Kurzdoku – Begehungstool / Chatbot (embedded)

Inhalt

[Einleitung 2](#_Toc187068939)

[Begehungstool 2](#_Toc187068940)

[Chatbot 2](#_Toc187068941)

[Tools / Services 2](#_Toc187068942)

[Funktionsweise 2](#_Toc187068943)

[Authentifizierung 3](#_Toc187068944)

[Datenbank 4](#_Toc187068945)

[Resend 5](#_Toc187068946)

# Einleitung

*Die folgende Dokumentation soll als grundlegender Anhaltspunkt zum Unterhalt des Begehungstool dienen. Sollten Probleme oder Fragen auftreten, die durch die Dokumentation nicht beantwortet werden können, könnt ihr euch unter* [*emmenegger@yanik.pro*](mailto:emmenegger@yanik.pro) *melden. Sollte ich Zeit haben, versuche ich zu helfen. Ich kann aber nicht zu viel Zeit investieren, da ich ja selbst noch arbeiten muss ;)*

*Alle Login-Daten für die genutzten Services können im Keepass abgerufen werden. Für den Zugang zum KeePass bitte bei Philipp Jent melden.*

*Aktuell wird bei allen Service die Gratis-Version genutzt. Diese reicht aktuell aus, sollte aber bei grösserem Benutzer-Zugriff ein Upgrade erhalten. Resend Pro kostet 20$ und Supabase Pro 25 pro Monat. Sollte das Limit erreicht wrden, muss dies mit Philipp angeschaut werden.*

## Begehungstool

Das Begehungstool wird von der Spitalhygiene dazu genutzt um Audits (Kontrollen) bei Operationen durchzuführen. Das Tool dient als Checkliste, welche während der Begehung geprüft wird. Nach Abschluss der Begehung kann ein Report erstellt werden, der per Mail an die Person gesendet wird, welche die Begehung durchgeführt hat. Der Report wird anschliessend weiterverarbeitet und schlussendlich abgelegt. Vor der Einführung des Tools, wurden Excel-Sheets verwenden, welche als Checkliste fungierten.

## Chatbot

In einem anderen Projekt (Luca Rolshoven), wurde ein Chatbot für die Spitalhygiene entwickelt. Da dieses Projekt bereits über eine Authentifizierung per @Insel.ch Mail beinhaltet und der Chatbot noch über keine derartige Authentifizierung verfügt, wurde der Chatbot ins Begehungstool integriert. Heisst, der Chatbot wird als Embedded App in dieses Projekt integriert. Da es sich aber nur um eine «IFrame» Integration handelt, wird darauf nicht näher eingegangen, der Code sollte einfach zu verstehen sein.

# Tools / Services

Damit das Begehungstool funktioniert, wurden verschiedene Tools und Services eingesetzt.

Ich werde in der Dokumentation nicht genauer auf die Programmierung des Codes eingehen, dieser sollte aber einheitlich und in sich selbst dokumentiert sein.

Das Tool wurde in Typescript geschrieben. Verwendet wurde das Webframework Nextjs. Um die Checklisten-Informationen zu speichern und die Authentifizierung gewährleisten zu können, wurde eine Supabase Instanz eingerichtet. Für das Versenden der Login-Mails und den Reports, wurde der Service von Resend eingesetzt. Genauere Dokumentation zu diesen Tools und Services sind hier:

[Supabase Docs](https://supabase.com/docs)

[Introduction - Resend](https://resend.com/docs/introduction)

# Funktionsweise

Wie bereits erwähnt handelt es sich beim Begehungstool um eine Checkliste, welche abgearbeitet werden muss. Dazu wurden in der Datenbank (siehe Kapitel Datenbank) Fragen, Abteilungen, Kategorien usw. erstellt. Die Daten stammen aus den Excel, welche zuvor genutzt wurden.

