Fachhochschule Dortmund
University of Applied Sciences and Arts

IKT
Institut für Kommunikationstechnik

Bachelorstudiengang Informations- und Kommunikationstechnik

- *IuK Projekt* -

„Aufbau eines Geofence Systems für das Cyberphysicalsystem“

„Development a Geofence system for the cyber physical system“

Tutor: M. Eng. Markus Kuller

M. Eng. Nursi Karaoglan

Autor: Anh Tu Nguyen

Matrikelnummer: 7096492

Vorgelegt am: 23.11.2020

Eigenständigkeitserklärung

Hiermit versichere ich an Eides statt, dass die von mir vorgelegte Prüfungsleistung selbständig und ohne unzulässige fremde Hilfe erstellt worden ist. Alle verwendeten Quellen sind in der Arbeit so aufgeführt, dass Art und Umfang der Verwendung nachvollziehbar sind.

_____

Ort, Datum Unterschrift

Zusammenfassung

Diese Projektarbeit wird das Geofence System für das Cyberphysical System erläutert. Außerdem wurde eine Plattform mit dem Namen Tile38 eingeführt. Um das Geofence System zu demonstrieren, die Tile38 Plattform wurde als den Geofence Server verwendet. In Rahmen dieser Projektarbeit wurde eine Java Applikation und eine Softwareoberfläche auf Node-RED Plattform entwickelt.

Abstract

This project work explains the Geofence system for cyber physical system. A platform called Tile38 was also introduced. To demonstrate the Geofence system, the Tile38 platform was used as the Geofence server. In this work, a Java application as well as a software interface in Node-RED were developed.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

HTTP	Hypertext-Übertragungsprotokoll
JSON	JavaScript Objekt Notation
KeyID	Schlüsselidentifikation des Geofence-Objekts
Field Name	Feldname des Geofence-Objekts
WWWW	Weltweites Netz
OSI-Modell	Offenes Systemverbindungsmodell
URL	Einheitlicher Ressourcenzeiger

1 Einleitung

Heutzutage mit dem Bedürfnis nach Kommunikation und der Entwicklung von Wissenschaft und Technologie der Menschheit werden ständig Technologien geboren, um dieses Bedürfnis zu befriedigen. Eines davon ist das Internet der Dinge (englisch: Internet of Things, IoT) eine Technologie, die in letzter Zeit sehr bekannt ist und mit vielen Anwendung in verschiedenen Ersatzgebieten wie Verbraucheranwendungen, organisatorische Anwendungen, industrielle Anwendungen, Infrastrukturanwendungen, Militärische Anwendungen, Produktdigitalisierung bietet.

Ziel des Internets der Dinge es ist, alle relevante Informationen aus der realen Welt automatisch aufbewahren, miteinander zu zusammenfassen und im Netzwerk publizieren. Darüber hinaus stellen viele reale Dinge die eigenen Zustandsinformationen für die Weiterverarbeitung im Netzwerk zur Verfügung. Ein Beispiel ist, dass der Standort eines Autos und die Menge des verfügbaren Parkplatzes dem Benutzer über eine mobile Anwendung kontinuierlich aktualisiert werden. Dadurch können Benutzer selbst den geeigneten Parkplatz auswählen und Verkehrsstaus reduzieren. Eine mögliche Lösung für dieses Problem ist Geofence. Ein Geofence ist eine virtuelle Grenze, die erkennen kann, wann ein Objekt den Bereich betritt oder verlässt. Diese Grenze kann ein Radius oder ein beliebiges Suchbereichsformat sein, z. B. ein Begrenzungsrahmen, ein GeoJSON-Objekt usw.

Der Geofence kann in vielfältigen Einsatzgebieten von Parkplatzbelegung, Zutrittskontrolle, E-Auto laden, Paketverfolgung bis Verkehrsvernetzung, Smart City eingesetzt werden. Das Ziel dieser Projektarbeit ist Aufbau eines Geofence Systems für das Cyberphysicalsystem mit Platzform Tile38, Java Standard, Node-RED.

