

# Integration eines Messenger-Servers für inner- und zwischenbetriebliche Kommunikation

## Projektdokumentation

**Auszubildender:**

Wilko Reins  
XXXXXXXXXX  
XXXXXXXXXX  
XXXXXXXXXX

**Ausbildungsbetrieb:**

SOL.Service Online  
GmbH & Co. KG  
Rheinstraße 49  
26382 Wilhelmshaven

**Ausbildungsberuf:**

Fachinformatiker für  
Systemintegration

**Dokument:**

18 Seiten Dokumentation  
14 Seiten Anhang

**Projektzeitraum:**

29.03.2022 – 05.04.2022  
Azubi-Ident-Nr.: XXXXXXXXXXXX  
Prüflingsnummer: XXXXX

## Inhaltsverzeichnis

1	Tabellenverzeichnis.....	III
2	Abbildungsverzeichnis.....	IV
3	Vorwort.....	1
4	Auftragsbeschreibung .....	1
4.1	Projektumfeld.....	1
4.2	Projektauftrag .....	1
5	Definitionsphase.....	1
5.1	Analyse des Ist-Zustandes .....	1
5.2	Sollkonzept.....	2
6	Planungsphase .....	2
6.1	Zu beachtende Datenschutzvorschriften .....	2
6.2	Vorstellung potentieller Softwarelösungen.....	3
6.2.1	Microsoft Teams.....	3
6.2.2	Matrix .....	3
6.2.3	Mattermost .....	4
6.2.4	Rocket.Chat .....	4
6.3	Ermittlung der Wirtschaftlichkeit – Kostenaufstellungen .....	4
6.3.1	Kosten/Nutzenanalyse der Softwarelösungen.....	6
6.4	Auswahl des Kommunikationsmittels.....	6
6.5	Hosting-Anbieter .....	7
6.5.1	Vorstellung potentieller Hosting-Anbieter .....	7
6.5.2	Nutzwertanalyse der Hosting-Anbieter.....	10
6.5.3	Auswahl des Hosting-Anbieters .....	10
6.6	Systemübersicht .....	11
6.7	Projektablaufplan .....	11
6.8	Analyse der Personalkosten .....	11
7	Durchführungsphase .....	11
7.1	Vorbereiten des Servers .....	11
7.2	Installation benötigter Dienste.....	11
7.3	Einrichtung der Docker-Umgebung.....	11
7.3.1	Anlegen der Verzeichnisse.....	12
7.3.2	Anlegen der Docker-Compose Datei.....	12
7.4	Erstellen der übrigen Firewall-Regeln.....	14
7.5	Generieren des SSL-Zertifikats.....	14
7.6	Konfigurationen anpassen .....	14
7.7	Testen der Konfigurationen .....	15

## Integration eines Messenger-Servers

7.7.1	Fehlerbehebung: TURN-Server startet nicht korrekt .....	16
7.7.2	Fehlerbehebung: TURN-Server startet nicht mit SSL .....	16
8	Abschlussphase .....	16
8.1	Soll-/Ist-Vergleich.....	16
8.1.1	Kostenvergleich.....	16
8.1.2	Zeitliche Abweichungen .....	17
8.1.3	Änderungen gegenüber dem Projektantrag .....	17
8.2	Anwenderdokumentation .....	17
8.3	Abnahme des Projekts.....	17
8.4	Fazit.....	17
9	Anhang.....	I
9.1	Glossar .....	I
9.2	Abbildungen.....	II
9.3	Tabellen.....	VII
9.4	Anwenderdokumentationen .....	IX
9.4.1	Anwenderdokumentation für Benutzer .....	IX
9.4.1.1	Anmelden .....	IX
9.4.1.2	Registrierung .....	X
9.4.1.3	Erste Schritte in Element .....	X
9.4.2	Anwenderdokumentation für Administratoren .....	XII
9.4.2.1	Starten und Stoppen der Docker-Instanzen.....	XII
9.4.2.2	Status der Docker-Instanzen einsehen.....	XII
9.5	Quellenangaben .....	XIII

## 1 Tabellenverzeichnis

Tabelle 1: Kostenaufstellung - Microsoft Teams .....	4
Tabelle 2: Kostenaufstellung - Matrix.....	5
Tabelle 3: Kostenaufstellung – Mattermost.....	5
Tabelle 4: Kostenaufstellung - Rocket.Chat.....	6
Tabelle 5: Nutzwertanalyse der Softwarelösungen .....	6
Tabelle 6: Mindestanforderung - Hosting-Server .....	7
Tabelle 7: Kostenaufstellung – IONOS .....	8
Tabelle 8: Kostenaufstellung – Alfahosting .....	8
Tabelle 9: Kostenaufstellung – Netcup .....	9
Tabelle 10: Kostenaufstellung – Hetzner .....	10
Tabelle 11: Nutzwertanalyse der Hosting-Anbieter .....	10
Tabelle 12: Projektablaufplan .....	VII
Tabelle 13: Personalkosten .....	VIII
Tabelle 14: Kostenvergleich.....	VIII

## 2 Abbildungsverzeichnis

Abbildung 1: Kostenaufstellung - Rocket.Chat.....	II
Abbildung 2: Systemübersicht und Containerstruktur .....	II
Abbildung 3: Update der Repositories .....	II
Abbildung 4: Upgrade der Server-Pakete .....	II
Abbildung 5: GPG-Key für Docker .....	III
Abbildung 6: Hinzufügen der Repository für Docker .....	III
Abbildung 7: Installation von Docker.....	III
Abbildung 8: Installation von Docker-Compose .....	III
Abbildung 9: Verzeichnisbaum .....	III
Abbildung 10: Docker-Compose: Matrix-Service .....	III
Abbildung 11: Docker-Compose: NGINX-Service.....	IV
Abbildung 12: Docker-Compose: TURN-Service .....	IV
Abbildung 13: Generieren der Matrix-Konfiguration.....	IV
Abbildung 14: Kopieren der NGINX-Konfiguration in das lokale Verzeichnis.....	IV
Abbildung 15: SSL-Zertifikat generieren .....	IV
Abbildung 16: Firewall-Konfigurationen .....	V
Abbildung 17: NGINX-Konfiguration .....	V
Abbildung 18: Coturn-Konfiguration .....	VI
Abbildung 19: Matrix-Konfiguration.....	VI
Abbildung 20: Federation Tester.....	VI
Abbildung 21: Willkommen bei Element.....	IX
Abbildung 22: Ändern des Heimservers.....	IX
Abbildung 23: Bearbeiten des Heimservers.....	IX
Abbildung 24: Eingabe der Benutzerdaten für die Anmeldung .....	X
Abbildung 25: Eingabe der Benutzerdaten für die Registrierung .....	X
Abbildung 26: Bearbeiten des Heimservers für die Registrierung.....	X
Abbildung 27: Startseite von Element nach der Erstanmeldung .....	XI
Abbildung 28: Seitenpanel des Element-Messengers .....	XI

## 3 Vorwort

Mein Name ist Wilko Reins und ich befinde mich seit dem 11.11.2019 in der Ausbildung zum Fachinformatiker für Systemintegration. Bis einschließlich Februar 2022 war ich Auszubildender in der Firma Brune-Mettcker Druck- und Verlagsgesellschaft mbH, wechselte den Betrieb nach internen Umstrukturierungen zur SOL.Service Online GmbH & Co. KG (im folgenden Text abgekürzt durch SOL). Im Zuge meiner betrieblichen Ausbildung habe ich den Auftrag bekommen, ein möglichst kosteneffizientes Kommunikationsmittel für zwischen- und innerbetriebliche Kommunikation im Betrieb zu integrieren.

Informationen, welche nicht für das Projekt relevant sind, z. B. IP-Adressen und Geheimschlüssel wurden mittels einer Bildbearbeitungssoftware unkenntlich gemacht.

## 4 Auftragsbeschreibung

### 4.1 Projektumfeld

Die Firma Brune-Mettcker Druck- und Verlags-GmbH ist ein Medienhaus in Wilhelmshaven und beschäftigt ca. 100 Mitarbeiter. Das Unternehmen produziert und gestaltet die Wilhelmshavener Zeitung, das Jeverische Wochenblatt und bis Ende 2021 den Anzeiger für Harlingerland sowie weitere Tageszeitungen und Druckerzeugnisse wie Briefpapier, Akzidenzen und weitere. Es ist an den Standorten Wilhelmshaven, Jever und Wittmund ansässig. Zusätzlich gab es bis Ende 2021 noch einen weiteren Produktionsstandort in Norden von Wilhelmshaven, der durch interne Umstrukturierungen geschlossen wurde.

Da ich den Ausbildungsbetrieb zum 01.03.2022 wechselte, setze ich das Projekt im neuen Betrieb um. Hierbei handelt es sich um die SOL.Service Online GmbH & Co. KG mit dem Hauptsitz in Hameln und einem neuen Standort in Wilhelmshaven. Die Firma hat 12 Angestellte und unterstützt Kunden verschiedenster Branchen, wie Arztpraxen, den öffentlichen Nahverkehr und Modehäuser. Dabei bietet sie u. a. Dienstleistungen, wie die App-Entwicklung, Webdesign oder dem Hosting mit über 1000 Produkten.

### 4.2 Projektauftrag

Für die SOL.Service Online GmbH & Co. KG soll ein Kommunikationsmittel für zwischen- und innerbetriebliche Kommunikation integriert werden. Es soll verschiedene Software auf Funktionsumfang und Wirtschaftlichkeit verglichen werden. Um den Datenschutz gewährleisten zu können, soll der Server selbst gehostet oder ein vertrauenswürdiger Hosting-Anbieter ausgewählt werden.

Am Ende wird das Projekt von dem Ausbilder und der Geschäftsleitung abgenommen.

## 5 Definitionsphase

### 5.1 Analyse des Ist-Zustandes

Die Mitarbeiter der SOL.Service Online GmbH & Co. KG arbeiten im oder aus dem Home-Office, dem neuen Standort in Wilhelmshaven oder aus Hameln und müssen auf Grund ihrer Projekte stetig miteinander kommunizieren. Für die Kommunikation wird die Plattform „Microsoft Teams“ verwendet. Viele Informationen, die zwischen den Kollegen und zwischen den verschiedenen Standorten ausgetauscht werden müssen, sind streng vertraulich.

Wählt man Microsoft Teams als Träger streng vertraulicher Daten, hat man nicht die unmittelbare Vollmacht darüber, unter welchen Sicherheitsbedingungen und wo diese Daten gespeichert werden.

### 5.2 Sollkonzept

Ziel des Projekts ist die Implementierung einer Softwarelösung, die einen Ersatz für die bisherige Lösung (Microsoft Teams) bietet. Es wird eine möglichst kosteneffiziente und hochverfügbare Lösung gesucht, die es den Mitarbeitern ermöglicht, sich untereinander und in Gruppen auszutauschen. Dazu gehören Funktionen wie Sprach- und Videotelefonie sowie die Nachrichtenübermittlung.

Der Server soll bei einem vertrauenswürdigen Hosting-Anbieter eingerichtet werden. Es wird ein Server mit geeignetem Betriebssystem für Containervirtualisierung<sup>1</sup> ausgewählt, um die Wartung und die Einrichtung der Dienste zu vereinfachen. Zudem wird eine gute Internetanbindung benötigt, da viele Benutzer zur gleichen Zeit über die (Web-)Anwendung auf den Dienst zugreifen müssen. Es soll ein Konzept für eine datenschutzkonforme Archivierung und Backuplösung entwickelt werden.

#### **Folgende Anforderungen sollen erfüllt sein:**

- Die Software muss für die Mitarbeiter leicht anwendbar sein
- Archivierung sowie Backup nach DSGVO
- Funktionen, die vom Home-Office sowie im Betrieb zur Verfügung stehen müssen:
  - Videotelefonie
  - Sprachtelefonie
  - Nachrichtenübermittlung
  - Bereitstellen von Dateien
  - Organisation von Abteilungen/Teams in eigenen Bereichen
  - Erstellen/Löschen von Konversationen
- Konversationen und Nachrichten müssen verschlüsselt übertragen werden
- Anstatt einer Neuentwicklung soll eine Standardlösung verwendet werden
- Das System muss mit Hilfe einer Containervirtualisierung auf einem Server installiert werden können
- Der Server muss über eine ausreichende Netzanbindung verfügen
- Es muss ein System sein, welches auf dem PC und dem Smartphone bedient werden kann

#### **Folgende Einschränkungen müssen berücksichtigt werden:**

- Es sollte eine Möglichkeit bestehen, auch offline auf Nachrichten zugreifen zu können, da man z. B. für außendienstliche Termine nicht immer online sein kann
- Fallen Wartungsarbeiten an der Software, der Containervirtualisierung oder dem Server an, wird die Kommunikation für den Zeitraum unterbrochen

## 6 Planungsphase

### 6.1 Zu beachtende Datenschutzvorschriften

Das Thema Datenschutz hat vor allem in den letzten zehn Jahren zunehmend an Bedeutung gewonnen. Da nach der Umsetzung meines Projekts unter anderem personenbezogene Daten von Mitarbeitern gesammelt werden, gilt es, die Datenschutzgrundverordnung (DSGVO) einzuhalten.

