Stadt Zürich
Schutz & Rettung

OST
Ostschweizer
Fachhochschule

Targeted Monitoring Tool

Studienarbeit

Studiengang Informatik
OST – Ostschweizer Fachhochschule
Campus Rapperswil-Jona

Herbstsemester 2020

Autoren:	Denis Nauli, Nadine Sennhauser
Betreuer:	Stefan Keller, Raphael Das Gupta
Projektpartner:	Schutz & Rettung Zürich – Christian Nüssli

Abstract

OpenStreetMap ist eine weit verbreitete Open-Source Karte. Mit dem stetigen Wachstum der Community werden auch die Anwendungsfälle immer kritischer. Deshalb wird auch der Wunsch nach Überwachung von Änderungen zentraler. Die Community ist zwar grösstenteils sehr vertrauenswürdig, es kann aber dennoch zu Fehlern kommen.

Eine umfassende Analyse der bestehenden Überwachungs-Tools ergab, dass keines ein Filtern nach Objekt-Tags erlaubt.

Da das Tool OSMCha bereits eine Menge Basisfunktionalitäten bot und die Struktur eine Filter-Erweiterung nach Objekt-Tags erlaubt, wurde darauf aufgebaut. Es wurde um die Möglichkeit erweitert, im Team OSM-Changesets nach Objekt-Tags zu filtern und so Änderungen an Objekten anhand des Filters zu überwachen. OSMCha wurde mit Python und dem Django Webframework entwickelt. Die Changesets stammen von Planet-OSM und werden in einer PostgreSQL Datenbank persistiert. Eine im Backend definierte Overpass-API-Abfrage retourniert aus den definierten Objekt-Tags eine Referenzfläche. Gefiltert werden alle Changesets, deren Changeset-Koordinaten die Referenzfläche überschneiden.

Das eigens entwickelte Frontend zum angepassten OSMCha-Backend wurde mit dem JavaScript-Framework React umgesetzt und kommuniziert über eine RESTful HTTP Schnittstelle mit dem Backend. Es wurde schlicht gehalten und konzentriert sich auf das Arbeiten im Team und das Überwachen von Changesets nach Objekt-Tags.

Management Summary

Ausgangslage

OpenStreetMap (OSM) ist eine weit verbreitete Open-Source Karte. Mit dem stetigen Wachstum der OSM-Community werden die Anwendungsfälle immer kritischer und auch der Wunsch nach Überwachung zentraler. Die Community ist grösstenteils sehr vertrauenswürdig, es kann aber dennoch zu Fehlern kommen. Solche Fehler können bei unserem Kunden "Schutz & Rettung Zürich" (SRZ) unter Umständen zu lebensbedrohlichen Situationen führen.

SRZ nutzt für ihre Einsätze OSM, da sie Daten in einem aussergewöhnlichen Detaillierungsgrad benötigen. Sie müssen in der Lage sein, direkt nach z.B. Restaurantnamen zu suchen. OSM ergänzt in diesen Bereichen die offiziellen Daten wie SwissNames optimal.

Einige Daten (z.B. Defibrillatoren) erfassen und pflegen die Datenkuratoren von SRZ in OSM selber. Leider werden die Daten, die für SRZ einsatzrelevant sind, teilweise von ungeübten Mappern verfälscht. Deshalb brauchen sie ein Tool, welches die Änderungen an diesen Objekten sichtbar macht.

Es sind aber keine Tools, die Änderungssätze (Changesets) nach Objekt-Tags filtern können verfügbar. Deshalb müssen die Notfalltelefonisten (Abbildung 1) von SRZ, die ihnen während dem Notruf aufgefallenen Fehler den internen Datenkuratoren melden, damit diese die Änderungen rückgängig machen können. Durch diesen Prozess können natürlich nicht alle Fehler gefunden werden.

Abbildung 1: Einsatzleitzentrale Schutz & Rettung Zürich (SRZ)

Vorgehen und Technologien

Es gibt bereits etliche Open-Source-Projekte, die Vandalismus erkennen können. Sehr viele dieser Tools wurden in dieser Studienarbeit eingehend analysiert. Die Analyse ergab, dass keines von diesen bestehenden Überwachungs-Tools ein Filtern nach Objekt-Tags erlaubte.

Da das Rad aber nicht neu erfunden werden muss, wurde entschieden auf dem bestehenden Tool OSMCha aufzubauen und dieses so zu erweitern, dass es das Filtern von Changesets anhand von Objekt-Tags erlaubt. OSMCha lässt sich dockerisieren und scheint ein aktiv unterhaltenes, weit verbreitetes Tool zu sein. Die in dieser Studienarbeit entwickelte Erweiterung könnte daher potentiell als Pull-Request an die Maintainer von OSMCha übergeben werden.

Am OSMCha-Backend, welches in Python mit Django aufgebaut wurde, wurde in der Grundstruktur nicht viel verändert. Die PostgreSQL Datenbank wurde leicht angepasst, da neu im Team gearbeitet werden kann. Die Nutzer können mit dem Status "in Bearbeitung" oder "abgeschlossen" anderen Nutzern klar machen, dass sich das Changeset gerade in Bearbeitung befindet oder bereits abgearbeitet wurde. Die RESTFul HTTP Schnittstelle auf Level 2 wurde ebenfalls erweitert, da die zusätzlichen Funktionen neue Abfragen verlangten.

Da Schutz und Rettung nur eine minimale Benutzeroberfläche für die Überwachung von Objekt-Tags benötigt, wurde mit React ein eigenes Frontend von Grund auf neu entwickelt. Für die rasche Orientierung seitens Benutzer wurde das Frontend in Anlehnung an bestehende Tools entwickelt. Die Karten-Ansicht mit aufbereiteter Änderungsansicht wurde als Node Package ebenfalls - wie schon das Backend (OSMCha) - von Map-

box bezogen. Dieses Element erlaubt eine interaktive Vorschau, sowie die Ansicht von Detail-Informationen der Changesets. Das Frontend kommuniziert über eine RESTful HTTP Schnittstelle mit dem Backend. Dadurch können Front- und Backend unabhängig voneinander weiterentwickelt werden.

Die Abbildung 2 zeigt die Technologien im Überblick, wobei noch zu erwähnen ist, dass die einzelnen Komponenten in eigenen Docker-Container laufen.

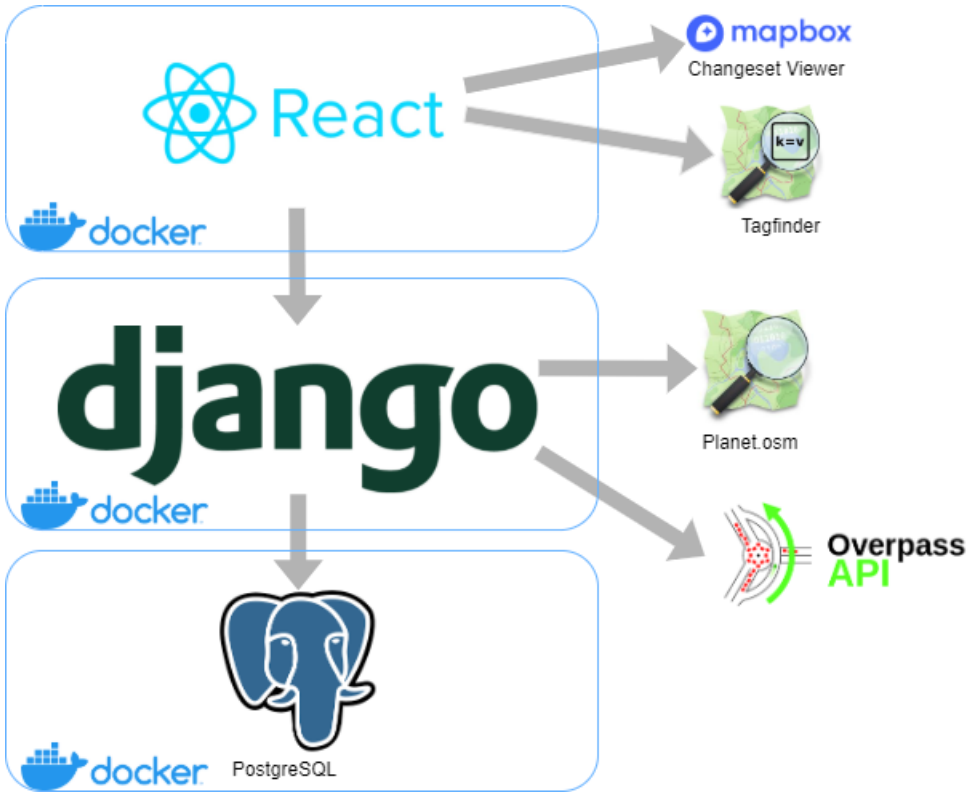


Abbildung 2: Technologien des Targeted Monitoring Tools im Überblick

Ergebnis

Das Targeted Monitoring Tool erlaubt endlich auch das Filtern nach Objekt-Tags bei Changesets. Wird ein Filter definiert kann dieser anderen Nutzern freigegeben werden. So können dann die gefilterten Changesets gemeinsam überprüft werden. Das Arbeiten im Team funktioniert problemlos, da durch die Status-Anzeige verhindert wird, dass mehrere Mitarbeiter gleichzeitig die gleichen Changesets bearbeiten.

Das Design des Frontends wurde auf Wunsch von Schutz & Rettung Zürich schlicht und übersichtlich gestaltet. Die Benutzeroberfläche erinnert an andere Monitoring-Tools, was die Bedienung deutlich vereinfacht. Auf der Startseite werden eigene und freigegebene Filter auflistet. Mit diesen können Changesets gefiltert werden. Die Abbildung 3 zeigt die Ansicht mit den gefilterten Changesets.

Für den Kunden wurde eine Installationsanleitung und ein Benutzerhadbuch geschrieben. Durch das bereitliegende Docker-Setup wird der Installationsprozess bedeutend einfacher.

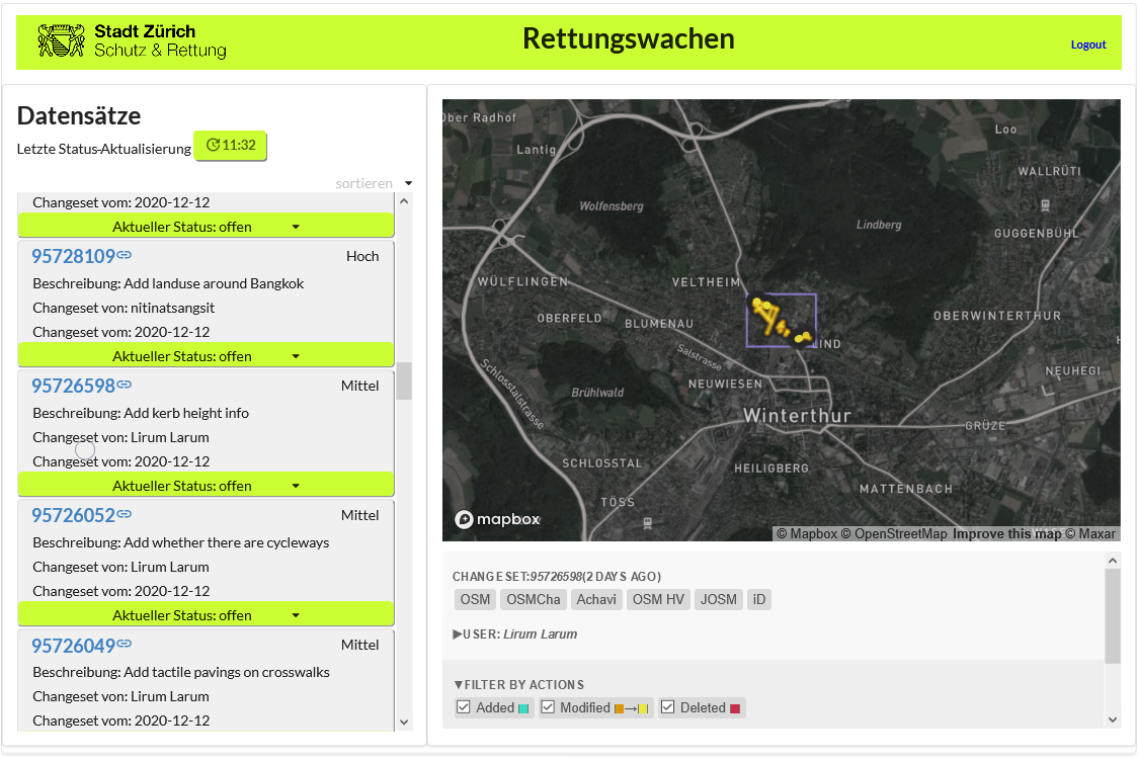


Abbildung 3: Benutzeroberfläche des Targeted Monitoring Tools

Ausblick

Der Code dieser Studienarbeit ist Open-Source und somit für jeden verfügbar. Erweiterungen und Verbesserungen können also von beliebigen Entwicklern angenommen werden.

Die Anzeige des Bearbeitungsstatus bei den Changesets wird nicht automatisch aktualisiert. Momentan ist das Targeted Monitoring Tool eher auf das Arbeiten in kleinen Teams ausgelegt. Ob in Zukunft auch grössere Mapping-Teams Interesse an der gemeinsamen Überwachung basierend auf Objekt-Tags haben, ist zwar fraglich, aber nicht ausgeschlossen. Es wäre trotzdem elegant, wenn die Status sich automatisch aktualisieren würden. Für Arbeiten in grösseren Teams wäre es zusätzlich praktisch, wenn sich der Benutzer selbst aus der Filter-Freigabe anderer löschen könnte.

Mail-Benachrichtigungen bei neuen Changesets wären ebenfalls eine mögliche Erweiterung. Ein Benachrichtigungsmechanismus wäre nur dann sinnvoll, wenn dieser aus- und eingeschaltet werden kann. Da bei gewissen Filtern täglich mehrere Changesets gefunden werden, würde es in Spam ausarten.

Die Karte im Frontend könnte selber implementiert werden, um mehr Kontrolle darüber zu haben.

Aufgabenstellung

Ein Werkzeug für die gezielte Überwachung von OpenStreetMap-Daten (Targeted Monitoring Tool)

- Studienarbeit im Herbstsemester 2020/2021 Bachelor Informatik
- Autoren: Nadine Sennhauser und Denis Nauli
- Betreuer: Prof. Stefan Keller, OST Campus Rapperswil
- Industriepartner: Christian Nüssli, Schutz und Rettung, Zürich

Aufgabenstellung

Grosse Community-Projekte wie OpenStreetMap werden immer mehr in geschäftskritischen Anwendungen eingesetzt, beispielsweise zur Geonamen-Suche in Notrufzentralen. Dabei stellt sich die Frage, wie Vandalismus erkannt werden kann. Dafür gibt es Monitoring-Tools, die möglichst viel automatisieren sollen.

Namentlich die überregionale Notruf- und Einsatzleitzentrale Zürich (SRZ) nutzt in ihrem Einsatzleitsystem seit einiger Zeit Daten von OpenStreetMap (OSM). Daneben werden selbstverständlich auch offizielle Daten wie Geonamen (SwissNames) oder Gebäudeadressen (GWR) verwendet. Zur Verwaltung der OSM-Daten gibt es beim SRZ ein kleines Datenkuratoren-Team beim SRZ. Dieses Team erfasst OSM-Daten teilweise selber und kommuniziert mit der OSM-Community wie üblich über Changeset-Kommentare. Das SRZ unterhält auch eine Wiki-Seite auf OSM.org.

Das Problem ist, dass es trotz des Vertrauens in die OSM-Community bei gewissen Datensätzen eine Kontrolle braucht, d.h. ein Monitoring der kritischen Datenobjekte. Hierfür gibt es Tools wie OSMCha (https://osmcha.org/) oder Overpass (https://overpass-turbo.osm.ch/), die eine Integration in eigene Applikationen wie diese ermöglichen.

Kritische Daten (Objektklassen) sind u.a. Flughafenareal, Flughafentore, Spitäler, Altersheime, Bahnhöfe / Haltestellen, Defibrillatoren, sowie Geonamen (Liste nicht abschliessend).

Die Aufgabe ist zweigeteilt:

- Teil 1 Evaluieren: Evaluieren bestehender Monitoring-Projekte (Open Source). Umsetzungsidee mithilfe der bestehenden Tools und den Requirements erarbeiten.
- Teil 2 Umsetzen: Adaptieren der geeignetsten Lösung durch Eigenentwicklung.

Funktionale Anforderungen an die Webapp sind:

- Dashboard mit Übersicht: Auffällige Datensätze (ID anzeigen)
- Listenansicht der zu prüfenden Elemente
- Link zu prüfendem Datensatz in OSM
- Möglichkeit den zu überwachenden Bereich räumlich zu definieren
- Möglichkeit Tags die überwacht werden sollen, einzutragen (Watchlist)
- Möglichkeiten der Benachrichtigung eines Supervisors (E-Mail, Dashboard)
- Benutzerverwaltung inkl. Rechtesystem (Administratoren und Supervisors)
- Branding mit SRZ Logo und Design (White Labeling)
- Gezielte Abfrage pro Mapper-Name

Lieferobjekte

1. Dokumentation, inkl. Textabstract (deutsch), Management Summary (deutsch), technischer Bericht und Software Engineering-Projekt (deutsch); Anhänge (Literaturverzeichnis, Inhalt).
2. Webapp Open Source "Targeted Monitoring Tool" funktionsfähig (als Docker) auf Server unter Kontrolle des Betreuers.

3. Die vom Studiengang geforderten bzw. empfohlenen Lieferobjekte: Poster (nur digital), Broschüren-Abstract, kein Kurzvideo.

Vorgaben/Rahmenbedingungen

- Frontend: HTML5, JavaScript; Unterstützung aktueller Browser (Chrome, Edge Chromium, Firefox, Safari); wenn möglich responsive (Mobile); React.
- Backend: Python.
- Persistenz: PostgreSQL.
- SW-Entwicklung auf OST-Server. Für das Hosting wird ein Docker-Konfig./Image an SRZ übergeben.
- Dokumentation: Latex

Vorgehen und Arbeitsweise: Die Studierenden wählen nach Rücksprache ein Vorgehensmodell zur Softwareentwicklung. Es gibt wöchentliche Meetings mit vorbereiteten Unterlagen; wobei Ausnahmen vereinbart werden können.

Dokumentation

Zur Dokumentation (vgl. auch Lieferobjekte oben):

- Die Abgabe ist so zu gliedern, dass die obigen Inhalte klar erkenntlich und auffindbar sind (einheitliche Nummerierung).
- Die Zitate sind zu kennzeichnen, die Quelle ist anzugeben.
- Verwendete Dokumente und Literatur sind in einem Literaturverzeichnis aufzuführen (nicht ausschliesslich Wikipedia-Links auflisten).
- Dokumentation des Projektverlaufes, Planung etc.
- Weitere Dokumente (z.B. Kurzbeschreibung, Eigenständigkeitserklärung, Nutzungsrechte) gemäss Vorgaben des Studiengangs und Absprache mit dem Betreuer.

Form der Dokumentation zuhanden Betreuer (Studiengang siehe separate Instruktionen):

- Bericht gebunden (1 Ex.)
- Alle Dokumente und Quellen der erstellten Software auf USB-Stick.

Bewertung

Es gelten die üblichen Regelungen zum Ablauf und zur Bewertung der Studienarbeit des Studiengangs Informatik mit besonderem Gewicht auf moderne Softwareentwicklung wie folgt:

- Projektorganisation (Gewichtung ca. 1/5)
- Bericht, Gliederung, Sprache (Gewichtung ca. 1/5)
- Inhalt inkl. Code (Gewichtung ca. 2/5)
- Gesamteindruck inkl. Kommunikation mit Industriepartner (Gewichtung ca. 1/5).

Ein wichtiger Bestandteil der Arbeit ist, dass eine lauffähige, getestete Software abgeliefert wird (inkl. Softwaredokumentation).

Weitere Beteiligte

Mitarbeiter des Instituts für Software.

Inhaltsverzeichnis

Teil I

Technischer Bericht

1 Einführung

1.1 Problemstellung

OpenStreetMap ist eine weltweite Open-Source Karte, die von jedem registrierten Benutzer bearbeitet und durch die Open-Source Lizenz in jedem Bereich verwendet werden kann. Inzwischen wird die Karte von vielen namhaften Unternehmen verwendet, wie z.B. Garmin, Apple, Niantic (PokemonGO) und vielen mehr.

Mit jedem neuen Verwendungszweck steigt auch die Erwartung an die Richtigkeit der Daten. Je kritischer die Anwendungszwecke sind, desto grösser wird das Verlangen nach automatisierter Überwachung kritischer Objekte. Leider kann unsachgemässe Bearbeitung oder Unwissenheit zu Fehlern in der Karte führen. Zudem gibt es auch in der Mappingcommunity schwarze Schafe, die in der Karte boshafte Manipulationen vornehmen.

Für Schutz & Rettung Zürich (SRZ) sind gewisse Daten in OSM einsatzrelevant. Falsche Daten können im schlimmsten Fall zu lebensbedrohlichen Situationen führen. Damit SRZ die sicherheitsrelevanten Daten überwachen kann, brauchen sie ein Monitoring-Tool, welches Änderungen überwacht.

Unter sicherheitsrelevante Daten fallen zum Beispiel Institutionen, die häufig angefahren werden. Das sind Krankenhäuser, Feuerwehranlagen, aber auch Altersheime und Eventlocations. Zudem pflegen sie ihre Daten in OSM selber. Dazu gehören Defibrillatoren oder Flughafentore. Auch diese Daten sollen nicht durch Fremdeinwirkung verfälscht

werden.

Zurzeit hat SRZ noch keine Möglichkeit, die Daten effizient zu überprüfen. Momentan werden Fehler erst erkannt, wenn ein Telefonist (Telefonisten nehmen Notrufe entgegen) einen Ort oder Objekt nicht mehr findet, die er bei einem früheren Einsatz noch finden konnte. Das leitet er dann an die internen Datenkuratoren bei SRZ weiter, welche sich der Sache annehmen. Mit dieser Variante werden Fehler erst (zu) spät oder gar nicht erst entdeckt.

1.2 Vision

In dieser Studienarbeit entsteht ein Monitoring-Tool für OSM-Daten. Dieses soll die Möglichkeit bieten, die Änderungssätze der Karte anhand ihrer Objekt-Tags zu überwachen.

Durch dieses Tool sollten einerseits die Telefonisten entlastet werden, damit sie sich besser auf ihre Hauptaufgabe - dem Lokalisieren des Anrufers - konzentrieren können und andererseits die Fehler in einsatzrelevanten Objekten effizient gefunden werden. Die Datenkuratoren sollten gemeinsam und zentral die Änderungen nachverfolgen können, ohne dass diese zuerst durch einen Mitarbeiter erkannt und gemeldet werden müssen.

1.3 Ziele und Unterziele

Der Auftrag definiert zwei Teilbereiche. Der erste Teil besteht aus dem Analysieren bestehender OSM-Monitoring-Tools, damit gegebenenfalls eine Lösung übernommen werden kann. Der zweite Teil der Aufgabe beinhaltet die Implementation der zusätzlich benötigten Funktionen, falls diese noch nicht im ausgewählten Tool vorhanden sind.

Das Tool soll über ein Dashboard mit einer Übersicht der zu prüfenden Elemente verfügen. Die Bearbeitung kann extern über einen Link in den OSM-Editor erfolgen. Die bezogenen Datensätze können räumlich und mit Tags eingegrenzt werden. Zudem soll es möglich sein, einzelne OSM-Benutzer zu überwachen. Über neue Datensätze sollte der Supervisor (Datenkurator) informiert werden. Die Benutzerverwaltung ermöglicht das Login von Administratoren und Supervisors. Die Darstellung erfolgt gemäss Vorgaben der SRZ.

1.4 Vorgehen

Bei einem Besuch der SRZ am Flughafen Zürich, wurden die Anforderungen des Kunden abgeholt und der Einsatz der Open-Source Daten miterlebt. Die Anforderungen die im Voraus in Papierform überreicht wurden, konnten so mit dem Kunden verfeinert und ausgearbeitet werden.

Mit den Anforderungen in Verbindung mit den erlebten Eindrücken vor Ort, wurde eine lange Liste bestehender Open-Source-Tools bewertet. Die Herausforderung dabei war zu verstehen, wie OSM genau funktioniert und wie die Daten zusammenhängen (Siehe Abschnitt 1.5 Für die Arbeit relevante OSM Strukturen). Weiter waren die Lizenzen ein zentrales Thema, in welchem wir uns erst einen Überblick verschaffen mussten. Erst dann konnten wir erkennen, welche Tools zu welchem Zweck verwendet werden dürfen (Siehe Abschnitt 1.6 Open-Source Lizenzen).

Eine sorgfältige Evaluation der bestehenden Überwachungs-Tools ergab, dass 1. keines ein Filtern nach Objekt-Tags erlaubte, und 2. keines ein Frontend enthielt, dass sich an die Bedürfnisse der SRZ anpassen liess. Ein Projekt, OSMCha, kam den gewünschten Funktionalitäten am nächsten. Darum wurde dessen Backend als Grundlage für diese Arbeit gewählt und das Frontend entsprechend den Anforderungen in React selber entwickelt.

Zum Abschluss des Projekts wurde ein Teil der ergänzten Funktionlität an OSMCha als Pull-Request zur Verfügung gestellt.

1.5 Für die Arbeit relevante OSM Strukturen

Um einige Entscheide und Realisierungen besser zu verstehen, sind die folgenden OSM Fakten zentral und werden deshalb kurz zusammengefasst:
Die Karte besteht aus Nodes, Ways, Closed Ways, Areas und Relations. Alle diese Strukturen können Tags besitzen. Diese Tags setzen sich aus Key=Value zusammen. Die Bearbeitung der Karte kann in zahlreichen Tools oder im Online-Editor gemacht werden. Die Änderungen, die in der Karte gemacht werden, sind in wenigen Minuten prozessiert und können gesichtet werden.[4]

Änderungsdaten eines Benutzer werden dabei in Changesets eingetragen. Diese bestehen aus mehreren Edits. Über diese Changesets kann die Geschichte zurückverfolgt werden. Jedes dieser Changesets hat eine Ausdehnung, die Bbox, in welcher alle Edits eingeschlossen sind. Auf diesen Changesets werden zudem Metainformationstags im

Key=Value Style abgelegt (Kommentar, Datum, Source, User,...). Nicht zu verwechseln sind die Changeset-Tags mit den Objekt-Tags.[3]

1.6 Open-Source Lizenzen

Es gibt etliche Lizenzen, die von der Open-Source Community verwendet werden. In der Evaluationsphase dieser Studienarbeit, musste ein besonderes Augenmerk darauf gelegt werden. Die häufigsten werden kurz erläutert:
GNU General Public License (kurz GNU GPL oder GPL) ist das Parade Beispiel einer Copyleft Lizenz. Dabei darf der Source Code nur weitergegeben werden, wenn er wieder mit der selben Lizenz versehen wird. Mit der Lizenz verfolgt man das Ziel, dafür zu sorgen, dass freie Software frei bleibt.[1]
Das **BSD** Lizenzmodell unterscheidet sich von der GNU General Public License (GPL) darin, dass es kein Copyleft enthält. Damit darf BSD-lizenzierte Software kopiert, verändert und verbreitet werden.[2]
Das spezielle an der **MIT** Lizenz ist, dass sie selbst nicht unter einem Copyright steht, sondern für den eigenen Gebrauch modifiziert werden kann. Sie zählt ebenfalls zu den freizügigsten Open-Source Lizenzen, die auf ein Copyleft verzichten.

2 Stand der Technik

Das Problem der unzureichenden Kontrolle über kritische Daten kennt nicht nur SRZ. Dem entsprechend gibt es etliche Tools, die auf verschiedene Art und Weise helfen können, Daten zu kontrollieren.

2.1 Bestehende allgemeine Tools für OSM

Die folgenden Tools wurden genauer analysiert. Eine komplette Liste der bestehenden allgemeinen Tools, die in dieser Studienarbeit angeschaut wurden, befindet sich im Anhang 1.

- **Overpass Turbo** (https://osm.li/OIG)

  Beschreibung: Overpass Turbo ist ein online Daten-Filterungs-Werkzeug für OpenStreetMap. Zum Frontend Overpass Turbo gibt es die Overpass-API. Die Abfragen

können über Tags und vieles mehr definiert werden und liefern passende OSM-Objekte.

- **OSM TagFinder** (https://tagfinder.herokuapp.com/)

  Beschreibung: Mit der Webseite lassen sich schnell und unkompliziert Begriffe nachschlagen. Die gefundenen Tags werden übersichtlich aufgelistet. Auf den ersten Blick werden die relevantesten Informationen über einen Tag dargestellt. Für mehr Informationen und Details sorgt eine Weiterleitung direkt zur Wiki-Seite und den Statistiken von TagInfo.org.