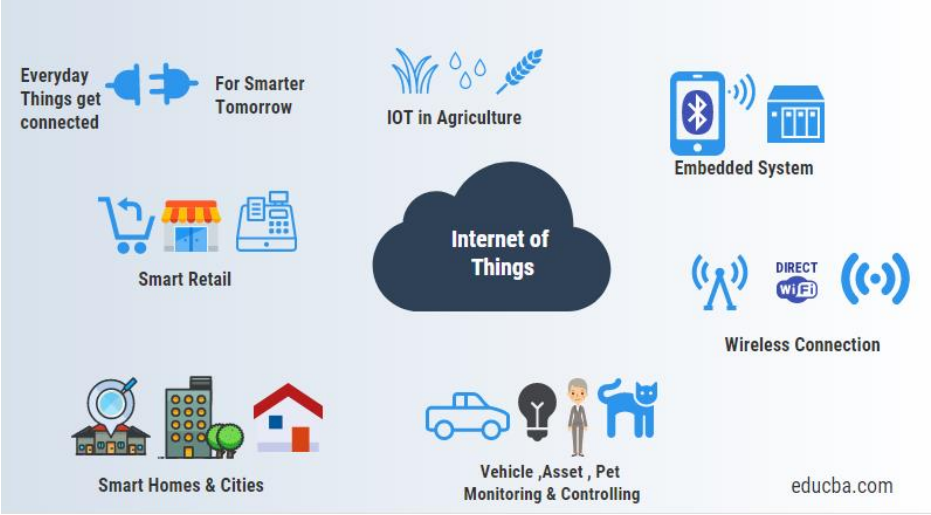

Abbildung 1 Anwendungsbereich von Internet der Dinge. [1]

1.1 Einführung in Internet der Dinge:

Laut der IoT Definition von einem Forschungsartikel:

> *„Internet of things (IOT) is a network of physical objects. The internet is not only a network of computers, but it has evolved into a network of device of all type and sizes , vehicles, smart phones, home appliances, toys, cameras, medical instruments and industrial systems, animals, people, buildings, all connected ,all communicating & sharing information based on stipulated protocols in order to achieve smart reorganizations, positioning, tracing, safe & control & even personal real time online monitoring , online upgrade, process control & administration."* [2]
>
> *„Das Internet der Dinge (IOT) ist ein Netzwerk physischer Objekte. Das Internet ist nicht nur ein Netzwerk von Computern, sondern hat sich zu einem Netzwerk von Geräten aller Art und Größe, Fahrzeugen, Smartphones, Haushaltsgeräten, Spielzeug, Kameras, medizinischen Instrumenten und industriellen Systemen, Tieren, Menschen, Gebäuden usw. entwickelt verbunden, alle kommunizieren und teilen Informationen basierend auf festgelegten Protokollen, um intelligente Reorganisationen, Positionierung, Rückverfolgung, Sicherheit und Kontrolle sowie persönliche Online-Überwachung in Echtzeit, Online-Upgrade, Prozesskontrolle und -verwaltung zu erreichen."*

Im einfachsten Sinne könnte das Internet der Dinge (englisch: Internet of Things, IoT), das digital verbundene Universum alltäglicher physischer Geräte ist, beschrieben werden. Diese Geräte sind mit Sensoren, Aktoren, Internetverbindungen und anderen Hardware ausgestattet, die die Kommunikation und Steuerung über das Web ermöglichen.

1.2 Anwendung von Internet der Dinge:

Wie bereits erwähnt, sind die Anwendungen des Internets der Dinge vielfältig und auf viele verschiedene Anwendungsgebiete verteilt. Einige Anwendungen sind Medizin-, Gesundheits-, Transport- und Automobilanwendungen sowie Anwendungen für die Hausautomation. Diese Anwendung werden immer beliebter und werden zu einem wesentlichen Bestandteil des modernen menschlichen Lebens.

Mit immer höheren menschlichen Anforderungen müssen Anträge noch mehr ausgefüllt werden. Eine der Grundvoraussetzungen ist Bequemlichkeit, die Einsparung von menschlichen Anstrengungen und Zeit. Wenn man beispielsweise nach Hause geht, generiert das Mobiltelefon automatisch einen QR-Code zum Öffnen der Tür und das Frontlicht wird automatisch eingeschaltet oder wenn man zur Arbeit geht, wählt die mobile App den nächsten