Ein Benutzer meldet sich im Tool an und begibt sich zum Internen Audit Tool. Dort besteht die Möglichkeit, eine neue Begehung zu starten. Als ersten Schritt muss er die Abteilung auswählen, für die die Begehung gedacht ist. Nach der Auswahl kann die Begehung gestartet werden. Dabei werden alle mit der Abteilung verknüpften Fragen geladen und es wird eine Begehung erstellt. Diese wird Lokal auf dem Endgerät in der IndexedDB des Browsers abgespeichert. Danach können die einzelnen Checkpunkte abgearbeitet werden. Sollte ein weiterer Checkpunkt dazukommen, können diese noch hinzugefügt werden. Sobald alle Checkpunkte abgearbeitet sind. Also erfüllt, nicht erfüllt oder nicht Anwendbar, kann die Begehung beendet werden. Diese kann dann entweder direkt oder zu einem späteren Zeitpunkt als Report versendet werden. Der Report wird immer an die Person versendet, die angemeldet ist.

*Siehe Benutzerdoku für mehr Info*

Technisch funktioniert es folgendermassen. Sobald eine neue Begehung gstartet wurde, werden die Abteilung von der API abgerufen und für den Benutzer bereitgestellt. Wenn dieser die Abteilung wählt und weiterfährt, wird ein Reivew-Objekt erstellt. Dies beinhaltet alle Informationen, die eine Begehung hat. Dafür werden alle Fragen, welche mit der Abteilung verbunden sind, geladen und als ReviewItem im Review-Objekt eingefügt. Das Review-Objekt erhält eine UUID und wird als Objekt in der lokalen Browser-IndexedDB abgespeichert. Somit ist das Review nun erstellt. Wenn der Benutzer die Review-Detail Seite öffnet, wird das Review aus der DB geladen. Dazu wird die ID als Slug von der URL extrahiert.

Das Review ist nun als Prop im ReviewComponent, welcher in ReviewContext ist. Über diesen Context und Component können nun verschiedene Aktionen durchgeführt werden. Alle Änderungen am Review werden direkt in der IndexedDB abgespeichert. Alle dazugehörigen Components sind unter /src/components/review zu finden.

Sobald das Review abgeschlossen ist, kann über die API /api/review/send ein Report generiert werden. Dabei wird das Review mit Hilfe des src/utils/documents/createWord.ts in eine Word-Datei umgewandelt und anschliessend mit Resend and den Benutzer versendet.

# Authentifizierung

Wie bereits erwähnt wird Supabase für die Authentifizierung genutzt. [Setting up Server-Side Auth for Next.js | Supabase Docs](https://supabase.com/docs/guides/auth/server-side/nextjs) – Anhand dieser Anleitung wurde die Authentifizierung aufgebaut.

Alle Requests werden von der Supabase-Middleware (/next-js-app/src/app/utils/supabase/middleware.ts) geprüft. Heisst, ist ein Benutzer nicht eingeloggt und die Url ist nicht «/login» oder «/api\*», wird er automatisch auf die Login-Seite weitergeleitet und hat keinen Zugriff auf das Tool. Die API Routes sind nicht direkt geschützt, werden aber durch Row-Level-Security auf Datenbank-Ebene geschützt.

Das Login funktioniert folgendermassen. Der Benutzer trägt seine @insel.ch Mail (andere Domain ist nicht möglich) ein und fordert einen Login-Link und OTP an. Supabase versendet über Resend eine Mail mit einem Link und dem OTP. Der Benutzer hat nun die Möglichkeit direkt den Link zu wählen oder den OTP einzugeben. Wenn der Link gewählt wird, wird der Benutzer auf dem Gerät eingeloggt, auf dem er den Link öffnet. Das OTP kann auf einem anderen Gerät eingegeben werden.

Welches Mail mit welchem Inhalt versendet wird, kann in Supabase unter Authentication>E-Mail-Templates nachgeschaut werden. Wichtig! Es muss dasselbe für Confirm Signup und Magic Link sein.