## Integration eines Messenger-Servers

Laut Artikel 7 der DSGVO muss der Mitarbeiter/die Mitarbeiterin über den Sachverhalt der Verarbeitung personenbezogener Daten aufgeklärt werden.

Die Einwilligung zum Sammeln der personenbezogenen Daten muss von den betroffenen Personen erfolgen. Da es ein Recht auf Widerruf gibt, muss dieses so einfach wie die Erteilung der Einwilligung gestaltet werden.

Da die SOL einen Server außerhalb des eigenen Hauses für die Softwarelösung verwenden, muss ein Vertrag zur Auftragsverarbeitung zwischen Hosting-Anbieter und Kunden abgeschlossen werden, der es garantiert, dass auch der Hosting-Anbieter gemäß der DSGVO (Artikel 28, Absatz 1) für die rechten Maßnahmen und den Schutz der personenbezogenen Daten sorgt.

Die DSGVO sieht auch vor, dass personenbezogene Daten der Datenminimierung (Artikel 5, Absatz 1) unterliegen, daher ist ein Backup- und Archivierungskonzept sehr wichtig. Wenn kein Speichergrund für diese personenbezogenen Daten mehr vorliegt, sind wir verpflichtet, diese zu löschen.

Im Falle einer Datenübertragung für das Archivieren und Backup muss die Vertraulichkeit und Integrität der Daten gewährleistet werden (Artikel 5, Absatz 1). Wählt man für die Übertragung ein externes Ziel-Verzeichnis, dann gelten die DSGVO-Bestimmungen für dieses Verzeichnis ebenfalls.

## 6.2 Vorstellung potentieller Softwarelösungen

Im folgenden Abschnitt werden verschiedene Softwarelösungen vorgestellt und miteinander verglichen. Es werden Lösungen vorgestellt, die vom Betrieb selbst zur Verfügung gestellt werden können und dann verglichen mit der derzeitigen SaaS<sup>3</sup>-Anwendung „Microsoft Teams“.

Alle Informationen bezüglich der Softwarelösungen wurden den Hersteller-Webseiten entnommen und die Kosten in diesem Projekt werden auf einen Zeitraum von 5 Jahren ermittelt.

### 6.2.1 Microsoft Teams

Microsoft Teams ist ein Kommunikations- und Kollaborationsdienst für die Zusammenarbeit in und zwischen Unternehmen und existiert bereits seit dem 14. März 2017 auf dem Markt. Funktionen, die von dem Vorgänger „Skype for Business“ angeboten wurden, sind auch in Microsoft Teams enthalten. Darunter fällt die Video- und Sprachtelefonie sowie die Organisation in Teams und der Austausch von Dateien. Es handelt sich bei Microsoft Teams um einen Dienst, der von Microsoft bereitgestellt und gehostet wird.

### 6.2.2 Matrix

Matrix ist ein dezentralisierter Kommunikationsdienst, welcher seit 2014 vom Entwicklungsteam der „Matrix.org Foundation C.I.C.“ entwickelt und aktualisiert wird. Die Stiftung stellt ihr Produkt der Öffentlichkeit kostenlos zur Verfügung. Benutzer können sich bei sogenannten „Homeservern“ registrieren und in Teams oder direkten Chaträumen miteinander austauschen. Matrix bietet die Möglichkeit, zwischen Benutzern und in Räumen Telefonie- und Videokonferenzen abzuhalten sowie Daten zu übertragen. Es können weitere Dienste, wie z. B. Bots, Kalender oder Monitoring-Dienste mittels Integrationen eingebunden werden. Mit Hilfe der Homeserver lässt sich der Dienst in vollem Funktionsumfang selbst hosten, alternativ stellt Matrix ebenfalls einen Server zur Verfügung. Es gibt für jede Plattform einen Client (Element<sup>2</sup>), mit dem man sich bei den Homeservern oder auf dem Matrix-Server anmelden kann.



### 6.2.3 Mattermost

Bei Mattermost handelt es sich um eine Open-Source Plattform, die seit 2015 entwickelt wird und Tools, wie die verschlüsselte Kommunikation für die organisierte Zusammenarbeit in Gruppen bietet. Es enthält Funktionen für die Softwareentwicklung, z. B. zur Entwicklung von Use-Case-Diagrammen, der Planung von Workflows und einem Inzidenz-Management-System, in dem Aufgaben zu Tickets umgewandelt und diese systematisch abgearbeitet werden können. Auf der Plattform können sich Kolleginnen und Kollegen untereinander schriftlich austauschen. Zusätzlich lassen sich Integrationen für die Sprach- und Videotelefonie hinzufügen. Man kann einen Mattermost-Server selbst hosten oder wählt das Cloud-Abonnement.

### 6.2.4 Rocket.Chat

Rocket.Chat ist eine Open-Source Plattform für die Vereinfachung von Echtzeit-Kommunikation und Zusammenarbeit von Kollegen, die 2015 als Nebenprojekt von vier Freunden entwickelt wurde. Unternehmen können sich Nachrichten zukommen lassen, Video- oder Audio-Anrufe halten und Dateien freigeben. Rocket.Chat bietet die Möglichkeit, Slack- und Microsoft Teams-Kanäle einzubinden und Kunden einzubeziehen. Als Unternehmer hat man die Auswahl zwischen einem selbstverwalteten Server oder Rocket.Chat als „Software as a Service“ in der Cloud zu hosten.

## 6.3 Ermittlung der Wirtschaftlichkeit – Kostenaufstellungen

In diesem Abschnitt werden die Kosten der Softwarelösungen für den Zeitraum von 5 Jahren (60 Monaten) aufgestellt. Es wird ein Durchschnittswert der Kosten aus dem Vergleich der Hosting-Server (wenn benötigt) aus *Abschnitt 6.5* hinzugezogen.

### Microsoft Teams

Microsoft bietet für verschiedene Zielgruppen unterschiedliche Varianten des Produkts „Microsoft Teams“ an. Es gibt Microsoft Teams als kostenlose Variante ohne Abonnement. Man kann Besprechungen organisieren und mit bis zu 100 Teilnehmern an Onlinebesprechungen teilnehmen. Möchte man Funktionen, wie die Erstellung von Gruppenräumen, das Aufzeichnen von Besprechungen, die Nutzung von Microsoft Word, Microsoft Excel oder zusätzliche Sicherheitsaspekte (Mehrstufige Authentifizierung) verwenden, dann muss man „Microsoft 365 Business Basic“ als Abonnement auswählen. Darin sind ebenfalls Administrationstools zur Verwaltung von Nutzern und Anwendungen enthalten sowie ein Kontingent für Unternehmen von einem Terrabyte und zehn weiteren Gigabyte pro Lizenz enthalten. Lizenzen müssen pro Benutzer erworben werden und kosten in diesem Abonnement 5,10 € pro Benutzer im Monat.

Produkt	Anzahl der Benutzer	Menge (Monate)	Einzelkosten monatlich (netto)	Gesamtkosten für 5 Jahre (netto)
Microsoft 365 Business Basic	12	60	5,10 € pro Benutzer	3672,00 €
Gesamt:				3672,00 €

Tabelle 1: Kostenaufstellung - Microsoft Teams

### Matrix

Da es sich bei Matrix.org um eine Stiftung handelt, finanziert sie sich nicht durch die Bereitstellung der Homeserver zur eigenen Einrichtung, sondern durch den Verkauf von Kleidungsstücken und Stickern. Bei der Installation und Verwaltung des eigenen Homeservers fallen

## Integration eines Messenger-Servers

hierbei lediglich Kosten für den Server, die Domain, dem Zertifikat und für das Personal an. Diese werden in Abschnitt 6.5 und unter Punkt 6.8 genauer erläutert.

Produkt	Anzahl der Benutzer	Menge (Monate)	Einzelkosten monatlich (netto)	Gesamtkosten für 5 Jahre (netto)
Matrix	12	60	0,00 €	0,00 €
Hosting-Server	/	60	10,62 €	637,36 €
<b>Gesamt:</b>				<b>637,36 €</b>

Tabelle 2: Kostenaufstellung - Matrix

## Mattermost

Für Mattermost gibt es mehrere Möglichkeiten, die zur Verfügung stehen. Für die Starter-Version von Mattermost fallen keine Kosten an. In diesem Paket sind jedoch nur die grundlegenden Funktionen gegeben, beispielsweise Video- und Sprachtelefonie. Dem Paket fehlt ein sehr wichtiges Sicherheitsfeature. Will man die Multi-Faktor-Authentifizierung als Sicherheitsfeature benutzen, ist man gezwungen, die Professional-Version dieser Software zu wählen. Sie enthält ebenfalls Funktionen, wie der Erstellung von Gastkonten oder der Vergabe von Gruppenrechten. Für das Abonnement von Mattermost-Professional fallen monatliche Kosten von 10 \$ pro Benutzer an. Das entspricht einem umgerechneten Wert von 8,96 € (Stand 30.03.2022).

Produkt	Anzahl der Benutzer	Menge (Monate)	Einzelkosten monatlich (netto)	Gesamtkosten für 5 Jahre (netto)
Mattermost Professional	12	60	8,96 € pro Benutzer	6451,20€
Hosting-Server	/	60	10,62 €	637,36 €
<b>Gesamt:</b>				<b>7088,56 €</b>

Tabelle 3: Kostenaufstellung – Mattermost

## Rocket.Chat

Vergleichbar zur Kostenaufstellung von Abschnitt *Tabelle 3*, bietet Rocket.Chat verschiedene Softwareversionen. Es gibt eine kostenlose Community-Version, eine Pro-Version, bei der jeweils 3 € pro Benutzer im Monat an Kosten anfallen und eine Enterprise-Variante, dessen Kosten im Monat pro Benutzer bei 5 \$ (umgerechnet 4,68 €) liegen. Die Kosten für das Enterprise-Abonnement können aus *Abbildung 1* entnommen werden. Darin werden die Kosten bei einer Benutzeranzahl von 12 gelistet. Um den gestellten Anforderungen an das Projekt gerecht zu werden, müsste ich „Rocket.Chat Enterprise“ wählen. Im Zwei-Jahres-Plan kostet jeder Benutzer monatlich 4,50 \$ (umgerechnet 4,03 €) und im Ein-Jahres-Plan liegen die Kosten bei 5 \$ pro Benutzer, was umgerechnet 4,48 € entspricht (Stand 30.03.2022).

Produkt	Anzahl der Benutzer	Menge (Monate)	Einzelkosten monatlich (netto)	Gesamtkosten (netto)
Rocket.Chat Enterprise (2y)	12	24	4,03 € pro Benutzer	1160,64 €
Rocket.Chat Enterprise (2y)	12	24	4,03 € pro Benutzer	1160,64 €
Rocket.Chat Enterprise (1y)	12	12	4,48 € pro Benutzer	645,12 €
Hosting-Server	/	60	10,62 €	637,36 €
<b>Gesamt (für 5 Jahre):</b>				<b>3603,76 €</b>

Tabelle 4: Kostenaufstellung - Rocket.Chat

## 6.3.1 Kosten/Nutzenanalyse der Softwarelösungen

In diesem Abschnitt werden die vier Softwarelösungen aus Abschnitt 6.3 anhand eines Scoring-Modells verglichen. Die Kriterien zur Bewertung der Softwarelösungen haben ein Bewertungsspektrum von 1 bis 5 Punkte, wobei 1 Punkt den geringsten Bewertungswert und 5 Punkte den höchsten Bewertungswert darstellen.