Parkplatz, der ausreichende Parkplätze gibt, aus. Ein weiteres Beispiel ist die derzeitige Covid-Epidemie. In den meisten Fällen hat die erste Reihe von Fällen, die mit dem Coronavirus infiziert sind, eine gewisse Reisegeschichte. Aus diesem Grund ist es äußerst wichtig, die Standorte und Bewegungen von Menschen zu verfolgen und ihre Gesundheit rechtzeitig zu testen. Solche Verfolgungsbemühungen sind oft mit Rückschlägen konfrontiert, da viele Menschen trotz der Sensibilisierung für die schwerwiegenden Auswirkungen des Virus einfach die Zusammenarbeit ablehnen und sogar versuchen, den Tests zu entkommen. Während die Verfolgung dieser Personen durch die Verwendung eines regelmäßigen Wachsamkeits- und Überwachungssystems schwierig sein kann, können mobile Tracking-Apps dabei eine wirklich nützliche Rolle spielen. Mobile Anwendungen können die registrierten Personen mithilfe der GPS-Tracker problemlos verfolgen, sodass die Behörden über einzelne Person benachrichtigen können.

Anhand der obigen Beispiele können wir den Komfort und die Notwendigkeit von IoT-Anwendungen erkennen und gleichzeitig die Notwendigkeit aufzeigen, eine Karte mit Standorten und Gebieten zu erstellen.

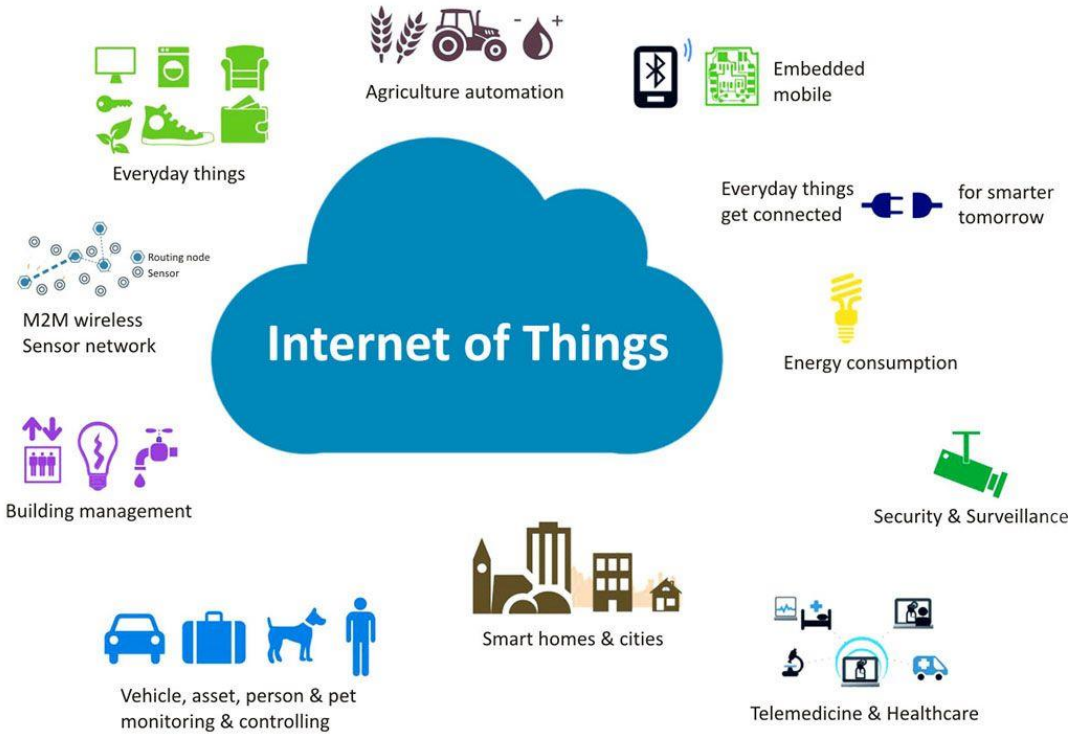

Abbildung 2 Anwendungen von Internet der Dinge. [3]

2 Geofencing:

2.1 Einführung:

Geofencing ist ein standortbasierter Dienst, bei dem eine Applikation oder eine andere Software Globales Positionsbestimmungssystem (GPS), Radiofrequenz-Identifikation (RFID), WLAN oder Mobilfunkdaten verwendet, um eine vorprogrammierte Aktion auszulösen.