- **Taginfo Schweiz** (https://taginfo.openstreetmap.ch)

  Beschreibung: OpenStreetMap benutzt Tags, um geografischen Objekten eine Bedeutung zuzuorden. Es gibt keine feste Liste dieser Tags. Wenn nötig, können neue Tags jederzeit erfunden und benutzt werden. Taginfo zeigt Statistiken, welche Tags in der Datenbank vorhanden sind, wie viele Mapper diese Tags benutzt haben, wo sie benutzt werden usw. Taginfo holt sich auch Informationen zu den Tags aus dem Wiki und aus anderen Quellen und bringt alle dies zusammen. So versteht der Benutzer, wie Tags benutzt werden und was sie bedeuten.

- **Taginfo** (https://taginfo.openstreetmap.org/)

  Beschreibung: Siehe Taginfo Schweiz (oben). Der Unterschied: Die Daten beziehen sich auf OSM weltweit.

2.2 Bestehende Fehlerreport-Tools OSM

Es wurden 2 Tools angesehen, siehe Anhang 1. Der Kunde wollte nur Changesets angezeigt haben und keine der OSM-Notes die auf Unstimmigkeiten hinweisen könnten. Deshalb wurden diese Tools nicht weiter analysiert.

2.3 Bestehende Tools zur Fehlererkennung OSM

Die folgenden, wichtigsten Fehlererkennungstools wurden genauer analysiert. Eine vollständige Liste der bestehenden Tools befindet sich im Anhang 1.

- **Osmose** (http://osmose.openstreetmap.fr)

  Beschreibung: Es werden technische Fehler gefunden (Strassenteile, die nicht miteinander verknüpft sind - Problematik beim Routen berechnen). Osmose bietet

Kartenansichten, um die Qualität zu verbessern und speziell aufbereitete Layer, die Probleme aufzeigen können. Teils nur in Frankreich und teils weltweit.

- **OSM Inspector** (https://tools.geofabrik.de/osmi/)

  Beschreibung: Wurde erstellt, um verschiedene Ansichten für die Fehlerbehebung zu erhalten. Eine View betreffend Adressen könnte eventuell interessant sein.

- **Map of Turn Restrictions** by Zartbitter (https://ahorn.lima-city.de/tr/)

  Beschreibung: Es zeigt die Beschilderung von Strassen (nur rechts/links, nur gerade aus, nicht links/rechts, Einbahn...). Dabei markiert es Fehler und warnt bei Widersprüchen dieser Bezeichnungen.

- **Restriction Analyzer** by MorbZ (https://restrictions.morbz.de)

  Beschreibung: Gleiche Funktionalität wie Map of Turn Restrictions by Zartbitter

- **Relation Analyzer** (http://ra.osmsurround.org/)

  Beschreibung: Der Relation Analyzer untersucht Relationen auf Lücken. Dies ist besonders für alle Arten von Routen-Relationen nützlich, einschliesslich derer, die gerade berbeitet wurden.

- **qa.poole.ch** (http://qa.poole.ch/)

  Beschreibung: Zeigt Strassen ohne Namen.

- **Staty** (https://staty.cs.uni-freiburg.de/)

  Beschreibung: Zeigt Unstimmigkeiten bei Public Transport Stops.

- **To-Fix** (https://github.com/osmlab/to-fix)

  Beschreibung: Ein Task Manager für Openstreetmap, um sich mit anderen zu koordinieren und eine Liste von Tasks abzuarbeiten, ohne sich gegenseitig zu stören.

2.4 Bestehende Monitoring-Tools für OSM

- **OSMCHA - OSM Changeset Analyzer** (https://osmcha.org)

  Beschreibung: Monitoring-Tool mit vielen Filtermöglichkeiten. Es enthält alle Changesets.

- **OSM Hall Monitor** (https://github.com/ethan-nelson/osmhallmonitor)

  Beschreibung: Mit OSM Hall Monitor können Änderungen verfolgt werden, die von bestimmten Benutzern, an bestimmten Objekten oder mit bestimmten Tags

vorgenommen wurden. Es können Benachrichtigungen aktiviert werden, um E-Mails zu empfangen, wenn eine der markierten Personen etwas bearbeitet oder eines der beobachteten Elemente bearbeitet wird. Grundlegende Funktionen für die Überwachung verdächtiger Änderungssätze sind ebenfalls enthalten.

3 Evaluation

3.1 Kriterien

In der Analyse der bestehenden Produkten hat sich gezeigt, dass eine komplette Eigenentwicklung nicht sinnvoll wäre. Es gibt bereits etliche ähnliche Tools, die gut um die benötigte Funktionalität erweitert werden können.

Wir haben die Tools bei der Evaluation in 5 Aufgabenbereiche unterteilt. Diese Teilgebiete kommen daher, dass wir zum einen eine Codebasis für die Erweiterung festlegen müssen und zum anderen passend dazu Tools evaluieren, mit denen wir die Erweiterungen umsetzen können.

Pro Aufgabengebiet werden die Tools in einer Tabelle verglichen. Die Überlegungen und Entscheidungen werden im Abschnitt darunter festgehalten.

In den farbigen Boxen können die finalen Entscheidungen mit Begründung in Form eines deutschen Y-Approach auf einen Blick nachvollzogen werden.[6]

Y-Approach

Backend: *Im Bereich Backend haben wir angesichts der knappen Zeit und der Vermeidung von Code-Nachbau entschieden, keine Eigenentwicklung zu erarbeiten, um unsere Kernfunktionalität - ein Filter nach Objekt-Tag - zu erreichen. Wir vernachlässigen dabei die entstehende Abhängigkeit und akzeptieren das Risiko einer aufwändigen Einarbeitung.*

3.2 Evaluation der Codebasis

Funktion	OSMCha	OSM Hall Monitor	To-Fix
Auffällige Datensätze anzeigen	Ja	Nein	Ja
OSM-Auth Login	Ja	Nein	Ja
Filter (Tag, Bereich, User)	Ja (Bereich, User und viele weitere nicht benötigte)	Ja (Single Node, Way, Relation mit Id und User)	Nein
Teambearbeitung	Nein	Nein	Ja (nur wenn man selber hostet)
Mail Benachrichtigung	Nein	Ja	Nein
Aktive Nutzer	Ja	Nein	Nein
Aktiver Unterhalt	Ja	Nein, Last 2018	Nein, Last 2017
Technologie Einsatz	Python, React	Python, React	Javascript, React

Table I.1: Code Grundlage

OSMCha wird die Grundlage für unser Backend bilden und darauf wird anschliessend aufgebaut. Ein Pull-Request an die Maintainer ist vorgesehen, kann aber nicht garantiert werden, da unsere Erweiterungen ziemlich einschneidend sind. Grund für den Entscheid OSMCha zu verwenden ist, dass das OSMCha Backend schon eine Menge der Basisfunktionalitäten liefert, die wir für unser Tool brauchen werden.

Keines dieser und der weiteren analysierten Tools lässt sich auf unsere Vorgaben an das Frontend anpassen. Deshalb wurde entschieden, dass das Frontend eine separate Eigenentwicklung geben soll. Das hat unter anderem den Vorteil, dass es nur die von uns vorgesehenen Funktionalitäten unterstützt und keine zusätzlichen, nicht benötigten Abhängigkeiten besitzt.

Die beiden Tools OSM Hall Monitor und To-Fix haben uns zur Funktionalität "Arbeiten im Team" inspiriert. Mit einem Gespräch mit dem Auftraggeber war klar, dass ihr Tool so eine Funktionalität auch besitzen sollte. Sie sind zwar nur ein kleines Team, aber dennoch sollte verhindert werden, dass zwei Datakuratoren die gleiche Änderung prüfen.

Y-Approach

Frontend: *Im Bereich Frontend haben wir angesichts der Design-Wünsche des Kunden entschieden, eine Eigenentwicklung zu erarbeiten, um eine schlichte und einfache Oberfläche zu erstellen. Wir vernachlässigen dabei, dass wir gegebenenfalls Code Fragmente nachbauen müssen und akzeptieren, dass wir mit Problemen kämpfen, die bereits einer in einem bestehenden Code gelöst hat.*

Y-Approach

Backend: *Im Bereich Backend haben wir uns angesichts des Funktionsumfangs entschieden, OSMCha als Grundlage zu verwenden, um uns den Start zu erleichtern. Wir vernachlässigen dabei, dass es für uns zu weitreichende Filtermöglichkeiten gibt und akzeptieren, dass OSMCha keine Teamarbeit und keine Benachrichtigungen bietet.*

3.3 Evaluation der auffälligen Datensätze

Funktion	Overpass Turbo	Planet osm.org
Erhalten von Changesets	Ja	Ja
Anzeigen von potentiellen Fehlern	Nein	Nein
Gefilterte Daten erhalten	Ja	Nein
API zur einfachen Integration	Ja	Ja
Hilfe zum Filtern	Nein	Nein
Geschätzte Zeitverzögerung*	Keine	1d
Wird aktiv unterhalten	Ja	Ja
Wird von OSMCha bereits unterstützt	Nein	Ja

Table I.2: Auffällige Datensätze alle Änderungen

Funktion	Map of Turn Restrictions by Zartbitter	Restriction Analyzer by MorbZ	Osmose	Relation Analyzer	qa.poole.ch	Staty
Erhalten von Changesets	Nein	Nein	Nein	Nein	Nein	Nein
Anzeigen von potentiellen Fehlern	Ja	Ja	Ja	Ja	Ja	Ja
Gefilterte Daten erhalten	Nein	Nein	Ja	Nein	Nein	Nein
API zur einfachen Integration	Nein	Nein	Ja	Nein	Nein	Nein
Hilfe zum Filtern	Nein	Nein	Nein	Nein	Nein	Nein
Geschätzte Zeitverzögerung*	1 Minute	1 Minute	1 Minute	1 Minute	5 Minute	2 Minute
Wird aktiv unterhalten	Nein	Nein	Ja	Nein	Jein, Dev ist aktiv, Tool nicht	Ja
Wird von OSMCha bereits unterstützt	Nein	Nein	Nein	Nein	Nein	Nein

Table I.3: Auffällige Datensätze Fehlererkennungstools

* Geschätzte Zeitverzögerung: Dabei ist gemeint wie lange es dauert, bis eine Änderung im OSM von diesm Tool erkannt wird.

Damit wir den Anforderungen des Kunden gerecht werden können, werden wir die Daten von Planet osm.org beziehen. Damit können wir alle Changesets anziehen. Für unseren Kunden sind alle Änderungen als auffällige Datensätze anzusehen, sofern sie den Filterkriterien entsprechen, denn jede Änderung im Bereich ist potentiell falsch. Deshlb wurden die Fehler- und Vandalismus-Erkennungstools nicht weiter beachtet.

Unser Kunde hat sich dennoch eine Klassierung der Daten gewünscht. Diese wurde mit ihm zusammen definiert. Da die Fehlererkennungstools für ihn nicht interessant waren, blieb nur noch die Möglichkeit, nach Changeset-Informationen zu klassieren. Es wurde entschieden, nach der Art der Änderung zu klassieren. Jedes Changeset mit

mindestens einer Löschung ist bedrohlicher, als eine Änderung oder eine Ergänzung. Dem entsprechend wird jedes Changeset mit mindestens einer Löschung als "Hoch" klassiert. Changesets mit mindestens einem Edit werden als "Mittel" eingestuft und der Rest der Changests (enthält nur neue Elemente) wird als "Gering" klassiert.

Y-Approach

Aufällige Datensätze beziehen: *Im Bereich aufällige Datensätze beziehen, haben wir angesichts der Entscheidung OSMCha zu verwenden, beschlossen, die Changesets von Planet osm.org zu beziehen. Wir vernachlässigen dabei, dass es etliche Tools zur Fehlerklassierung gäbe und akzeptieren, dass unser Monitoring Tool alle Änderungen enthält und nicht nur auffällige Datensätze.*

Y-Approach

Klassierung: *Im Bereich Klassierung der Datensätze haben wir mit dem Kunden angesichts der grössten Problematik einer Löschung entschieden, dass hinzufügen als Gering, modifizieren Mittel und löschen als Hoch klassiert wird. Wir vernachlässigen dabei, dass es weitere sinvolle Klassierungen gäbe und akzeptieren die minimale Lösung.*

3.4 Evaluation Filter nach Tag

Funktion	Changeset Analyser
Nach Tags filtern	Ja
Buffer um Tags	Nein
Filter dynamisch anpassbar	Nein
Einfache Integration	Ja
Wird aktiv unterhalten	Ja
Wird von OSMCha bereits unterstützt	Ja

Table I.4: Evaluation Changeset Analyser

Die Filtrierung nach Tag mit der Möglichkeit eines Buffers um die selektierten Objekte gibt es zurzeit nicht. Es gibt den Changeset Analyser, der eine Filtrierung nach Tag

ermöglicht. Diese Filter werden aber fix im Source-Code definiert und erlauben keinen Buffer. Deshalb und wegen der Zeitvorgabe, in der das Projekt fertig abgegeben werden muss, haben wir uns für eine einfache Implementation über eine Referenzfläche entschieden. Dabei werden die Tags in eine Referenzfläche umgewandelt und bei Wunsch ein passender Buffer um die Objekte gelegt. Mit dieser Referenzfläche werden anschliessend die Changesets gefiltert.

Funktion	overpass turbo	Overpassalternativen
Unbegrenzte Anzahl Abfragen	Ja	Ja
Unbegrenzte Anzahl Rückgabe Objekte	Nein	Nein
Weltweit einsetzbar	Ja	Nein
Einfache Integration	Ja	Ja
Wird aktiv unterhalten	Ja	Ja
Wird von OSMCha bereits unterstützt	Nein	Nein

Table I.5: Evaluation Overpass und co.

Die Referenzflächen werden kaum so häufig aktualisiert, dass es in diesem Projekt zu Problemen mit der Overpass-Abfrage kommen sollte. SRZ wird voraussichtlich zwischen 4-6 aktive Benutzer haben. Wir können Overpass somit einsetzen ohne, dass wir die Anfragelimite überschreiten. Die Abfrage dauert dadurch etwas länger und wurde mit einem entsprechenden Lade-Symbol dem Benutzer angezeigt.

Y-Approach

Evaluation Filter nach Tag: *Im Bereich Filter nach Tag haben wir angesichts der Statik und der verwendeten Technologie entschieden, den Changeset Analyser nicht zu verwenden. Wir vernachlässigen dabei, dass es in OSMCha bereits verwendet wird und akzeptieren, dass wir somit die Änderungen direkt im Django-Backend erarbeiten und keine schöne Trennung haben werden.*

Y-Approach

Referenzfläche berechnen: *Im Bereich Referenzfläche berechnen haben wir angesichts der eingeschränkten Alternativen entschieden, Overpass zu verwenden. Wir vernachlässigen dabei, dass es zu einem Problem führt, wenn zu viele Anfragen in zu kurzer Zeit abgeschickt werden und akzeptieren das Risiko einer fehlerhaften Antwort bei einer Abfrage, die nach zu vielen Elementen sucht.*

3.5 Evaluation Frontend Inhalte

Funktion	Overpass Turbo	Planet osm.org	OSMCha Changeset Viewer
Map Data	Ja	Ja	Ja
Details Data	Ja	Ja	Ja
Einfachen Integration	Data Ja; View nein	Karten Ansicht Ja, Details nein	Ja (Details und Karte)
Wird aktiv unterhalten	Ja	Ja	Ja

Table I.6: Evaluation Frontend Inhalte

Der Changeset Viewer weist die besten Inhalte vor. Dabei werden die Changeset-Änderungen direkt in der Karte dargestellt und es werden einige Changeset-Informationen dargestellt. Zudem ist es ein junges, sehr aktiv weiter entwickeltes Tool, bei welchem auch ein von uns erstellter Issue in kürzester Zeit angegangen und repariert wurde. Somit scheint es nicht nur die passendste, sondern auch die Lösung mit der einfachsten Implementierung zu sein. Die Nachteile, dass sich im jungen Code noch Fehler finden lassen, lässt sich bei älteren Tools oder einer eigenen Nachimplemetierung ebenfalls nicht ausschliessen. Deshalb wurde der Changeset Viewer gewählt. (Nachtrag: Am Ende des Projekts wurde uns bekannt, dass die Hersteller vielleicht bald keine Open-Source Projekte mehr anbieten. Die Implementierung konnte aus Zeitgründen nicht mehr verändert werden.)

Y-Approach

Evaluation Frontend Inhalte: *Im Bereich Vorschau haben wir angesichts der hilfreichen Ansichten entschieden, OSMCha Changeset Viewer zu implementieren. Wir vernachlässigen dabei, dass es sich um ein Tool handelt, dass noch sehr jung ist und akzeptieren, dass es zu Fehlern kommen kann, die aber mit einem Package-Update durch Mapbox eliminiert werden können.*

3.6 Evaluation Frontend Inputvalidierung

Funktion	TagfInfo	Tagfinder
Tags Eingabe Hilfe	Ja	Nein
Wird aktiv unterhalten	Ja	Nein

Table I.7: Evaluation Inputvalidierung Frontend

Die Inkludierung einer solchen Prüfung der Tags ist für eine dynamische Eingabe der Überwachungstags kaum wegzudenken. Die einfache Implementierung der Tagfinder-API hat uns dazu bewogen, diesen zu verwenden. Dabei kann eine Anfrage mit Key und Value gemacht werden und wenn die Antwort einen Inhalt entält, ist es ein gültiger Tag. Wenn trotz 200er Status kein Inhalt vorhanden ist, ist der Tag nicht gültig. Da das Tool aber nicht immer erreichbar sein wird und Anfragen zu lange dauern könnten, wurde ein passendes Errorhandling mit implementiert.

Y-Approach

Evaluation Frontend Eingabehilfe: *Im Bereich Vorschau haben wir angesichts der einfachen API entschieden, Tagfinder zu implementieren. Wir vernachlässigen dabei, dass es zu Ausfällen des Tools kommen kann und akzeptieren den Bedarf eines passenden Errorhandlings.*

3.7 Architektur des Lösungsansatzes

Die Architektur baut auf einer PostgreSQL-DB auf, welche mit Changesets von Plant-OSM gefüllt wird. Dabei werden die Changesets zum ersten Mal gefiltert, sodass nur Schweizer Changesets gespeichert werden.

Die nächste Schicht bildet das Django-Framework, welches Data Models Business-Logik und eine Rest Schnittstelle zur Präsentation beinhaltet. Zudem werden aus dem Django Filter-Model die Overpass-Abfragen gesendet.

Das React Frontend ist stark entkoppelt und kommuniziert über RESTful HTTP Abfragen mit dem Backend. Über drei Seiten kann der Benutzer hier auf einfachste Art und Weise mit dem Backend kommunizieren. Die drei Seiten setzen sich aus verschiedenen, widerverwendbaren Komponenten zusammen. Die Daten werden Dabei intern in einem Reduxstore gehalten. Das Login erfolgt über Oauth mit einem OSM-Benutzerkonto.

4 Umsetzungskonzept

Abbildung I.1 zeigt den Aufbau der Umsetzung des Targeted Monitoring Tools.

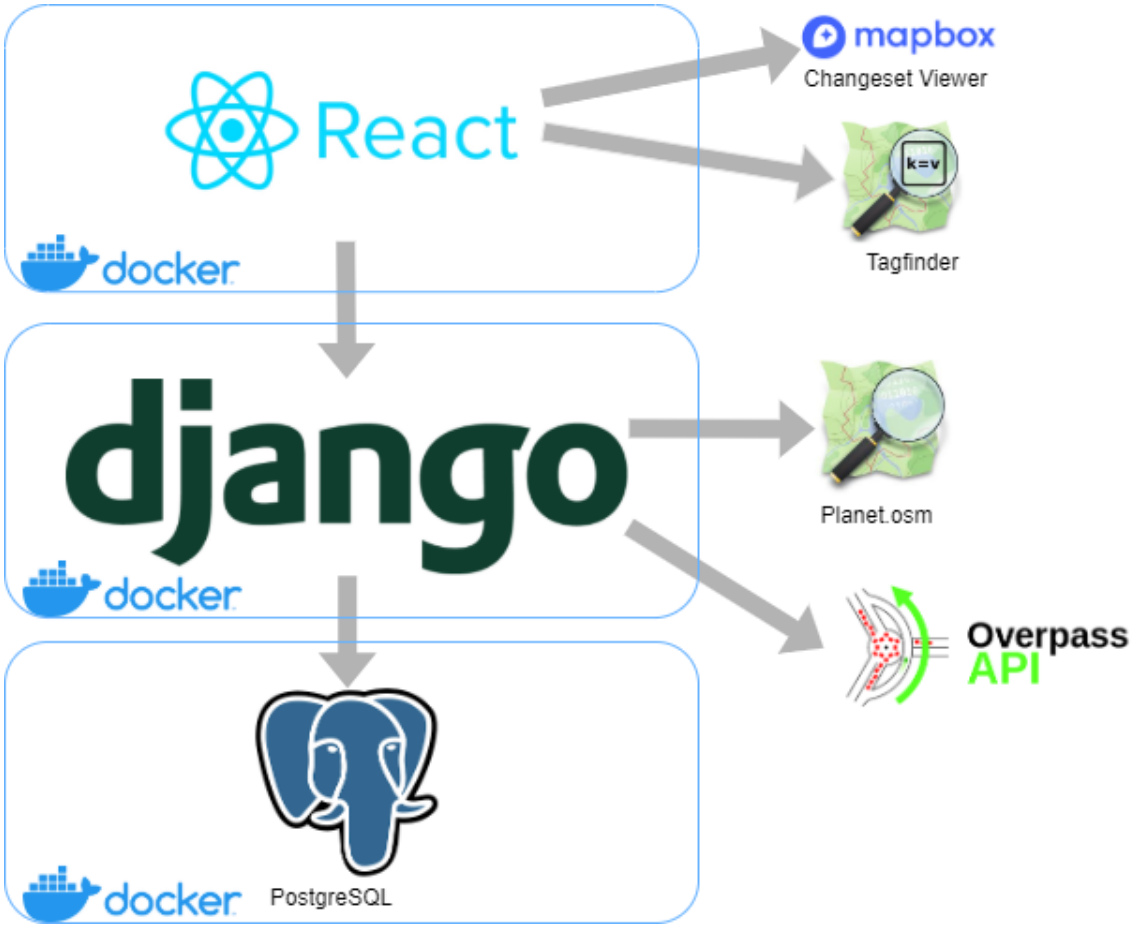


Abbildung I.1: Überblick über den Aufbau der Lösung

4.1 Filter nach Tag

Die Filtererweiterung im Backend berechnent via Overpass eine Referenzfläche. Diese alleine reicht aber nicht. Damit auch die gelöschten Objekte angezeigt werden, wird die neue Referenzfläche der Overpass Abfrage mit der Fläche der gespeicherten Fäche der letzen Filterabfrage vereinigt. Zudem wird anschliessend ein kleiner Buffer um die Elemente gerechnet, damit auch Punkt-Objekte eine Grundfläche haben. Diesen Buffer kann der User auch noch erweitern, wenn er neben seinen Objekten zum Beispiel auch noch die Anfahrtsstrassen überwachen möchte.

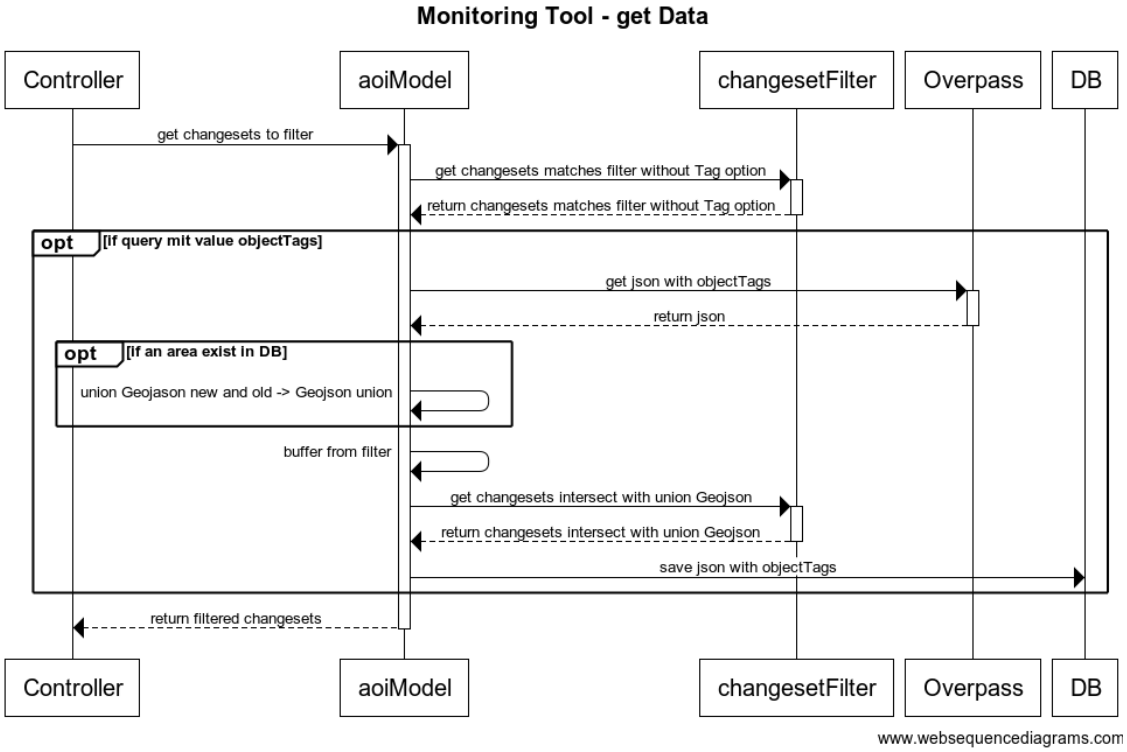


Abbildung I.2: Sequenzdiagramm Filter nach Objekt-Tag

Diese berechnete Gesamtfläche und die BBox-Flächen der Changesets werden einem "Rest Framework Gis Geometrie Filter" übergeben. Dieser behält alle Changesets, die mit der Referenzfläche intersecten, die restlichen werden entfernt. Anschliessend werden wie bei OSMCha üblich die weiteren Filter durchgefürt. Die Changesets, die am Ende noch übrig sind und auf alle Filtereinstellungen passen, werden via RESTFul HTTP an das Frontend gesendet.

Die Abbildung I.3 veranschaulicht, wieso es notwendig ist, die neue und alte Overpass-Fläche zu verschmelzen.

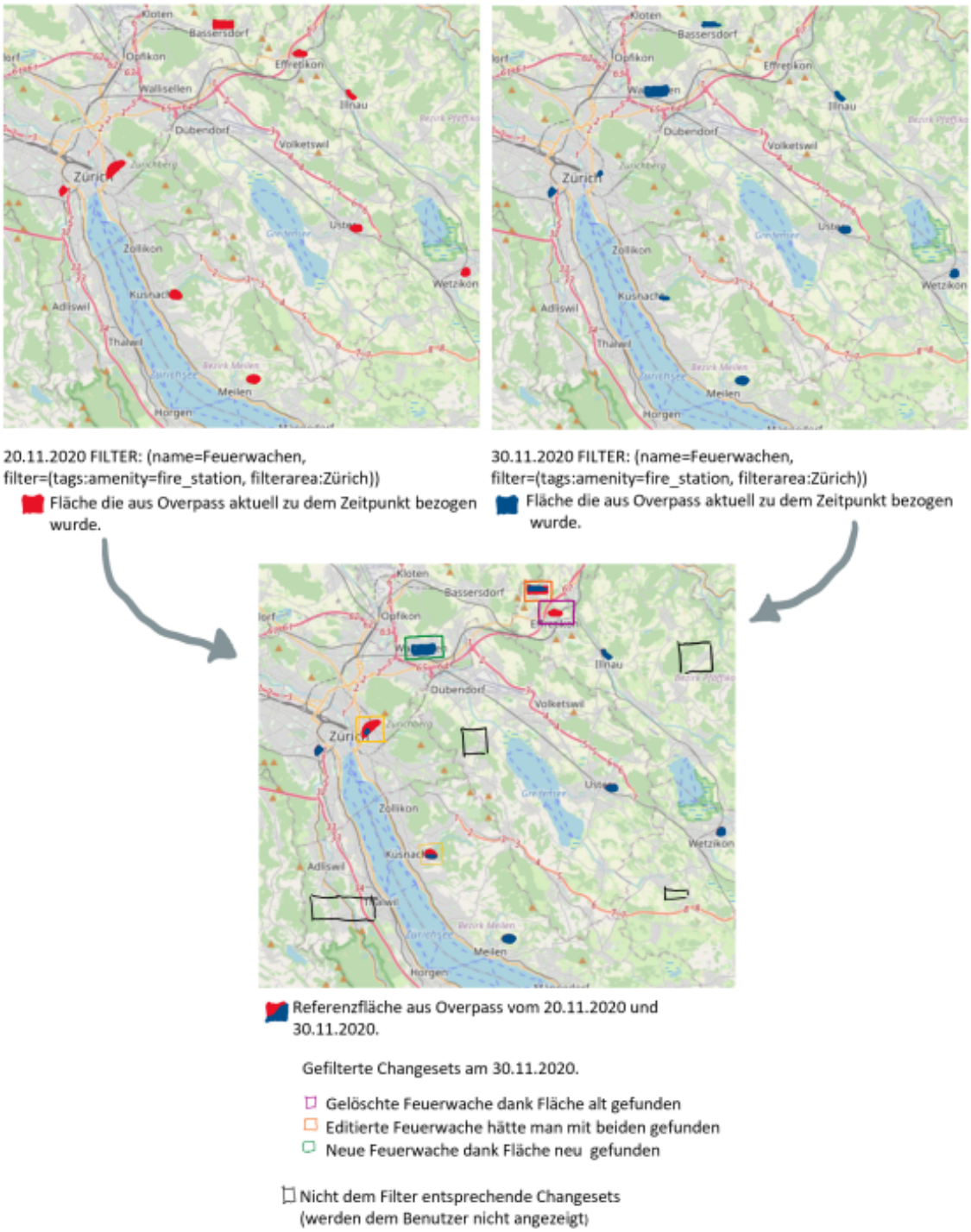


Abbildung I.3: Referenzfläche Veranschaulichung

Eine Overpass-Query mit dem Tag emergency=defibrillator und der Region Zürich und Bern sieht so aus:

```
1  [out:json];
2  area[name="Zürich"]->.searchZurich;
3  area[name="Bern"]->.searchBern;
4  (
5  nwr["emergency"="defibrillator"](area.searchZurich);
6  nwr["emergency"="defibrillator"](area.searchBern);
7  );
8  out body;
9  >;
10 out skel qt;
```

Abbildung I.4: Overpass-Query

Overpass liefert dann alle Objekte mit dem Tag emergency=defibrillator als JSON.