Kriterien	Gewicht	MS Teams		Matrix		Mattermost		Rocket.Chat	
		Wertung	Gesamt	Wertung	Gesamt	Wertung	Gesamt	Wertung	Gesamt
Kosten der Softwarelösung für fünf Jahre	30 %	3	0,9	5	1,5	1	0,3	3	0,9
Benutzerfreundlichkeit	20 %	4	0,8	4	0,8	4	0,8	4	0,8
Technische Anforderungen	20 %	5	1	3	0,6	3	0,6	3	0,6
Erweiterungsfähigkeit	15 %	2	0,3	3	0,45	5	0,75	5	0,75
Verfügbarkeit	15 %	5	0,75	5	0,75	5	0,75	5	0,75
<b>Gesamt</b>	<b>100 %</b>	<b>20</b>	<b>3,75</b>	<b>25</b>	<b>4,1</b>	<b>20</b>	<b>3,2</b>	<b>24</b>	<b>3,8</b>

Tabelle 5: Nutzwertanalyse der Softwarelösungen

## 6.4 Auswahl des Kommunikationsmittels

Im Vergleich von Abschnitt 6.3 fiel auf, dass die Kosten für Mattermost doppelt so hoch sind wie für Rocket.Chat. Die Kosten für Rocket.Chat sind geringer als für Microsoft Teams, jedoch werden im Scoring-Modell von Absatz 6.3.1 zusätzlich zu den Kosten noch weitere Kriterien einbezogen. Die hauptsächliche Gewichtung liegt hier weiterhin bei den Kosten für die reine Softwarelösung, jedoch ist bei der Auswahl der Softwarelösung ebenfalls zu beachten, wie benutzerfreundlich die Software ist, welche technischen Mittel benötigt werden, ob und wie sehr die Software erweitert werden kann und wo die Software dem Endbenutzer zur Verfügung steht. Abgesehen von Microsoft Teams, ähneln sich die anderen Softwarelösungen in vielen Punkten. Lediglich die Erweiterbarkeit und die Kosten machen dort den Unterschied.

## Integration eines Messenger-Servers

Da Matrix mit 4,1 Punkten am besten bewertet wurde, werde ich diese Softwarelösung in im Betrieb einsetzen.

### 6.5 Hosting-Anbieter

Der Vergleich der Hosting-Anbieter in diesem Abschnitt ist für alle Softwarelösungen von Relevanz abgesehen von Microsoft Teams.

#### 6.5.1 Vorstellung potentieller Hosting-Anbieter

In diesem Abschnitt werden Angebote verschiedener Hosting-Anbieter auf einen Zeitraum von 5 Jahren verglichen, um den Wirtschaftlichsten für den Betrieb zu ermitteln. Zusätzlich zum Server wird eine Domain und ein SSL-Zertifikat benötigt, damit der Server sicher adressiert werden kann und die Kommunikation zum Server verschlüsselt wird, daher werden diese Positionen in der Kostenermittlung integriert.

Auf dem Server muss ein Betriebssystem installiert werden, welches die Containervirtualisierung mittels „Docker“ unterstützt.

Da mehrere Docker-Dienste auf dem Server laufen werden und es sich bei Docker um Virtualisierung handelt, muss der Server mit ausreichendem Arbeitsspeicher ausgestattet werden. Die Kommunikation vom Server zu den Clients sowie vom Client zum Server verläuft mittels asymmetrischer Verschlüsselung über SSL, weshalb zwei oder mehr Prozessorkerne zur Verfügung stehen sollten.

Damit die Dienste langfristig laufen können, sollte der Server folgende Mindestanforderungen erfüllen:

#### Mindestanforderung

Serverkomponente	Konfiguration
Arbeitsspeicher	6 Gigabyte
Prozessor	2 Kerne
Speicher	80 Gigabyte
Netzwerkanbindung	100 Mbit/s
Standort	Deutschland
Betriebssystem	Linux Ubuntu 20.04 LTS

Tabelle 6: Mindestanforderung - Hosting-Server

### IONOS SE

IONOS ist ein deutscher Hosting-Anbieter, 2018 gegründet von 1&1 durch den Zusammenschluss von 1&1 Internet und dem Berliner IaaS-Anbieter ProfitBricks. Mit Produkten, beispielsweise Domains, Webservern und Root-Servern bis zu ganzen Cloud-Infrastrukturen. IONOS wirbt für ihre Server mit einem Standort in Baden-Baden (Deutschland).

Auf der Webseite wird ein Serverpaket mit der Bezeichnung „Virtual Server Cloud XL“ angeboten. Es enthält über die Mindestanforderung hinaus einen Prozessor mit 4 virtuellen Kernen, acht Gigabytes an Arbeitsspeicher und 160 Gigabyte SSD-Speicher. Die Installation von Linux Ubuntu ist im Angebot inklusive. Der Netzwerk-Traffic ist für die Serverpakete unbegrenzt und eine Bandbreite von bis zu 400 Mbit/s wird dem Server zur Verfügung gestellt. Dazu kommen Kosten für eine .de-Domain, jedoch keine Kosten für das SSL-Zertifikat, da es im Domain-Paket enthalten ist.

## Integration eines Messenger-Servers

In der folgenden *Tabelle 7* sind alle anfallenden Positionen gelistet.

Produkt	Menge	Einzelkosten (netto)	Gesamtkosten (netto)
Virtual Server Cloud XL	6 (Monate)	0,84 €	5,04 €
Virtual Server Cloud XL	54 (Monate)	13,45 €	726,30 €
Domain (.de)	12 (Monate)	0,07 €	0,84 €
Domain (.de)	48 (Monate)	1,09 €	52,44 €
SSL-Zertifikat	60 (Monate)	-	<i>Im Domain-Paket enthalten</i>
<b>Gesamt:</b>			<b>784,62 €</b>

*Tabelle 7: Kostenaufstellung – IONOS*

## Alfahosting GmbH

Die Alfahosting GmbH ist ein deutsches Unternehmen, welches 1999 gegründet wurde. Es erhielt in den letzten zwei Jahrzehnten mehrere Auszeichnungen für Nachhaltigkeit und wurde mehrfacher Testsieger im Bereich Hosting. Sie stellt ihren Kunden Webserver, Cloud Server, E-Shops sowie auch Domains und SSL-Zertifikate zur Verfügung. Der Standort für das Rechenzentrum der Alfahosting GmbH befindet sich in Leipzig.

Das Angebot „Cloud-Server M v3“ enthält eine Prozessorleistung von vier Kernen und 300 Gigabyte Speicher sowie sechs Gigabyte Arbeitsspeicher. Der Daten-Traffic ist hierbei unbegrenzt und es wird eine Bandbreite von 120 Mbit/s garantiert. Zusätzliche Dienste, beispielsweise eine Lizenz für die Software „Plesk“ sind im Angebot des Servers enthalten und die Installation bzw. die Einrichtung von Linux Ubuntu 20.04 LTS ist kostenlos. Für die Domain und das SSL-Zertifikat fallen jeweils monatliche Kosten an.

In der folgenden *Tabelle 8* sind alle anfallenden Positionen gelistet.

Produkt	Menge	Einzelkosten (netto)	Gesamtkosten (netto)
Cloud Server M v3	60 (Monate)	8,40 €	504,00 €
Domain (.de)	60 (Monate)	0,84 €	50,40 €
Power SSL-Zertifikat	60 (Monate)	1,50 €	90,00 €
<b>Gesamt:</b>			<b>634,40 €</b>

*Tabelle 8: Kostenaufstellung – Alfahosting*

### Netcup GmbH

Die Netcup GmbH wurde 2008 als Einzelunternehmen gegründet. Das Unternehmen ist in Karlsruhe ansässig und besitzt je ein Rechenzentrum in Wien (Österreich) sowie in Nürnberg (Deutschland). Auf der Webseite wird angegeben, dass die Netcup GmbH zu 100 % Ökostrom, gewonnen aus Wasserkraft, verwendet. Angebote für Webhosting, Server, Domains und SSL-Zertifikate werden in verschiedene Kategorien für unterschiedliche Zielgruppen unterteilt.

In meinem Fall kommt das Serverpaket „VPS 1000 G9 12M“ in Frage, da es zwei Prozessorkerne, acht Gigabytes an Arbeitsspeicher und 160 Gigabyte Festplattenspeicher zur Verfügung stellt. Der Daten-Traffic ist limitiert auf 80 Terrabyte im Monat und der Server verfügt über eine 1000 Mbit/s Netzanbindung mit einer garantierten Bandbreite von 200 Mbit/s. Die Domain sowie das SSL-Zertifikat müssen dazu gebucht werden. In einer Liste werden die verfügbaren Betriebssysteme gelistet und darunter befindet sich auch Linux Ubuntu 20.04 LTS.

Nachfolgend werden die Kostenstellen der Positionen und die Gesamtkosten dargestellt.

Produkt	Menge	Einzelkosten (netto)	Gesamtkosten (netto)
VPS 1000 G9 12M	60 (Monate)	5,04 €	302,40 €
Domain (.de)	60 (Monate)	0,35 €	21,00 €
Thawte SSL 123	5 Stk	44,54 €	222,70 €
<b>Gesamt:</b>			<b>546,10 €</b>

*Tabelle 9: Kostenaufstellung – Netcup*

### Hetzner Online GmbH

Hetzner Online wurde 1997 gegründet und bietet privaten und geschäftlichen Kunden verschiedenste Hosting Angebote und eine zuverlässige Infrastruktur. Mit Rechenzentren in Nürnberg, Falkenstein, Helsinki und mittlerweile auch einem neuen Standort in Ashburn (Virginia, USA), ist der deutsche Hosting-Anbieter nicht mehr nur europaweit, sondern weltweit vertreten.

Bei Hetzner Online werden unter Anderem schon mehrere Server von der Firma SOL gehostet, daher hatte man mir nahegelegt, das Serverpaket „CX31“ anzusehen. Dieses Paket enthält zwei Prozessorkerne, acht Gigabyte Arbeitsspeicher sowie 80 Gigabyte SSD-Speicher. Wird mehr Speicher benötigt, lässt sich dieser leicht und flexibel erweitern. Der Daten-Traffic ist limitiert bei 20 Terrabyte, aber wenn mehr verbraucht werden, wird für jeden weiteren angebrochenen Terrabyte 1,19 € abgerechnet. Der Server besitzt eine 10 Gbit Netzwerkanbindung. Eine Domain kostet bei Hetzner Online 0,84 € (netto) monatlich. Das SSL-Zertifikat lässt sich auf Servern von Hetzner mit Hilfe von „Let’s Encrypt“ kostenlos erstellen und verwenden, sodass auch dafür keine Kosten anfallen.

Produkt	Menge	Einzelkosten (netto)	Gesamtkosten (netto)
<b>CX31</b>	60 (Monate)	8,90 €	534,00 €
<b>Domain (.de)</b>	60 (Monate)	0,84 €	50,40 €
<b>SSL-Zertifikat</b>	60 (Monate)	-	<i>Im Domain-Paket enthalten</i>
<b>Gesamt:</b>			<b>584,40 €</b>

Tabelle 10: Kostenaufstellung – Hetzner

## 6.5.2 Nutzwertanalyse der Hosting-Anbieter

In diesem Abschnitt stelle ich ein Scoring-Modell auf, anhand dessen sich die Hosting-Anbieter nicht nur wirtschaftlich, sondern auch in ihrem Funktionsumfang und den genannten Leistungen vergleichen lassen. Den Kriterien habe ich Noten von 1 bis 5 zugeteilt, wobei der niedrigste Bewertungswert die 1, und der Höchste die 5 ist.

Kriterien	Gewicht	IONOS		Alfahosting		Netcup		Hetzner	
		Wertung	Gesamt	Wertung	Gesamt	Wertung	Gesamt	Wertung	Gesamt
Kosten des Hosting-Anbieters für fünf Jahre	30 %	2	0,6	3	0,9	5	1,5	4	1,2
Verfügbarkeit	20 %	5	1	5	1	4	0,8	3	0,6
Supportumfang des Anbieters	10 %	2	0,2	5	0,5	3	0,3	5	0,5
Verfügbare Datentransfer	15 %	5	0,75	5	0,75	4	0,6	4	0,6
Kundenzufriedenheit (Trustpilot.de)	5 %	4	0,2	5	0,25	3	0,15	3	0,15
Flexibilität der Hostingprodukte	20 %	2	0,4	2	0,4	2	0,4	5	1
<b>Gesamt</b>	<b>100 %</b>	<b>20</b>	<b>3,15</b>	<b>25</b>	<b>3,8</b>	<b>20</b>	<b>3,85</b>	<b>24</b>	<b>4,05</b>

Tabelle 11: Nutzwertanalyse der Hosting-Anbieter

## 6.5.3 Auswahl des Hosting-Anbieters

Im Abschnitt 6.5.2 kann man dem Scoring-Modell entnehmen, dass Hetzner mit 4,05 Punkten am besten bewertet wurde. An zweiter Stelle ist Alfahosting mit 3,8 Punkten, danach kommt Netcup mit 3,55 Punkten und den letzten Platz belegt IONOS mit nur 3,15 Punkten.