Geofencing wird in einer Vielzahl von Bereichen eingesetzt, um Verwaltungsaufgaben zu verwalten, das Marketing zu ergänzen oder sicherheitsrelevante Aspekte zu überprüfen. Im Prinzip funktionieren solche Systeme wie Positionierungs- und Navigationssysteme. Der Unterschied liegt in den Grenzkoordinaten, die einen bestimmten Bereich in Form eines Rechtecks oder Kreises einschließen und als Geofilter fungieren. Diese virtuelle Positionierung ist aus der Fahrzeugposition per GPS bekannt. Durch die Unterscheidung zwischen dem Inneren und dem Äußeren eines genau definierten Bereichs ist es möglich, Aktionen beim Betreten oder Verlassen dieses definierten Bereichs auszulösen.

Abhängig davon, wie ein Geofence konfiguriert ist, kann der Push-Benachrichtigungen für Mobilgeräte auslösen, Textnachrichten oder Warnungen auslösen, gezielte Werbung in sozialen Medien senden, die Verfolgung von Fahrzeugen ermöglichen, bestimmte Technologien deaktivieren oder standortbezogene Marketingdaten liefern. Einige Geofence sind eingerichtet, dass die Aktivitäten in sicheren Bereichen überwacht werden, sodass das Management Warnungen sehen kann, wenn jemand einen bestimmten Bereich betritt oder verlässt.

2.2 Funktionsprinzip:

Um Geofence nutzen zu können, muss ein Administrator oder Entwickler zunächst eine virtuelle Grenze um einen bestimmten Ort in einer GPS- oder RFID-fähigen Software festlegen. Dies kann einfach wie ein Kreis sein, der 100 Meter um einen Ort in Google Maps gezogen wird, wie dies mit Hilfe von APIs bei der Entwicklung einer mobilen App angegeben wurde. Dieser virtuelle Geofence löst dann eine Rückmeldung aus, wenn ein autorisiertes Gerät diesen Bereich betritt oder verlässt, wie vom Administrator oder Entwickler angegeben wurden.

Ein Geofence wird am häufigsten im Code einer mobilen Anwendung definiert, insbesondere da Benutzer sich für Ortungsdienste anmelden müssen, damit der Geofence funktioniert.

Wenn man zu einer Messe geht, kann man die Informationen über die Veranstaltung durch eine App bekommen. Oder ein Einzelhändler zeichnet einen Geofence, um Ihre Angebote für Kunden auszulösen, wenn die Kunden in der Nähe von Ihren Verkaufsstellen sind.

Ein Geofence kann auch von Endbenutzern mit Hilfe von Geofence Funktionen in ihren mobilen Apps eingerichtet werden. Mit diesen Apps, z. B. Erinnerungen-App, kann man eine Adresse oder einen Ort auswählen, an dem man eine bestimmte Warnung oder Push-Benachrichtigung auslösen will. Dies wird als "Wenn dies, dann das" -Befehl bezeichnet, bei dem eine App so programmiert ist, dass sie eine Aktion auslöst, die auf einer anderen Aktion basiert. Zum Beispiel: "Wenn ich fünf Fuß von meiner Haustür entfernt bin, schalte das Licht ein."