Bei erfolgreichen authentifizierung (/api/auth/confirm) wird ein Cookie (userAuthenticated) gesetzt mit dem Access Token. Dies ist notwendig, damit die Authentifizierung im Chatbot ebenfalls funktioniert. Anschliessend wird der Benutzer auf die Startseite, der Chatbot weitergeleitet.

# Datenbank

Wie bereits erwähnt, sind die Daten Checkpunkte und dazugehörigen Informationen in einer Supabase-Datenbank abgespeichert. Es handelt sich hierbei um eine PostgreSQL-DB. Folgendes Schema wird genutzt.

A screenshot of a computer

Description automatically generated

Die Benutzer selbst werden im Auth-Schema abgespeichert, dies ist ein separates Schema, welches von Supabase automatisch erstellt und betrieben wird.

Damit die Daten fürs Begehungstool zur Verfügung stehen, wurden eigene API-Routes erstellt. Eine OpenAPI Doku ist im Verzeichnis «doc» verfügbar.

Auf die Datenbank zugegriffen wird über die Supabase-Nodejs Library die einfache und sichere Datenbank Zugriffe erlaubt.

Damit die Datensicherheit gewährleistet werden kann, wurden für alle Tabellen Row-Level-Security (RLS) Policies erstellt.

**SELECT -> Authenticated Users**Alle Benutzer, die angemeldet sind, können per SELECT Befehl auf die Tabelle zugreifen.

**INSERT, UPDATE, DELETE -> Authenticated Users & In Admin User Tabelle**Destruktive Befehle (Einfügen, Aktualisieren und Löschen) können nur von Benutzern durchgeführt werden, welche in der admin\_users Tabelle sind. Die Benutzer müssen dort manuell eingetragen werden.

# Resend

Der Service von Resend wird für den Mail verkehr genutzt. Damit eine Mail versendet werden kann, muss eine Domain verknüpft werden. Zurzeit ist jedoch noch keine Domain der Insel verknüpft, sondern eine von mir (yanik.pro)

A screenshot of a computer

Description automatically generated

So bald wie möglich sollte hier eine Insel eigene Domain verknüpft werden. Dazu einfach den Schritt für Schritt Anleitungen von Resend folgen. Sobald eine neue Domain verknüpft ist, kein ein neuer API-Key erstellt werden. Dies kann ebenfalls in wenigen Schritten im Resend Dashboard gemacht werden. Der API-Key muss dann im env File der Nextjs App und in den Supabase Einstellungen ausgetauscht werden:



A screenshot of a black screen

Description automatically generated

# Admin Tool

Damit die Checklisten-Daten durch die Spitalhygiene angepasst werden kann, ohne dass dies direkt in der Datenbank gemacht werden muss, wurde ein Admin-Bereich erstellt. Dieser kann via Navigations-Buttons erreicht werden. Der Button wird nur Benutzern angezeigt, die in der admin\_user Tabelle eingetragen sind. Im Admin Bereich können bestehende Informationen angepasst, geändert oder gelöscht werden. Bei Bedarf können auch neue Informationen darüber hinzugefügt werden.

<https://infectioncontrol.ch/begehung/admin>

# Docker

Das Begehungstool wird auf einer virtuellen Maschine betrieben, die Logindaten befinden sich ebenfalls im Keepass. Durch Luca Rolshoven (Chatbot-Projekt) wurden bereits Zertifikate erstellt, um eine verschlüsselte Verbindung zu gewährleisten. Mit einem Docker Compose und einer Nginx Konfiguration wird eine Multicontainer-App aufgebaut. Sie besteht aus einem Nginx Server, welcher als Reverse Proxy dient. Alle Anfragen bis auf /chat werden an die NEXTJS App weitergeleitet. Die Anfragen nach Chat, werden an den Chatbot von Luca weitergeleitet.