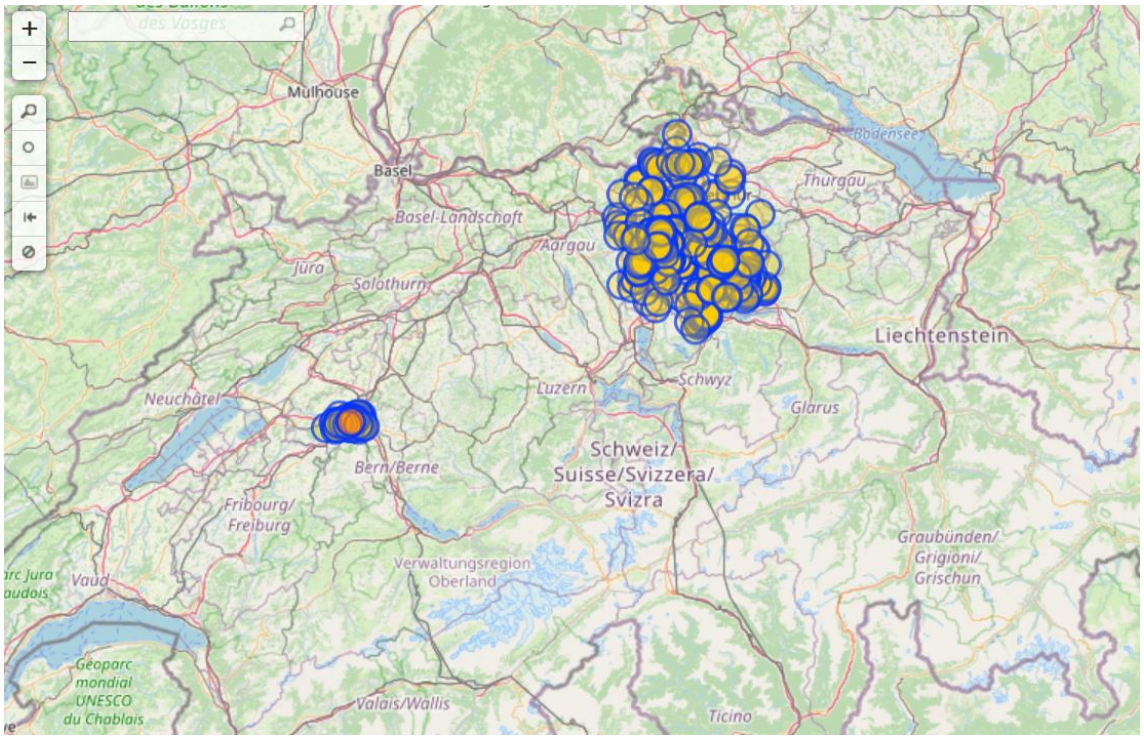


Abbildung I.5: Resultat einer Overpass-Abfrage

4.2 Status-Meldungen

Jedes Changeset, welches einem Filter zugewiesen wird, erhält den Status "neu". Sobald nun ein Benutzer mit der Arbeit beginnt, setzt er den Änderungssatz den er bearbeiten möchte auf "in Bearbeitung". Dies sendet eine PATCH Nachricht ans Backend und die Änderung wird im Filter gespeichert. Leider hat die Zeit nicht mehr gereicht, um

eine Variante mit Server Push zu implementieren. Deshalb wurde ein Update Button im Frontend platziert. Mit diesem kann man die bei sich angezeigten Status-Meldungen mit denen im Backend abgleichen. Dem entsprechend aktualisiert sich die Anzeige der Status. Dieser Lösungsansatz reicht unserem Kunden, da sie nur ein kleines Team von wenigen Datenbearbeitern sind und sich somit kaum gegenseitig in die Quere kommen werden.

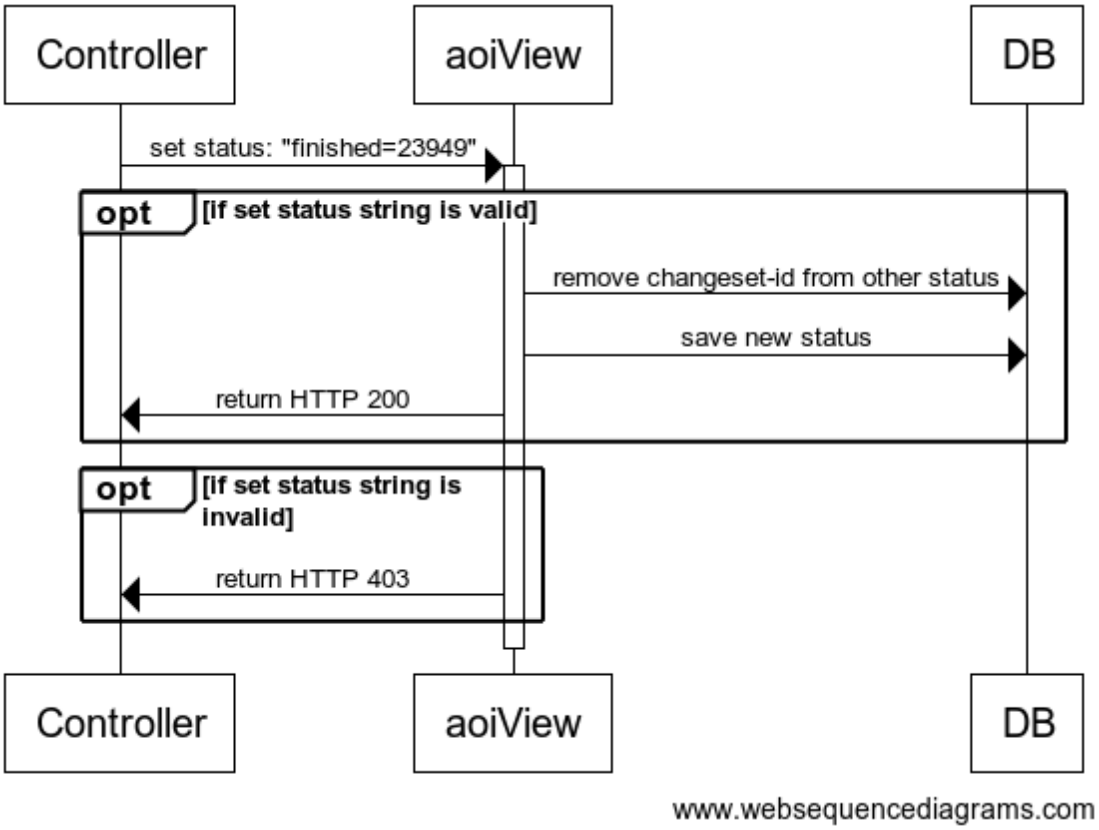


Abbildung I.6: Sequenzdiagramm Status setzen

5 Schlussfolgerung

5.1 Zielerreichung

Unser Kunde war bei der Abnahme am 11.12.2020 sehr begeistert vom Targeted Monitoring Tool. SRZ hat nun ein einsatzfähiges Tool, welches sie uns mit Freude abgenommen haben. Sie werden es nun ausgiebig testen, aber der Funktionsumfang scheint für sie erreicht. Sie müssen noch bis im nächsten Jahr auf die Auslieferung warten, da dies den Rahmen dieser Arbeit gesprengt hätte. Das gesamte SRZ-Team hat Zugriff auf die Instanz, die auf unserem OST-Server läuft und nutzt das Targeted Monitoring Tool

bereits.

Es wurde keine klassische Rollenverteilung im System hinterlegt, sondern jeder kann Administrator und Datenkurator sein. Jeder kann Filter erstellen und anderen freigeben. Der Filter-Ersteller ist automatisch Administrator des Filters. Die einem Filter hinzugefügten Benutzer sind die Datenkuratoren des Filters. Den Filter bearbeiten und löschen kann nur der Administrator, der Datenkurator kann ihn lediglich ansehen.

In unserer Endversion kann nach Changesets gesucht werden, die über einem bestimmten Objekt liegen, welches das gewünschte Key-Value-Tag besitzt. Diese Changesets werden in einem aufgeräumten Dashboard aufgelistet. Zusätzlich verfügt das Dashboard über eine Karte, in welcher das ausgewählte Changeset ersichtlich ist. Um die Gefahr einer Änderung schneller abschätzen zu können, werden noch einige Informationen des Changesets dargestellt (Beschreibung, Benutzernamen des Changeset-Erstellers und Datum). Als kleines Extra des Changeset Viewers, kann man sich noch den Karten-Hintergrund einstellen, oder Changeset Elemente ein- und ausblenden.

Man kann gut im Team arbeiten, ohne dass alle mit der Überprüfung des gleichen Changesets beschäftigt sind. Dazu kann man in der Liste der Changesets pro Filter den Status von jedem Changeset ändern, sobald man mit der Arbeit beginnt. Um zu erfahren, welche Datensätze gerade durch die Kollegen bearbeitet werden, braucht es lediglich einen Klick auf den Status-Update-Knopf und keinen Spaziergang ins Büro des Kollegen.

Mit einem Filter nach OSM-Benutzer kann man die Änderungen einer bestimmten Person überwachen. Somit können die neuen Praktikanten vor der harschen Kritik der Community geschützt werden. Der verantwortliche Filter-Administrator kann so nachsehen, ob und was für Fehler die Praktikanten beim Mappen noch machen.

Die Changesets der Datenkuratoren tauchen nicht im Dashboard auf, ausser sie wurden zusätzlich in die Liste der zu überwachenden Personen aufgenommen.

Da die Filtereingabe leicht zu Fehlern führen kann, gibt es eine passende Eingabeprüfung. Beim Hinzugefügen eines Tags in Form von Key=Value, überprüft das Tool via Tagfinder die Gültigkeit der Eingabe. Ist sie ungültig, wird der Tag nicht hinzugefügt. Somit wird verhindert, dass keine Referenzfläche berechnet werden kann. Auch auf die Pflichtfelder wird der Benutzer hingewiesen und ohne deren Eingabe kann der Filter nicht gespeichert werden.

Da die verwendeten Open-Source API's nicht immer verfügbar sind, besitzt das Programm über eine Fehler-Informationsseite, um einen Absturz zu verhindern.

Nicht eingebaut wurde eine Benachrichtigung per Mail. Dies wurde mit dem Kunden SRZ abgesprochen und ist vertretbar, da sie sich noch unsicher sind, ob dies einen bedeutenden Mehrwert generieren würde oder nicht. Sie möchten das Tool nun so testen und sich überlegen, ob sich eine spamartige Benachrichtigung überhaupt lohnen würde.

Da die Benachrichtigung an Wichtigkeit für unseren Kunden verloren hat, ist auch die automatische Prüfung 1x am Tag weggefallen. Zu Beginn der Arbeit war noch nicht ganz klar, ob die Changesets 1x täglich, oder ständig während dem Benutzen des Tools gefiltert werden. Zweiteres ist jetzt der Fall.

Wie in Abbildung X und X zu erkennen ist, erfolgt das Login über Oauth mit dem OSM-Benutzerkonto.

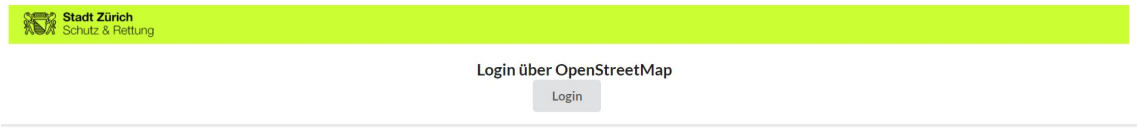


Abbildung I.7: Login Screen

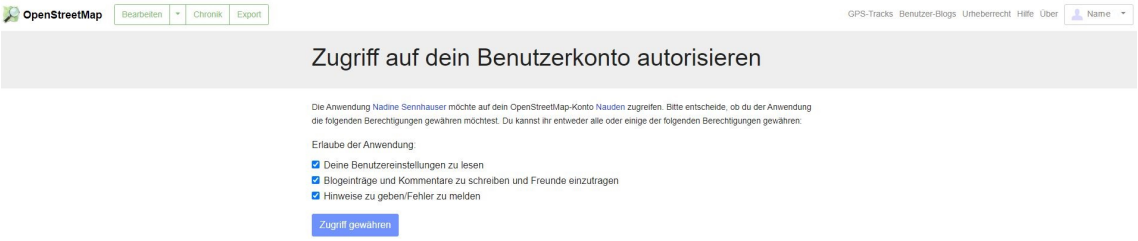


Abbildung I.8: Bestätigung der Berechtigungen

5.2 Weiterentwicklung

Zur Weiterentwicklung des Tools ist jeder willkommen und Pull-Requests können im Git-Repository von Schutz & Rettung platziert werden.

Die Anzeige des Bearbeitungsstatus bei den Changesets wird nicht automatisch aktualisiert. Momentan ist das Targeted Monitoring Tool auf das Arbeiten in kleinen Teams ausgelegt. Ob in Zukunft auch grössere Mapping-Teams Interesse an der gemeinsamen Überwachung basierend auf Objekt-Tags haben, ist zwar fraglich, aber nicht ausgeschlossen. Es wäre trotzdem elegant, wenn die Status sich automatisch aktualisieren würden. Für Arbeiten in grösseren Teams wäre es zusätzlich praktisch, wenn sich der Benutzer selbst aus der Filter-Freigabe anderer löschen könnte.

Mail-Benachrichtigungen bei neuen Changesets wären ebenfalls eine mögliche Erweiterung. Ein Benachrichtigungsmechanismus wäre nur dann sinnvoll, wenn dieser aus- und eingeschaltet werden kann, da bei gewissen Filtern täglich mehrere Changesets gefunden werden und das so in Spam ausarten würde.

Die Karte im Frontend könnte selber implementiert werden, um mehr Kontrolle darüber zu haben.

Die Filtrierung nach Objekt-Tag kann noch optimiert werden und statt mit der BBox zu filtern, könnten die Nodes der bearbeiteten Elemente im Changeset verwendet werden. Da wir aber leider ein fixe Zeitvorgabe hatten, war dies nicht möglich.

Ebenso könnte man das Tool so erweitern, dass man die Regionen nicht nur nach den vordefinierten Kantonen einschränken kann. Dazu müsste man aber eine Ortssuche und Validierung einbauen, damit die Overpass-Abfrage, die diesen Bereich als Einschränkung benötigt, keinen Fehler generiert.

Im Frontend kann noch am Design geschraubt werden, indem gerade angewählte Changesets besser markiert werden. Momentan wird zwar ein grüner Rahmen um das gerade angewählte Changeset (in der Liste) gelegt, dieser ist aber noch zu unscheinbar.

5.3 Wir sagen Danke

Wir danken besonders unseren Unterstützern.

Stefan Keller danken wir für die Betreuung und die vielen Tipps während der gesamten Studienarbeit.

Marcel Huber danken wir für seine Geduld mit welcher er uns beim Aufsetzen der GitLab-Pipeline, den Dockerfiles und den damit verbundenen Server-Firewall-Problemen geholfen hat. Seine Unterstützung haben wir sehr geschätzt und auch seine Anwesenheit beim Besuch bei Schutz & Rettung war eine sehr angenehme Unterstützung.

Raphael Das Gupta danken wir besonders für die Unterstützung mit Django und PostgreSQL, sowie für die vielen weiteren Antworten rund um die Studienarbeit.

Yael Schärer danken wir für die Hilfe beim Design des Browser-favicon.

Christan Nüssli danken wir für die Geduld und die angenehme zusammenarbeit.

Teil II

Projektdokumentation

Dieser Teil der Dokumentation beschäftigt sich mit den projektbezogenen Entscheidungen und Abläufen, sowie dem Projektmanagement. Die Projektdokumentation gibt einen Überblick über die Entstehung des in dieser Studienarbeit erarbeiteten Monitoring Tools.

1 Vision

Die Vision wird in Kapitel 1.2 beschrieben.

2 Anforderungsspezifikation

2.1 Requirements

Die folgenden Requirements wurden aus dem Auftrag erarbeitet und vom Kunden akzeptiert:

Nr	Anforderung	Gewichtung
1	Auffällige Datensätze sind im Dashboard ersichtlich.	Muss
2	Auffällige Datensätze sind nach Gefahr klassifiziert.	Soll
3	Auffällige Datensätze sind nach Klassifizierung sortierbar.	Soll
4	Auffällige Datensätze sind nach Klassifizierung filterbar.	Kann
5	Auffällige Datensätze sind nach Datum sortierbar.	Kann
6	Auffällige Datensätze enthalten einen Link ins OSM zum bearbeiten.	Muss
7	Auffällige Datensätze sind mit einem Status zu versehen.	Kann
8	Datenkuratoren und Administrator prüfen auffällige Datensätze.	Soll
9	Der Benutzer loggt sich mit seinem OSM-Konto ein.	Soll
10	Das Filterset wird vom Administrator bearbeitet.	Soll
11	Das Filterset basiert auf OSM-Tags.	Muss
12	Spezifische OSM-User sind überwachbar.	Kann
13	Das Design des Dashboards beinhaltet das SRZ Logo und ihre standartisierte Farbe.	Kann
14	Die auffälligen Datensätze werden von OSM und weiteren Open Source Tools bezogen.	Muss
15	Die neuen Änderungen aus OSM werden laufend, oder aber mindestens 1x täglich ins Dashboard übernommen, sofern sie dem Filterbereich entsprechen.	Muss
16	Die Änderungen der internen Datenkuratoren werden nicht ins Dashboard übernommen.	Soll
17	Das Tool wird lokal bei SRZ gehostet.	Muss
18	Das Tool informiert die Datenkuratoren über neu erschienene auffällige Datensätze.	Soll

Tabelle II.1: Requirements

2.2 Benutzer-Analyse

Die Benutzer-Gruppe wurde mit 2 Personas beschrieben. Wobei die Persona 1 die Auftraggeber darstellt und deren Bedürfnisse an erster Stelle stehen. Die Persona 2 sollte aber aufzeigen, dass das Projekt durchaus von weiteren Personen sinnvoll eingesetzt werden kann. Dadurch sollte auch begründet sein, warum das Projekt unbedingt Open-Source und dadurch allen zur Verfügung stehen sollte.

2.2.1 Persona 1

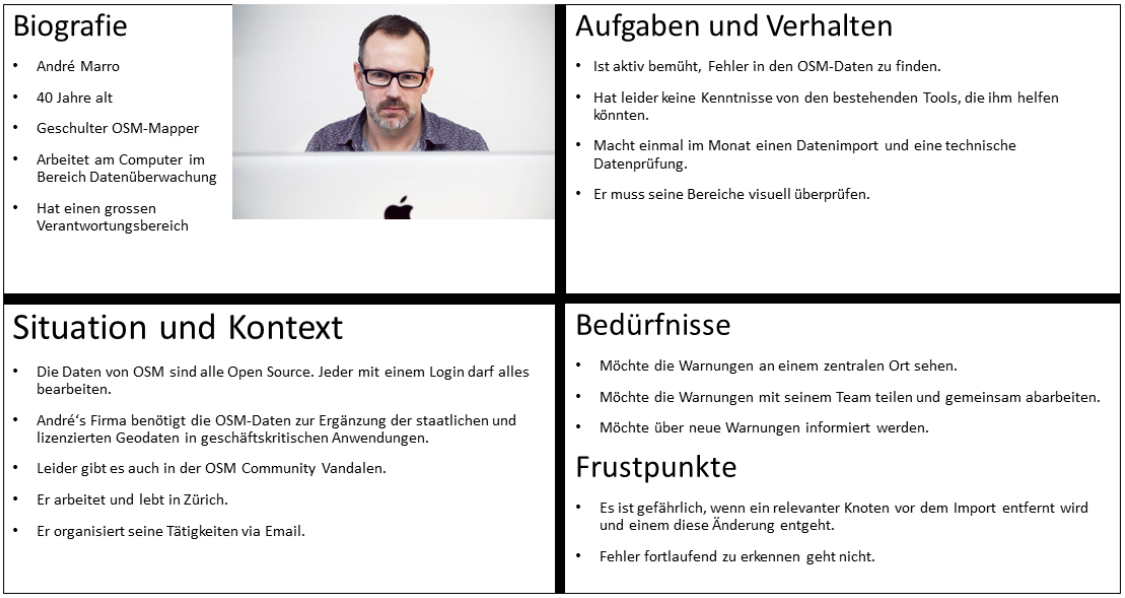


Abbildung II.1: Beschreibung von Persona 1

2.2.2 Persona 2

Abbildung II.2: Beschreibung von Persona 2

2.2.3 Ist-Customer Journey Map

Schritt 1: Vorbereiten

1. Katia, welche die Notrufe entgegennimmt und lokalisiert bemerkt, dass sie einen Suchbegriff nicht mehr findet und meldet dies dem Administrator André.
2. André muss wiederum, einen Datenkurator darauf ansetzen.
3. Eine automatische Prüfung der Dateninhalte wird noch keine durchgeführt.
4. Nur beim Import der Daten wird eine technische Prüfung der OSM-Daten gemacht. Die gemeldeten Probleme sind aber rein importtechnischer Natur.
5. Diese Fehler teilt André den Datenkuratoren zu.

Schritt 2: Fehler finden

1. Die Fehler aus dem monatlichen Import werden sehr rasch gefunden.
2. Die Meldung von Katia ist zum Glück geografisch in einem Bereich eingegrenzt, weil sie vom letzten Mal noch ungefähr wusste, wo es war.
3. Um aber weitere Fehlbearbeitungen an den Flughafen-Toren zu bemerken, muss André den Flughafen und seine Attribute visuell nach Fehlern absuchen.

Schritt 3: Fehler korrigieren

1. Die Fehler bearbeitet der neue Datenkurator Marc im OSM Web-Editor oder im JOSM.

Schritt 4: Abschliessen

1. Die neuen Datenkuratoren, wie Marc einer ist, werden von der Community meist sehr direkt auf Fehler hingewiesen, sodass sie schnell die Motivation verlieren.

2.2.4 Soll-Customer Journey Map

Schritt 1: Vorbereiten

1. André und seine Datenkuratoren müssen sich in das Tool einarbeiten.
2. Als Administrator muss André den Filterbereich festlegen.
3. Die Datenkuratoren werden über auffällige Datensätze im Bereich informiert.

Schritt 2: Fehler finden

1. Der neue Datenkurator Marc öffnet nach erhaltener Mail das Dashboard und sieht die Änderungen.
2. Er sieht, dass sein Kollege den Obersten bereits bearbeitet und setzt den Nächsten auf "in Bearbeitung".
3. Nun entscheidet er, ob sein Eingreifen nötig ist, oder ob die Bearbeitung korrekt ist.
4. Kann er sie ignorieren, wird sie für alle aus dem Dashboard entfernt.

Schritt 3: Fehler korrigieren

1. Bemerkt er einen Fehler, der seinen Eingriff verlangt, öffnet er den Link ins OSM und bearbeitet ihn dort.

Schritt 4: Abschliessen

1. Ist der Fehler behoben, kann er den auffälligen Datensatz auf beendet setzen.
2. Der Administrator André kann nun über den Filter nach Personen die Änderungen seines Mitarbeiters Marc ansehen.
3. Damit kann er ihm eine Rückmeldung geben, bevor die Community über ihn herfällt.

2.3 Aktoren

Aus der Benutzer-Analyse und dem Gespräch mit dem Auftraggeber, konnten zwei Systemrollen erkannt werden. Zum einen gibt es die **Administratoren**, welche die Filter definieren, um den Überwachungsbereich festzulegen. Andererseits gibt es die **Datenkuratoren**, welche die auffälligen Datensätze abarbeiten und informiert werden, wenn neue aufällige Datensätze im System erscheinen.

Actor	Beschreibung
Datenkurator	Ein Benutzer, welcher einem Filter hinzugefügt wurde.
Administrator	Ein Benutzer, der einen Filter erstellt hat.

Table II.2: Aktoren Beschreibung

2.4 Use Cases

Die Use Cases wie sie in Abbildung II.3 und im nachfolgenden Abschnitt dokumentiert sind, wurden zu Beginn erarbeitet.

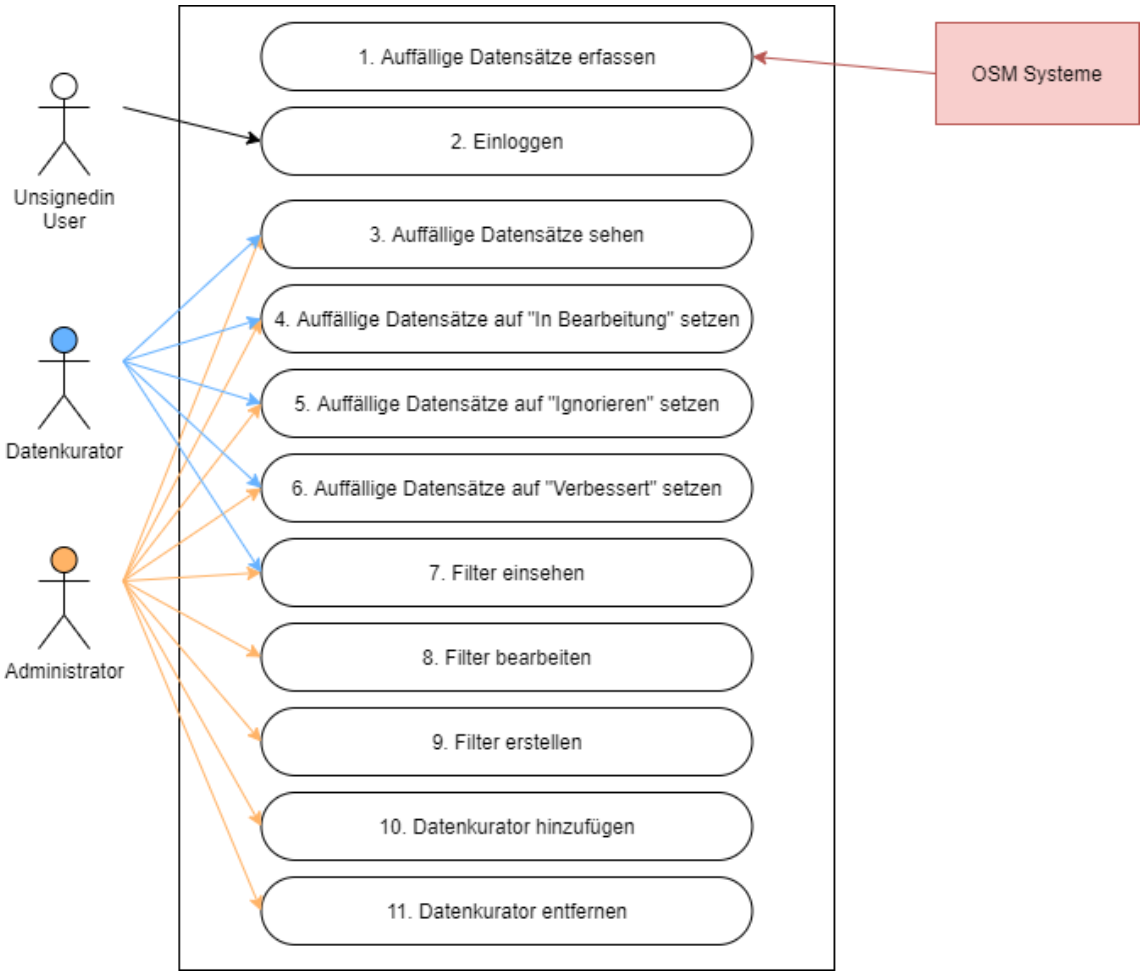


Abbildung II.3: UseCase-Diagramm

1. **Auffällige Datensätze erfassen** - Im OSM System erfassen OSM-User Daten, welche im Monitoring Tool erscheinen sollen.
2. **Einloggen** - Der unsignedin User loggt sich mit seinem OSM-Account im Tool ein.
3. **Auffällige Datensätze sehen** - Der Datenkurator sieht im Dashboard die Datensätze, welche das Tool anhand des Filters als auffällig gekennzeichnet hat.
4. **Auffällige Datensätze auf "In Bearbeitung" setzen** - Der Benutzer kann einen auffälligen Datensatz im Dashboard auf "In Bearbeitung" setzen, damit die anderen Benutzer sehen, dass dieser Datensatz gerade abgearbeitet wird.