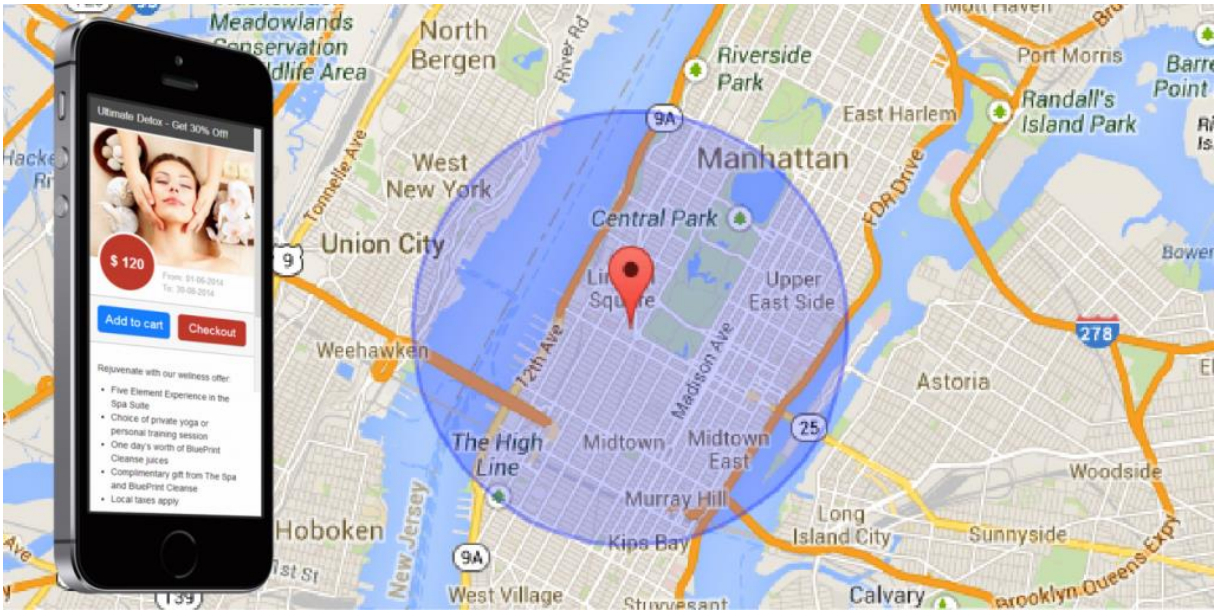

Abbildung 3 Benachrichtigung von App mit Geofence. [4]

2.3 Anwendungsfall:

2.3.1 Parkplatzbelegung:

Um einen guten Parkplatz für ein Auto zu finden, ist ein praktisches Problem, mit dem Millionen von Fahrern täglich konfrontiert sind. Auf persönlicher Ebene geht es um Angst und Unsicherheit, und auf der Ebene der Gesellschaft verschwendet es begrenzte Ressourcen - Zeit, Straßenraum und Kraftstoff -, wenn Fahrer auf der Suche nach freien Parkplätzen zirkulieren. Diese Probleme könnten gelindert werden, wenn die Fahrer vorab über freie Parkplätze informiert würden. Die Informationen können mit einem Geofence Systemen gesammelt werden, der den Reservierungsstatus von Parkplätzen verfolgen.

Abbildung 4 Intelligente Parkplatzbelegung mit Geofence. [5]

Zum Beispiel fährt ein Benutzer mit seinem Auto zur Arbeit und die aktuelle Standortdaten von Benutzer wird durch sein Handy in einem vorgebbaren Zeitintervall übermittelt. Danach wird die aktuelle Standortdaten mit dem Geofence System abgleichen. Wenn der Geofence erkannt ist, fragt der die Anzahl freier Parkplätze ab. Wenn es noch freie Parkplätze gibt, dann benachrichtigt sein Handy die freie Parkplätze. Hernach sein Auto wird geparkt und die Sensoren erkennen die Belegung von seinem Auto.