In den Bereichen Flexibilität der Hosting-Produkte und dem Supportumfang des Anbieters erreichte Hetzner 5 Punkte, weil man das Paket nach Buchung noch anpassen und erweitern kann. Der Support für das Rechenzentrum in Falkenstein ist 24 Stunden/7 Tage die Woche erreichbar, sodass man immer Auskunft oder Unterstützung erwarten kann. Die Strukturen sind zudem für meine Kollegen bereits bekannt, daher entscheide ich mich für Hetzner Online als Hosting-Anbieter für mein Projekt.

### 6.6 Systemübersicht

In der *Abbildung 2* habe ich eine Übersicht über die benötigten Container und Komponenten sowie der Netzwerkstruktur innerhalb der Docker-Umgebung erstellt. Damit der Anwender auf den Matrix-Server zugreifen und ihn zu Kommunikationszwecken verwenden kann, muss die Anwendung „Element“ heruntergeladen oder als Webanwendung benutzt werden. Eine Dokumentation zur Anwendung befindet sich unter *Punkt 9.4.1*.

### 6.7 Projektablaufplan

Den Projektablaufplan habe ich in Form einer Tabelle (*Tabelle 12: Projektablaufplan*) erstellt, um die Soll- und Ist-Zeiten der Phasen zu vergleichen und Abweichungen zu notieren, damit ein zeitlicher Überblick über das Projekt dargestellt werden kann. Unter *Punkt 8.1.3* werden die abweichenden Zeiten und die Änderungen zum Projektantrag genauer erläutert.

### 6.8 Analyse der Personalkosten

In *Tabelle 13* habe ich die Personalkosten und die investierten Arbeitsstunden den verschiedenen Phasen zugeordnet und mit den Stundensätzen des jeweiligen Personals verrechnet.

## 7 Durchführungsphase

### 7.1 Vorbereiten des Servers

Der Server wurde bei Hetzner Online GmbH durch meinen Ausbilder Herrn Fähnders gemietet und mir wurde ein bereits aufgesetzter Server mit Ubuntu 20.04 LTS zur Verfügung gestellt. Zunächst habe ich einen Befehl zum Aktualisieren der Repositories<sup>6</sup> (*Abbildung 3*) ausgeführt, damit mir bei einer Aktualisierung der aktuelle Stand der Produkte zum Herunterladen angeboten wird. Daraufhin führe ich den Upgrade-Befehl (*Abbildung 4*) aus, um sicherzustellen, dass ich das Projekt auf einem sicheren und aktuellen System umsetzen konnte. Danach habe ich die Firewall-Regeln für die Ubuntu-Firewall erstellt, sodass ich mich per SSH (Port 22) und SSH-Schlüssel-basierter Authentifizierung verbinden konnte.

### 7.2 Installation benötigter Dienste

Um die benötigten Dienste auf dem Server zu installieren, wird eine Containervirtualisierung vorausgesetzt. In meinem Fall verwende ich Docker. Für die Installation von Docker muss ich mehrere Schritte durchführen. Der erste auszuführende Befehl im Linux-Terminal dient der Registrierung des Schlüssels für das verschlüsselte Herunterladen der Docker-Pakete (*Abbildung 5*). Jetzt kann das Repository von Docker in die Liste der Ubuntu-Repositories hinzugefügt werden. Der Befehl hierfür befindet sich in *Abbildung 6*. Damit die Pakete von dem neuen Repository heruntergeladen werden können, muss ich zunächst den Befehl zur Aktualisierung dieser (*Abbildung 3*) erneut ausführen. Jetzt kann ich Docker mit Hilfe des Befehls in *Abbildung 7* installieren. Daraufhin führe ich noch einen weiteren Befehl zur Installation von Docker-Compose aus (*Abbildung 8*), damit ich meine Docker-Anwendungen, wie es der Name des Pakets schon vermuten lässt, in Kompositionen zusammenstellen und diese voneinander abhängig machen kann. Dem Benutzer muss ich jetzt die Berechtigung anhand des Befehls „`sudo chmod +x /usr/local/bin/docker-compose`“ erteilen, damit ich Docker-Compose ausführen kann.

### 7.3 Einrichtung der Docker-Umgebung

In diesem Abschnitt werde ich genauer auf die Struktur und den Projektaufbau eingehen und erläutern, weshalb ich mich für diese Art der Virtualisierung entschieden habe.



Zuerst muss eine geeignete Verzeichnishierarchie ausgewählt werden. Danach werden die benötigten Docker-Dienste installiert und zuletzt werde ich die Dienste konfigurieren. Abschließend wird das Projekt auf seine Funktionalität getestet, um sicherzustellen, dass alle Konfigurationen korrekt sind.

### 7.3.1 Anlegen der Verzeichnisse

Um das Projekt übersichtlich zu gestalten, muss eine Ordnerstruktur erstellt werden, die möglichst schnell verständlich und für die Containervirtualisierung leicht abzubilden ist. Die Daten, die in den Containern erstellt werden, werden auf dem Host in Form von Volumes<sup>7</sup> abgelegt (bzw. „gemappt“<sup>8</sup>), damit diese persistent und auch noch nach dem Neustart der Container vorhanden sind. In meinem Fall erstelle ich auf der Root-Ebene einen neuen Ordner „APPL“ für Applikationen bzw. Anwendungen. Darunter erstelle ich den Ordner „docker“, damit in diesem Verzeichnis alle Docker-Anwendungen angelegt werden können. Für die Softwarelösung „Matrix“ lege ich ein Verzeichnis („matrix“) an und darin einen Ordner namens „DATA“, in dem die Daten, Konfigurationen und die Datenbank des Projekts gespeichert werden. Diese Ordner werden beim ersten Start der Docker-Compose Dienste automatisch erstellt, da sie in der Docker-Compose Datei in *Abschnitt 7.3.2* des Projekts als Unterverzeichnisse des DATA-Verzeichnisses zugewiesen und benötigt werden.

Zur Veranschaulichung wird der Verzeichnisbaum in *Abbildung 9* dargestellt.

### 7.3.2 Anlegen der Docker-Compose Datei

Im Matrix-Verzeichnis lege ich nach der Erstellung der Verzeichnisstruktur eine Datei mit der Bezeichnung „docker-compose.yml“ an. YAML ist eine sogenannte Modelling-Language und mit Hilfe dieser Datei kann ich strukturiert darstellen, welche Dienste ich in Docker starte, auf welche Abbilder ich mich dabei beziehen möchte, wohin die Volumes gemappt werden und welchem Dienst welche Ports zur Verfügung stehen sollen. Das Ziel dieser Datei ist also, die Dienste in einer Datei zusammenzustellen, damit diese gemeinsam gestartet, gestoppt und im Falle einer Migration als Ganzes auf ein neues System übertragen werden können.

Für mein Projekt werden drei Dienste benötigt. Neben der hauptsächlichen Anwendung „Matrix“ werden in der Compose-Datei zusätzlich ein Proxy-Server<sup>9</sup>, sowie ein TURN-Server<sup>10</sup> benötigt. In einer Docker-Compose Datei werden Dienste als Services angelegt.

### Dienst 1: Matrix

Wie in *Abbildung 10* zu sehen ist, lege ich zuerst den Service „matrix“ an. Das Image, auf welchem der Container aufgebaut wird, beziehe ich über „Docker-Hub“, einer Plattform für vorgefertigte Docker-Abbilder, sodass ich lediglich die Konfigurationen vornehmen und nicht noch das Image anfertigen muss.

Der Container bekommt den Namen „matrix-chatapp“. Mit Hilfe des Namens lassen sich Docker-Container besser auseinanderhalten, daher benenne ich die Container nach ihrer Funktion.

Die Restart-Funktion (restart: always) sorgt dafür, dass der Container automatisch startet, sollte der Server oder Docker neu starten, sodass die Kommunikation über Matrix möglichst früh wieder gegeben ist.

Das definierte Volume sorgt dafür, dass der Container die Daten für das interne Verzeichnis „/data“ aus dem Host-Verzeichnis „/DATA/matrix/data“ bezieht und diese auch dort wieder ablegt, sodass sie dauerhaft gespeichert werden. In dem Ordner werden die Konfigurationen automatisch nach dem ersten Start angelegt, sodass diese nur noch bearbeitet werden müssen.

## Integration eines Messenger-Servers

Die Umgebungsvariablen müssen gesetzt werden, damit Docker anhand dieser Informationen die Konfigurationen und das SSL-Zertifikat erstellen kann.

Zuletzt muss ich noch den Port 8008 freigeben. Hierbei wird der containerinterne Port 8008 zum Host auf Port 8008 weitergeleitet, sodass Matrix im Container darüber Kommunizieren kann.

Um die Konfigurationen von Matrix zu Laden ist ein Befehl (*Abbildung 13*) auszuführen, welcher aus den Umgebungsvariablen die Datei erstellt und im Verzeichnis ablegt.

## Dienst 2: NGINX

Eine übliche Software zur Einrichtung eines Reverse-Proxies ist „NGINX“. Der Reverse-Proxy nimmt Anfragen auf Domains entgegen und kann diese auf lokale Ports im Server umleiten, sodass der Matrix-Server nicht immer mit Port 8008 erreicht werden muss, sondern einfach mit der Server-Domain. NGINX lege ich, wie in *Abbildung 11* abgebildet, in der Docker-Compose-Datei an. Ähnlich wie bei *Dienst 1: Matrix*, beziehe ich mich auf ein bereits angefertigtes Docker-Image.

Bei der Vergabe des Container-Namens folge ich derselben Namenskonvention und nenne den Dienst „matrix-proxy“.

Auch hier soll der Proxy wieder starten, wenn er unterbrochen wird, da Matrix ohne den Proxy ebenfalls nicht erreichbar ist.

Der Proxy nimmt die Anfragen über HTTP (Port 80), HTTPs (Port 443) sowie für Port 8448 entgegen.

Dem Proxy weise ich zwei Verzeichnisse unter Volumes zu, weil er zum einen aus den Web-Konfigurationen und andererseits aus dem SSL-Zertifikat besteht. Das SSL-Zertifikat erstelle ich auf dem Host daher befindet es sich nicht direkt im DATA-Verzeichnis.

Die Konfigurationsdatei kopiere ich mit Hilfe des Befehls aus *Abbildung 14* aus dem laufenden Container auf das Verzeichnis des Hosts.

## Dienst 3: Coturn

Zuletzt braucht der Server einen TURN-Service. Seine Aufgabe ist das Routen von Paketen, vor Allem bei Voice-over-IP<sup>11</sup> als Relay, da diese Pakete auf Grund von NAT über nur eine einzige öffentliche IPv4-Adresse geroutet werden müssen. Zwei Geräte, die sich in unterschiedlichen privaten Netzwerken befinden, können sich daher nicht direkt über VoIP anrufen. Der TURN-Server steht als Relay-Server zwischen beiden Gesprächsteilnehmern und routet VoIP-Pakete anhand einer großen Spanne an Ports von Teilnehmer A nach Teilnehmer B.

In *Abbildung 12* ist zu sehen, dass ich für den Turn-Server „Coturn“ verwende.

Die Benennung folgt auch beim Turn-Server der Namenskonvention und den Restart gebe ich dem Container auch hier als Parameter mit.

Der „network\_mode: host“ bedeutet, dass dieser Service zwar in Docker läuft, jedoch im Netzwerk des Hosts aktiv ist. Das trage ich ein, damit der TURN-Server nur auf die IP-Adresse des Hosts „hört“ und ihm eine große Bandbreite an Ports für die Umsetzung von Gesprächen zur Verfügung steht.

Der TURN-Server benutzt den Port 5349 für das „Ansprechen“ des Dienstes. Da der Container allerdings mit dem Host-Netzwerk „arbeitet“, muss der Port nicht in der Compose-Datei zur Weiterleitung deklariert werden.

Wichtig für den Container ist ein Volume für die Konfiguration des Servers und auch hier müssen die SSL-Zertifikate verlinkt werden.

### 7.4 Erstellen der übrigen Firewall-Regeln

In *Abschnitt 7.3.2* setze ich die benötigten Ports für die Dienste, sodass ich in diesem Abschnitt die Firewall-Regeln für Dienste in die Ubuntu Firewall übertragen kann.

In *Abbildung 16* liste ich die angelegten Firewall-Regeln und den Befehl zum Anzeigen der Regeln, die für dieses Projekt notwendig sind. Den Matrix-Server sprechen die Anwender lediglich über den Port 443 (HTTPS) an und über NGINX werden die Anfragen intern an die Container weitergeleitet. Somit muss der Endbenutzer keinen Port mehr bei Eingabe des Servers für den Verbindungsaufbau beachten. Der Port 5349 muss von außerhalb erreichbar und ansprechbar sein, weil darüber die Anrufe gelenkt werden. Zusätzlich schalte ich den Port-Bereich 49152 bis 65535 frei, damit diese für Telefonate reserviert werden können. Zur Erstellung der Regeln verwende ich jeweils den Befehl `ufw allow <Port:Port>/tcp` und `ufw allow <Port:Port>/udp`.

