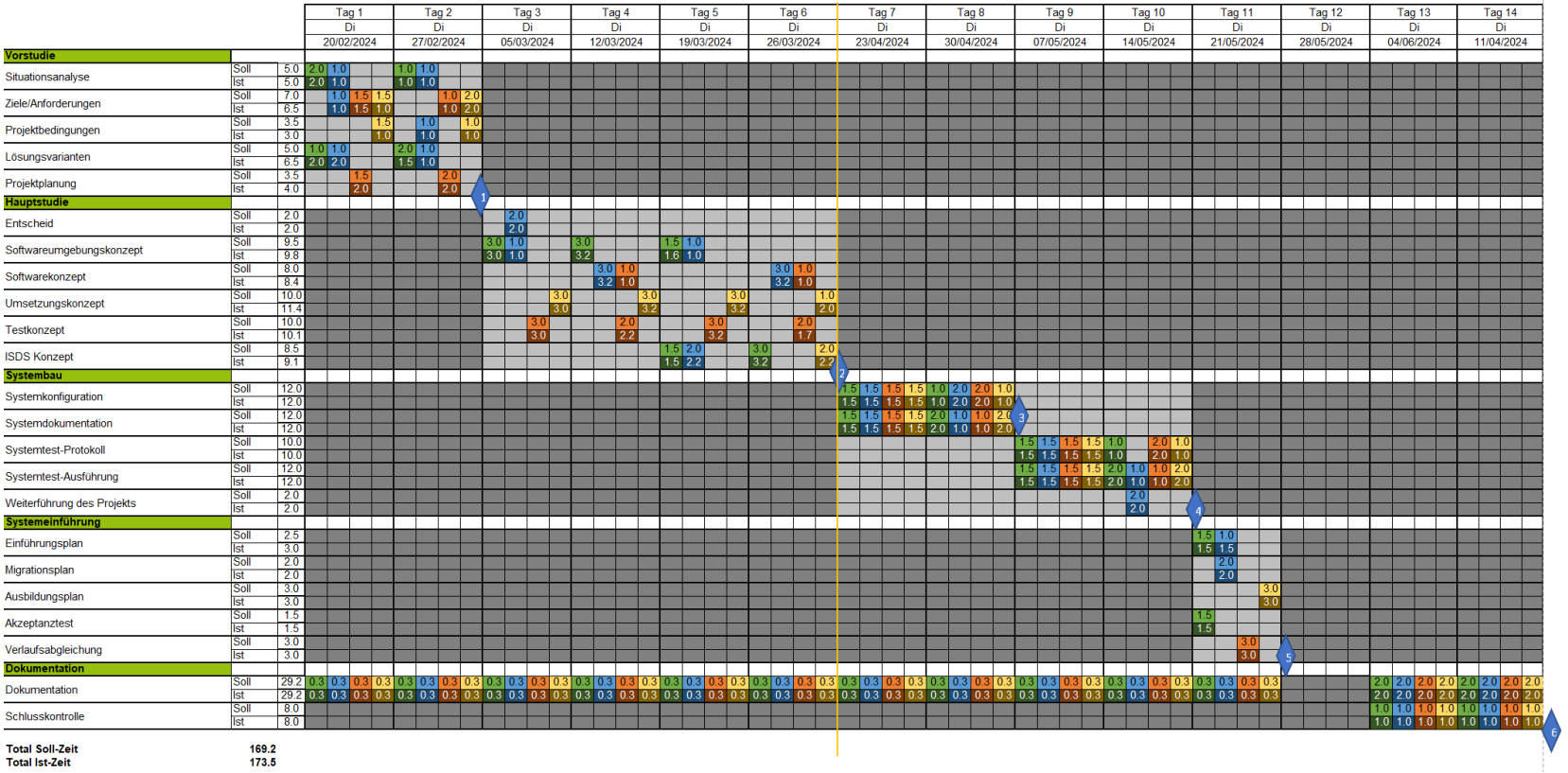
Die Ergebnisse weichen von den Anforderungen ab bzw. sind teilweise nicht ganz erfüllt. Der Grund ist, dass wir das Projekt nicht so umsetzen konnten wie Anfangs geplant.

Die Anforderungen, welche definiert worden sind, sind folgende:

| ID | Anforderung | Ziel |
|---|---|---|
| A1 | Unsere Webapplikation muss ohne Subscription oder anderen ähnlichen Gewinn einbringenden Methoden verwendet werden können. | Wir halten die Webapplikation für den Benutzer kostenfrei. Ziel (Z1) |
| A2 | Die Webseite / Webapplikation soll leicht und einfach zu bedienen sein. Es ist benutzerfreundlich gestaltet und es hat nicht zu viele «unnötige» Funktionen. | Mockups sind bis zur Realisierungsphase erstellt. Die Mockups zeigen den genauen Aufbau der Webapplikation, diese sind leicht überschaubar und es sind keine komplexen Menüs und Untermenüs vorhanden. Ziel (Z4) |
| A3 | Alle VMs, Systeme, Daten und so weiter, sind ausschliesslich in der Schweiz und auch nur in der Schweiz verfügbar. Daten VMS usw. dürfen nicht ausserhalb der Schweiz gelagert / vorhanden sein. | Um eine Datensicherheit zu gewährleisten, haben wir entschieden, jegliche Daten, Dienste und Dazugehörigkeiten in der Schweiz auf unseren eigenen Systemen zu hosten. Wir verzichten somit auf die Unterstützung externen Datacenter-Anbieter. So können wir eine strikte Einhaltung des Datenschutzgesetzes (DSG, DSGVO, revDSG) gewährleisten und können das SCVL3 elimieren. Ziel (Z5) |
| A4 | Die Hardware stellt David Reymond und Noah Isenschmid zur Verfügung. Das ganze Projekt wird zu Hause gehosted und bereitgestellt. | Die Hardware wird vor der Realisierungsphase aufgesetzt und konfiguriert. Ziel (Z3) |
| A5 | Das Projekt wird mit dem nötigen Wissen und know how erstellt und geleitet. Wir bilden uns in der Applikationsentwicklung weiter, damit das Projekt realisiert werden kann. | Bis zu der Realisierungsphase des Projektes haben wir uns alle das nötige Wissen für die Applikationsentwicklung, welche es im Rahmen des Projektes benötigt, angeeignet. Ziel (Z1) |

Abweichungen zu den Anforderungen gibt es bei A4 und A5. Bei A4, weil wie den Virtualisierungsserver nicht wie geplant erhalten haben. Bei A5, weil wir das nötige Knowhow nicht hatten, um das Projekt umzusetzen.

# 7 Zeitplan


Tag 1 Di 20/02/2024
Tag 2 Di 27/02/2024
Tag 3 Di 05/03/2024
Tag 4 Di 12/03/2024
Tag 5 Di 19/03/2024
Tag 6 Di 26/03/2024
Tag 7 Di 23/04/2024
Tag 8 Di 30/04/2024
Tag 9 Di 07/05/2024
Tag 10 Di 14/05/2024
Tag 11 Di 21/05/2024
Tag 12 Di 28/05/2024
Tag 13 Di 04/06/2024
Tag 14 Di 11/04/2024
Vorstudie
Situationsanalyse Soll 5.0 Ist 5.0
Ziele/Anforderungen Soll 7.0 Ist 6.5
Projektbedingungen Soll 3.5 Ist 3.0
Lösungsvarianten Soll 5.0 Ist 6.5
Projektplanung Soll 3.5 Ist 4.0
Hauptstudie
Entscheid Soll 2.0 Ist 2.0
Softwareumgebungskonzept Soll 9.5 Ist 9.8
Softwarekonzept Soll 8.0 Ist 8.4
Umsetzungskonzept Soll 10.0 Ist 11.4
Testkonzept Soll 10.0 Ist 10.1
ISDS Konzept Soll 8.5 Ist 9.1
Systembau
Systemkonfiguration Soll 12.0 Ist 12.0
Systemdokumentation Soll 12.0 Ist 12.0
Systemtest-Protokoll Soll 10.0 Ist 10.0
Systemtest-Ausführung Soll 12.0 Ist 12.0
Weiterführung des Projekts Soll 2.0 Ist 2.0
Systemeinführung
Einführungsplan Soll 2.5 Ist 3.0
Migrationsplan Soll 2.0 Ist 2.0
Ausbildungsplan Soll 3.0 Ist 3.0
Akzeptanztest Soll 1.5 Ist 1.5
Verlaufsabgleichung Soll 3.0 Ist 3.0
Dokumentation
Dokumentation Soll 29.2 Ist 29.2
Schlusskontrolle Soll 8.0 Ist 8.0
Total Soll-Zeit 169.2
Total Ist-Zeit 173.5
Soll-Arbeitszeit von Reymond David
Ist-Arbeitszeit von Reymond David
Soll-Arbeitszeit von Isenschmid Noah
Ist-Arbeitszeit von Isenschmid Noah
Soll-Arbeitszeit von Lalicata Paulo
Ist-Arbeitszeit von Lalicata Paulo
Soll-Arbeitszeit von Fleury Lucien
Ist-Arbeitszeit von Fleury Lucien
Meilensteine
1 Meilenstein Vorstudie
2 Meilenstein Konzept
3 Meilenstein Konfiguration
4 Meilenstein Testing / Realisierung
5 Meilenstein Einführung
6 Meilenstein Schluss