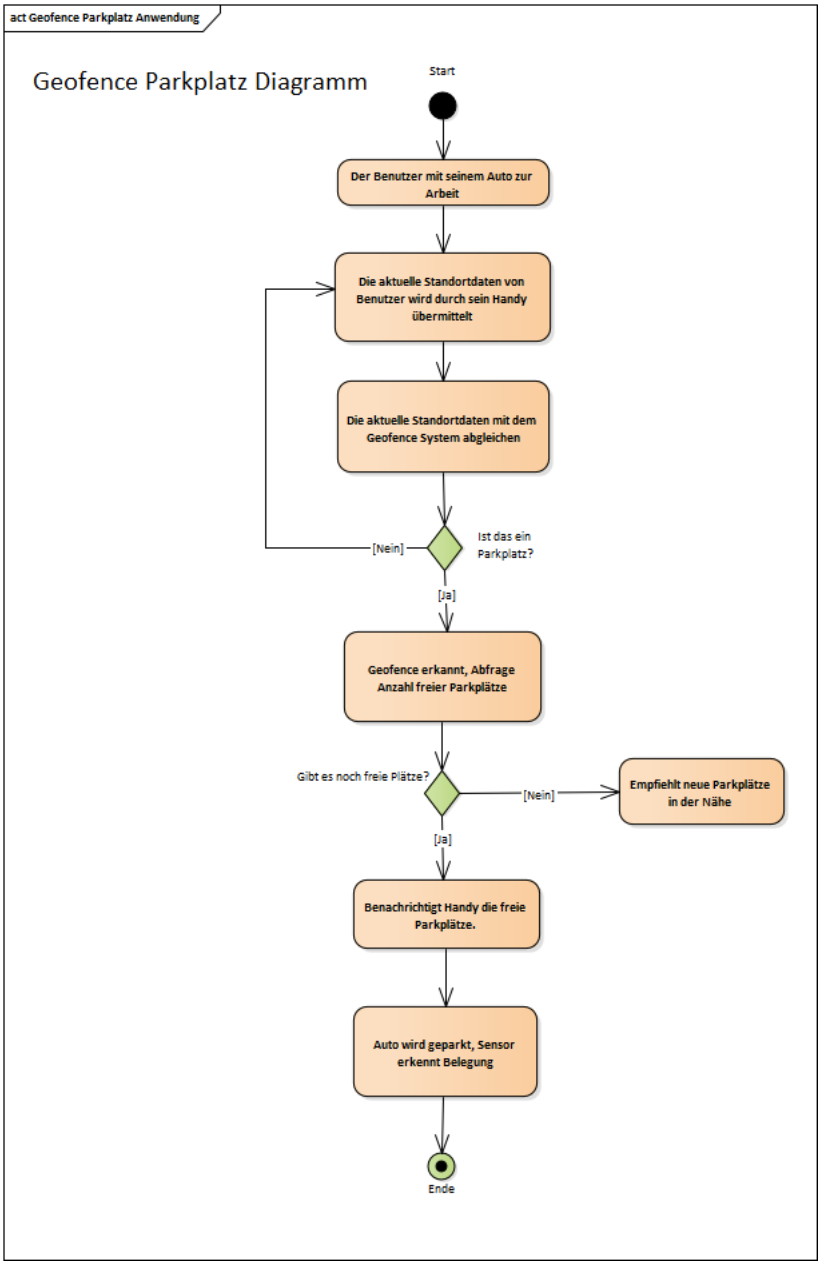

Abbildung 5 Diagramm von Parkplatzbelegung mit Geofence.

2.3.2 Zutrittskontrolle:

Außerdem kann Geofence nicht nur für Parkplatzbelegung sondern auch für Zutrittskontrolle eingesetzt werden. Das Szenario könnte sein, dass der Benutzer nach dem Parken des Autos auf dem Parkplatz in das Bürogebäude geht. Wenn der Benutzer an der Vorderseite des Büros ankommt, das sich in einem zuvor festgelegten Geofence befindet, sendet das Handy eine Anfrage an das System, um einen QR-Code zu generieren. Dieser QR-Code wird dann per E-Mail, Telegramm oder WhatsApp an diese Person gesendet. Mit dem empfangenen QR-Code kann die Tür 15 Minuten lang geöffnet und dann automatisch gelöscht werden.

Abbildung 6 Zutrittskontrolle mit QR-Code. [6]

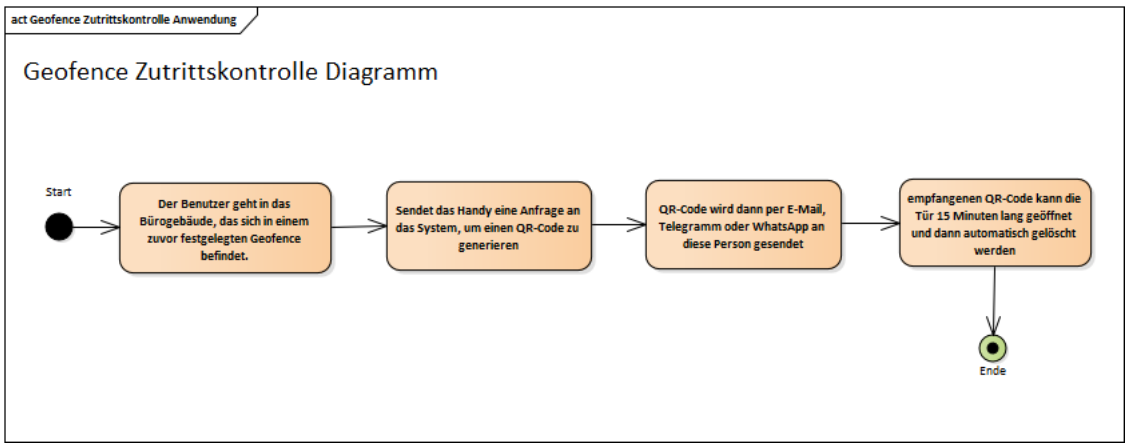

Abbildung 7 Diagramm von Zutrittskontrolle mit Geofence.

2.3.3 E-Auto laden:

Eine weitere Anwendung von Geofence ist E-Auto Aufladung. Der Einsatz von Elektroautos ist der Trend der Zukunft. Laut Statistik aus dem Jahr 2020 sind weltweit rund 5,6 Millionen Elektroautos im Einsatz, ein Anstieg von mehr als 64 Prozent gegenüber 2018. Das bedeutet, dass die Nachfrage nach dem Aufladen von Elektroautos in Zukunft steigen wird, besonders wenn das Auto geparkt ist, weil es dem Besitzer viel Zeit spart.