### 7.5 Generieren des SSL-Zertifikats

Wie bereits unter *Abschnitt 7.3.2* erwähnt, muss ich für die sichere Kommunikation zwischen Server und Client ein SSL-Zertifikat generieren. Hierfür bietet sich die Verwendung von „Let’s Encrypt“ an. Es handelt sich dabei um einen Dienst, welcher SSL-Zertifikate kostenfrei zur Verfügung stellt und diese automatisch verlängert.

Um ein SSL-Zertifikat für meine Server-Domain zu generieren, gebe ich zuerst den Befehl aus *Abbildung 15* im Terminal ein. Daraufhin wird unter dem Root-Verzeichnis „/etc“ ein neuer Ordner namens „letsencrypt“ angelegt, in welchem die SSL-Zertifikate abgelegt werden. Um diese in den Containern später verwenden zu können, muss ich diesen Ordner als Volume für den Proxy und für den TURN-Server in der Compose-Datei verlinken.

### 7.6 Konfigurationen anpassen

In diesem Abschnitt erläutere ich, welche Konfigurationsdateien ich anpasse und welche Konfigurationen ich vornehme, damit der Server erreichbar und voll funktionstüchtig ist.

#### NGINX-Konfiguration

Zuerst muss ich den NGINX-Server konfigurieren. Die Konfigurationen befinden sich in der Datei `nginx.conf` in meinem Docker-Volume.

In den ersten 7 Zeilen legte ich fest, dass der Server Anfragen über HTTP zu HTTPS weiterleitet.

Von Zeile 8 bis 20 definiere ich, dass NGINX die HTTPS-Anfragen zum Docker-Container von Matrix weiterleitet.

Die Zeilen 21 bis 32 füge ich zusätzlich hinzu, damit ich Matrix-Integrationen, beispielsweise ein gemeinsamer Kalender oder Monitoring, über Port 8448 dem Server zuweisen kann.

Ich weise den Verbindungen u. a. in Zeile 14 und 15 das SSL-Zertifikat zu, welches ich zuvor über das Volume in den Container verlinke, sodass die Kommunikation über den Proxy-Server verschlüsselt wird.

Die vollständige Konfiguration für NGINX kann in *Abbildung 17* eingesehen werden.

### Coturn-Konfiguration

Die Konfigurationsdatei des TURN-Servers kopiere ich aus dem GitHub<sup>5</sup>-Repository und füge sie im Volume des Coturn-Containers als „turnserver.conf“ ein.

Zuerst muss ich die öffentliche IP-Adresse des Servers als „Listening-IP“ und als „External-IP“ eintragen, damit der Server über eine Adresse erreichbar ist. Der „Listening-Port“ ist 5349 und bleibt damit der Standard-Port für verschlüsselte Kommunikation.

Die Port-Range habe ich bereits bei der Vergabe der Firewall-Regeln festgelegt, daher muss ich hier ebenfalls die Port-Range von Port 49152 bis Port 65535 eintragen.

Da der Matrix-Dienst mit dem Coturn-Server kommunizieren soll, vergebe ich einen „Hex-Code“, welcher die Kommunikation zwischen den Diensten verschlüsselt. Hierfür verwende ich den Befehl „openssl rand -hex 32“.

Den Schlüssel trage ich bei „static-auth-secret“ ein und aktiviere die Nutzung des Schlüssels, über die Konfigurationsanweisung „use-auth-secret“ (die im Standard auskommentiert ist).

Eine weitere Konfiguration, die ich anpasse, ist der Realm, welchen ich zur Domain vom Server ändere, damit der Server Anfragen für diese Domain empfängt.

Alle vorgenommenen Änderungen können in *Abbildung 18* eingesehen werden.

### Matrix-Konfiguration

Zuletzt passe ich noch die Konfigurationsdatei von Matrix („homeserver.yaml“) an.

Beim Servernamen trage ich meine Domain ein.

Danach lege ich den „Listener“ auf Port 8008 mit HTTP-Protokoll fest und dass der Server für (Web-)Clients zugänglich ist, damit der Proxy den Matrix-Server über HTTP über den Port erreicht und Anfragen von Clients weiterleitet.

Damit der Matrix-Server Voice-over-IP unterstützt weise ich den zuvor eingerichteten TURN-Server nun dem Matrix-Server zu. Dafür trage ich den Link zum Turn-Server, den bereits erstellten Hex-Code und einen Parameter, der es Gästen erlaubt, dem Telefonat beizutreten in die Konfiguration ein.

Die Einstellung zur Registrierung neuer Benutzer setze ich, damit sich die Kollegen und Kunden selbstständig registrieren können.

Die restlichen Konfigurationen lasse ich unverändert. Alle Änderungen habe ich auf Grund der umfangreichen Datei an den Anfang gesetzt und in *Abbildung 19* veranschaulicht.

## 7.7 Testen der Konfigurationen

Nachdem ich die Dienste konfiguriert habe, starte ich die Docker-Container auf dem Server. Zuerst rufe ich die Webseite „Federation Tester“ auf und gebe meine Domain ein. Auf *Abbildung 20* kann man sehen, dass alle Überprüfungen zur Erreichbarkeit und zur Verschlüsselung des Matrix-Servers erfolgreich sind.

Um den Funktionsumfang von Matrix zu testen, registriere ich zuerst einen Admin-Benutzer. Dafür führe ich im Matrix-Container den Befehl „register\_new\_matrix\_user -c /data/homeserver.yaml http://localhost:8008“ aus. Mit diesem Benutzer melde ich mich erfolgreich an und kann innerhalb der Applikation agieren. Ich melde mich testweise sowohl über die Web-Applikation von Element am Laptop, sowie an der mobilen Applikation Element mit meinen Benutzerdaten an. Danach richte ich einen zweiten Testbenutzer ein, mit dem ich erfolgreich Räume und Konversationen starte. Das Hinzufügen von Integrationen, wie beispielsweise ein Youtube-Video oder eine Jitsi-Videokonferenz, zu einem Raum ist erfolgreich. Zunächst überprüfe ich die Funktionalität der (Video-)Telefonie, wobei es zu Problemen kommt, die ich in den folgenden Abschnitten genauer erläutere.

### 7.7.1 Fehlerbehebung: TURN-Server startet nicht korrekt

Nachdem erstmaligen Starten der drei Container konnte ich mich in der Applikation anmelden und alle Funktionen, die die Applikation Element zur Verfügung stellt, testen und ausprobieren. Ich führte zuerst einen Test durch, ob die Videotelefonie zwischen zwei Geräten, die nicht im selben Netzwerk sind, funktioniert. Hierbei fiel mir auf, dass der Anruf nicht aufgebaut werden konnte und an beiden Geräten nur die Meldung „Verbindungsaufbau...“ auftrat.

Ich schaute mir die Logs des Coturn-Containers mit dem Befehl „`docker logs <Container-ID>`“ an und stellte fest, dass dem Prozess im Container keine ID für den TURN-Server vergeben werden konnte, da die PID-Datei aus benutzerrechtlichen Gründen nicht in „`/var/run`“ angelegt werden konnte. Daraufhin änderte ich das Verzeichnis zu „`/var/tmp`“ in der Konfigurationsdatei des TURN-Servers. Die Meldung tauchte nicht erneut auf.

### 7.7.2 Fehlerbehebung: TURN-Server startet nicht mit SSL

Nachdem ich den Fehler aus *Abschnitt 7.7.1* behoben habe, gab es einen weiteren Fehler. Zunächst kam die Meldung, dass das SSL-Zertifikat nicht gefunden werden konnte.

Daraufhin überprüfte ich in der „`turnserver.conf`“, ob der Pfad zum privaten Schlüssel korrekt eingetragen wurde, allerdings war dieser richtig. Zunächst überprüfte ich mit dem Befehl „`ls -l`“ im Verzeichnis des SSL-Zertifikats, welche Rechte dieser Datei für Benutzer und Gruppen zugewiesen wurden. Für andere Benutzer war diese Datei nicht lesbar, weshalb ich die Rechte für diese Datei mit dem Befehl „`sudo chmod 644 privkey1.pem`“ änderte. Nach dem Neustart des Containers, wurde der Schlüssel erkannt und der erneute Test, Videotelefonie aus zwei unterschiedlichen Netzen durchzuführen, war erfolgreich.

## 8 Abschlussphase

### 8.1 Soll-/Ist-Vergleich

Die gewählte Softwarelösung bietet alle Möglichkeiten, die in der Soll-Konzeption als Anforderungen an das Projekt gestellt wurden. Benutzer der Software können Dateien teilen, sich schriftlich miteinander austauschen und sowohl Videokonferenzen, als auch Sprachtelefonie abhalten. Eine sehr wichtige Anforderung war die sichere Datenübertragung von Teilnehmer A nach Teilnehmer B und diese wird durch die verschlüsselte Signatur mittels eines SSL-Zertifikats sichergestellt. Die Archivierung und ein mögliches Backup nach DSGVO sind in diesem Projekt umsetzbar, da unser Betrieb mit dem gewählten Hosting-Anbieter einen Auftragsverarbeitungs-Vertrag unterschrieben hat, der es gesetzlich verankert, dass auch auf den Servern nach DSGVO-Vorschrift mit den personenbezogenen Daten gehandelt wird. Mit Hilfe von Element kann man mit dem Smartphone und jedem Gerät, welches über einen Webbrowser verfügt, auf die Software zurückgreifen.

Die Dienste und Konfigurationen wurden auf Basis der Containervirtualisierung erstellt und umgesetzt, wodurch sich das Projekt als sehr portabel und leicht zu verwalten herausstellt. Lediglich die Erstellung des SSL-Zertifikats basiert direkt auf dem Host.

#### 8.1.1 Kostenvergleich

In der *Tabelle 14* habe ich die Soll-Zeiten mit den Ist-Zeiten verglichen sowie die angefallenen Personalkosten für diese berechnet. Da ich eine halbe Stunde bei der Durchführung des Projekts weniger benötigt habe, haben sich die Kosten um 12,50 € reduziert.

### 8.1.2 Zeitliche Abweichungen

Für den Vergleich der verschiedenen Softwarelösungen habe ich ungefähr eine Stunde länger benötigt, da ich versuchte, telefonischen Kontakt mit dem Vertrieb von Mattermost aufzunehmen und das mehr Zeit in Anspruch genommen hat. Mit dem Vertrieb von Mattermost habe ich ein etwas längeres Telefongespräch geführt, in dem mir die Optionen für meinen Betrieb vorgestellt wurden.

Die Installation und Konfiguration der Dienste dauerte sechs Stunden länger als ich ursprünglich eingeplant habe, jedoch brauchte ich auch sechs Stunden weniger Zeit, um die Anwender- und Betriebsdokumentation zu erstellen.

Für die Erstellung und das Einbinden des SSL-Zertifikats in der Durchführungsphase brauchte ich die gesamte Zeit, die im Projektantrag für die Erstellung der Domain und des SSL-Zertifikats eingeplant war, da das SSL-Zertifikat vom Host in den Container übertragen werden musste und ich vorher noch nicht mit „Let's Encrypt“ gearbeitet habe.

Da ich in meiner Ausbildung schon des Öfteren mit Docker gearbeitet habe, konnte ich bei der Einrichtung des Servers etwas Zeit „einsparen“ und das ganze Projekt ließ sich sehr gut umsetzen.

### 8.1.3 Änderungen gegenüber dem Projektantrag

In meinem Projektantrag plante ich eine Stunde für die Erstellung und Beantragung einer Domain beim Hosting-Anbieter sowie die Erstellung des SSL-Zertifikats ein, jedoch wurde mir die Domain bereits eingerichtet von meinem Ausbilder zugewiesen, sodass innerhalb des Projekts kein Zeit- oder Kostenaufwand dadurch von mir entstand.

In der Durchführung meines Projekts habe ich zuerst die Firewall-Regeln gesetzt und erst im Test erstellte ich den administrativen Benutzer. Abgesehen von der zeitlichen Umverteilung im Vergleich zum Antrag, konnte ich den Projektantrag erfüllen.

## 8.2 Anwenderdokumentation

Damit die Endnutzer darin geschult werden, wie sie ihre Benutzerkonten in Matrix anlegen können und wo sie welche Funktionen nutzen können, wird ihnen eine Dokumentation (*siehe 9.4 Anwenderdokumentation*) bereitgestellt. Ebenfalls ist dort eine Dokumentation für Administratoren zu finden.