# 8 Schlussbericht

## 8.1 Vergleich Ist/soll

Die Ergebnisse weichen von den Anforderungen ab bzw. sind teilweise nicht ganz erfüllt. Der Grund ist, dass wir das Projekt nicht so umsetzen konnten wie Anfangs geplant.
Die Anforderungen, welche definiert worden sind, sind folgende:

| ID | Anforderung | Ziel |
|---|---|---|
| A1 | Unsere Webapplikation muss ohne Subscription oder anderen ähnlichen Gewinn einbringenden Methoden verwendet werden können. | Wir halten die Webapplikation für den Benutzer kostenfrei. Ziel (Z1) |
| A2 | Die Webseite / Webapplikation soll leicht und einfach zu bedienen sein. Es ist benutzerfreundlich gestaltet und es hat nicht zu viele «unnötige» Funktionen. | Mockups sind bis zur Realisierungsphase erstellt. Die Mockups zeigen den genauen Aufbau der Webapplikation, diese sind leicht überschaubar und es sind keine komplexen Menüs und Untermenüs vorhanden. Ziel (Z4) |
| A3 | Alle VMs, Systeme, Daten und so weiter, sind ausschliesslich in der Schweiz und auch nur in der Schweiz verfügbar. Daten VMS usw. dürfen nicht ausserhalb der Schweiz gelagert / vorhanden sein. | Um eine Datensicherheit zu gewährleisten, haben wir entschieden, jegliche Daten, Dienste und Dazugehörigkeiten in der Schweiz auf unseren eigenen Systemen zu hosten. Wir verzichten somit auf die Unterstützung externen Datacenter-Anbieter. So können wir eine strikte Einhaltung des Datenschutzgesetzes (DSG, DSGVO, revDSG) gewährleisten und können das SCVL3 elimieren. Ziel (Z5) |
| A4 | Die Hardware stellt David Reymond und Noah Isenschmid zur Verfügung. Das ganze Projekt wird zu Hause gehosted und bereitgestellt. | Die Hardware wird vor der Realisierungsphase aufgesetzt und konfiguriert. Ziel (Z3) |
| A5 | Das Projekt wird mit dem nötigen Wissen und know how erstellt und geleitet. Wir bilden uns in der Applikationsentwicklung weiter, damit das Projekt realisiert werden kann. | Bis zu der Realisierungsphase des Projektes haben wir uns alle das nötige Wissen für die Applikationsentwicklung, welche es im Rahmen des Projektes benötigt, angeeignet. Ziel (Z1) |

Abweichungen zu den Anforderungen gibt es bei A4 und A5.
Bei A4, weil wir den Virtualisierungsserver nicht wie geplant erhalten haben.
Bei A5, weil wir das nötige Knowhow nicht hatten, um das Projekt umzusetzen.

## 8.2 Mittelbedarf

Wir bedienten uns an verschiedenen Mitteln, wir hatten folgendes zur Verfügung:
Raspberry Pi Model 4 mit 8GB RAM
Opensource Software: Authentik, Docker, Portainer, Proxmox

## 8.3 Fazit zum Projekt

Wir konnten Teile vom Projekt abschliessen und andere wiederum nicht.
Der Teil mit der Virtualisierung hatte nicht ganz so funktioniert, weil uns das Know-How der Applikationsentwicklung fehlte. Es fiel uns schwer, API, Coding etc. uns anzueignen und es in dieser Zeit umzusetzen. Das Projekt war grundsätzlich zu gross skaliert, die ganzen Anforderungen und Ziele waren zu komplex, um diese in dieser Zeit umzusetzen. Jedoch konnten wir alle, Neues über die Projektmanagement Methode lernen und wissen nun, wie sich die IPA in etwa anfühlen wird.
Es ist eher eine gute Vorbereitung für die Probe IPA.
Dass das mit dem physischen Virtualisierungsserver nicht funktioniert hatte, frustrierte uns, da uns dadurch eine ganze Komponente fehlte und somit unser geplantes «Netzwerk» nicht vollständig war.
Wir haben zwar kein fertiges Produkt, welches wir abgeben / präsentieren können, dafür eine saubere und ausführliche Dokumentation, in welcher wir Erfahrungen sammeln konnten.

## 8.4 Persönliches Fazit

Unser persönliches Fazit ist, dass wir zufrieden mit der Dokumentation sind.
Beim Projekt sehen wir aber noch ganz klar Verbesserungspotenzial. Uns allen war das Arbeitsjournal eine «Lehre» in dem Sinne, wie man ein Arbeitsjournal zu schreiben hat und was ein gutes Arbeitsjournal ausmacht.
Durch das Projekt konnten wir als Team in verschiedenen Rollen arbeiten und so ein besseres Gefühl für HERMES bekommen.
Damit die Motivation durch das ganze Projekt gleich bleibt würden wir beim nächsten Mal das Projekt anders skalieren und uns vorher überlegen, wie gross wir es machen wollen.

## 8.5 Schlussreflexion

Abschliessend können wir sagen, dass das ganze Projekt / die Phasen lehrreich war. Was wir definitiv besser machen hätten können, wäre folgendes:

- Kein grosser Coding Teil, bei welchem wir uns viel Wissen aneignen müssen.
- Projekt kleiner skalieren, nach dem Motto weniger ist mehr.
- Klarere Vision vom Projekt selbst, wie realistisch ist es wirklich, kann man es in der Zukunft noch ausbauen.