5. **Auffällige Datensätze auf "Ignorieren" setzen** - Der Benutzer kann einen auffälligen Datensatz im Dashboard auf "Ignorieren" setzen, damit dieser nicht mehr im Tool auftaucht. Den anderen Benutzern wird dieser dann auch nicht mehr angezeigt.

6. **Auffällige Datensätze auf "Verbessert" setzen** - Wenn der Benutzer einen auffälligen Datensatz abgearbeitet hat, kann er diesen auf "Verbessert" setzen. Andere Benutzer sehen dann, dass dieser Datensatz korrigiert wurde.

7. **Filter einsehen** - Der Benutzer kann den ausgewählten Filter anschauen. Er sieht da Informationen, welche Tags oder Personen überwacht werden, ob ein Buffer um die Objekte gesetzt wurde, sowie den Namen des Filters und die hinzugefügten Datenkuratoren.

8. **Filter bearbeiten** - Der Administrator kann den Filter bearbeiten. Er kann den Namen des Filters ändern, Tags und zu überwachende Personen hinzufügen oder entfernen, den Buffer verändern oder entfernen und Datenkuratoren hinzufügen oder entfernen (Siehe Use Case 10 und 11).

9. **Filter erstellen** - Ein Benutzer kann einen neuen Filter erstellen und ist so automatisch Administrator des neu erstellten Filters. Dabei muss ein Filtername und mindestens ein zu überwachender Tag (mit oder ohne Buffer) angegeben werden.

10. **Datenkuratoren hinzufügen** - Der Administrator kann einem Filter über "Filter bearbeiten" oder direkt beim Erstellen einen oder mehrere Datenkuratoren hinzufügen. Diese können ihm bei Use Case 3-6 helfen.

11. **Datenkuratoren entfernen** - Der Administrator kann einen oder mehrere Datenkuratoren über "Filter bearbeiten" aus dem Filter entfernen.

2.5 User Stories

1. Als Administrator will ich ein tagbasiertes Filterset definieren, sodass die Datenkuratoren damit arbeiten können.

2. Als Administrator will ich im Filterset auch zu überwachende Personen festlegen, damit die Änderungen der Praktikanten überwacht werden können.

3. Als Datenkurator will ich die auffälligen Datensätze in einem Dashboard sehen, damit ich diese überprüfen kann.

4. Als Datenkurator will ich die Datensätze nach Klassifizierung sortieren, damit ich die kritischsten Datensätze zuerst anpacken kann.

5. Als Datenkurator will ich die Datensätze nach Datum sortieren, damit die älteren Datensätze nicht vernachlässigt werden.
6. Als Datenkurator will ich auf einen Link der ins OSM führt klicken, damit ich dort Änderungen vornehmen kann.
7. Als Datenkurator will ich bei der Beurteilung eines auffälligen Datensatzes genügend Informationen erhalten, damit ich schnell entscheiden kann, ob die Änderung gut oder schlecht war.
8. Als Datenkurator will ich den Status eines Datensatzes auf "in Bearbeitung" setzen, damit andere Datenkuratoren wissen, dass sich dieser Datensatz gerade in Bearbeitung befindet.
9. Als Datenkurator will ich den Status eines Datensatzes auf "abgeschlossen" setzen, damit allen klar ist, dass dieser abgearbeitet wurde.

2.6 Nicht-funktionale Anforderungen

Security:

- Der Zugang zur App ist per Authentifikation über OSM gesichert.
- Persönliche Daten der User können nicht von Unbefugten eingesehen werden.
- Nur der Administrator kann den Filterbereich festlegen.
- Benachrichtigungen enthalten keine sensitiven Daten.

Availability:

- Im laufenden Betrieb und bei ausreichender Netzwerkverbindung werden 95% aller Anfragen an das Backend fehlerfrei beantwortet.
- Bei einer Betriebszeit von 8760 Stunden pro Jahr ist das System 95% der Zeit verfügbar. Das entspricht 8322 Stunden Uptime im Jahr. (Nicht garantiert, dass die Datenlieferanten diese Betriebszeiten einhalten können)

Reliability:

- Hardware Fehler[800 000 FIT] + Software Fehler[800 000 FIT] + nicht verfügbare Daten von den Lieferanten[5 000 000 FIT] (FIT - Anzahl Fehler, die in 1x10^9 Stunden auftreten)

Fault Tolerance:

- Die Applikation darf nicht aufgrund einer instabilen Netzwerkverbindung zum Backend abstürzen.
- Wenn ein Datenlieferungstool ausfällt, darf nicht die ganze Applikation abstürzen.
- Wenn ein Datenlieferant längere Zeit ausfällt (nicht mehr zur Verfügung steht), ist es einfach möglich, eine Alternative einzubauen. Für einen Entwickler, der die Repo-Beschreibung gelesen hat und Erfahrung mit OSM Tools, sowie Python hat, ist es in 2 Arbeitstagen möglich, diese zu ersetzen.

Recoverability:

- Im Falle eines vollständigen Systemausfalls, kann das Backend innerhalb eines Personentages aus der Sicherung wiederhergestellt werden.
- Bei Schäden am Frontend kann dies ebenfalls in einem Tag wider aus dem Source-Code wiederhergestellt werden.
- Bei einer Wiederherstellung kann es vorkommen, dass die am gleichen Tag bearbeiteten Datensätze noch einmal im Monitoring Tool erscheinen.

Usability:

- Das Definieren eines Filters gelingt dem Administrator mit Hilfe des Handbuches in weniger als zehn Minuten.
- In Eingabefeldern können nur Daten im passenden Format eigegeben werden. Alle Eingaben werden validiert, sodass falsche Daten verhindert werden.
- Es werden verständliche Fehlermeldungen ausgegeben
- Auffällige Datensätze landen spätestens 24h nach Eintrag im OSM im Tool.
- Das Benutzen mit einer Sehschwäche ist möglich. (Optional)

Performance: Definition Hardware: Unser Backend läuft auf einem Rechner mit vier 2.2 Ghz Prozessorkernen, 8GB RAM und 30GB Festplattenspeicher, sowie einer Gigabit-Ethernet Netzwerkkarte.

Definition 1: Die Webseite ist in Betrieb. Es sind 5 User registriert (Es arbeiten nicht mehr als 5 Personen im SRZ als Datenpfleger). 5 User nutzen die Webseite zurzeit aktiv und sind eingeloggt. Die Netzwerkverbindung beträgt mindestens 50Mbps.

Definition 2: Die Webseite ist in Betrieb. Es sind 5 User registriert. Einer dieser User ist aktiv oder eingeloggt. Die Netzwerkverbindung beträgt mindestens 50Mbps.

- Das Login in der Applikation dauert unter Annahme der Definition 1 nicht länger als 5 Sekunden.
- Beim Öffnen des Dashboardes werden unter der Annahme der Definition 2 innerhalb von 10 Sekunden die ersten auffälligen Datensätze (oder eine Information "Zurzeit konnten keine auffälligen Datensätze in Ihrem Bereich gefunden werden.") angezeigt.
- Vom Absenden des Filtersets bis zur Anzeige im Home dauert es nicht länger als 10 Sekunden.

Resource Behavior:

- Die Webseite wird bis zu 20 registrierte User unterstützen.
- Die Webseite verwendet maximal 2GB RAM. (Browserverbrauch miteinbezogen)
- Der textbasierte Teil eines auffälligen Datensatzes wird 1MB nicht überschreiten.

Analysability:

- Bei einem Benutzererror wird dem User eine Meldung ausgegeben. Diese fordert ihn auf, die Aktion zu wiederholen.

Testability:

- Alle Usecases sind mindestens mit einem System-Test getestet.
- Alle Testfälle müssen erfolgreich sein, bevor Änderungen des Quellcodes in die stabile Software übernommen werden.

Installability:

- Der Source Code ist Open-Source und die Installation sollte mittels Docker genau beschrieben sein.

- Mit der Dokumentation kann jeder Softwareentwickler diesen in 1 Tag installieren und zum Laufen bringen.

Modifiability:

- Die Erweiteung um Fehler-Erkennungs-Tools ist einfach möglich. Für einen Developer der die Repo-Beschreibung gelesen hat und Erfahrung mit OSM Tools, sowie Python hat, ist es in 2 Arbeitstagen möglich diese zu erweitern.

3 Design

3.1 Architektur

Die Architekturbeschreibung wird anhand des C4-Architekturmodells veranschaulicht. C4 bietet 3 Levels, mit denen die Architektur von oben bis unten nachvollzogen werden kann.[5]

3.1.1 C4 Level 1

Die Abbildung II.4 zeigt eine Gesamtübersicht über die Architektur auf dem Level 1 des C4-Architekturmodell und bildet ab, in welchem Kontext sich das Targeted Monitoring Tool befindet. Darin ersichtlich sind die Systeme und User mit welchen es kommuniziert.[5]

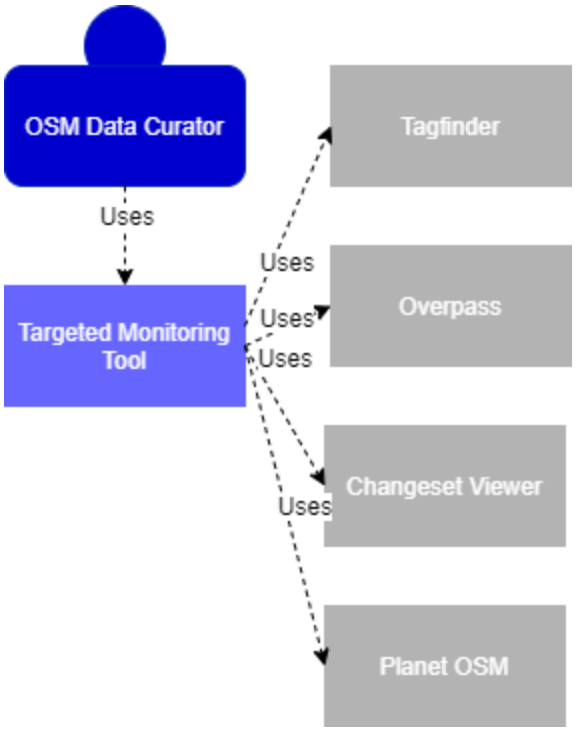


Figure II.4: Architektur im C4 Modell Level 1

3.1.2 C4 Level 2

Wird beim C4-Modell auf die nächste Ebene (Level 2) gezoomt, entsteht das Container-Diagramm. Hier werden die Teile des Systems gezeigt, die seperat voneinander laufen. In der Grafik II.5 werden diese abgebildet.[5]

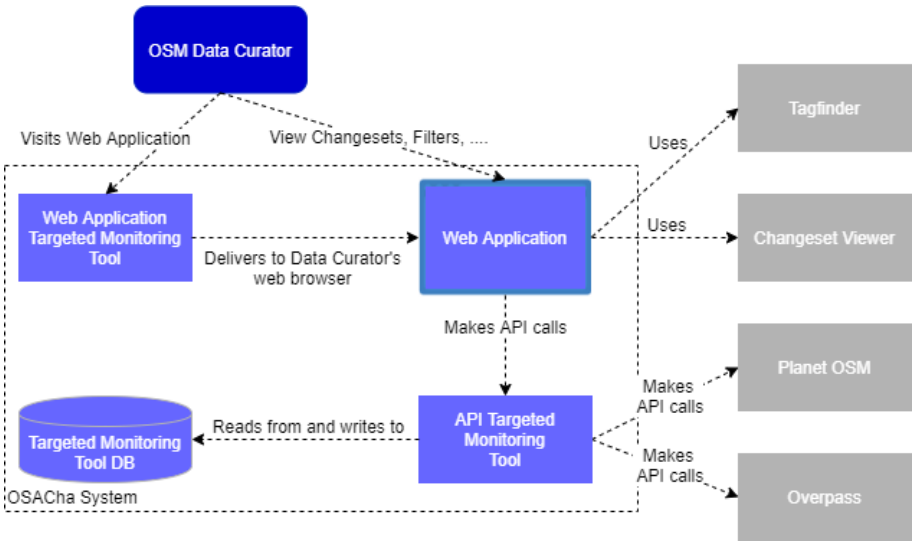


Figure II.5: Architektur im C4 Modell Level 2

3.1.3 C4 Level 3

Frontend

Das Frontend ist eine Eigenentwicklung mit React und Redux. Die Abbildung X zeigt die wichtigsten Komponenten des Frontends, um einen Überblick zu bieten. Das Frontend baut auf 3 Pages auf, die je mehrere Components enthalten. Der interne State wird dabei von einem Redux Store verwaltet und kann nur über einen Actions-Dispatch verändert werden. Dafür kann aus allen Komponenten auf den State zugegriffen werden.

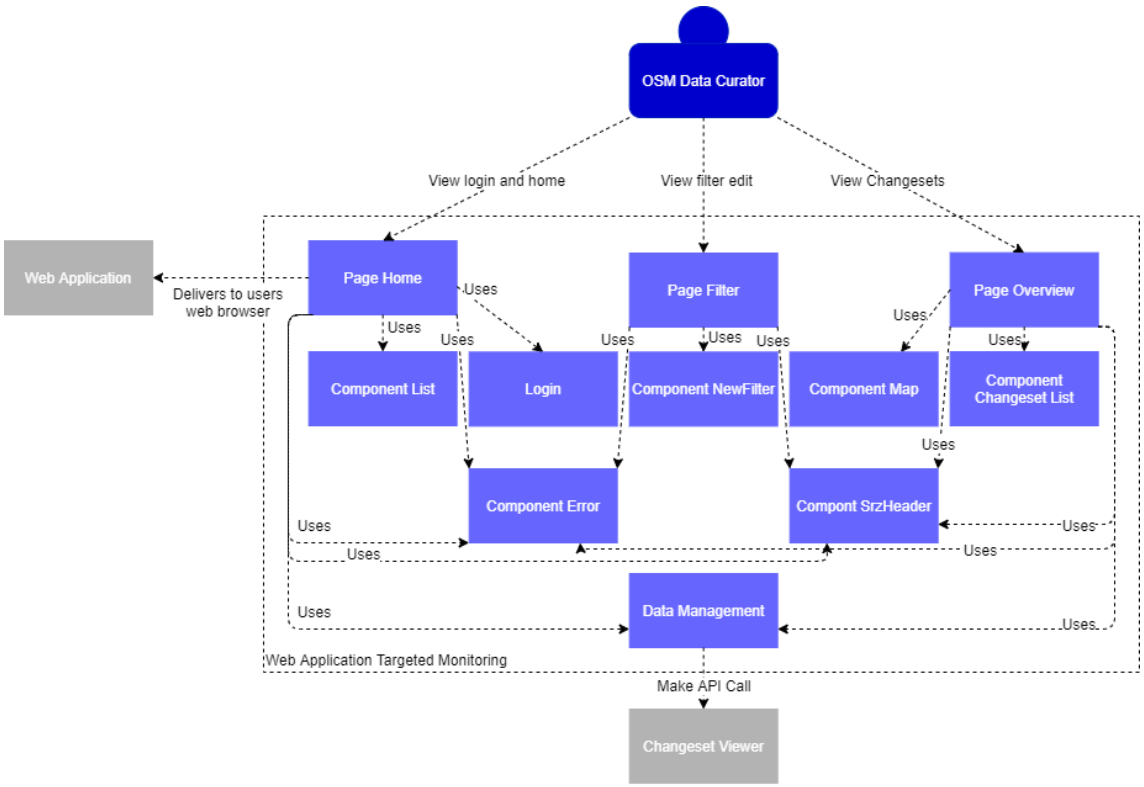


Figure II.6: Frontend Architektur im C4 Modell Level 3

Backend

Das Backend baut auf dem Open-Source Tool OSMCha-Django Version 4.9.2 auf. Es basiert auf dem Web-Framework Django und ist dem entsprechend in Python geschrieben. Abbildung X zeigt die wichtigsten Komponenten des Backend.

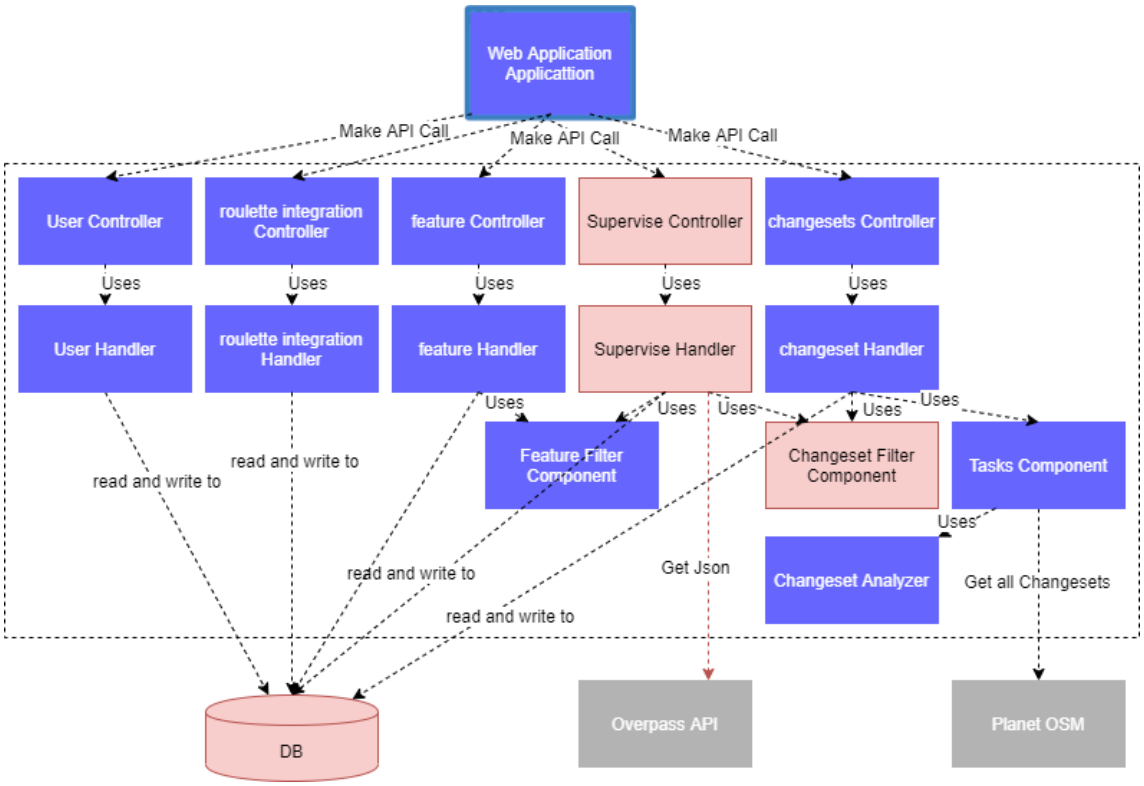


Figure II.7: Backend Architektur im C4 Modell Level 3

Da es sich um keine komplette Eigenentwicklung handelt, wurden in der Abbildung II.X die von uns veränderten Komponenten rot eingefärbt. Die beiden Supervise-Komponenten enthalten die neue Filter-Möglichkeit nach Objekt-Tags. Deshalb wurde im Supervise Handler die Overpass-API eingebunden. Das tatsächliche Filtern der Changesets findet in der Changeset Filter Component statt, welche ebenfalls angepasst werden musste. Neue API Abfragen für das Setzen und Anfordern der Status, sowie dem Holen von freigegebenen Filtern, wurden im Supervise Controller definiert. Da pro Filter eine Referenzfläche und die Status pro Changeset gespeichert werden müssen, wurden Änderungen an der Datenbank vorgenommen.

Kommunikation zwischen Frontend und Backend

Frontend und Backend kommunizieren über eine RESTFul HTTP Schnittstelle des Level 2 (HTTP Verbs). Das Frontend startet dabei jeweils asynchrone Anfragen, um den Benutzer nicht zu blockieren.

Die API wurde mit Swagger dokumentiert. Trotz vielen Bemühungen kann die API-Dokumentation nur über ein JSON, anstatt über das vorgesehene Swagger-UI angesehen werden.

3.2 Wichtige Abläufe

Filter definieren
Beim Abfüllen des Formulars gibt es Hilfestellungen. Explizit bei der Eingabe von Key-Value-Paaren für das Filtern nach Tag. Die Eingaben werden mittels einer Anfrage an Tagfinder sofort geprüft. Die Daten werden vom Benutzer via Formular an das Backend gesendet und in der DB abgelegt. Die Tags werden aus dem dafür angelegten Django json-Field als als Liste von Key=Value Strings in die DB gespeichert.

Filtern der Datensätze
Problem: Gelöschte Elemente und neue Elemente müssen in den Filterbereich einfliessen. Mit Hilfe der Overpass-API wird ein Geojson aus den Key-Values und dem festegelegten Bereich gewonnen. Wir benötigen dabei nur die Geometrien und keine weiteren Attribute. Die Filterfläche der letzten Anfrage wird aus der Datenbank gholt, wo sie nach der letzten Berechnung abgelegt wurde. Damit können wir die Anfragemenge an Overpass halbieren, müssen dafür etwas Speicherplatz einbüssen. Wenn es gewünscht wird, wird von unserem Tool noch ein Buffer um jedes der Objekte aus beiden Dateien gerechnet. Die Vereinigung (Union) der beiden Filterflächen wird als Filterreferenzfläche verwendet. Somit wird das Problem beseitigt und sowohl Objekte, die seit der letzten Prüfung dazugekommen sind, aber auch diese die gelöscht wurden, werden gefunden. Das neu generierte Json von Overpass, ohne Vereinigung, überschreibt das vorherige Geojson in der Datenbank.

Siehe Kapitel I.4.

Datensätze beziehen
Die auffälligen Daten werden laufend von Planet.osm bezogen. Dabei werden nur Schweizer Changesets gespeichert. Auch diese Filtrierung wird über einen Referenzflächenverschnitt gemacht.

Daten klassieren
Die Daten werden nur zur Anzeige klassiert. Die Anzeige basiert auf den Changeset-informationen zur Anzahl neuen, bearbeiteten und gelöschten Nodes.
Hohes Risko = mindestens 1 gelöschtes Objekt
Mittleres Risiko = mindestens 1 Bearbeitetes Objekt
Geringes Risiko = nur neue Objekte

3.3 UI Design

Das erste UI Design auf Papier wurde vom Kunden für seine Einfacheit gelobt und befindet sich im Anhang 2. Während der Entwicklung wurde das Design mehrfach angepasst. Die Anpassungen wurde fortlaufend mit dem Kunden abgesprochen. Die finale Ansicht des Dashboards ist in der Abbildung II.8 und die finale Ansicht des Home-Bildschirms in Abbildung II.9 zu sehen. Weitere Screenshots werden im Anhang 3 abgebildet.

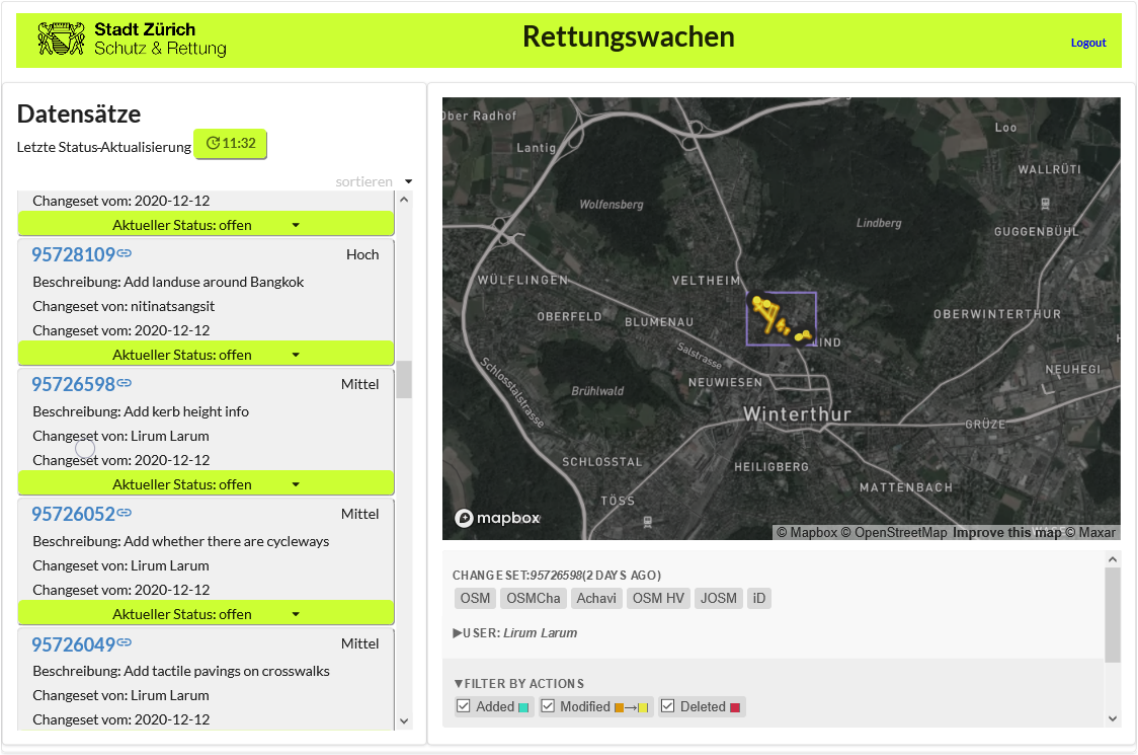


Figure II.8: Dashboard Ansicht der Endversion

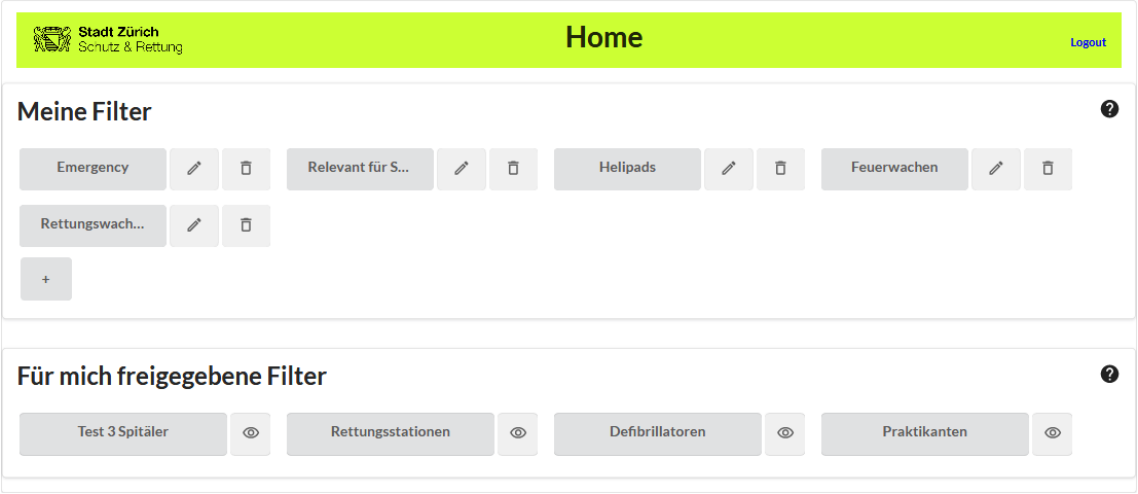


Figure II.9: Home Ansicht der Endversion

4 Testing

4.1 Automatische Testverfahren

Das Backend wurde mit automatisierten Tests getestet und hatte am Ende eine Test-Abdeckung von 96.5%. Leider konnte die Funktion mit der Overpass Abfrage in der gegebenen Zeit nicht sinnvoll Unit getestet werden.

Im Frontend wurden einige automatische Tests erarbeitet. Die API Abfragen wurden zu diesem Zweck gemockt.

Cod Abdeckung Frontend:

- Statements 38.48%
- Branch 45.77%
- Functions 33.65%
- Lines 38.82%

4.2 Manuelle Testverfahren

Jede Version der Software wurde einem manuellen System Test unterzogen. Die Systemtestprotokolle liegen im Anhang 4.