## 8.3 Abnahme des Projekts

Abschließend habe ich das Projekt meinem Projektbetreuer und Ausbilder vorgestellt. Einen Termin zur Vorstellung des Projekts bei der Geschäftsleitung werde ich nach Anfertigung der Projektpräsentation vereinbaren.

## 8.4 Fazit

Da sich die Ausgangslage meines Projekts durch den Wechsel des Betriebs (geringfügig) änderte, habe ich mich mit dem Projekt ausführlich auseinandersetzen können. Bei der Brune-Mettcker Druck- und Verlags-GmbH war es das Ziel, ein möglichst günstiges Kommunikationsmittel zu implementieren, weil es zuvor keine angewandte Möglichkeit gab. Bei der SOL.Service Online GmbH & Co. KG war Microsoft Teams bereits in Verwendung, aber da ich die Möglichkeit hatte, Microsoft Teams und Matrix zu verwenden, konnte ich gute Vergleiche durchführen und bin zu dem Entschluss gekommen, dass Matrix im Funktionsumfang sehr vergleichbar mit Microsoft Teams ist und wir bei der Verwendung von Matrix deutlich gewissenhafter im Umgang mit personenbezogenen Daten sind.

## Integration eines Messenger-Servers

Das Thema „Sicheres Messaging“ ist in den vergangenen Jahren immer bedeutsamer geworden und die Anzahl an Applikationen auf dem Markt hat sich vervielfacht. Eine self-hosted<sup>12</sup> Variante der Kommunikationsmittel war mir in diesem Umfang bis zur Umsetzung des Projekts nicht geläufig, aber mit dem Blick hinter eine Softwarelösung für Messenger habe ich viel über das Geschehen und die Funktionsweise lernen können. Vor allem die Funktionsweise eines TURN-Servers zu verstehen, hat mich sehr gefordert und ich habe viel darüber lernen können.

Ich kann sagen, dass das Projekt ein sehr großer Erfolg für mich war und mir die Umsetzung in Form der Containervirtualisierung sehr viel Freude bereitet hat, weil ich mich im Laufe meiner gesamten Ausbildung mehrfach mit Docker auseinandersetzen durfte und ich mein dazu-gewonnenes Wissen mit diesem Projekt anwenden konnte.

## 9 Anhang

### 9.1 Glossar

<sup>1</sup>**Containervirtualisierung:** Eine Methode, um mehrere Instanzen eines Betriebssystems isoliert voneinander den Kernel des Hostsystems nutzen zu lassen.

<sup>2</sup>**Element:** Matrix-basierte Messenger-Applikation

<sup>3</sup>**SaaS:** „Software as a Service“, Cloud-basierte Softwarelösung

<sup>4</sup>**Docker:** Docker ist eine freie Software zur Isolierung von Anwendungen mit Hilfe von Container-virtualisierung

<sup>5</sup>**GitHub:** GitHub ist ein netzbasierter Dienst zur Versionsverwaltung für Projekte.

<sup>6</sup>**Repository:** Ein Repository ist ein verwaltetes Verzeichnis zur Speicherung und Beschreibung digitale Objekte für ein digitales Archiv.

<sup>7</sup>**Volume:** Mit Volumes ist es in Docker möglich, Daten zwischen Containern auszutauschen und persistent zu speichern.

<sup>8</sup>**Mapping:** Ein Verzeichnis auf dem Host in den Container zu verknüpfen, sodass die Daten in diesem Verzeichnis sowohl vom Host, als auch im Container bearbeitet werden können.

<sup>9</sup>**Proxy-Server:** Ein Vermittler, der Anfragen entgegennimmt und diese über seine eigene Adresse weiterleitet.


<sup>10</sup>**TURN-Server:** Bedeutet “Traversal Using Relays around NAT” und übernimmt die Aufgabe, zwei (oder mehr) Adressen aus verschiedenen Netzen miteinander kommunizieren zu lassen.

<sup>11</sup>**Voice-over-IP:** Die Telekommunikation erfolgt hierbei über das IP-Protokoll.

<sup>12</sup>**self-hosted:** Bereitstellung von Software oder Datenbanken auf eigener Server-Infrastruktur.



## 9.2 Abbildungen

Number of Users	12	SOL.Service Online  - Self hosted Environment						
Self-Managed Enterprise	€5.00							
Valid Until March the 20th	Investment	Discount	Total Price	Cost per user*	You'd Save	Yearly Price	Guest Users	Payment Frequency **
1 year plan	€720.00	0%	€720	€5.00	€0.00	€720	60	Whole Term Payment
2 year plan	€1,440.00	10%	€1,296	€4.50	€144.00	€648	60	Payment Flexibility

\*Only for comparison purposes  
 \*\*Payment Flexibility can be whole term or yearly - Discounts may vary depending on it  
 Prices expressed in Euros

Abbildung 1: Kostenaufstellung - Rocket.Chat

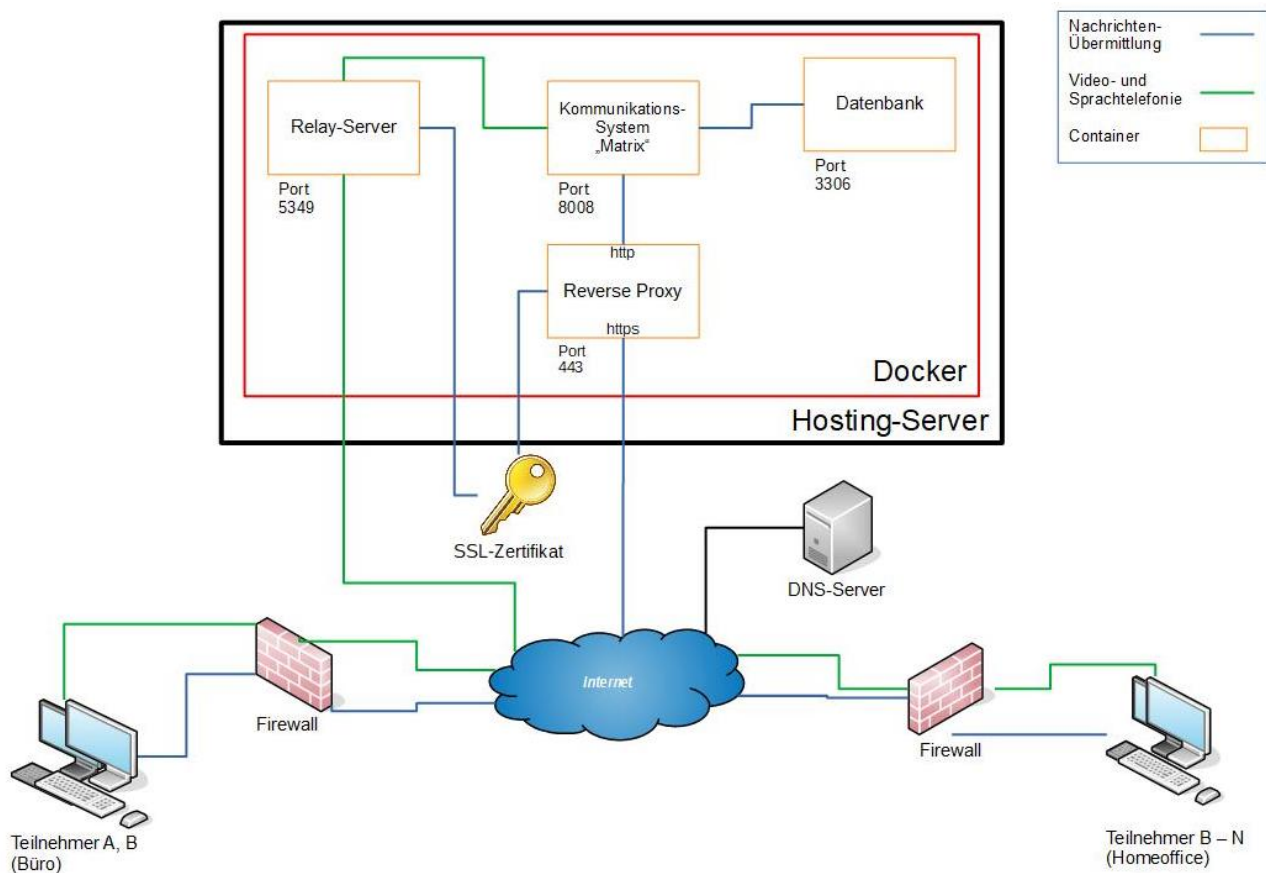


Abbildung 2: Systemübersicht und Containerstruktur

```
root@wilko:/APPL/docker# sudo apt-get update
```

Abbildung 3: Update der Repositories

```
root@wilko:/APPL/docker# sudo apt-get upgrade
```

Abbildung 4: Upgrade der Server-Pakete

## Integration eines Messenger-Servers

```
root@wilko:/APPL/docker# curl -fsSL https://download.docker.com/linux/ubuntu/gpg | \
> sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg
```

Abbildung 5: GPG-Key für Docker

```
root@wilko:/APPL/docker# echo \
> "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu \
> $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

Abbildung 6: Hinzufügen der Repository für Docker

```
root@wilko:/APPL/docker# sudo apt-get install docker-ce docker-ce-cli containerd.io
```

Abbildung 7: Installation von Docker

```
root@wilko:/APPL/docker# curl -L "https://github.com/docker/compose/releases/download/1.29.2/docker-compose-$(uname -s)-$(uname -m)" \
> -o /usr/local/bin/docker-compose
```

Abbildung 8: Installation von Docker-Compose

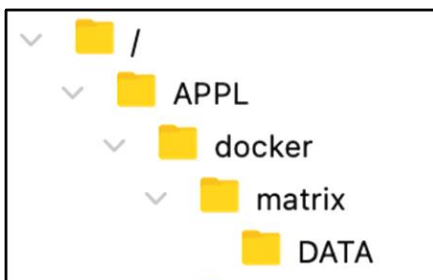


Abbildung 9: Verzeichnisbaum

```
# Matrix-Container-----
matrix:
  image: matrixdotorg/synapse:latest
  container_name: matrix-chatapp
  restart: always
  volumes:
    - ./DATA/matrix/data:/data
  environment:
    LETSENCRYPT_HOST: "wi[REDACTED].de"
    SYNAPSE_SERVER_NAME: "wi[REDACTED].de"
    SYNAPSE_REPORT_STATS: "no"
  ports:
    - "8008:8008"
```

Abbildung 10: Docker-Compose: Matrix-Service

## Integration eines Messenger-Servers

```
# Proxy-Container-----
nginx:
  image: nginx:latest
  container_name: matrix-proxy
  restart: always
  ports:
    - 80:80
    - 443:443
    - 8448:8448
    - 5349:5349
  volumes:
    - ./DATA/nginx/etc/nginx/conf/nginx.conf:/etc/nginx/conf.d/default.conf
    - /etc/letsencrypt/archive/wi[REDACTED].de:/etc/nginx/certs
```

Abbildung 11: Docker-Compose: NGINX-Service

```
# TURN-Relay-Server-Container-----
matrix-voip-relay:
  image: coturn/coturn:latest
  container_name: matrix-relay
  restart: always
  network_mode: host
  volumes:
    - ./DATA/coturn/etc/coturn/conf/turnserver.conf:/etc/coturn/turnserver.conf
    - /etc/letsencrypt/archive/wi[REDACTED].de:/certs
```

Abbildung 12: Docker-Compose: TURN-Service

```
root@wilko:/APPL/docker/matrix# docker-compose run --rm matrix generate
```

Abbildung 13: Generieren der Matrix-Konfiguration

```
root@wilko:/APPL/docker/matrix# docker cp 7007373ff008:/etc/nginx/nginx.conf DATA/nginx/etc/nginx/conf
```

Abbildung 14: Kopieren der NGINX-Konfiguration in das lokale Verzeichnis

```
root@wilko:/APPL/docker/matrix# certbot certonly -d wi[REDACTED].de
```

Abbildung 15: SSL-Zertifikat generieren

## Integration eines Messenger-Servers

```
22/tcp (OpenSSH (v6))      ALLOW IN    Anywhere (v6)
22/tcp (v6)                ALLOW IN    Anywhere (v6)
443/tcp (v6)               ALLOW IN    Anywhere (v6)
443/udp (v6)               ALLOW IN    Anywhere (v6)
5349/udp (v6)              ALLOW IN    Anywhere (v6)
5349/tcp (v6)              ALLOW IN    Anywhere (v6)
49152:65525/tcp (v6)       ALLOW IN    Anywhere (v6)
49152:65525/udp (v6)       ALLOW IN    Anywhere (v6)

root@wilko:/APPL/docker# ufw status verbose
```

Abbildung 16: Firewall-Konfigurationen

```
matrix > DATA > nginx > etc > nginx > conf > ⚙ nginx.conf
1  server {
2      listen 80;
3      server_name wi[REDACTED].de;
4      return 301 https://wi[REDACTED].de$request_uri;
5      #access_log /dev/null;
6      #error_log /dev/null;
7  }
8  server {
9      listen 443 ssl;
10     listen [::]:443 ssl;
11     server_name wi[REDACTED].de;
12     #access_log /dev/null;
13     #error_log /dev/null;
14     ssl_certificate /etc/nginx/certs/fullchain1.pem;
15     ssl_certificate_key /etc/nginx/certs/privkey1.pem;
16     location /_matrix {
17         proxy_pass http://matrix-chatapp:8008;
18         proxy_set_header X-Forwarded-For $remote_addr;
19     }
20 }
21 server {
22     listen 8448 ssl;
23     server_name wi[REDACTED].de;
24     #access_log /dev/null;
25     #error_log /dev/null;
26     ssl_certificate /etc/nginx/certs/fullchain1.pem;
27     ssl_certificate_key /etc/nginx/certs/privkey1.pem;
28     location / {
29         proxy_pass http://matrix-chatapp:8008;
30         proxy_set_header X-Forwarded-For $remote_addr;
31     }
32 }
33
```

Abbildung 17: NGINX-Konfiguration

## Integration eines Messenger-Servers

```
DATA > coturn > etc > coturn > conf > turnserver.conf
# Coturn TURN SERVER configuration file
#
tls-listening-port=5349
listening-ip=142.
external-ip=142.
min-port=49152
max-port=65535
verbose
fingerprint
use-auth-secret
static-auth-secret=11539
realm=wi.de
cert=/certs/fullchain1.pem
pkey=/certs/privkey1.pem
syslog
no-stun
no-cli
```

Abbildung 18: Coturn-Konfiguration