Mit dem Kunden wurden entgegen der Planung aus Zeitgründen nur 2 Usability Tests durchgeführt. Einer mit dem Design auf Papier zum Start und einer am Ende zur Abnahme am laufenden Produkt. Das Protokoll dazu befindet sich im Anhang 5.

Zudem wurde am Ende des Projekts noch die Performance über ein einfaches Pythonscript getestet. Da SRZ nur im kleinen Team arbeitet und ihnen die Ladezeiten nicht so wichtig sind, wurde hier entsprechend auch nur im kleinen Rahmen getestet. Das Protokoll befindet sich im Anhang 6.

5 Resultate und Weiterentwicklung

5.1 Resultat

Die Software ist nun einsatzbereit und wartet noch auf die Installierung auf den Servern von SRZ.

Wir konnten alle unsere Requirements der Stufe "muss" abdecken. Auch viele der weniger relevanten Requirements konnten in der gegebenen Zeit erarbeit werden, jedoch nicht der volle Funktionsumfang.

Es fehlt noch die aktive Benachrichtigung der User durch das Tool. Davon sind im Wesentlichen die Requirements 15 und 18 betroffen. Damit wir eine E-mail an alle Filter-Mitglieder senden können, müsste die Aktualisierung der gefilterten Changesets automatisch gestartet und anschliessend eine E-Mail an die Interessierten gesendet werden. Damit verbunden wäre aber auch die Implementation einer Möglichkeit, Spam zu vermeiden. Es wurde mit dem Kunden beschlossen, die E-Mail Funktion vorerst wegzulassen. Dadurch können die Changesets erst mit dem Starten der Filteransicht eingesehen werden.

Auch eine Filtrierung der Datensätze nach Status wurde, der Einfachheit des Frontendes geschuldet, ausgelassen. Wir haben festgestellt, dass das Filtern im Filter zu Verwirrung führte und haben diese Funktion bewusst für den Start weggelassen. Sollte sich aber herausstellen, dass die Sortierung alleine nicht ausreicht, wäre eine Changeset Status Filtrierung durchaus umsetzbar.

Die Non-Functional Requierements, welche anfangs definiert wurden, können so abgedeckt werden, dass es seitens SRZ akzeptiert wurde. Die Usability könnte noch im Sinne der Sehschwäche verbessert werden, denn die Markierung des gewählten Changesets ist nur schwach zu erkennen. Und bei der Installation in einem Tag ist wichtig, dass die festgelegten Voraussetzungen erfüllt sind.

Ein automatisches Backup der DB wird nicht ausgeführt, sollte aber auf Wunsch einfach zu implementieren sein.

Während des Projekts sind weitere Interessenten aufgetaucht. Deshalb wurde das Frontend auf eine Internationalisierung vorbereitet und kann einfach übersetzt werden.

Weiteres zur Lösung finden Sie im Teil I Kap. 5.

5.2 Weiters Vorgehen

Die Auslieferung des Systems an den Kunden ist noch ausstehend. Die Code Grundlage ist bereits im GIT-Repository von SRZ. Auch die Abnahme und alle Benutzer- und Installationsanleitungen wurden übermittelt. Die Auslieferung wird vermutlich das Institut für Software übernehmen.

Die weitere Unterhaltung ist noch nicht abschliessend bestimmt und ein Wartungsvertrag steht noch aus.

Das System ist aber bereits Open-Source und die in Teil I Kap. 5 beschriebenen Erweiterungen, können jederzeit per Pull-Request ergänzt werden.

6 Projektmanagement

6.1 Steakholder

Christian Nüssli als Vertreter von Schutz & Rettung
Stefan Keller und Raphael Das Gupta als Betreuer
Nadine als Hauptverantwortliche im Bereich Frontend
Denis Nauli als Hauptverantwortlicher im Bereich Backend

6.2 Zeitplan

Im Projekt wurde iterativ gearbeitet. Dabei wurde die Scrum+ Variante aus SE2 mit 4 Phasen verwendet.

Phase	Beschreibung	Von	Bis	Dauer [Wochen]
Inception	Projektidee finalisieren	Mo 14.09.	Mo 21.09.	1
Elaboration	Projektplanung und Organisation	Di 22.09.	Mo 12.10.	3
Construction	Konstruktion der Applikation	Di 13.10.	Sa 14.12.	9
Transition	Demo Endprodukt	So 15.12.	Fr 18.12.	1

Table II.3: Projekt Phasen

Die Iterationen sind in Tabelle II.4 ersichtlich.

Iteration	Dauer [Wochen]/ Enddatum	Beschreibung	Assessment
M1.1: Projektplanung	1/ 21.09.20	Projektidee erarbeiten und Projekt organisieren	Genehmigung der Projektidee
M1.2: Requirements	1/ 28.09.20	Produktanforderungen erarbeiten, Use Cases in Brief-Format	Requirements dokumentiert
M2: End of Elaboration	2/ 12.10.20	End of Elaboration Liste fertigstellen, Starten der Arbeitsaufteilung	Paperbased Usabilitytest
M3: Kommunikations Pipeline	2/ 26.10.20	Architektur Grundgerüst erarbeiten und Kommunikation ermöglichen	Systemtest des Architektur-Prototyps
M4: Alpha	2/ 09.11.20	Alpha-Version coden, Hosting Übergabe vorbereiten	System-Test und Usability-Test der Alpha Version
M5: Beta	2/ 23.11.20	Erstellen Beta Version	System-Test der Beta Version
M6: Endprodukt	2/ 07.12.20	Wepapp finalisieren, komplett testen	Webapp final
M7: Abgabe Endprodukt und Dokumentation	2/ 18.12.20	-	-

Table II.4: Projekt Iterationen

6.3 Risiken

Risiken bei der Projekt Initialisierung

1. Datenverlust:
   Menschliches Versagen, Hardware Fehler, etc
2. Nicht zuverlässige Datenlieferanten:
   Verwendete Tools für die Datengewinnung liefern keine oder falsche Daten
3. Schnittstellen zu anderen Tools:
   Beim Einbinden von den passenden Tools treten Probleme auf
4. Benachrichtigungen:
   Die Benachrichtigungen können nicht wie gewünscht implementiert werden.
5. RTT:
   Roundtriptime mit Anfrage, Algorithmus und Antwort ist zu langsam
6. Unbekannte Technologien:
   Das Team muss sich in unbekannte Technologien wie Docker, Python einarbeiten
7. Zeitplanung:
   Das Team wird mit der Software im vorgegebenen Zeitraum nicht fertig

Gewichteter Schaden: 1-4
Eintrittswahrscheinlichkeit: unwahrscheinlich, wahrscheinlich, sehr wahrscheinlich

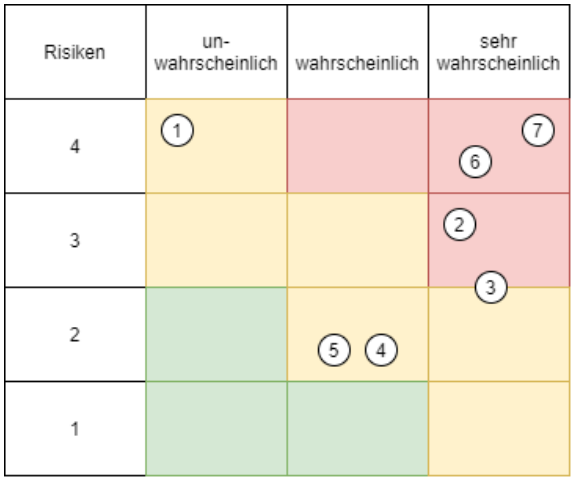


Figure II.10: Risiken zu Beginn des Projekts

Die folgenden Massnahmen wurden ergriffen um die Risiken zu reduzieren.

1. Datenverlust:
   aus GitLab Versionierung widerherstellbar

2. Nicht zuverlässige Datenlieferanten:
   Tool mit den besseren Verfügbarkeiten implementieren, wenn mehrere möglich sind; Fault Tolerance Möglichkeiten überlegt

3. Schnittstellen zu anderen Tools:
   Genaue Analyse der Tools und deren Schnittstellen-Dokumentation; Alternativen ermittelt

4. Benachrichtigungen:
   -

5. RTT:
   -

6. Unbekannte Technologien:
   Gute Einarbeitung; Hilfsbereite Spezialisten gefunden (Namen: Marcel Huber, Raphael Das Gupta), die bei Problemen helfen

7. Zeitplanung:
   Viel Zeit in eine gute Planung investiert; Requirements mit Gewichtung versehen, damit wir die wesentlichen Dinge sicher umsetzen können und gegebenenfalls nur Unwesentliches weglassen

Gewichteter Schaden: 1-4
Eintrittswahrscheinlichkeit: unwahrschinlich, warscheinlich, sehr wahrscheinlich

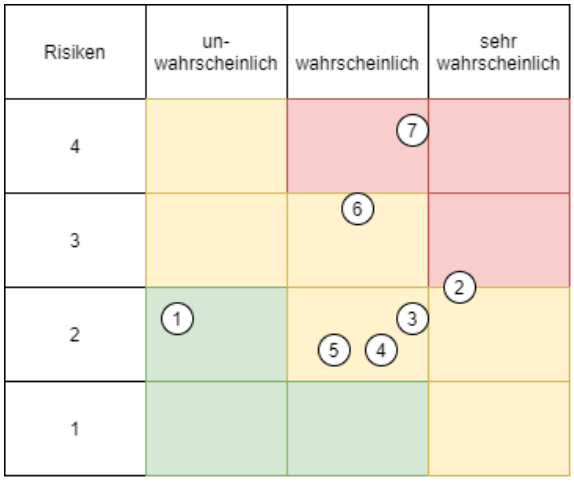


Figure II.11: Reduzierte Risiken nach End of Elaboration

7 Projektmonitoring

7.1 Ereignisse

7.1.1 HSR GitLab und Ost Server

Das erste Problem hatten wir mit dem GitLab (HSR-Netz), welches auf einen OST-Server zugreifen wollte. Deren Firewalls haben sich ständig gegenseitig ausgeschlossen. Dies wurde zum Glück schnell behoben, aber wir konnten die Funktionalität des GitLab mit seiner Docker-Regisry bis zum Ende nicht verwenden. Für weitere Projekte nehmen wir mit, dass man vor der Beginn der Arbeit seine Umgebung testet.

7.1.2 Design

Das Design musste vom ersten Entwurf, bis hin zur fertigen Seite in mehreren Durchläufen verbessert werden. Die Usability des ersten Entwurfes war nicht gegeben und auch viele Details wurden darauf nicht abgebildet. Für ein nächstes Projekt, wäre es sicher hilfreich, dieses detaillierter auszuarbeiten, damit weniger Anpassungen am Code nötig sind.

7.1.3 OSMCha

Beim ersten Versuch, OSMCha ohne Docker aufzusetzen, reichte die Anleitung von Mapbox nicht, um zum Ziel zu gelangen. Wir mussten erst die Grundlagen verstehen und erkennen, welche Schritte sie "einfachheitshalber" unterschlagen haben.

7.1.4 Docker

Die nächste Hürde war das Konfigurieren der Dockerfiles inklusive Docker-Compose. Dabei hat uns Marcel Huber sehr geholfen. Die Files, wie sie bereits von Mapbox zur Verfügung standen, konnte wir nicht 1:1 bei uns verwenden. Für uns Docker-Anfänger war es unmöglich ohne Hilfe alles zu verstehen.

7.1.5 DB Setup

Als endlich alle Docker-Container initialisiert waren, die Konfiguration für die Befüllung der DB korrekt war und alle Environment-Variablen passten, wurde die DB nicht wie erhofft erstellt. Da das Volume für die Datenbank schon in einem früheren Startversucht noch vor der eigentlichen DB-Initialisierung abgebrochen wurde, konnte die DB nicht mehr erstellt werden. Erst als Raphael Das Gupta herausfand, dass wir das unvollständige Volume löschen müssen, konnte das Problem behoben und die Datenbank endlich initialisiert werden.

7.1.6 Auslieferung

Zwei Wochen verspätet wollten wir die erste Version auf dem Server der SRZ laufen lassen. Aber nach etlichen Versuchen wurde das Vorhaben abgebrochen, da die Firewall der SRZ nicht passend vorbereitet war.

7.1.7 Swagger

Das letzte zeitfressende Problem konnten wir auch mit Hilfe nicht lösen. Im Backend dokumentiert Swagger die API. Probleme mit der Docker-Konfiguration und den Static-Files verhinderten es jedoch, dass das UI von Swagger angezeigt wird. Ohne UI, mit /swagger.json, kann die API Dokumentation eingesehen werden.

7.2 Soll-Ist-Zeit-Vergleich

7.2.1 Planung

Leider hat uns hier das Risiko Nummer 7 (Unbekannte Technologien) sehr stark getroffen. Trotz intensiver Einarbeitung zu Beginn der Arbeit konnte dies nicht verhindert werden. Wir konnten mit dem Programmieren erst in der Iteration Beta beginnen, da das Einarbeiten und Aufsetzen des doch sehr umfangreichen OSMCha viel Energie und Zeit in Anspruch nahm. Für ein nächstes Projekt müssen wir dies besser abschätzen. In den Bereichen Django, Python, Docker und React konnte viel Erfahrung gesammelt werden, sodass ein weiteres Projekt mit ähnlichen Technologien bestimmt besser laufen würde.

7.2.2 Zeit schätzen

Entsprechend der nicht eingehaltenen Planung war unsere Zeitschätzung auf den Issues überhaupt nicht genau. In der folgenden Abbildung II.12 wird dies mit dem Issue "Auslieferung Betaversion" aufgezeigt. Es wurden 6h geplant, nach 22h wurde das Vorhaben aus Zeitgründen jedoch aufgegeben und auf nach der Arbeit verschoben.

Abbildung II.12: Issue mit Zeitschätzung

7.2.3 Aufwand

Resultierend aus den Hindernissen ist auch die Gesamt-Zeit etwas über dem Budget. Wir waren Insgesamt 505 Stunden für das Projekt tätig. Die Abbildung II.13 zeigt sehr schön, in welchen Phasen die Probleme auftraten.

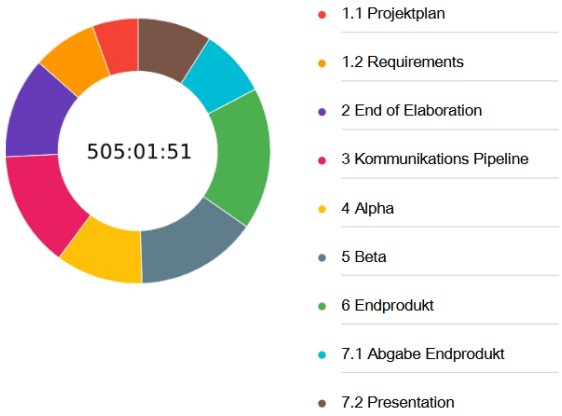


Abbildung II.13: Zeit nach Iteration

Die nächste Abbildung II.14 veranschaulicht, dass wir viel Zeit beim Setup, Dokumentieren, Analysieren und Evaluieren gebraucht haben (Im Bild alles unter Documentation zusammengefasst.) und der Anteil des Codings nur etwas höher als die Meetingzeit ist.

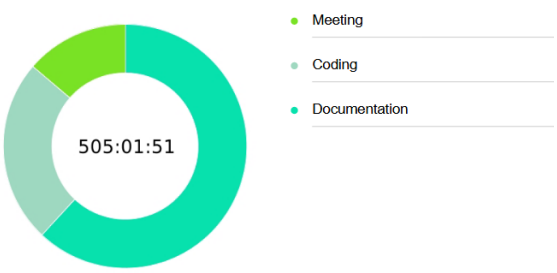


Abbildung II.14: Coding Anteil

Da die vorgängigen Ansichten kein Aufschluss über die Verteilung in der Zeit geben, ist das in der Abbildung II.15 sichtbar.

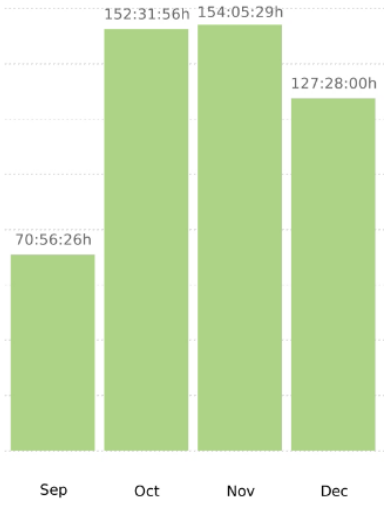


Abbildung II.15: Zeit pro Monat

7.3 Codestatistik ohne Tests

Die Zahlen des Backends:

Code Zeilen Backend gesamt: 6782
Davon HTML: 1081
Code Zeilen von uns bearbeitet ca: 220
Davon HTML von uns bearbeitet: keine
Anzahl Klassen gesamt: 256
Anzahl Klassen von uns: keine komplett
Anzahl Kommentare: 554
Anzahl Kommentare von uns: 1
Zudem wurden einige Confic Dateien angepasst und 2 Migration Files generiert.

Die Zahlen im Frontend:

Code Zeilen: 2154
Davon CSS Zeilen: 261
Anzahl Pages: 3
Anzahl Components: 11
Anzahl Kommentare: 25
Anzahl Funktionen: 183

7.4 Fazit

Wir haben uns enorm in das Projekt hinein gearbeitet und würden das Projekt gerne weiter pflegen. Wir hatten viele Rückschläge, aber sind mit dem Resultat dennoch zufrieden. Wir freuen uns sehr, dass es von SRZ aktiv verwendet wird. Ideen zur Verbesserung haben wir noch viele und werden uns um den möglichst langen Erhalt des Programms bemühen.

8 Softwaredokumentation

8.1 Installationsanleitung

Die Installationsanleitung wurde direkt im Repository von SRZ abgelegt und ist im Anhang 7 zu finden.

8.2 Benutzerhandbuch

Das Benutzerhandbuch wurde direkt im Repository von SRZ abgelegt und ist im Anhang 8 zu finden.

Anhang

1 Evaluation

Tools

(Beachten zur Einbettung in die Dokumentation: Teile der Beschreibungen und Relevanz wurden aus About der jeweiligen Seite kopiert!)

Allgemeine Tools

IFTTT -!-

Beschreibung: Hilfstools wie IFTTT können nützlich sein, um (möglicherweise eine Teilmenge von) z.B. QA-RSS-Feed-Einträge in eine E-Mail-Benachrichtigung umzuwandeln.

Relevanz für unsere Arbeit: Vielleicht bezüglich Benachrichtigung...

OSMyBiz:

https://osmybiz.osm.ch/#/19/47.22749/8.81816

Beschreibung: Zur einfachen Bearbeitung von Attributen für Geschäftsbesitzer. Es ermöglicht Laien mit wenig Zeit zum Beispiel ihre Öffnungszeiten zu pflegen. Die Informationen werden als Hinweise festgehalten, welche durch Andere Mapper im OSM erfasst werden können.

Relevanz für unsere Arbeit: Keine (Besseres Verständnis dafür, was OSM-Tools bieten)

GeoJSON: -!-

https://geojson.io/#map=13/47.2848/8.9962

Beschreibung: Schnell und einfach Webkarten erstellen, verändern und publizieren. Das Tool ermöglicht es GeoJasons einzulesen und diese zu bearbeiten, oder neu zu erstellen. Diese Eigenkreationen können gespeichert werden und in der eigenen Webseite eingebettet werden.

Relevanz für unsere Arbeit: Falls man Filter nach Bereich mit einer grafischen Oberfläche implementieren möchte, könnte dies ein Ansatz sein. Man könnte den definierten Bereich als GeoJason speichern und eventuell jeweils für die Überwachung verwenden.

overpass turbo -!!-

https://osm.li/OlG

Beschreibung: Overpass turbo ist ein online Daten-Filterungs-Werkzeug für OpenStreetMap. Mit overpass turbo kannst du Overpass-API-Abfragen ausführen und die gewonnenen OSM Daten interaktiv auf einer Karte analysieren. Man kann auch time-based Queries erstellen. (showing amenities that existed on a given date: http://overpass-turbo.eu/s/fsj showing items that were edited during a period of time: http://overpass-turbo.eu/s/fso)

Relevanz für unsere Arbeit: Die overpass API scheint eine Möglichkeit zu bieten, die Daten von OSM mit gewünschten Filterkriterien zu erhalten. Die Overpass API erlaubt es OSM-Daten nach selbst definierten Kriterien zu durchsuchen. Dies ermöglicht eine eigens dafür geschaffene Abfragesprache.

OSM TagFinder -!!-

https://tagfinder.herokuapp.com/

Beschreibung: Mit der Website lässt sich schnell und unkompliziert einen Begriff nachschlagen und die gefundenen Tags werden übersichtlich aufgelistet. Auf den ersten Blick werden die relevantesten Informationen über einen Tag dargestellt und wenn man weitere Details will, gelangt man direkt zur Wiki-Seite und den Statistiken von TagInfo.org.

Relevanz für unsere Arbeit: Auf die TagFinder Daten kann per REST API zugegriffen werden. Dies kann uns helfen, um für die Filter passende Tag-Vorschläge zu generieren. Siehe API Dokumentation für mehr Details.

Taginfo Schweiz -!!-

- https://taginfo.openstreetmap.ch/tags/amenity=toilets#combinations

Beschreibung: OpenStreetMap benutzt Tags, um geografischen Objekten eine Bedeutung zuzuorden. Es gibt keine feste Liste dieser Tags. Wenn nötig, können neue Tags jederzeit erfunden und benutzt werden. Taginfo hilft bei diesen und weiteren Fragen, indem es Statistiken zeigt, welche Tags wirklich in der Datenbank vorhanden sind, wie viele Mapper diese Tags benutzt haben, wo sie benutzt werden usw. Taginfo holt sich auch Informationen zu den Tags aus dem Wiki und aus anderen Quellen und bringt all dies zusammen, um dir dabei zu helfen, zu verstehen, wie Tags benutzt werden und was sie bedeuten.

Relevanz für unsere Arbeit: Könnte interessant sein, die API einzubinden um Filterung nach Tags zu verbessern. Dabei könnten aus Eingaben weitere eventuell interessante Tags vorgeschlagen werden, oder die Anzahl Verwendungen mitgeteilt werden. Siehe API Dokumentation für mehr Details.

Taginfo -!-

https://taginfo.openstreetmap.org/tags/amenity=toilets

Beschreibung: Siehe oben. Unterschied: Die Daten beziehen sich auf OSM weltweit.

Relevanz für unsere Arbeit: Könnte interessant sein die API einzubinden um Filterung nach Tag zu verbessern. Dabei könnten aus Eingaben weitere eventuell interessante Tags vorgeschlagen werden, oder die Anzahl Verwendungen mitgeteilt werden. Siehe API Dokumentation für mehr Details.

Bug reporting Tools

Notes -(!!)-

https://wiki.openstreetmap.org/wiki/Notes

Beschreibung: Notes ist eine Kernfunktionalität von OSM. Diese sind dafür vorgesehen, Fehler in der Karte/den Attributen zu kennzeichnen. Dadurch kann ein Mapper sich das Problem ansehen und es gegebenenfalls korrigieren. Nach der Korrektur sollte die Nachricht als "Bearbeitet" gekennzeichnet werden.

Relevanz für unsere Arbeit: Wenn ich ein Objekt überwache, möchte ich eventuell auch die Information erhalten, wenn jemand einen Verbesserungsvorschlag für dieses Objekt hat. Vieleicht kann man mit einem Koordinaten-Buffer kontrollieren, ob es Notes hat und diese ins Monitoring aufnehmen. API

MapDust -(!)-

http://www.mapdust.com/

Beschreibung: Da können auch Fehler gemeldet werden, die mit OSM zusammenhängen und anschliessend durch einen Mapper bearbeitet werden können. Die Fehler werden hier kategorisiert erfasst.

Relevanz für unsere Arbeit: Auch hier kann man es sich überlegen die gemeldeten Fehler mittels Buffer zu beobachten.

Error detection Tools

Keep Right

https://www.keepright.at/

Beschreibung: Das Tool vermerkt sehr viele potenziell technische Fehler. Da es aber sehr viele Freiheiten bei OSM gibt, ist nicht jeder dieser Fehler auch wirklich falsch. Fehler Beispiele: Ungeschlossene Flächen, fixme items, ...

Relevanz für unsere Arbeit: Es gäbe ein API, aber: "This project is no longer developed, with last activity on February 2017[1] but lists of detected errors are getting updated as of January 2019." Deshalb glaube ich nicht, dass wir da Daten anziehen werden. Ob solche Meldungen relevant sind, ist mit dem Auftaggeber abzuklären.

Osmose -(!!)-

http://osmose.openstreetmap.fr/de/map/#zoom=14&lat=47.45017&lon=8.55337&item=xxxx&level=1&tags=&fixable=

Beschreibung: Ähnlich wie oben. Es werden technische Fehler gefunden (Strassenteile, die nicht miteinander verknüpft sind -> Problematik beim Routenberechnen). Kartenansichten um die Qualität zu verbessern. Speziell aufbereitete Layer, die Probleme aufzeigen können. Teils nur in Frankreich und teils weltweit.

Relevanz für unsere Arbeit: Mit dem Auftraggeber klären, ob er solche Meldungen in der Nähe seiner Beobachtungsgeometrien erhalten möchte. Vorteil: Man kann eigene Checks schreiben und es gibt eine API, sowie Source Code vom Frontend. (Auch als CRM Schritt möglich) Da es gute Ansichten sind, wäre ein Link vom Fehler zu dieser Ansicht zur Verständlichkeit des Fehler denkbar.

JOSM/Validator

https://josm.openstreetmap.de/

Beschreibung: Geometrie Error detector

Relevanz für unsere Arbeit: Eher nicht Monitoring Tool, eher für Vandalismus Fehler und nicht unerkennbare Geometrie Fehler. (als CRM Schritt)

Gray Tools

Beschreibung: Gray68 hat sehr viele Tools entwickelt. Seine Tools sind in Pearl geschrieben und er ist leider nicht mehr aktiv in der Community dabei. Tools wie Osmdiv, waycheck, todo, ...

Relevanz für unsere Arbeit: Es hätte interessante Tools darunter, aber da sie nicht mehr laufen und in Pearl geschrieben sind, denke ich nicht, dass wir davon etwas einbauen.

OSM Inspector -!-

https://tools.geofabrik.de/osmi/

Beschreibung: Wurde erstellt um verschiedene Ansichten für die Fehlerbehebung zu erhalten. Eine View betreffend Adressen könnte eventuell interessant sein.

Relevanz für unsere Arbeit: Vielleicht können wir bei Adressfehleren einen Link auf diese Seite einbauen, um das Problem besser zu erkennen.

TIGER Edited Map

Beschreibung: Zeigt Flächen die seit dem letzten TIGER Import verändert wurden.

Relevanz für unsere Arbeit: Keine, da nur USA verfügbar

Coarse Highway

Beschreibung: Ein grosses Problem sind ungenau erfasste Higways. Dieses Tool erkennt zu stark generalisierte Highways.

Relevanz für unsere Arbeit: Ich denke keine. In der Schweiz wird dies nicht so ein grosses Problem sein.

Map of Turn Restrictions by Zartbitter -(!)-

https://ahorn.lima-city.de/tr/

Beschreibung: Es zeigt die Beschilderung von Strassen (nur rechts/links, nur geradeaus, nicht links/rechts, Einbahn...). Dabei markiert es Fehler und Warnungen bei genau diesen Bezeichnungen.

Relevanz für die Arbeit: Abklären ob solche Fehler den Kunden interessieren. Falls ja, herausfinden aufgrund von welchen Erkentnissen die Fehler erkannt werden. keep right, Weitere Tools

Restriction Analyzer by MorbZ -(!)-

https://restrictions.morbz.de/#14/52.45000/13.35000

Beschreibung: Siehe oben

Relevanz für unsere Arbeit: Siehe oben

US Interstate refs

https://harrywood.dev.openstreetmap.org/waychains/htmlreport/

Beschreibung: Liste der Referenznummern und der entsprechenden OSM interstates.

Relevanz für unsere Arbeit: Keine, da US-Daten

layers.openstreetmap.fr

Beschreibung: Base Layer und Overlays mit verschiedenen Daten zur Verbesserung der Qualität

Relevanz für unsere Aerbeit: Kaum Relevant. Vielleicht links Vom Fehlerstandort in Karten, die zur Verifikation verwendet werden dürfen.(Nur Ansichten die in der Schweiz verfügbar sind)

Maproulette aka Remap-a-Tron

https://maproulette.org/browse/challenges/13527

Beschreibung: Da werden Challenges generiert, die die Qualität der Karte erhöhen. Begonnen ohne Challenge frontend um Bereiche hervorzuheben, die aufgrund der Lizenzänderung gelöscht wurden.

Relevanz für die Arbeit: Ansicht der Callenges/Errors und die damit verbundene Motivation diese zu lösen kann man vielleicht adaptieren. Ganz gelöschte Bereiche in der Schweiz bezweifle ich.

osmlab's to-fix -!!-

https://github.com/osmlab/to-fix/blob/master/LICENSE.md

Beschreibung: Ein Task Manager für Openstreetmap, um sich mit anderen zu koordinieren und eine Liste von Tasks abzuarbeiten, ohne sich gegenseitig zu stören.