```
matrix > DATA > matrix > data > ! homeserver.yaml
1 # Configuration file for Synapse.
2 #
3 - port: 8008
4   tls: false
5   type: http
6   resources:
7     - names: [client, federation]
8
9   turn_uris: [ "turn:142.:5349?transport=udp", "turn:142.:5349?transport=tcp" ]
10  turn_shared_secret: "11539
11  turn_user_lifetime: 1h
12  turn_allow_guests: true
13
14  enable_registration: true
```

Abbildung 19: Matrix-Konfiguration

Connection Reports	
142.:8448	
Checks	Success
MatchingServerName	Success
FutureValidUntilTS	Success
HasEd25519Key	Success
AllEd25519ChecksOK	Success
Ed25519Checks	Success
ed25519:a_XHSQ	Success
ValidEd25519	
MatchingSignature	
ValidCertificates	Success

Abbildung 20: Federation Tester

### 9.3 Tabellen

Phase	Tätigkeit	Soll-Zeit	Ist-Zeit	Differenz	Datum (2022)
<b>Definitionsphase</b>	Ist-Zustand analysieren	1	0,5	-0,5	29.03
	Entwickeln eines Soll-Konzeptes	2	1,5	-0,5	29.03
<b>Planungsphase</b>	Ermittlung der Datenschutzvorschriften	1	1	0	29.03
	Vergleich mehrerer Softwarelösungen	3	4	1	29.03
	Erstellen einer Kosten-/Nutzenanalyse der Softwarelösungen	1	1	0	30.03
	Erstellung eines Kriterienkatalogs inklusive Gewichtung	2	1	-1	30.03
	Vergleich verschiedener Webhosting-Anbieter	1	2	1	30.03
	Erstellen einer Kosten-/Nutzenanalyse der Hosting-Anbieter	1	1	0	30.03
	Ermittlung des benötigten Betriebssystems für Containervirtualisierung	1	0,5	-0,5	31.03
<b>Durchführungsphase</b>	Server beim Webhosting-Anbieter mieten	0,5	0,5	0	31.03
	Erstellung eines SSL-Zertifikats	1	1	0	31.03
	Installation der Containervirtualisierung auf dem Server	1	0,5	-0,5	31.03
	Installation des Konferenzsystems und der zugehörigen Dienste	4	10	6	01.04
	Registration der administrativen Benutzer	0,5	0,5	0	01.04
	Erstellen der Firewall-Regeln	0,5	0,5	0	01.04
	Testen der Konfiguration und Fehlerbehebungen	1,5	1,5	0	01.04
	Erstellen der Anwenderdokumentation	8	2	-6	04.04 - 05.04
<b>Abschlussphase</b>	Soll-/Ist-Vergleich	1	0,5	-0,5	05.04
	Einweisung der Administratoren in das System	2	3	1	05.04
	Abnahme und Reflexion	2	2	0	05.04
<b>Gesamt</b>		<b>35</b>	<b>34,5</b>	<b>-0,5</b>	<b>29.03.22 – 05.04.22</b>

Tabelle 12: Projektablaufplan

## Integration eines Messenger-Servers

Personal	Tätigkeit	Dauer (Stunden)	Kosten pro Stunde (netto)	Gesamt (netto)
Wilko Reins	Projektdurchführung	35	25,00 €	875,00 €
Ausbilder	Bereitstellen des Servers	0,5	50,00 €	25,00 €
	Einweisung in die Administration der Softwarelösung	2	50,00 €	100,00 €
	Abnahme des Projekts	1		50,00 €
<b>Gesamtkosten</b>				<b>1050,00 €</b>

Tabelle 13: Personalkosten

Personal	Tätigkeiten	Soll Dauer (Stunden)	Ist Dauer (Stunden)	Kosten pro Stunde (netto)	Gesamt Soll (netto)	Gesamt Ist (netto)
Wilko Reins (Azubi)	Projektdurchführung	35	34,5	25,00 €	875,00 €	862,5 €
Ausbilder	Bereitstellen des Servers	0,5	0,5	50,00 €	25,00 €	25,00 €
	Einweisung in die Administration der Softwarelösung	2	2		100,00 €	100,00 €
	Abnahme des Pro- jekts	1	1		50,00 €	50,00 €
Personalkosten						1037,00 €
Posten				Menge (Monate)	Kosten pro Monat (netto)	Gesamt (netto)
Hosting- Server (Hetzner)				60	9,74 €	584,40 €
Softwarelö- sung (Mat- rix)				60	0,00 €	0,00 €
Gesamtkosten						1621,90 €

Tabelle 14: Kostenvergleich

### 9.4 Anwenderdokumentationen

#### 9.4.1 Anwenderdokumentation für Benutzer

Element ist eine Messenger-Applikation, zur Kommunikation über Matrix-Heimserver. Sie ist verfügbar als Webapplikation, für Android, iOS, und als Client für MacOS, Linux und Windows.

Damit sich neue Benutzer in Element registrieren und anmelden können, muss zuerst die Webseite oder die Applikation auf dem jeweiligen Endgerät aufgerufen werden. Die Webseite kann unter dem Link: „<https://app.element.io>“ aufgerufen werden. Daraufhin erscheint folgende Maske:

Je nachdem, ob man nun einen neuen Benutzer registrieren, oder einen bereits vorhandenen Benutzer anmelden möchte, wählt man „Anmelden“ oder „Konto erstellen“.

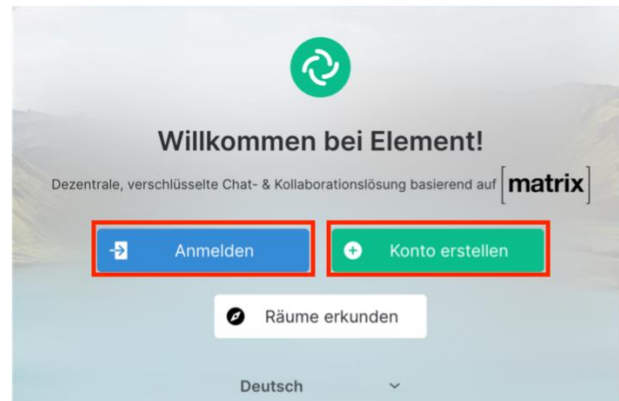


Abbildung 21: Willkommen bei Element

##### 9.4.1.1 Anmelden

Hierfür muss zunächst der Heimserver unter „Bearbeiten“ geändert werden (Abbildung 23). Eingegeben wird hier die Domain des Servers (Abbildung 22) und danach klickt man auf „Fortfahren“.

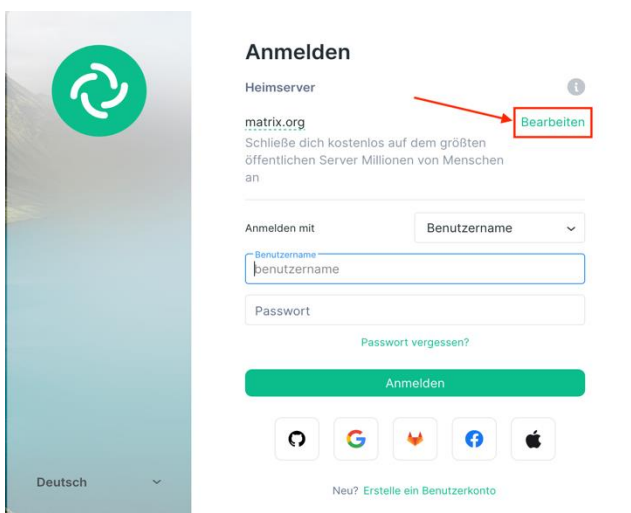


Abbildung 23: Bearbeiten des Heimservers

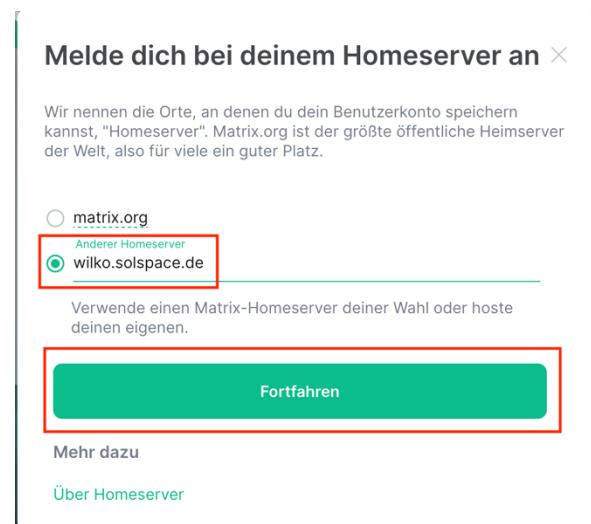


Abbildung 22: Ändern des Heimservers

Jetzt können die Benutzerdaten des Benutzers eingetragen werden und „Anmelden“ geklickt werden (Abbildung 24).



## Integration eines Messenger-Servers

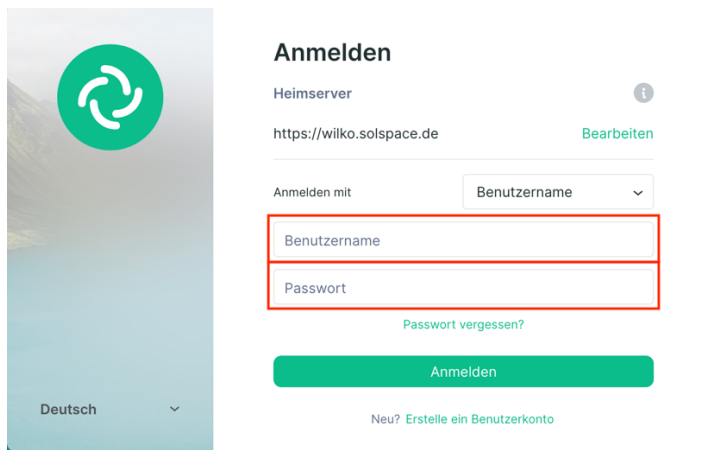


Abbildung 24: Eingabe der Benutzerdaten für die Anmeldung

### 9.4.1.2 Registrierung

Zur Registrierung wählt man die Option „Konto erstellen“ im Willkommensfenster (Abbildung 21). Daraufhin erscheint die Maske „Konto anlegen“, in der zuerst der Heimspace bearbeitet werden muss (Abbildung 26).

Dann kann der Heimspace wie in Abbildung 23 geändert werden und die gewünschten Benutzerdaten können eingegeben werden (Abbildung 25).

Abschließend kann die Registrierung durch das klicken auf den Knopf „Registrieren“ bestätigt werden.

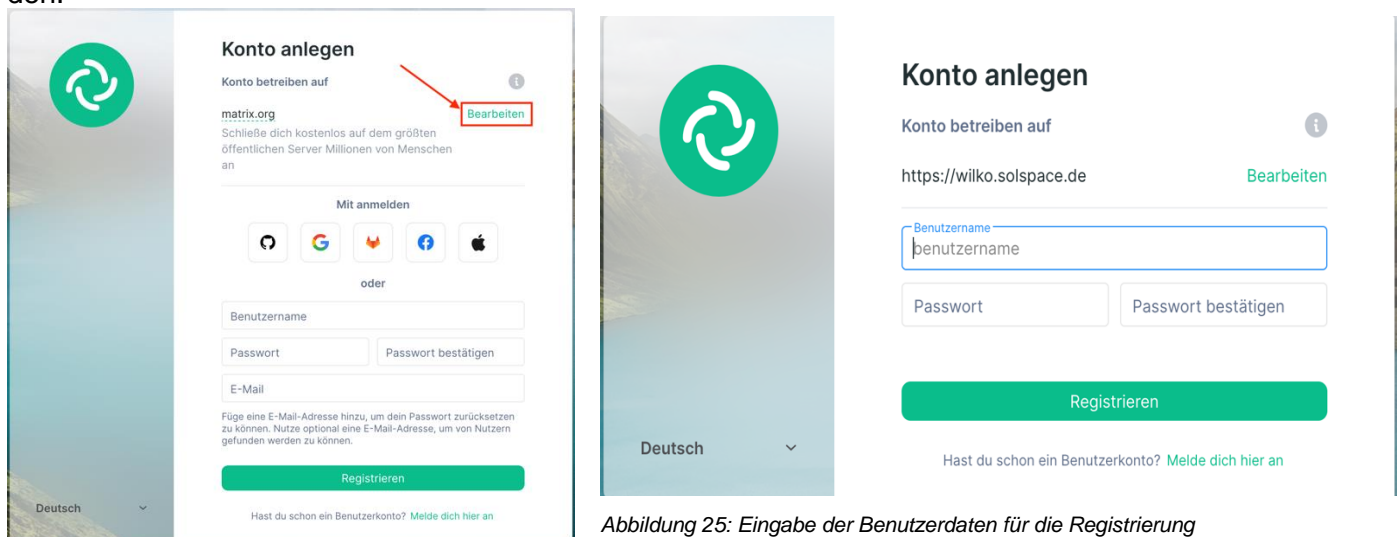


Abbildung 26: Bearbeiten des Heimspace für die Registrierung

Abbildung 25: Eingabe der Benutzerdaten für die Registrierung

### 9.4.1.3 Erste Schritte in Element

Hat man sich erfolgreich in Element angemeldet, bekommt man die Möglichkeit, neue Räume und Direktnachrichten zu erstellen sowie öffentliche Räume zu erkunden.

## Integration eines Messenger-Servers

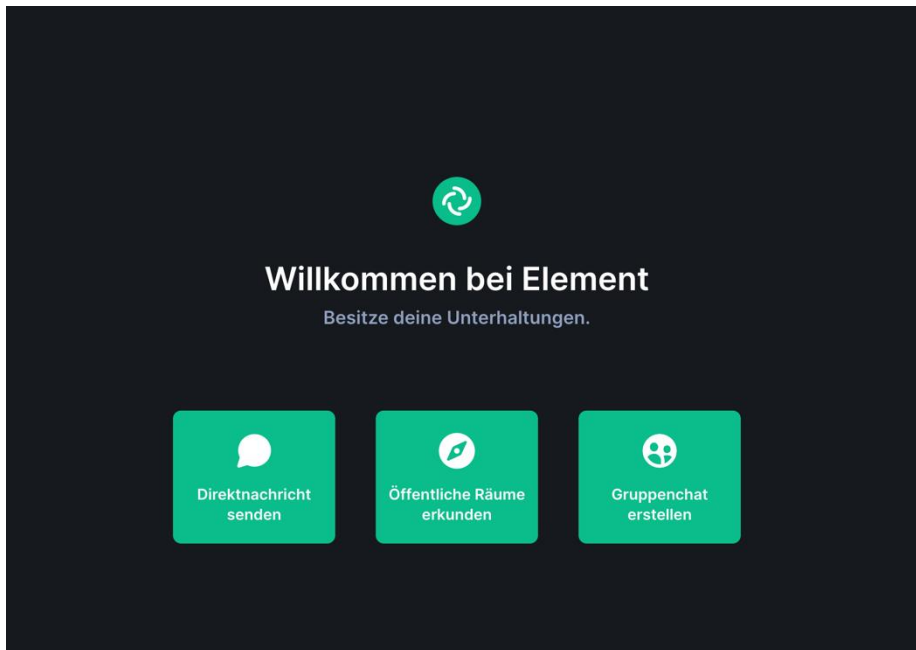


Abbildung 27: Startseite von Element nach der Erstanmeldung

Im Seitenpanel von Element (Abbildung 28) gibt es ebenfalls die Funktion, Kontakte und Räume zu verwalten.

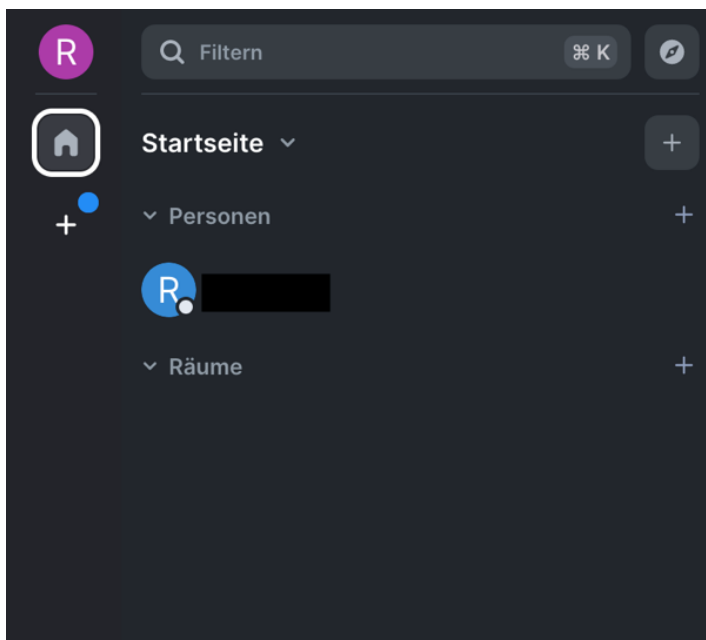


Abbildung 28: Seitenpanel des Element-Messengers

## 9.4.2 Anwenderdokumentation für Administratoren

### 9.4.2.1 Starten und Stoppen der Docker-Instanzen

Der Start der Docker-Container kann durch den Befehl „`docker-compose up -d`“ oder mit der Bash-Datei „`start.sh`“ initiiert werden. Der Befehl muss innerhalb des Matrix-Ordners ausgeführt werden, weil dieser sich auf die Compose-Datei bezieht und alle Services darin hoch und herunterfährt. Will man einzelne Container hochfahren, muss der Befehl „`docker-compose up -d <SERVICE_NAME>`“ ausgeführt werden.

Um die Docker-Container zu stoppen, muss der Befehl „`docker-compose down`“ oder die Bash-Datei „`stop.sh`“ ausgeführt werden. Dieser Befehl muss ebenfalls im Matrix-Ordner ausgeführt werden.

### 9.4.2.2 Status der Docker-Instanzen einsehen

Will man den Status der jeweiligen Docker-Container einsehen, dann muss der Befehl „`docker ps -a`“ ausgeführt werden. Daraufhin erhält man eine Ansicht, welche Container gestartet wurden, welche Ports sie belegen und welche Container-ID sie besitzen, seit wann diese laufen und ob diese nach wie vor laufen.

## 9.5 Quellenangaben

**[Quelle 1]** DSM-Online: „Datenschutz im Unternehmen“ (Online) unter: <https://www.dsm-online.eu/ratgeber/datenschutz-im-unternehmen/> [Zugriff am 29.03.2022]

**[Quelle 2]** Datenschutz-Praxis: „Download Checkliste: Datensicherung & Backup unter der DSGVO“ (Online) unter: <https://www.datenschutz-praxis.de/tom/download-checkliste-datensicherung-backup-unter-der-dsgvo/> [Zugriff am 29.03.2022]

**[Quelle 3]** Microsoft: „Die passende Microsoft Teams-Version für Ihre Anforderungen“ (Online) unter: <https://www.microsoft.com/de-de/microsoft-teams/compare-microsoft-teams-options> [Zugriff am 29.03.2022]

**[Quelle 4]** Matrix: „Introduction“ (Online) unter: <https://matrix.org/docs/guides/introduction> [Zugriff am 29.03.2022]

**[Quelle 5]** Matrix: „Rules“ (Online) unter: <https://matrix.org/media/2019-06-10%20-%20Matrix.org%20Foundation%20CIC%20Rules.pdf> [Zugriff am 29.03.2022]

**[Quelle 6]** Matrix: „Articles of Association“ (Online) unter: <https://matrix.org/media/2019-06-10%20-%20Matrix.org%20Foundation%20CIC%20Articles%20of%20Association.pdf> [Zugriff am 29.03.2022]

**[Quelle 7]** Mattermost: „Mattermost overview“ (Online) unter: <https://docs.mattermost.com/about/product.html> [Zugriff am 29.03.2022]

**[Quelle 8]** Mattermost: „Mattermost Pricing“ (Online) unter: <https://mattermost.com/pricing/> [Zugriff am 29.03.2022]

**[Quelle 9]** Rocket.Chat: „Startseite“ (Online) unter: <https://de.rocket.chat/> [Zugriff am 29.03.2022]

**[Quelle 10]** IONOS: „Wer wir sind“ (Online) unter: <https://www.ionos.de/unternehmen> [Zugriff am 29.03.2022]

**[Quelle 11]** IONOS: „VPS Tarife“ (Online) unter: <https://www.ionos.de/server/vps#tarife> [Zugriff am 30.03.2022]

**[Quelle 12]** Hetzner: „Über uns“ (Online) unter: <https://www.hetzner.com/de/unternehmen/ueber-uns> [Zugriff am 30.03.2022]

**[Quelle 13]** Hetzner: „Cloud-Server“ (Online) unter: <https://www.hetzner.com/de/cloud> [Zugriff am 30.03.2022]

**[Quelle 14]** Alfahosting: „Das Unternehmen Alfahosting“ (Online) unter: <https://alfahosting.de/alfahosting-das-unternehmen/> [Zugriff am 30.03.2022]

**[Quelle 15]** Alfahosting: „vServer-Hosting“ (Online) unter: <https://alfahosting.de/vserver-hosting/> [Zugriff am 30.03.2022]

**[Quelle 16]** Netcup: „Über Netcup“ (Online) unter: <https://www.netcup.de/ueber-netcup/> [Zugriff am 30.03.2022]

**[Quelle 17]** Netcup: „virtuelle Server“ (Online) unter: <https://www.netcup.de/vserver/vps.php> [Zugriff am 30.03.2022]

**[Quelle 18]** Netcup: „Rechenzentren“ (Online) unter: <https://www.netcup.de/ueber-netcup/rechenzentren.php> [Zugriff am 30.03.2022]

**[Quelle 19]** Trustpilot: „Bewertung von Unternehmen“ (Online) unter: <https://de.trustpilot.com/> [Zugriff am 30.03.2022]

## Integration eines Messenger-Servers

**[Quelle 20]** Docker: „Install Docker Engine on Ubuntu“ (Online) unter: <https://docs.docker.com/engine/install/ubuntu/> [Zugriff am 31.03.2022]

**[Quelle 21]** Docker: „Install Docker Compose“ (Online) unter: <https://docs.docker.com/compose/install/> [Zugriff am 31.03.2022]

**[Quelle 22]** Docker: „Matrix Docker“ (Online) unter: <https://hub.docker.com/r/matrixdotorg/synapse> [Zugriff am 01.04.2022]

**[Quelle 23]** Docker: „NginX Docker“ (Online) unter: [https://hub.docker.com/\\_/nginx](https://hub.docker.com/_/nginx) [Zugriff am 01.04.2022]

**[Quelle 24]** Docker: „Coturn TURN Server Docker Image“ (Online) unter: <https://hub.docker.com/r/coturn/coturn> [Zugriff am 01.04.2022]

**[Quelle 25]** Atlantic.net: „How to install Matrix Synapse and NginX with Let's Encrypt SSL on Debian 10“ (Online) unter: <https://www.atlantic.net/dedicated-server-hosting/how-to-install-matrix-synapse-with-nginx-and-lets-encrypt-ssl-on-debian-10/> [Zugriff am 31.03.2022]

**[Quelle 25]** Matrix: „Federation Tester“ (Online) unter: <https://federationtester.matrix.org/> [Zugriff am 05.04.2022]