Relevanz für unsere Arbeit: Sehr interessant! Tool ziemlich genau das, was wir brauchen. Meherere aus dem Team von Herrn Nüssli wollen eine Liste von Fehleren abarbeiten. Was adaptiert werden müsste wäre, dass Fehler nicht aus gewissen Filterkriterien geladen werden. Nicht einfach ein Bereich der Karte als falsch markieren. Lizenz BSD 3 (Commercial use, Modification, Distribution, Private use alles io)

netzwolf's opening_hours & co. map -(!)-

https://www.netzwolf.info/osm/time_domain/map_opening.html

Beschreibung: Karte mit Kreisen bei Betrieben ohne Öffnungszeiten.

Relevanz für unsere Arbeit: Keine Öffnungszeiten interessieren uns nicht direkt, aber über den Filter sollte man vielleicht bei Interesse auch solche Fehler filtern können.

Rutino error log

http://www.routino.org/

Beschreibung: Generiert einen Error-Log zu der Routenberechnung, was auf Fehler hinweisen könnte.

Relevanz für unsere Arbeit: Keine. Im Monitoring Tool lassen wir Routing-Fehler aus zeitlicher Begrenzung weg.

Bicycle tags map

http://mijndev.openstreetmap.nl/~ligfietser/fiets/index.html

Beschreibung: Findet Bicycle Tags in OSM, die nicht in der Cyclemap vorhanden sind.

Relevanz für unser Tool keine: Nicht die Art von Fehlern, die wir suchen.

Max Speed Overpass

http://mijndev.openstreetmap.nl/~peewee32/maxspeed/Maxspeed.htm

Beschreibung: Die Maximalgeschwindigkeiten der Strassen werden passend eingefärbt.

Relevanz für unsere Arbeit: Keine.

CheckAutopista

http://k1wiosm.github.io/checkautopista2/?id=189802&lat=47.3935&lon=8.4300&z=14

Beschreibung: Kann Informationen über Higways herunterladen und anzeigen. ZB die Maximalgeschwindigkeiten der Strassen werden passend eingefärbt.

Relevanz für unsere Arbeit: Keine.

Relation Analyzer -!-

http://ra.osmsurround.org/

Beschreibung: Der Relation Analyzer untersucht Relationen auf Lücken. Dies ist besonders nützlich für alle Arten von Routen-Relationen, einschliesslich derer, die gerade berbeitet wurden.

Relevanz für unsere Arbeit: Wen man eine Route unter Beobachtung hat, kann man sie hier vielleicht auch analysieren und diese Fehler mit einspeisen.

analyser.openstreetmap.fr

Beschreibung: Auch Analyse von relationalen Objekten.

Relevanz für unsere Arbeit: Service down. Hilfe (Python) ist willkommen. Kontakt tech@listes.openstreetmap.fr

qa.poole.ch -!-

https://wiki.openstreetmap.org/wiki/Qa.poole.ch_(QA_tool)

Beschreibung: Zeigt Strassen ohne Namen.

Relevanz für unsere Arbeit: Fehler, die zu melden sind, wenn in Fehler Kriterien enthalten.

Improve OSM

http://improve-osm.org/

Beschreibung: Erkennen von fehlenden Strassen aus GPS Daten...?

Relevanz für unsere Arbeit: Keine. (experiencing problems or is undergoing routine maintenance)

OSM-Sidewalker by Mapbox -!-

https://www.mapbox.com/osm-sidewalker/#14.83/48.8112/9.1217

Beschreibug: Eine Overlayfunktion um sideway Attributierung zu erfassen.

Relevanz für unsere Arbeit: Fehlerart anzeigen Ja/Nein? 404 nicht erreichbar

Unmapped Places

https://resultmaps.neis-one.org/unmapped#11/46.7264/9.3212

Beschreibung: Markiert Bereiche ohne Strassen.

Relevanz für unssere Arbeit: Keine. Aktuellste Daten: Unmapped places (2019-12-30)

Errors in Alternativas libres

https://mapas.alternativaslibres.es/downloads.php

Beschreibung: Generiert Text-Feiles pro Land mit Fehlern vom Garmin Import.

Relevanz für unsere Arbeit: Keine. Page or file not found

OSMsuspects

https://osm-suspects.gbconsite.de/#16/52.5249/13.2141/osm

Beschreibung: Adressen prüfen von Deutschland.

Relevanz für unsere Arbeit: Keine.

Is OSM up-to-date

Beschreibung: Zeigt alte Daten, die potentiell veraltet sind.

Relevanz für unsere Arbeit: Fehler, die unser Tool nicht anzeigen wird, sonst erhält man die immer wieder -> nicht geeignet für laufende Kontrolle.

BRouter Suspect Manager

http://brouter.de/osmoscope/#map=14/8.54286/47.37673&tab=s&l=http://brouter.de/osmoscope/active_deferred.json

Beschreibung: Korrekturen auf das Routing optimiert. (Osmoscope)

Relevanz für unsere Arbeeit: Keine

housenumbervalidators

Beschreibung: OSM Inspector adress layer und NoName Map.

Relevanz für die Arbeit: In OSM Inspector sehen oder NoName

Osmoscope -!-

http://osmoscope.openstreetmap.de/

Beschreibung: Zeigt Errors. Man kann Layers selber hinzufügen. Beta Version

Relevanz für unsere Arbeit: Hübsche Oberflächen Gestaltung.

OSM POI Analyzer (OPA) -(!)-

http://openstreetmap.me/

Beschreibug: Dieses Analysetool bietet Metriken und Empfehlungen für die Positionsplausibilität von POIs in OpenStreetMap. Die analytische Technik basiert auf der räumlichen Assoziationsregeln.

Relevanz für unsere Arbeit: Klären, ob dies zu den Errors kommt. (Städte auf den ersten Blick nicht in der Schweiz)

staty -!-

https://staty.cs.uni-freiburg.de/

Beschreibung: Public Transport stops

Relevanz für unsere Arbeit: Bei passendem Filter mit entsprechenden Attributen vielleicht beziehen.

Monitoring Tools

-!- Planet History beinhaltet die ganze Geschichte mit all ihren Änderungen. (Latest Full History Planet XML File 142 GB, created 4 days ago Extracts & Mirrors sind möglich) Dadurch kann man auch ein Rollback der Änderungen ermöglichen.

osm.org -!-

https://www.openstreetmap.org/history

Beschreibung: Das API v0.6 beinhaltet die Änderungen. Es gibt dazu zwei Frontends:

Relevanz für unsere Arbeit: Die Änderungen von gefilterten Bereichen und Objekten sollten angezeigt werden.

LiveEditMapViewerJ

Beschreibung: Ein Java Tool, um Live-Änderungen basierend auf Planet.osm diff zu sehen.

Relevanz für unsere Arbeit: Keine, wir werden die veränderten Geometripunkte nicht so genau analysieren.

OSMZmiany

Beschreibung: OSMZmiany ist ein Tool zur Überwachung der Bearbeitungsaktivität. Es ähnelt LiveMapViewer und basiert darauf. Es zeigt jeden editierten node.

Relevanz für unsere Arbeit: Keine, siehe oben

History Browser -!-

Beschreibung: Über die ID wird zu einem Objekt die History angezeigt. Dadurch kann alt und neu nebeneinander angesehen werden.

Relevanz für unsere Arbeit: Vielleicht auch einen Link um die Unterschiede sichtbar zu machen und dem Beobachter bei der Entscheidung zu helfen, ob er eingreifen muss.

magOSM

https://magosm.magellium.com/portail/#/changements

*__Beschreibung:__ Änderungen gruppiert nach Thematik (Heatmap Style). Nur in Frankreich.

*__Relevanz für unsere Arbeit:__ Keine

Mapki's Deep Diff

Beschreibung: Tabelle der Änderung nach ID.

Relevanz für unsere Arbeit: Wir möchten keine Langzeit-History im Monitoring Tool. Nur neue Änderungen.

OSM Deep History

Beschreibung: Tabelle der Änderung nach ID.

Relevanz für unsere Arbeit: Wir möchten keine Langzeit-History im Monitoring Tool. Nur neue Änderungen.

OSM Visual History

Beschreibung: Tabelle + Karte der Änderung nach ID (Node, Way, Relation).

Relevanz für unsere Arbeit: Wir möchten keine Langzeit-History im Monitoring Tool. Nur neue Änderungen.

OSM History Viewer (by PeWu) -(!)-

https://pewu.github.io/osm-history/#/way/259444761

Beschreibung: Um die Änderungen eines Objekts für verschiedene Versionen zu visualisieren, funktioniert für Elemente, Knoten und Weg (Fläche ähnlich achavi). Es zeigt pro Element eine Liste mit Änderungen und je einem passenden Kartenausschnitt und viele Links, um es zu bearbeiten oder in andere Monitoring tools.

Relevanz für unsere Arbeit: Frage ob unsere Liste auch so aussehen sollte und diese Links zur Bearbeitung übernehmen?

OSM History Viewer (OSMHV)

Beschreibung: Ist ein Tool zur Änderungsüberwachung und zum Debuggen, um die Änderungen in einem einzelnen Änderungssatz zu visualisieren und die Geschichte einer Beziehung zu analysieren.

Relevanz für unsere Arbeit: Keine. Old styled Page mit wenig Nutzerfreundlichkeit.

OSM Aware

Beschreibung: OSMaware ist ein Python-Befehlszeilentool, das eine OSM-ASC-Datei verwendet und eine KML-Datei der Mapper-Aktivität erstellt.

Relevanz für unsere Arbeit: Keine. Man darf keine Google-Karten zu Erstellung der OSM verwenden.

Historical Coverage

Beschreibung: This service is no longer available. Dienst um GIFs zu erstellen, um die OSM-Abdeckung zu beobachten.

Relevanz für unsere Arbeit: Keine.

UserActivity

Beschreibung: Statistiken zur Benutzeraktivität zur Erkennung von Vandalismus. 2 Files in und bsp PNG. (osm dif mit mehr Details)

Relevanz für unsere Arbeit: Wir müssen Vandalismus erkennen, aber dies scheint mir zu umständlich.

WhoDidIt 3 Tools

Beschreibung: -> Funktioniert nicht? Verstehe nicht was man sehen sollte

Relevanz für unsere Arbeit: Keine.

Latest OSM Edits per Tile

Beschreibung: Es werden die Änderungen pro Tile der max. letzten 14 Tagen auf einer Art Heatmap dargestellt.

Relevanz für unsere Arbeit: Keine, da wir nur an spezifischen Daten interessiert sind.

RSS History Filter

Beschreibung: -> Funktioniert nicht? Verstehe nicht was man sehen sollte

Relevanz für unsere Arbeit: Keine

OSM controltool

Beschreibung: -> Seite nicht verfügbar

Relevanz für unsere Arbeit: Keine

tyrasd's RSS feed link creator -!-

https://tyrasd.github.io/osm-qa-feeds/

Beschreibung: Man kann Regionen kennzeichnen und dann Exports von verschiedenen Seiten herunterladen.

Relevanz für unsere Arbeit: Exports vielleicht im Tool einbauen.

tyrasd's Latest Changes service

https://tyrasd.github.io/latest-changes/#14/47.3790/8.5374

Beschreibung: Zeigt bei mir nichts an.

Relevanz für unsere Arbeit: Keine

achavi

https://overpass-api.de/achavi/

Beschreibung: Zeigt OpenStreetMap-Änderungen basierend auf der Adiff-Abfrage (Overpass API Augmented Delta) und dem Augmented Diff-Format.

Relevanz für unsere Arbeit: Keine. Es gibt keine Geometrie-Änderungs-Bilder

OpenStreetMap Analytic Difference Engine -(!)-

Beschreibung: Tool mit Change-List und passender Diff Ansicht.

Relevanz für unsere Arbeit: Code einsehen. Tool ist nicht on und deshalb sicher zeitaufwändig um sich nur inspirieren zu lassen.

OSMarelmon - The OSM Relation Monito

Beschreibung: Link geht nicht

Relevanz für unsere Arbeit: Keine

Where are the new OSM Contributors? -!-

https://resultmaps.neis-one.org/newestosm.php#10/47.1841/8.7197

Beschreibung: Neuste Mapper der letzten 7 Tage.

Relevanz für unsere Arbeit: Vielleicht um die Fehler als wichtiger zu Kategorisieren, wenn der Mapper neu ist.

OSMCHA - OSM Changeset Analyzer -!!-

https://osmcha.org/changesets/91462372

Beschreibung: Monitoringtool mit Filtermöglichkeiten

Relevanz für unsere Arbeit: Basis für unsere Arbeit. Was noch ergänzen (Regeln für Fehler festlegen und so klassieren? Gemeinsam arbeiten und auf bearbeiten setzen, damit nicht die Gleichen bearbeitet werden...)

Osmlab Changeset-map - OSM Changeset Viewer

Beschreibung: Filter nach einem Chngeset und darstellen des Changeset auf einer Map.

Relevanz für unsere Arbeit: Keine

osm-suspicious -!!-

https://resultmaps.neis-one.org/osm-suspicious#5/48.894/11.404

Beschreibung: Suche nach verdächtigen OpenStreetMap-Änderungssätzen

Relevanz für unsere Arbeit: Interessant. Was ist verdächtig?

OSM Hall Monito -!!-

Beschreibung: Mit OSM Hall Monitor können Sie Änderungen verfolgen, die von bestimmten Benutzern, an bestimmten Objekten oder mit bestimmten Tags vorgenommen wurden. Sie können auch Benachrichtigungen aktivieren, um E-Mails zu empfangen, wenn eine der markierten Personen etwas bearbeitet oder eines der Beobachteten Elemente bearbeitet wird. Grundlegende Funktionen für die Überwachung verdächtiger Änderungssätze sind ebenfalls enthalten.

Relevanz für unsere Arbeit: Sehr relevant, sind unsere Anforderungen bereits abgedeckt?

https://wiki.openstreetmap.org/wiki/Quality_assurance

-!- Relevant für unsere Arbeit -!!- Sehr relevant für unsere Arbeit -(!)- Vielleicht relevant für unsere Arbeit -(!!)- Vielleicht sehr relevant für unsere Arbeit

2 UI Design Entwurf

Filter Übersicht

Stadt Zürich
Schutz & Rettung

Meine Filter

Praktikant Hero

Keine neuen Datensätze

\+

Neuer Filter

Für mich freigegebene Filter

Flughafen

8 neue Datensätze

---

zurück zur Filter Übersicht

Praktikant Hero Filter

Stadt Zürich
Schutz & Rettung

aufällige Datensätze:

Sortierung: Wichtigkeit▼

aktuellen Filter Bearbeiten

ErrorName Id — High
LINK OSM
Description
llaslkdsnc hsdhb sidd
f hsdjsdfsdffdd...
In Bearbeitung

ErrorName Id — High
LINK OSM
Description
llaslkdsnc hsdhb sidd
f hsdjsdfsdffdd

ErrorName Id — Middle
LINK OSM
Description
llaslkdsnc hsdhb sidd
f hsdjsdfsdffdd

ErrorName Id — Middle
LINK OSM
Description
llaslkdsnc hsdhb sidd
f hsdjsdfsdffdd

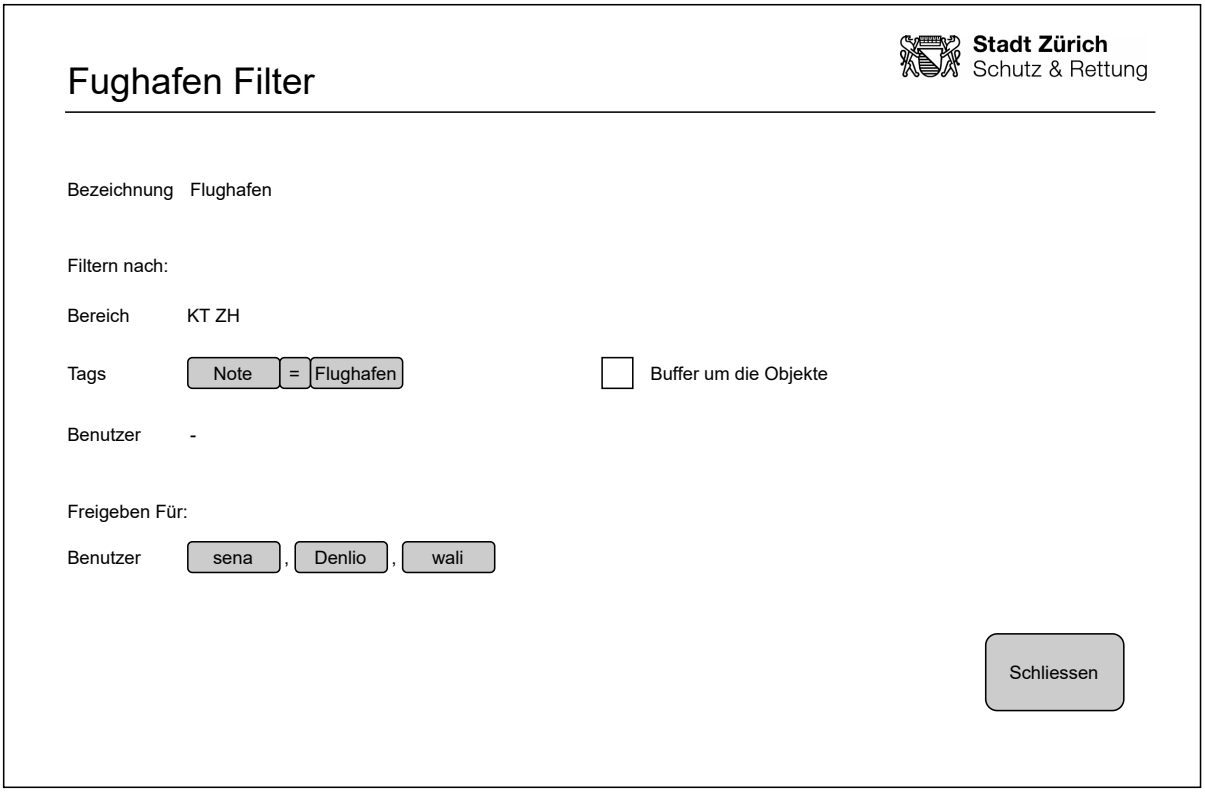
Vorschau

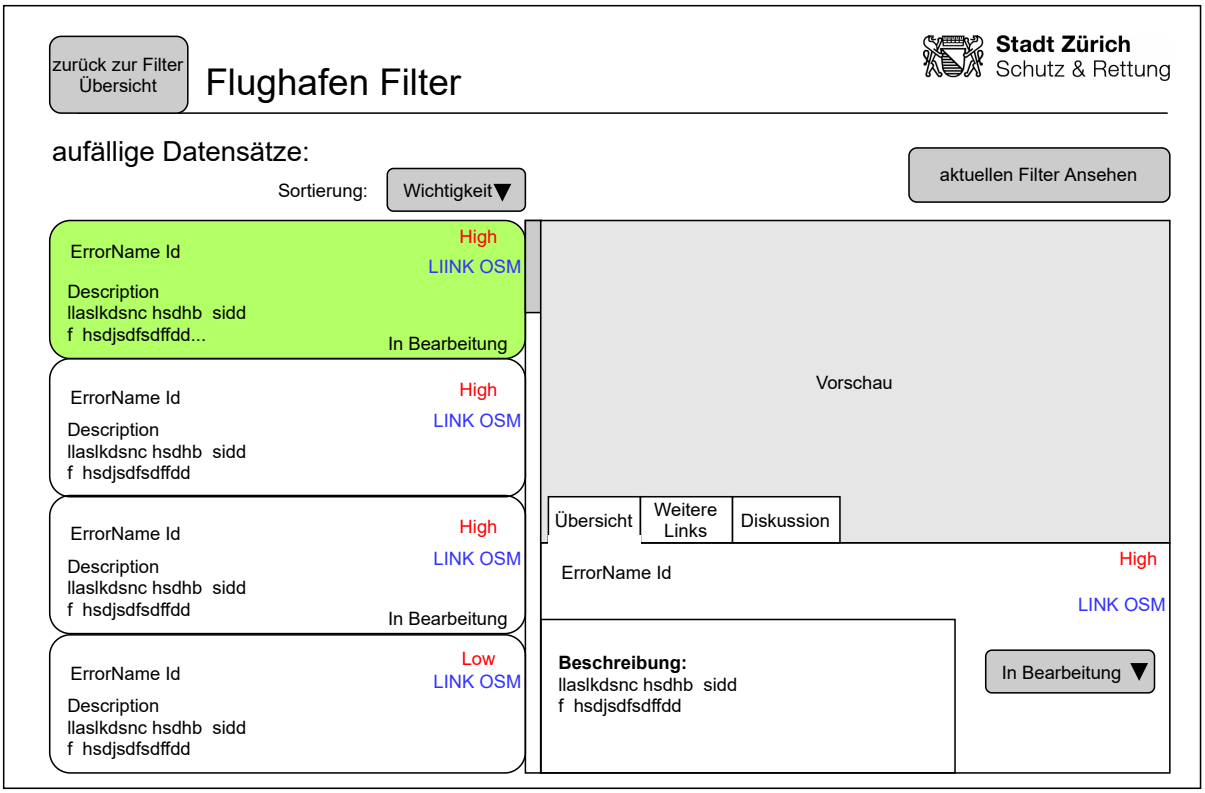
Übersicht | Weitere Links | Diskussion

ErrorName Id — High
LINK OSM

Beschreibung:
llaslkdsnc hsdhb sidd
f hsdjsdfsdffdd

In Bearbeitung ▼


Fughafen Filter
Stadt Zürich
Schutz & Rettung
Bezeichnung Flughafen
Filtern nach:
Bereich KT ZH
Tags Note = Flughafen
Buffer um die Objekte
Benutzer -
Freigeben Für:
Benutzer sena , Denlio , wali
Schliessen


zurück zur Filter Übersicht
Flughafen Filter
Stadt Zürich
Schutz & Rettung
aufällige Datensätze:
Sortierung: Wichtigkeit
aktuellen Filter Ansehen
ErrorName Id
High
LIINK OSM
Description
llaslkdsnc hsdhb sidd
f hsdjsdfsdffdd...
In Bearbeitung
ErrorName Id
High
LINK OSM
Description
llaslkdsnc hsdhb sidd
f hsdjsdfsdffdd
ErrorName Id
High
LINK OSM
Description
llaslkdsnc hsdhb sidd
f hsdjsdfsdffdd
In Bearbeitung
ErrorName Id
Low
LINK OSM
Description
llaslkdsnc hsdhb sidd
f hsdjsdfsdffdd
Vorschau
Übersicht
Weitere Links
Diskussion
ErrorName Id
High
LINK OSM
Beschreibung:
llaslkdsnc hsdhb sidd
f hsdjsdfsdffdd
In Bearbeitung

Neuer Filter

Stadt Zürich
Schutz & Rettung

Bezeichnung

Filtern nach:

Bereich

Tags

☐ Buffer um die Objekte

10 m ▼

Benutzer

Freigeben Für:

Benutzer , ,

Abbrechen | Speichern

Spitäler Filter Bearbeiten

Stadt Zürich
Schutz & Rettung

Bezeichnung Spitäler

Filtern nach:

Bereich KT ZH

Tags Note = Spital

☑ Buffer um die Objekte

10 m ▼

Benutzer

Freigeben Für:

Benutzer sena , Denlio , wali

Abbrechen | Speichern

Praktikant Hero Filter Bearbeiten

Stadt Zürich
Schutz & Rettung

Bezeichnung Spitäler

Filtern nach:

Bereich KT ZH

Tags

☐ Buffer um die Objekte

10 m

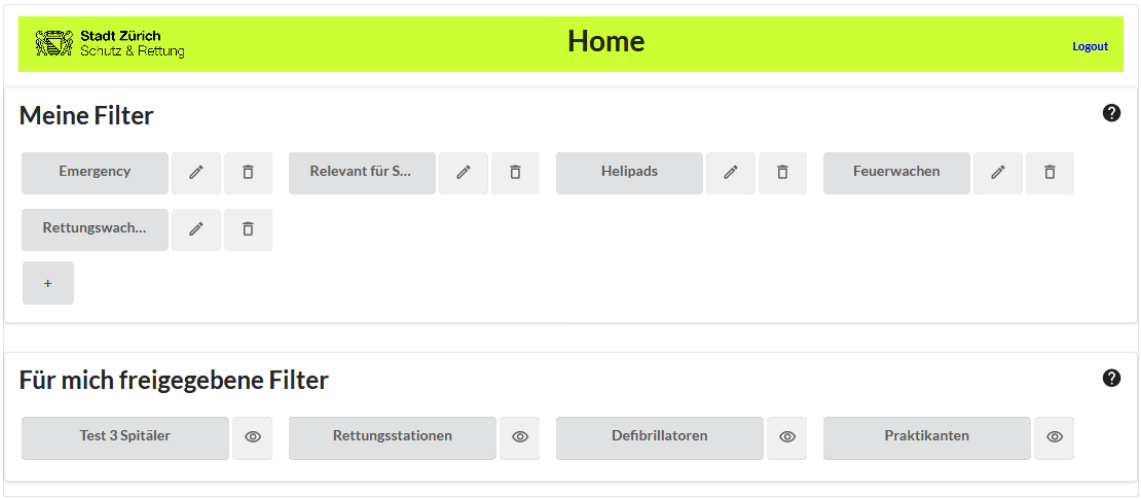
Benutzer , Hero

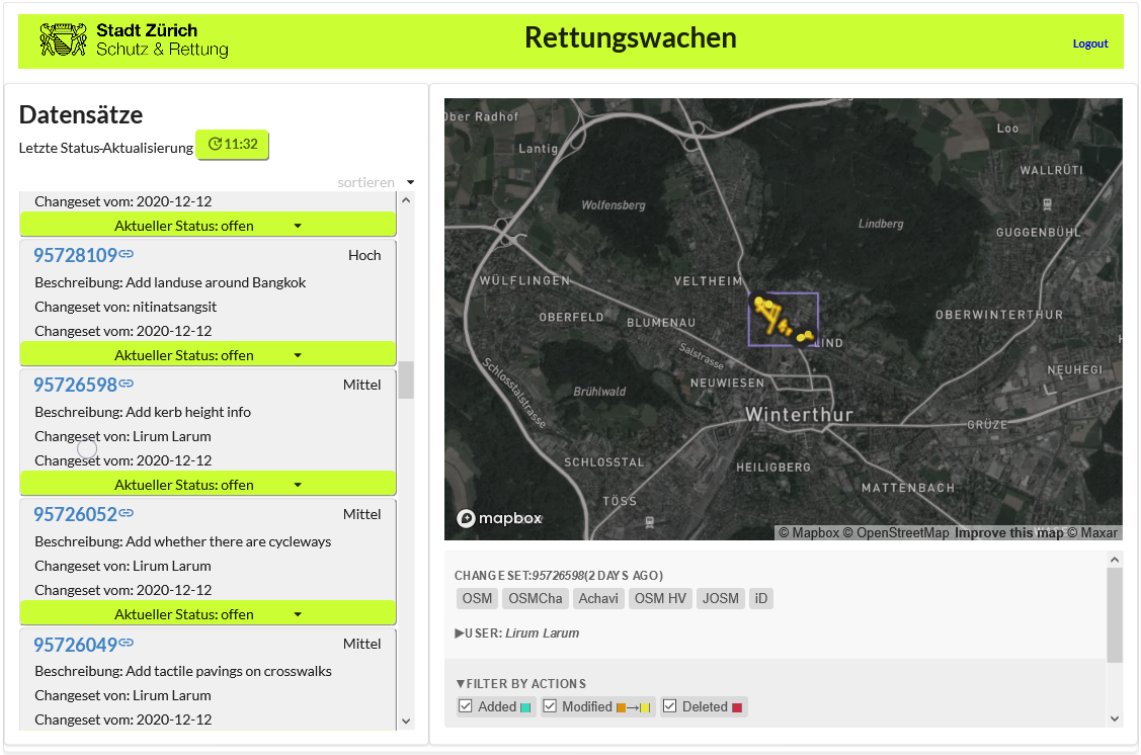
Freigeben Für:

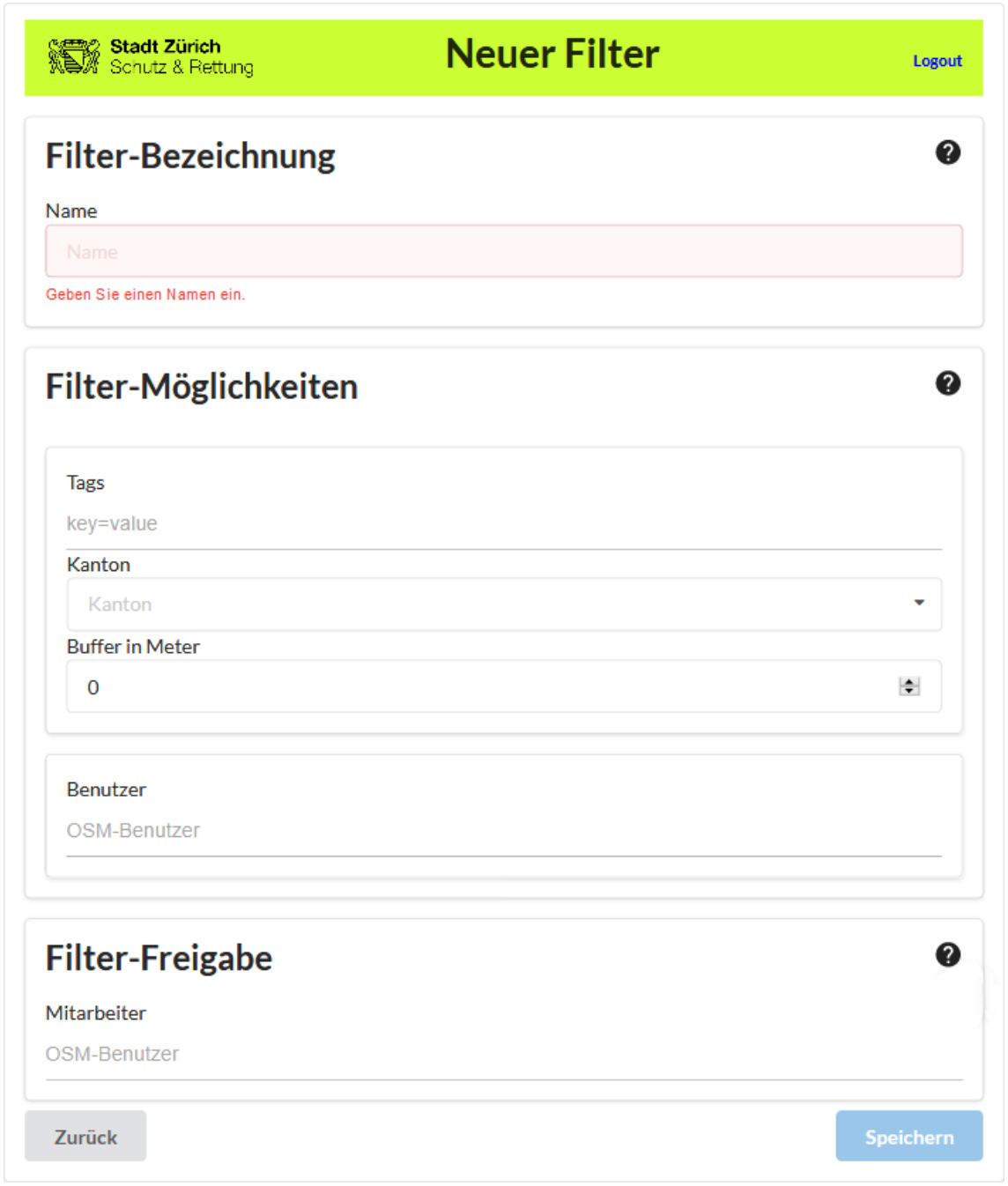
Benutzer

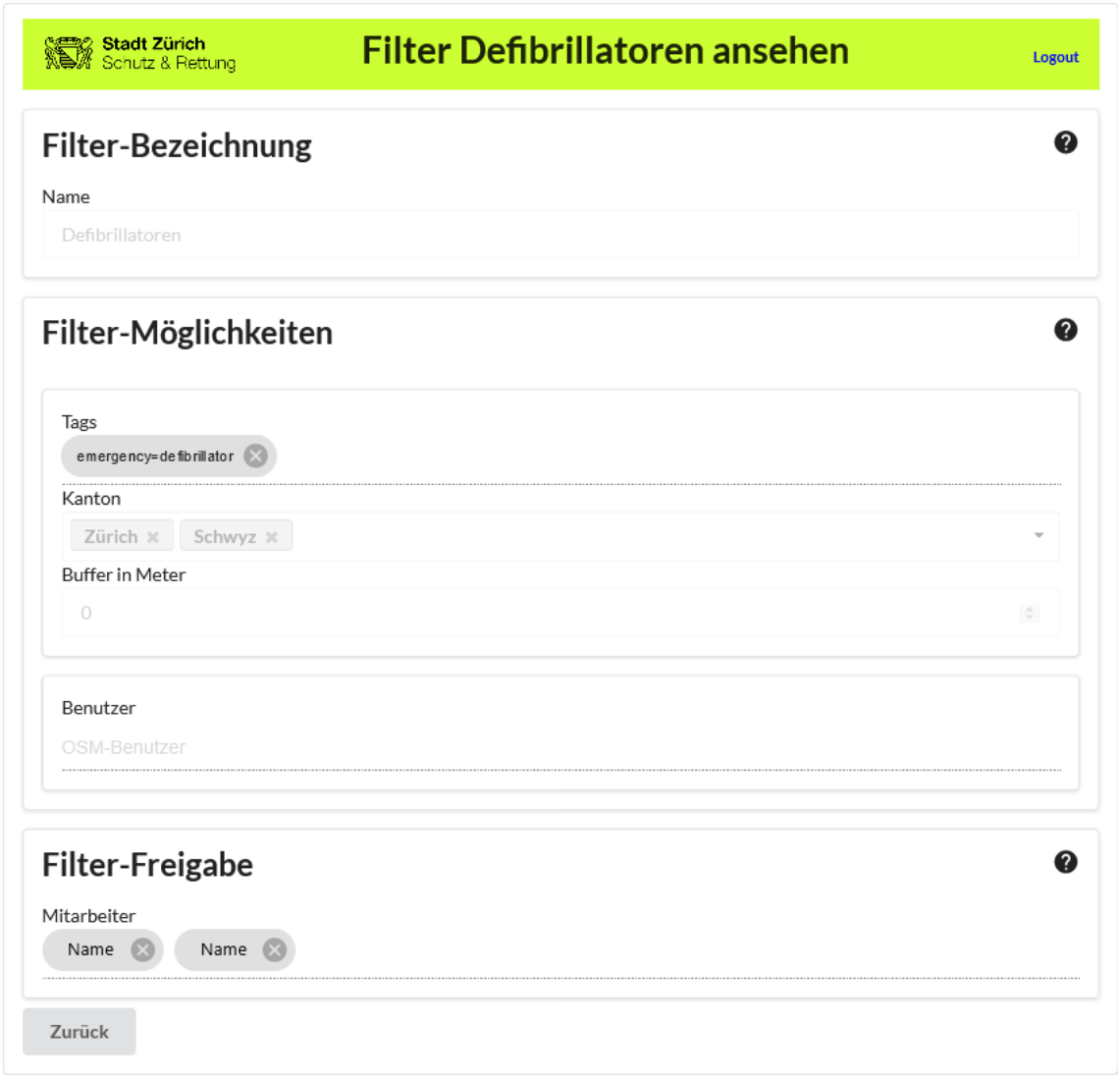
Abbrechen Speichern

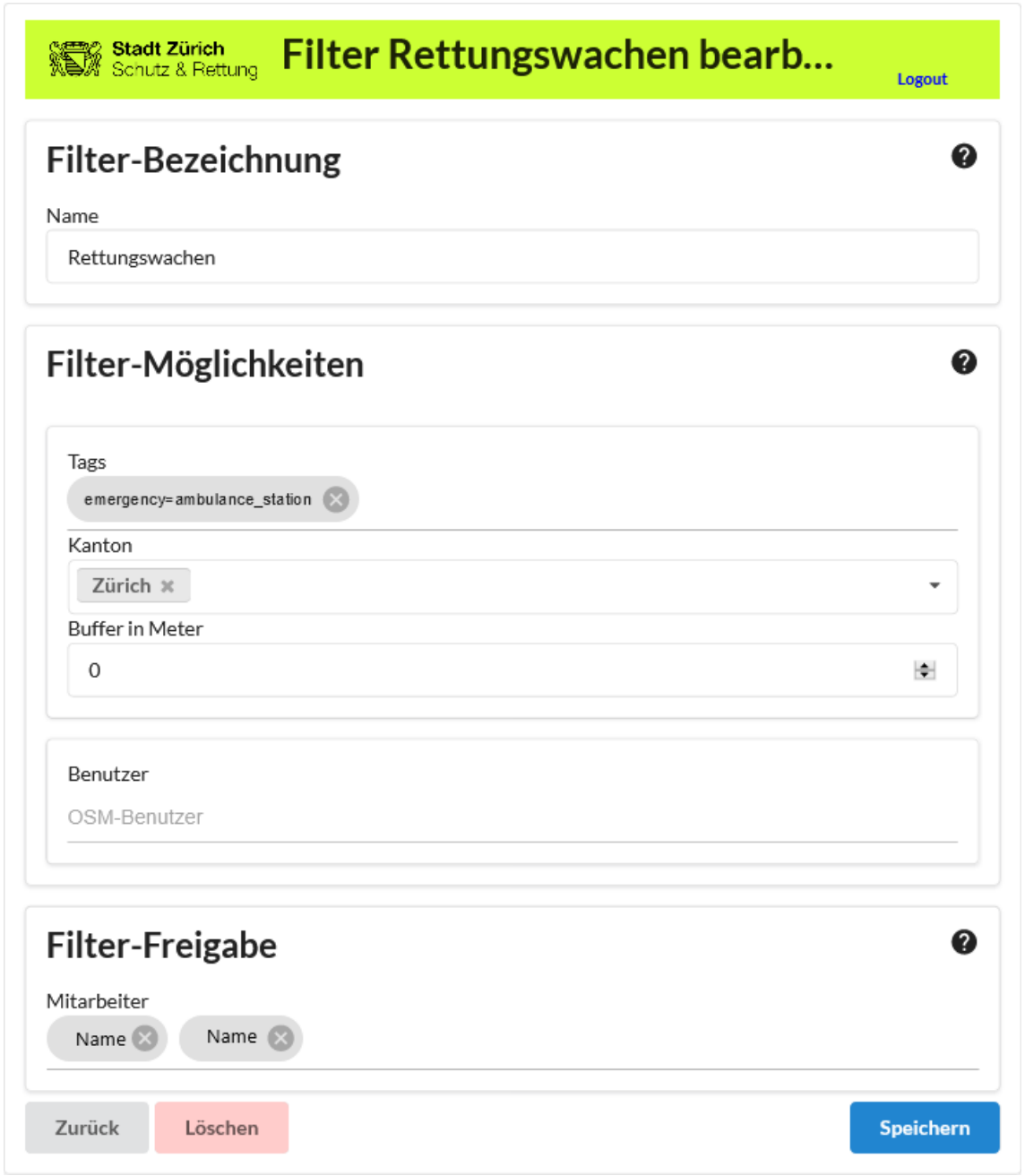
3 UI Endversion






Stadt Zürich
Schutz & Rettung
Neuer Filter
Logout
Filter-Bezeichnung
Name
Name
Geben Sie einen Namen ein.
Filter-Möglichkeiten
Tags
key=value
Kanton
Kanton
Buffer in Meter
0
Benutzer
OSM-Benutzer
Filter-Freigabe
Mitarbeiter
OSM-Benutzer
Zurück
Speichern


Stadt Zürich
Schutz & Rettung
Filter Defibrillatoren ansehen
Logout
Filter-Bezeichnung
Name
Defibrillatoren
Filter-Möglichkeiten
Tags
emergency=defibrillator
Kanton
Zürich
Schwyz
Buffer in Meter
0
Benutzer
OSM-Benutzer
Filter-Freigabe
Mitarbeiter
Name
Name
Zurück


Stadt Zürich
Schutz & Rettung
Filter Rettungswachen bearb...
Logout
Filter-Bezeichnung
Name
Rettungswachen
Filter-Möglichkeiten
Tags
emergency=ambulance_station
Kanton
Zürich
Buffer in Meter
0
Benutzer
OSM-Benutzer
Filter-Freigabe
Mitarbeiter
Name
Name
Zurück
Löschen
Speichern

4 Systemtests

Beta 26.11.2020

Was	Implementiert	Kommentar	Verbesserung
1 Auffällige Datensätze erfassen: Im OSM System erfassen OSM-User Daten.	x	Changesets Schweizweit	-
2 Einloggen: Der unsignedin User loggt sich mit seinem OSM-Account im Tool ein.	x	-	Login seite verschönern.
3 Auffällige Datensätze sehen: Der Datenkurator sieht im Dashboard die Datensätze, welche das Tool anhand des Filters als auffällig gekennzeichnet hat.	-	Erst Admin Filter	--
4 Auffällige Datensätze auf "In Bearbeitung" setzen: Der Benutzer kann einen auffälligen Datensatz im Dashboard auf "In Bearbeitung" setzen, damit die anderen Benutzer sehen, dass dieser Datensatz gerade abgearbeitet wird.	-	---	--
5 Auffällige Datensätze auf "Ignorieren" setzen: Der Benutzer kann einen auffälligen Datensatz im Dashboard auf "Ignorieren" setzen, damit dieser nicht mehr im Tool auftaucht. Den anderen Benutzern wird dieser dann auch nicht mehr angezeigt.	-	---	--
6 Auffällige Datensätze auf "Verbessert" setzen: Wenn der Benutzer einen auffälligen Datensatz abgearbeitet hat, kann er diesen auf "Verbessert" setzen. Andere Benutzer sehen dann, dass dieser Datensatz korrigiert wurde.	-	---	--
7 Filter einsehen: Der Benutzer kann den ausgewählten Filter anschauen. Er sieht da Informationen, welche Tags oder Personen überwacht werden, ob ein Buffer um die Objekte gesetzt wurde, den Namen des Filters und die hinzugefügten Datenkuratoren.	-	---	--
8 Filter bearbeiten: Der Administrator kann den Filter bearbeiten. Er kann den Namen des Filters ändern, Tags und zu überwachende Personen hinzufügen oder entfernen, den Buffer verändern oder entfernen und Datenkuratoren hinzufügen oder entfernen (Siehe Use Case 10 und 11).	-	---	--

Was	Implementiert	Kommentar	Verbesserung
9 Filter erstellen: Ein Benutzer kann einen neuen Filter erstellen und ist so automatisch Administrator vom neu erstellten Filter. Dabei muss ein Filtername und mindestens ein zu überwachender Tag (mit oder ohne Buffer) angegeben werden.	x	Filter nach Tag ist noch required (mit Region und Buffer)	Nach Tag nicht required
10 Datenkuratoren hinzufügen: Der Administrator kann einem Filter über "Filter bearbeiten" oder direkt beim Erstellen einen oder mehrere Datenkuratoren hinzufügen. Diese können ihm bei Use Case 3-6 helfen.	-	---	--
11 Datenkuratoren entfernen	-	---	--

Endversion 09.12.2020 (wurde noch nach gebessert -> neuster Test unten!)

Was	Implementiert	Kommentar	Verbesserung
1 Auffällige Datensätze erfassen: Im OSM System erfassen OSM-User Daten.	x	Changesets Schweizweit	-
2 Einloggen: Der unsignedin User loggt sich mit seinem OSM-Account im Tool ein.	x	-	-
3 Auffällige Datensätze sehen: Der Datenkurator sieht im Dashboard die Datensätze, welche das Tool anhand des Filters als auffällig gekennzeichnet hat.	x	-	-
4 Auffällige Datensätze auf "In Bearbeitung" setzen: Der Benutzer kann einen auffälligen Datensatz im Dashboard auf "In Bearbeitung" setzen, damit die anderen Benutzer sehen, dass dieser Datensatz gerade abgearbeitet wird.	x	Kein server Push sondern mit Update-Knopf	-
5 Auffällige Datensätze auf "Ignorieren" setzen: Der Benutzer kann einen auffälligen Datensatz im Dashboard auf "Ignorieren" setzen, damit dieser nicht mehr im Tool auftaucht. Den anderen Benutzern wird dieser dann auch nicht mehr angezeigt.	x	"	-
6 Auffällige Datensätze auf "Verbessert" setzen: Wenn der Benutzer einen auffälligen Datensatz abgearbeitet hat, kann er diesen auf "Verbessert" setzen. Andere Benutzer sehen dann, dass dieser Datensatz korrigiert wurde.	x	"	-

Was	Implementiert	Kommentar	Verbesserung
7 Filter einsehen: Der Benutzer kann den ausgewählten Filter anschauen. Er sieht da Informationen, welche Tags oder Personen überwacht werden, ob ein Buffer um die Objekte gesetzt wurde, den Namen des Filters und die hinzugefügten Datenkuratoren.	x	-	-
8 Filter bearbeiten: Der Administrator kann den Filter bearbeiten. Er kann den Namen des Filters ändern, Tags und zu überwachende Personen hinzufügen oder entfernen, den Buffer verändern oder entfernen und Datenkuratoren hinzufügen oder entfernen (Siehe Use Case 10 und 11).	x	-	-
9 Filter erstellen: Ein Benutzer kann einen neuen Filter erstellen und ist so automatisch Administrator vom neu erstellten Filter. Dabei muss ein Filtername und mindestens ein zu überwachender Tag (mit oder ohne Buffer) angegeben werden.	x	-	-
10 Datenkuratoren hinzufügen: Der Administrator kann einem Filter über "Filter bearbeiten" oder direkt beim Erstellen einen oder mehrere Datenkuratoren hinzufügen. Diese können ihm bei Use Case 3-6 helfen.	x	-	-
11 Datenkuratoren entfernen	x	-	-
12 Benachrichtigung per mail	-	-	-

Endversion 17.12.2020

Was	Implementiert	Kommentar	Verbesserung
1 Auffällige Datensätze erfassen: Im OSM System erfassen OSM-User Daten.	x	Changesets Schweizweit	-
2 Einloggen: Der unsignedin User loggt sich mit seinem OSM-Account im Tool ein.	x	-	-
3 Auffällige Datensätze sehen: Der Datenkurator sieht im Dashboard die Datensätze, welche das Tool anhand des Filters als auffällig gekennzeichnet hat.	x	-	-
4 Auffällige Datensätze auf "In Bearbeitung" setzen: Der Benutzer kann einen auffälligen Datensatz im Dashboard auf "In Bearbeitung" setzen, damit die anderen Benutzer sehen, dass dieser Datensatz gerade abgearbeitet wird.	x	Kein server Push sondern mit Update-Knopf	-

Was	Implementiert	Kommentar	Verbesserung
5 Auffällige Datensätze auf "Ignorieren" setzen: Der Benutzer kann einen auffälligen Datensatz im Dashboard auf "Ignorieren" setzen, damit dieser nicht mehr im Tool auftaucht. Den anderen Benutzern wird dieser dann auch nicht mehr angezeigt.	x	Ignorieren = abgeschlossen	-
6 Auffällige Datensätze auf "Verbessert" setzen: Wenn der Benutzer einen auffälligen Datensatz abgearbeitet hat, kann er diesen auf "Verbessert" setzen. Andere Benutzer sehen dann, dass dieser Datensatz korrigiert wurde.	x	Verbessert = abgeschlossen	-
7 Filter einsehen: Der Benutzer kann den ausgewählten Filter anschauen. Er sieht da Informationen, welche Tags oder Personen überwacht werden, ob ein Buffer um die Objekte gesetzt wurde, den Namen des Filters und die hinzugefügten Datenkuratoren.	x	-	-
8 Filter bearbeiten: Der Administrator kann den Filter bearbeiten. Er kann den Namen des Filters ändern, Tags und zu überwachende Personen hinzufügen oder entfernen, den Buffer verändern oder entfernen und Datenkuratoren hinzufügen oder entfernen (Siehe Use Case 10 und 11).	x	-	-
9 Filter erstellen: Ein Benutzer kann einen neuen Filter erstellen und ist so automatisch Administrator vom neu erstellten Filter. Dabei muss ein Filtername und mindestens ein zu überwachender Tag (mit oder ohne Buffer) angegeben werden.	x	-	-
10 Datenkuratoren hinzufügen: Der Administrator kann einem Filter über "Filter bearbeiten" oder direkt beim Erstellen einen oder mehrere Datenkuratoren hinzufügen. Diese können ihm bei Use Case 3-6 helfen.	x	-	-
11 Datenkuratoren entfernen	x	-	-
12 Benachrichtigung per mail	-	-	-

5 Usability Test

Usability Test

Testszenarien

Kurzbeschrieb	Szenario	Fragestellung (Hypothese)
Filter finden	Stellen Sie sich vor, sie sind Datenkurator und haben eine Benachrichtigung erhalten über neue auffällige Datensätze am Flughafen.	Findet der Benutzer diesen Filter
Gewichtung	Da sie in wenigen Minuten eine Sitzung haben, möchten sie die mit "High risk" zuerst bearbeiten. Können sie solche erkennen?	Kann der Tester die Sortierung und die Wertung erkennen?
Status	Mit welchem Datensatz würden Sie mit der Arbeit beginnen?	Einer nicht in Bearbeitung
Detail Infos	Zum Bearbeiten - Wie gehen Sie vor?	Wird der Status auf "In Bearbeitung" umgeschaltet und aufden Link ins OSM geklickt (Weitere Details ansehen.)?
Filter wechseln	Nun möchten sie noch kurz in ihrem Filter die Änderungen der Praktikanten ansehen.	Zurück zu alle Filter und in den Praktikanten Filter
Filter bearbeiten	Fügen Sie dem Filter noch den neuen Praktikanten Alfred hinzu.	Findet bearbeiten, kann Alfred adden und speichern.
Filter anlegen	Zum Abschluss noch schnell den neuen Filter für die Sitzung vorbereiten. Es sollte einen neuen Überwachungsbereich geben, der Spitäler im Kanton Zürich und einen Umkreis von 10m beobachtet. Der sollte den Datenkuratoren sena, Denlio, wali zur Verfügung stehen.	Neuen Filter anlegen und wissen wo was abfüllen.

Protokoll Christian Nüssli

Filter finden:

- Dauer bis ausgeführt: schnell
- [x] gefunden
- [ ] Verrwirung wegen Eigenen und Freigegebenen Filter
- [x] Inhalte gelesen
- [x] Genügend Inhalte sichtbar
- [ ] Umständlicher Einstieg
- Bemerkungen: Interpretation eigene Filter als priorisierte Filter. Ich kann freigegebene Filter zu meinen Filtern hochnehmen.

Gewichtung:

- Dauer bis ausgeführt: schnell
- [x] Gewichtung erkannt
- [x] Sortierung gesehen
- Bemerkungen: Auf der Seite zurechtfinden klappt gut. Beschreibt: links Änderungen und rechts Vorschau (sehr toll). Erster wird angewählt sein. Ist sortiert, es sind die High-Qualified der 8 angegebenen Fehler angezeigt. Gewichtung etwas grösser gewünscht. Meint er sieht nur Status neu nicht in Bearbeitung.

Status:

- Dauer bis ausgeführt: schnell
- [ ] Erkannt
- [ ] Beachtung geschenkt
- Bemerkungen: Keine wirkliche Beachtung. Er kann sich vorstellen, dass nur solche mit Status offen ersichtlich sind und deshalb den obersten. Mit Nachfragen, was ihm das In Bearbeitung sagt, wurde es erkannt.

Detail Infos:

- Dauer bis ausgeführt:
- [x] Link gefunden
- [x] Status-Change gefunden

- [ ] Es fehlen Infos
- [ ] Unübersichtlich
- [ ] Zu viele Infos
- Bemerkungen: Noch nicht so klar, welche Informationen. Da dargestellt werden aber in diesem Status auch noch keine realen Daten abgefüllt. Changeset Disskusion erkannt. Nicht den direkt link in Liste verwendet über Details. Status wechsel jetzt noch klar, aber ich hatte nicht den Eindruck, dass dies mit der Zeit nicht vergessengeht.

Filter wechseln:

- Dauer bis ausgeführt: schnell
- [x] Zurück zur Übersicht gefunden
- [x] Nächster Filter gefunden
- [ ] Zu Umständlich
- [ ] Verwirrend
- Bemerkungen: Schema mit Admin und Datacurator noch nicht ganz klar. (Unterschied meine / freigegebene Filter). Ist sehr strikt im Denkmuster Login als Admin und dann muss da eine andere Sicht kommen. Benennung statt meine Filter geleistete Arbeiten und pro Praktikant ein Kästchen. Brauchte eine kurze Erklärung, wie wir uns das vorgestellt haben und dann war die Idee mit der Unterteilung ganz ok.

Filter bearbeiten:

- Dauer bis ausgeführt: schnell
- [x] Bearbeiten gefunden
- [x] Klar wo was angeben
- [ ] Zu umständlich
- [ ] Verwirrend
- [ ] Funktionalität fehlt
- Bemerkungen: Sofort Benutzer ergänzt.

Filter anlegen:

- Dauer bis ausgeführt: schnell
- [x] Neu gefunden
- [ ] Klar wo was angeben
- [ ] Zu umständlich
- [ ] Verwirrend
- [ ] Funktionalität fehlt
- Bemerkungen: Das Benutzerfeld war plötzlich nicht mehr so klar. Wollte sich dort als Benutzer des Filters eingeben. Buffer war ihm ein unbekannter Begriff, musste erklärt werden. Er wünst sich Hover zur Eingabehilfe und Erklärung. Sieht Schwierigkeiten bei der Eingabe der Tags mit korrektem Key-Value-Pair. Wünscht sich Hilfen.
- Wie sind Sie mit der Navigation zurecht gekommen?
- [x] ++
- [ ] +
- [ ] /
- [ ] -
- [ ] - -
  Bemerkung:
- Waren die Angaben in den Previews für Sie aufschlussreich genug??
- [ ] ++

- [ ] +
- [ ] -
- [ ] - -
  Bemerkungen: Noch nicht beurteilbar da Informationen noch nicht abgebildet werden.
- Wie gefällt Ihnen das Design? (Achtung Geschmacksfrage)
- [x] ++
- [ ] +
- [ ] /
- [ ] -
- [ ] - -
  Bemerkungen: Schlicht und einfach
- Was war besonders gut: Nicht so vollgepackt sehr aufgeräumt, Filter ansehen als Datacurator
- Was besonders schlecht:
- Probleme aus dem Test aufgreifen:
- Frage: Kann ich die Datacurators vom Filter überwachen. Antwort: Nur mit einem passenden Filter mit den Datacurators Changesets. Kein Status vorgesehen, um Administrator eine Bestätigung zu verlangen.

Positiv

- Nicht so vollgepackt sehr aufgeräumt
- Filter ansehen als Datacurator

Verbesserungen

Probem	Änderungsvorschlag Testperson	Änderungsvorschlag Team	Umsetzung
Differenz eigene Filter und freigegebene Filter nicht intuitiv	-	Neu bennenen von "Von mir Erstellte Filter" zu "Für mich Freigegebene Filter"	Bessere Bezeichnung wird umgesetzt
Admin gleiche Ansicht wie Datacurator	Benennung von "Meine Filter" zu "Geleistete Arbeiten" ändern und pro Praktikant ein Kästchen	-	Team hat beschlossen, dies generischer zu lassen und mit Erklärungen das Denkmuster auf zu brechen. Bei uns ist jeder Admin von seinen eigenen Filtern und kann als Datacurator Fehler bei den Freigegebenen abarbeiten. Nach Erklärung des Grundgedanken war das für den Befragten auch eine tolle Lösung.
Gewichtung geht unter	Gewichtung grösser	-	Wird in der Grösse bez. Auffälligkeit verbessert.
In Bearbeitung nicht sehen	Filtern	Neben der Sortierung noch ein Knopf ausblenden	Filter ist Adminsache und sollte nicht durch den Kurator gemacht werden. Deshalb die Idee mit dem Ausblenden, damit es keine Verwirrung gibt. Die Priorität ist niedrig wird nur bei genug Zeit umgesetzt.
In Bearbeitung setzen könnte vergessen	-	Automatisch wenn man den Link anklickt und nicht bereits umgestellt wurde?	Die Priorität ist niedrig und wird nur bei genügend Zeit umgesetzt.

gehen.

Benutzer Feld im Filter	Nicht mit Beschreibungen überladen, lieber per Hover	Vielleicht I-Button mit Hover Funktion und Abgrenzung mit Kästchen, damit die Zusammengehörigkeit klarer wird.	Nur Hover UX mässig optimal, da nicht ersichtlich ist, das es Infos hat. Auch Touch Bedienung mit Hover nicht gegeben.
Buffer Wort unbekannt	Umkreis oder Hover Info	-	I-Button mit Hover-Funktion mit Erklärung
Tags Eingabe	Unterstützung und Vorschläge gewünscht.	Tagfinder einbauen	Ist bereits im Plan enthalten.

Protokoll Abnahme Test Nüssli

*Dauer schnell, angemessen, langsam

Login:

- Dauer bis ausgeführt: schnell
- [x] gefunden
- [ ] Verrwirung
- [ ] Umständlicher Einstieg
- Bemerkungen: Eigenständig gecklickt, ohne Probleme im Chrome.

Filter finden:

- Dauer bis ausgeführt: schnell
- [x] gefunden
- [ ] Verrwirung wegen Eigenen und Freigegebenen Filter
- [ ] Inhalte gelesen
- [ ] Genügend Inhalte sichtbar
- [ ] Umständlicher Einstieg
- Bemerkungen: Eigenständig gecklickt war von der übersicht Begeistert. Keine Einwände.

Gewichtung:

- Dauer bis ausgeführt: schnell sortiert gerade alle sortierungen getestet
- [x] Gewichtung erkannt
- [ ] Sortierung gesehen
- Bemerkungen: Wollte nocheinmal wissen wie die Gewichtung zustande kam. Gering=neues hinzugefügt, Mittel=min 1 Element geändert, Hoch=min 1 Element gelöscht. Wünscht das in Dokumentationsform, damit dies für alle benutzer klar ist. gerade alle sortierungen getestet.

Status:

- Dauer bis ausgeführt: schnell
- [x] Erkannt
- [ ] Beachtung geschenkt
- Bemerkungen: Ist gut so man sieht nicht wer daran arbeitet oder? bei ihnen nicht so relevant nur kleines Team und ohne Corona sowieso immer miteinander in Kontakt. Kann nicht einem verbieten das zu bearbeiten? Statusupdat klick sichtbarer machen.

Detail Infos:

- Dauer bis ausgeführt: angemessen
- [-] Link gefunden
- [x] Status-Change gefunden
- [ ] Es fehlen Infos
- [ ] Unübersichtlich
- [ ] Zu viele Infos

- Bemerkungen: Links hate nur die in Detailview gesehen und den über die Id nicht ohne Hinweise.

Filter anlegen:

- Dauer bis ausgeführt:schnell
- [x] Neu gefunden
- [x] Klar wo was angeben
- [ ] Zu umständlich
- [x] Verwirrend: Buffer
- [ ] Funktionalität fehlt
- Bemerkungen: hilfen nicht gesehen. schnell eingegeben rückmeldung tagfinder fehlt wenn anfrage zulange dauert.

Filter wechseln:

- Dauer bis ausgeführt: schnell
- [x] Zurück zur Übersicht gefunden
- [x] Filter gefunden
- [ ] Zu Umständlich
- [ ] Verwirrend
- Bemerkungen: Ohne probleme.

Filter bearbeiten:

- Dauer bis ausgeführt: schnell
- [x] Bearbeiten gefunden
- [x] Klar wo was angeben
- [ ] Zu umständlich
- [ ] Verwirrend
- [ ] Funktionalität fehlt
- Bemerkungen: Toll und alle Varrianten getestet.

Logut:

- Dauer bis ausgeführt:schnell
- [x] gefunden
- [ ] Zu umständlich
- [ ] Verwirrend
- Bemerkungen: Selbständig getestet und wider angemeldet.
- Wie sind Sie mit der Navigation zurecht gekommen?
- [x] ++
- [ ] +
- [ ] /
- [ ] -
- [ ] - -
  Bemerkung:
- Waren die Angaben in den Previews für Sie aufschlussreich genug??
- [x] ++
- [ ] +
- [ ] -
- [ ] - -

Bemerkungen: Map Coool

- Wie gefällt Ihnen das Design? (Achtung Geschmacksfrage)
- [x] ++
- [ ] +
- [ ] /
- [ ] -
- [ ] - -
  Bemerkungen: Schlicht und übersichtlich wie gewünscht

Sonstiges: Interesse an Note und Weiterentwicklung als Opensource Software.

Positiv

- Einfacheit
- Map vorschau
- Filtermöglichkeiten

Verbesserungen

Probem	Änderungsvorschlag Testperson	Änderungsvorschlag Team	Umsetzung
Status Update	klickmöglichkeit sichtbar machen (Sybol oder so)	Hoverfearbe	Symbol und Hoverfarbe
Gewichtung Dokumentieren	Möchte die Gewichtung im Benutzerhandbuch	-	Erklärung
Status in Bearbeitung	Erkennen wer am Bearbeiten ist	-	Nein als Weiterentwicklung festgehalten
in Bearbeitung verbieten	Kann ich jemandem verbieten ein spezielles Changeset auf in Bearbeitung zusetzen?	-	Nein als Weiterentwicklung festgehalten
BearbeitungsLink	-	Sichtbarer machen da nicht erkannt dass es ein link ist.	

6 Performance Test

Preformance Test

Test 1 Filter ansehen

Filter der abgefragt wird.
{"name":"Feuer2","filters":{"filterArea":"Zürich","users":"","filterMember":"Nauden","objectTags":"amenity=fire_station","buffer":0}}
Antwort mit 111 changesets

```
startTime = datetime.now()
headers = {'Authorization': 'Token 05f7f233a3ec851deb82e87df58a8f1d19f0a923'}
x = 0
faile = 0
while x <= 10:
    requestStart = datetime.now()
    response = requests.get("http://152.96.56.12/api/v1/aoi/cb193079-857b-42b6-aea1-b0f1f599a131/ch
    print(datetime.now() - requestStart)
    print(x, ": ", response.status_code)
    if response.status_code != 200:
        faile += 1
    x +=1
```

Responses:

```
0:00:02.180108
0 :  200
0:00:03.062462
1 :  200
0:00:02.196539
2 :  200
0:00:02.581364
3 :  200
0:00:02.868121
4 :  200
0:00:03.413978
5 :  200
0:00:04.335350
6 :  200
0:00:04.166852
7 :  200
0:00:03.319641
8 :  200
0:00:03.932801
9 :  200
0:00:01.981734
10 :  200
0:00:34.066944
0
```

```
0:00:02.978223
0 :  200
0:00:03.614340
1 :  200
0:00:02.074738
2 :  200
0:00:02.205956
3 :  200
0:00:02.716548
4 :  200
0:00:03.435629
5 :  200
0:00:03.116232
6 :  200
0:00:02.344976
7 :  200
0:00:02.173097
8 :  200
0:00:02.651573
9 :  200
0:00:02.488927
```

```
10 :  200
0:00:29.808288
0
```

```
0:00:02.576289
0 :  200
0:00:03.414322
1 :  200
0:00:01.987922
2 :  200
0:00:02.785725
3 :  200
0:00:02.433361
4 :  200
0:00:02.856121
5 :  200
0:00:02.111533
6 :  200
0:00:04.219550
7 :  200
0:00:02.019822
8 :  200
0:00:02.755733
9 :  200
0:00:01.979561
10 :  200
0:00:29.172672
```

Durchschnitt der Antwort: 2.5s
Anzahl der Abfragen in Folge ohne Fehler: 10

Test 2 Filter speichern

```
headers2={ 'Authorization': 'Token 05f7f233a3ec851deb82e87df58a8f1d19f0a923', 'content-type': 'appl

y = 0
faile = 0
startTime = datetime.now()
while y <= 10:
    requestStart = datetime.now()
    name= "Performe3" + str(y)
    response = requests.post("http://152.96.56.12/api/v1/aoi", headers=headers2, data={"name":name,
    print(datetime.now() - requestStart)
    print(y, ": ", response.status_code)
    if response.status_code != 200:
        faile += 1
    y +=1
print(datetime.now() - startTime)
print(faile)
```

Responses:

```
0:00:00.406689
0 :  200
0:00:00.137337
1 :  200
0:00:00.148206
2 :  200
0:00:00.248200
3 :  200
0:00:00.158806
4 :  200
0:00:00.229413
5 :  200
0:00:00.160033
6 :  200
0:00:00.141072
7 :  200
0:00:00.184615
8 :  200
0:00:00.172091
```

```
9 :  200
0:00:00.276839
10 :  200
0:00:02.304217
0
```

```
0:00:00.192637
0 :  200
0:00:00.151172
1 :  200
0:00:00.136814
2 :  200
0:00:00.147110
3 :  200
0:00:00.155982
4 :  200
0:00:00.212814
5 :  200
0:00:00.167362
6 :  200
0:00:00.133666
7 :  200
0:00:00.156553
8 :  200
0:00:00.287686
9 :  200
0:00:00.245573
10 :  200
0:00:02.022360
0
```

```
0:00:00.287511
0 :  200
0:00:00.165223
1 :  200
0:00:00.165542
2 :  200
0:00:00.158476
3 :  200
0:00:00.139385
4 :  200
0:00:00.168663
5 :  200
0:00:00.228569
6 :  200
0:00:00.153607
7 :  200
0:00:00.149742
8 :  200
0:00:00.153610
9 :  200
0:00:00.161205
10 :  200
0:00:01.983440
0
```

Durchschnitt der Antwort: 1s
Anzahl der Abfragen in Folge ohne Fehler: 10

7 Installationsanleitung

Targeted Monitoring Installation

Allgemein

- Frontend: https://gitlab.dev.ifs.hsr.ch/se-na/sa-monitoring-osm-frontend
- Backend: https://gitlab.dev.ifs.hsr.ch/DNA/sa-monitoring-osm-backend
- Bedienungsanleitung: Hier

Installation

Sie können den Code über die oben verlinkten Repositories downloaden und so Ihre eigene Backend- und Frontendinstanz laufen lassen.

Voraussetzung sind 2 Ports. Der für das Frontend muss vom Browser des Benutzers erreichbar sein. Und der vom Backend muss das Beziehen von Changesets im Planet-OSM erlauben, sowie den zugriff durch das Frontend.

Run in Docker

Voraussetzung

- Linux mit installiertem Docker und Docker-compose
- 4GB freier Festplattenspeicher
- 2 Ports, die vergeben werden können

OAuth Tool registrieren

1. Auth Consumer registrieren (Dokumentation)[https://wiki.openstreetmap.org/wiki/DE:OAuth]

Backend

1. Download Code vom oben verlinkten Backend-Repository
2. Die .env Datei anpassen, dabei gibt es folgende Möglichkeiten:

Environment Variable	Django Setting	Developme
DJANGO_CACHES	CACHES (default)	locmem
DJANGO_DEBUG	DEBUG	True
DJANGO_SECRET_KEY	SECRET_KEY	CHANGEME!!!
DJANGO_SECURE_BROWSER_XSS_FILTER	SECURE_BROWSER_XSS_FILTER	n/a
DJANGO_SECURE_SSL_REDIRECT	SECURE_SSL_REDIRECT	n/a
DJANGO_SECURE_CONTENT_TYPE_NOSNIFF	SECURE_CONTENT_TYPE_NOSNIFF	n/a
DJANGO_SECURE_FRAME_DENY	SECURE_FRAME_DENY	n/a
DJANGO_SECURE_HSTS_INCLUDE_SUBDOMAINS	HSTS_INCLUDE_SUBDOMAINS	n/a
DJANGO_SESSION_COOKIE_HTTPONLY	SESSION_COOKIE_HTTPONLY	n/a
DJANGO_SESSION_COOKIE_SECURE	SESSION_COOKIE_SECURE	n/a
DJANGO_DEFAULT_FROM_EMAIL	DEFAULT_FROM_EMAIL	n/a
DJANGO_SERVER_EMAIL	SERVER_EMAIL	n/a
DJANGO_EMAIL_SUBJECT_PREFIX	EMAIL_SUBJECT_PREFIX	n/a
DJANGO_CHANGESETS_FILTER	CHANGESETS_FILTER	None
POSTGRES_USER	POSTGRES_USER	None
POSTGRES_PASSWORD	POSTGRES_PASSWORD	None
PGHOST	PGHOST	localhost

OAUTH_OSM_KEY	SOCIAL_AUTH_OPENSTREETMAP_KEY	None
OAUTH_OSM_SECRET	SOCIAL_AUTH_OPENSTREETMAP_SECRET	None
OSM_VIZ_TOOL_LINK	VIZ_TOOL_LINK	https://osmlab.gith map/# https://osmlab.gith map/#
DJANGO_ANON_USER_THROTTLE_RATE	ANON_USER_THROTTLE_RATE	None
DJANGO_COMMON_USER_THROTTLE_RATE	COMMON_USER_THROTTLE_RATE	None
DJANGO_NON_STAFF_USER_THROTTLE_RATE	NON_STAFF_USER_THROTTLE_RATE	3/min
OAUTH_REDIRECT_URI	OAUTH_REDIRECT_URI	http://localhost:80 landing.html
OSMCHA_FRONTEND_VERSION	OSMCHA_FRONTEND_VERSION	oh-pages
DJANGO_ENABLE_CHANGESET_COMMENTS	ENABLE_POST_CHANGESET_COMMENTS	False
DJANGO_OSM_COMMENTS_API_KEY	OSM_COMMENTS_API_KEY	''

Dabei zwingend sind folgende Variablen:

- POSTGRES_USER
- POSTGRES_PASSWORD
- DJANGO_SETTINGS_MODULE
- OAUTH_OSM_KEY
- OAUTH_OSM_SECRET
- OAUTH_REDIRECT_URI

3. `docker-compose up --build` ausführen

Frontend

1. Download Code vom oben verlinkten Frontend-Repository
2. .env richtig konfigurieren:

Variable	Wert
REACT_APP_HOME_URL	URL zu ihrer Frontendpage "wird im docker run angegeben"
REACT_APP_API	URL zu ihrem Backend

3. docker build `docker build --tag targetedmonitoring:latest ~/sa-monitoring-osm-frontend`
4. docker run `docker run -p yourPort:80 -d --name frontendcontainer targetedmonitoring:latest`

Run ohne Docker

OAuth Tool registrieren

1. Auth Consumer registrieren (Dokumentation)[https://wiki.openstreetmap.org/wiki/DE:OAuth]

Backend

Folgendes muss installiert sein:

- pip
- virtualenv
- PostgreSQL (mit Benutzer und Passwort wie in env festgelegt)

1. Download Code vom oben verlinkten Backend-Repository
2. Die .env Datei anpassen, dabei gibt es folgende Möglichkeiten:

Environment Variable	Django Setting	Developme
DJANGO_CACHES	CACHES (default)	locmem
DJANGO_DEBUG	DEBUG	True

DJANGO_SECRET_KEY	SECRET_KEY	CHANGEME!!!
DJANGO_SECURE_BROWSER_XSS_FILTER	SECURE_BROWSER_XSS_FILTER	n/a
DJANGO_SECURE_SSL_REDIRECT	SECURE_SSL_REDIRECT	n/a
DJANGO_SECURE_CONTENT_TYPE_NOSNIFF	SECURE_CONTENT_TYPE_NOSNIFF	n/a
DJANGO_SECURE_FRAME_DENY	SECURE_FRAME_DENY	n/a
DJANGO_SECURE_HSTS_INCLUDE_SUBDOMAINS	HSTS_INCLUDE_SUBDOMAINS	n/a
DJANGO_SESSION_COOKIE_HTTPONLY	SESSION_COOKIE_HTTPONLY	n/a
DJANGO_SESSION_COOKIE_SECURE	SESSION_COOKIE_SECURE	n/a
DJANGO_DEFAULT_FROM_EMAIL	DEFAULT_FROM_EMAIL	n/a
DJANGO_SERVER_EMAIL	SERVER_EMAIL	n/a
DJANGO_EMAIL_SUBJECT_PREFIX	EMAIL_SUBJECT_PREFIX	n/a
DJANGO_CHANGESETS_FILTER	CHANGESETS_FILTER	None
POSTGRES_USER	POSTGRES_USER	None
POSTGRES_PASSWORD	POSTGRES_PASSWORD	None
PGHOST	PGHOST	localhost
OAUTH_OSM_KEY	SOCIAL_AUTH_OPENSTREETMAP_KEY	None
OAUTH_OSM_SECRET	SOCIAL_AUTH_OPENSTREETMAP_SECRET	None
OSM_VIZ_TOOL_LINK	VIZ_TOOL_LINK	https://osmlab.gith map/# https://osmlab.gith map/#
DJANGO_ANON_USER_THROTTLE_RATE	ANON_USER_THROTTLE_RATE	None
DJANGO_COMMON_USER_THROTTLE_RATE	COMMON_USER_THROTTLE_RATE	None
DJANGO_NON_STAFF_USER_THROTTLE_RATE	NON_STAFF_USER_THROTTLE_RATE	3/min
OAUTH_REDIRECT_URI	OAUTH_REDIRECT_URI	http://localhost:80 landing.html
OSMCHA_FRONTEND_VERSION	OSMCHA_FRONTEND_VERSION	oh-pages
DJANGO_ENABLE_CHANGESET_COMMENTS	ENABLE_POST_CHANGESET_COMMENTS	False
DJANGO_OSM_COMMENTS_API_KEY	OSM_COMMENTS_API_KEY	''

Dabei zwingend sind folgende Variablen:

- POSTGRES_USER
- POSTGRES_PASSWORD
- DJANGO_SETTINGS_MODULE
- OAUTH_OSM_KEY
- OAUTH_OSM_SECRET
- OAUTH_REDIRECT_URI

3. Installiere einige Packages auf dem System `sudo ./install_os_dependencies.sh install`
4. Aktivieren der virtualenv
    1. `source venv/bin/activate`
    2. `source settings.env`
5. Installiere die lokalen Requirements
   `pip install -r requirements/local.txt`
6. Datenbank erstellen
   `createdb osmcha`
7. Datenbank abfüllen
   `python manage.py migrate`

8. Server starten
   `python manage.py runserver_plus`

Frontend

Folgendes muss installiert sein:

- Node.js

1. Download Code vom oben verlinkten Frontend-Repository
2. Installiere die benötigten Packages. Dazu soll der Befehl `npm i` im Root-Ordner ausgeführt werden.
3. .env richtig konfigurieren

Variable	Wert
REACT_APP_HOME_URL	http://localhost:3000/
REACT_APP_API	URL zu ihrem Backend

4. Webapplikation im Developmentmodus starten `npm start`
   Starten der App im Produktionsmodus `npm run build`
   Unter http://localhost:3000 können Sie die App ansehen. Die Seite wird neu geladen, wenn sie Änderungen vornehmen

Login ohne Frontend

- Post-Request an `<your_base_url>/api/v1/social-auth/` dann erhalten Sie die folgenden Keys: `oauth_token`, `oauth_token_secret`
- Mit dem `oauth_token` starten Sie eine Anfrage zu `https://www.openstreetmap.org/oauth/authorize?oauth_token=<oauth_token>`
- Dort muss sich der Benutzer einloggen und den Zugriff genehmigen. Danach wird man zur im env definierten Rücksprung-Adresse geführt und erhält ein `oauth_verifier` per URL-String.
- Noch einmal senden Sie einen Post-Request an `<your_base_url>/api/v1/social-auth/`, aber dieses Mal mit den `oauth_token`, `oauth_token_secret` und `oauth_verifier` im Body.
- Im Anschluss erhalten Sie ein Token, welches Sie dann immer im Header angeben müssen: `Authorization: Token 9944b09199c62bcf9418ad846dd0e4bbdfc6ee4b`

8 Benutzerhandbuch

Benutzer-Handbuch Targeted Monitoring Tool

Einleitung

Das Targeted Monitoring Tool wurde im Auftrag von Schutz & Rettung Zürich (SRZ) entwickelt, welche für sicherheitsrelevante Zwecke OpenStreetMap verwendet. Trotz des Vertrauens in die OSM-Community, braucht es für heikle Kartenelemente eine Kontrolle, d.h. ein Monitoring der kritischen Datenobjekte. Da die Überwachung von Changesets über Objekt-Tags bei diversen bestehenden Monitoring-Tools nicht gegeben ist, kommt das Targeted Monitoring Tool zum Einsatz, welches Changesets nach Objekt-Tags filtert. So können ungewollte Änderungen erkannt und korrigiert werden.

- Installationsanleitung: Hier
- Frontend-Code: https://gitlab.dev.ifs.hsr.ch/se-na/sa-monitoring-osm-frontend
- Backend-Code: https://gitlab.dev.ifs.hsr.ch/DNA/sa-monitoring-osm-backend

Anwendung

Login

Beim erstmaligen Aufruf der Webseite erscheint die Login-Seite. Mit einem Klick auf den Login-Knopf werden Sie zu OpenStreetMap weitergeleitet. Dort können Sie sich mit Ihrem bestehenden OSM-Account anmelden, wenn Sie nicht bereits angemeldet sind. Es gibt nur die Möglichkeit, sich über OpenStreetMap anzumelden. Sollten Sie also noch keinen Account besitzen, müssen Sie diesen erst erstellen. Haben Sie sich mit Ihrem OSM-Account angemeldet, müssen Sie dem Targeted Monitoring Tool mit dem Klick auf "Zugriff gewähren" folgendes erlauben:

- Deine Benutzereinstellungen zu lesen
- Blogeinträge und Kommentare zu schreiben und Freunde einzutragen
- Hinweise zu geben/Fehler zu melden

Home

Nach dem Login landen Sie automatisch auf der Home-Seite (Startseite). Diese Seite zeigt Ihnen ihre eigenen, sowie die Ihnen freigeschalteten Filter an. Sollten Sie noch keine Filter haben, ist die Liste leer. Im Bereich "Meine Filter" können Sie sich über den Button "+" einen neuen Filter anlegen. Wenn Sie bereits einen oder mehrere Filter angelegt haben, können Sie mit einem Klick auf den gewünschten Filter in die Ansicht wechseln, welche die zum aktuellen Filter passenden Changesets anzeigt. Zudem besteht die Möglichkeit, die eigenen Fiter zu bearbeiten oder zu löschen. Die freigegebenen Filter können Sie nicht bearbeiten oder löschen, aber dafür die Filter-Definition anschauen.

Home Screen

Filter definieren

Einen neuen Filter können Sie über den Knopf "+" auf dem Home anlegen. Eigene Filter können Sie mit einem Klick auf das Stift-Symbol neben dem entsprechenden Filter bearbeiten und mit einem Klick auf das Mülltonnen-Symol löschen (Der Löschvorgang muss in einem seperaten Popup bestätigt werden).

Nach dem Klick auf das +-Symbol erscheint eine neue Seite, in der Sie den neuen Filter erfassen können. Die Filterdefinition ist in 3 Bereiche unterteilt:

Filterbezeichnung

Die Filter-Bezeichnung besteht aus einem Namen. Der Name muss eindeutig sein, Sie können also nicht schon einen Filter mit diesem Namen besitzen. Um einen Filter zu erstellen, muss dieses Feld zwingend ausgefüllt werden. Der Name des Filters dient im Home als Filter-Bezeichnung.

Filtermöglichkeiten

Die Filter-Möglichkeiten besteht aus zwei Bereichen. Sie können nach einem oder mehreren Tags in einem Kanton mit einem

Buffer oder einem oder mehreren OSM-Benutzern suchen.

Für einen Filter nach Tags, müssen Sie die Felder Kanton und Tags ausfüllen und können zusätzlich einen Buffer angeben. Die Tags werden in der Form 'key=value' im entsprechenden Eingabefeld angegeben und mit der Enter-Taste bestätigt. Es können mehrere Tags angegeben werden. Bei mehreren Tags handelt es sich um eine Oder-Verknüpfung, die Suchergebnisse liefern also Changesets, die mindestens über einem Objekt mit einem der angegebenen Tags liegt.

Beispiel: amenity=hospital amenity=fire_station -> Changesets die ein Spital oder eine Feuerstation überlagern, werden angezeigt.

Da die Tags mit Hilfe von Overpass in eine Referenzfläche umgewandelt werden, müssen Sie die Region eingrenzen. Die Region können Sie im Dropdown mit der Überschrift "Kanton" auswählen. Der Buffer ermöglicht es Ihnen, auch Changesets zu erhalten, deren Ausdehnung in die Nähe der zu überwachenden Objekte fällt.

Beispiel: Spital wird beobachtet, aber Änderungen an der Zufahrt sollten auch bemerkt werden.

Für einen Filter nach Benutzer können Sie den OSM-Benutzernamen des zu überwachenden Nutzers im Feld mit der Überschrift "Benutzer" eingeben und mit der Enter-Taste bestätigen. Beim Filter werden dann kantonsunabhängig alle Changesets dieser Person angezeigt. Es können mehrere Personen gleichzeitig überwacht werden.

Werden Objekt-Tags, sowie OSM-Benutzer eingegeben, werden beim Filtern nur Changesets zurückgegeben, die zum Tag passen UND von einem der angegebenen Benutzern stammen.

Filter-Freigabe

Im Bereich Filter-Freigabe können Sie Mitarbeiter hinzufügen. Geben Sie dazu den OSM-Namen des Mitarbeiters in das Feld mit der Überschrift "Mitarbeiter" ein und drücken die Enter-Taste. Die Changesets der hinzugefügten Mitarbeiter werden beim Filtern nicht angezeigt, ausser Sie werden zusätzlich bei den Filter-Möglichkeiten unter Benutzer eingetragen.

neuer Filter

Filter bearbeiten

Filter ansehen

Freigegebenen Filter können Sie zwar nicht bearbeiten, aber mit einem Klick auf das Augen-Symbol die Filter-Definition einsehen.

Filter andehen

Changesets überwachen

Mit dem Klick auf den Filter werden die Changesets gefiltert dargestellt. Die Filter zeigen passende Changesets seit dem Aufsetzen des Tools an. Changesets mit dem Status "abgeschlossen" werden nicht mehr angezeigt.

Mit einem Klick auf die einzelnen Datensätze, werden rechts im Fenster auf der Karte die passenden Änderungen eingezeichnet.

Changesets abarbeiten

Wenn Sie mit dem Überprüfen eines Changesets beginnen, ist es zu empfehlen, zuerst die Statusmeldungen zu aktualisieren. Dazu müssen Sie auf den Button "Letzte Status Aktualisierung" klicken. Changesets mit dem Status "neu" sind noch zu überprüfen. Solche "in Bearbeitung" werden schon von einem ihrer Mitarbeiter überprüft. Bevor Sie also ein Changeset überprüfen, sollten Sie zuerst den Status "in Bearbeitung" auswählen, damit ihre Mitarbeiter darüber Bescheid wissen.

Um die Zeit effizient zu nutzen, können Sie die Changesets auch nach Gewichtung abarbeiten. Dabei werden Changesets die Löschungen beinhalten als "Hoch" klassiert und solche mit Bearbeitungen als "Mittel". Wurden neue Objekte hinzufügt, zeigt die Gewichtung "Gering".

Wenn Sie sich mit einem Changeset befassen und den Status auf "in Bearbeitung" gesetzt haben, können Sie bei einem Klick auf die Changeset-Nummer (blau) in einem neuen Fenster in OpenStreetMap das Objekt bearbeiten. Wenn es nichts anzupassen gibt, oder die Anpassungen vollendet sind, können Sie dem Changeset den Status "abgeschlossen" geben. Solange Sie sich in der Ansicht befinden, bleiben auch die abgeschlossen Changesets sichtbar. Haben Sie also versehentlich ein Changeset auf "abgeschlossen" gesetzt, können Sie das noch ändern. Beim Neuladen, werden die Abgeschlossenen aber nicht mehr dargestellt.

Overview Screen

9 Glossar

Glossary

Bbox Ausdehnung eines Changesets. 13

Changesets Ein Changeset enthält Änderungsdaten eines Benutzers an einem oder mehreren Kartenelementen.. 1

Copyleft Copyleft Lizenzen schreiben vor, dass der Source-Code nur dann weitergegeben werden darf, wenn dieser wieder mit der selben Lizenz versehen wird.. 14

Copyright Urheberrechtlich geschützt. 14

Datenkuratoren Datenpfleger, Supervisor. 2

dockerisieren Wenn eine Anwendung containerisiert, also Docker-fähig gemacht wird.. 3

Mapper Ein OSM-Benutzer, der an der Karte mitarbeitet.. 15

Notfalltelefonisten Ein Notfalltelefonist bei SRZ nimmt Notrufe entgegen und koordiniert den Rettungseinsatz. Er ist deshalb auf eine korrekte und detailreiche OSM-Karte angewiesen.. 2

Objekt-Tags Tags in Form von key=value eines Objektes in OSM. 1

OpenStreetMap Stellt kostenlose Kartendaten (Open-Source) zur Verfügung.. 1

Punkt-Objekte Im OSM Node genannt. Das ist ein Punkt auf der Karte.. 26

Y-Approach Architekturentscheidungs-Dokumentations-Variante. 17

10 Quellenverzeichnis

[1] GNU General Public License, https://de.wikipedia.org/wiki/GNU_General_Public_License#Copyleft-Prinzip, letzter Zugriff am 18.12.2020
[2] Open Source Software: Lizenzkategorien, https://www.rentschpartner.ch/ict-law/open-source-software/lizenzkategorien, letzter Zugriff am 18.12.2020
[3] Changeset, https://wiki.openstreetmap.org/wiki/Changeset, letzter Zugriff am 18.12.2020
[4] Beginners Guide 1.3, https://wiki.openstreetmap.org/wiki/Beginners_Guide_1.3, letzter Zugriff am 18.12.2020
[5] The C4 model for visualising software architecture, https://c4model.com/, letzter Zugriff am 18.12.2020
[6] Architectural Decisions — The Making Of, https://ozimmer.ch/practices/2020/04/27/ArchitectureDecisionMaking.html#y-statements-and-other-templates, letzter Zugriff am 18.12.2020

11 Abbildungsverzeichnis

Abbildungsverzeichnis

12 Tabellenverzeichnis

List of Tables