Ein mögliches Szenario besteht darin, dass der Benutzer den Autolademodus aktiviert hat und das Fahrzeug auf einem aufgeladenen Parkplatz geparkt ist. Da sich der Parkplatz in einem Geofence befindet, analysiert das Auto nach dem Parken des Autos automatisch die Batteriemenge, die verbleibende Nutzungsdauer und die geschätzte Ladezeit, um zu

entscheiden, ob die Batterie aufgeladen werden soll oder nicht. Wenn dies der Fall ist, wechselt das Auto in den Lademodus, schaltet die Batterieladeanzeige ein und zeigt Informationen zur Batterie sowie zur Ladezeit in der App für den Benutzer an.

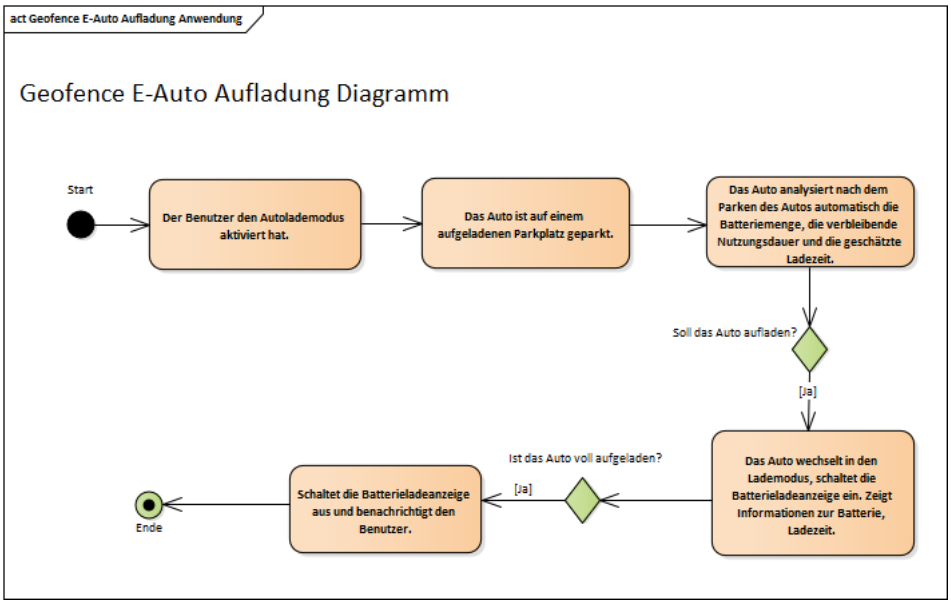

Abbildung 8 Diagramm von E-Auto Aufladung mit Geofence.

3 Tile38 Plattform:

Laut der Tile38 Definition von Tile38 Webseiten und Github von Tile38 Projekt:

> *„Tile38 is an open source (MIT licensed), in-memory geolocation data store, spatial index, and realtime geofence. It supports a variety of object types including lat/lon points, bounding boxes, XYZ tiles, Geohashes, and GeoJSON.“* [7]
>
> *„Tile38 ist ein Open-Source, speicherinterner Geolokalisierungsdatenspeicher, räumlicher Index und Echtzeit-Geofence. Es unterstützt eine Vielzahl von Objekttypen, einschließlich Breiten- und Längengrad Punkten, Begrenzungsrahmen, XYZ-Kacheln, Geohashes und GeoJSON.“*

Kurz gesagt ist Tile38 eine Open Source Plattform zum Erstellen, Speichern und Verwalten von Geofence-Objekten. Diese Geofence-Objekte können ein Ort mit Breiten- und Längengrad sein, können ein Bereich jeder Polygonform mit Breiten- und Längengradscheitelpunkten sein. Diese Geofence-Objekte können in vielen verschiedenen Datentypen gespeichert werden, am häufigsten in GeoJSON.

Und was ist GeoJSON? Nochmals stellt die Definition von GeoJSON in der Tile38 Webseiten:

> *„GeoJSON is an industry standard format for representing a variety of object types including a point, multipoint, linestring, multilinestring, polygon, multipolygon, geometrycollection, feature, and featurecollection.“* [7]
>
> *„GeoJSON ist ein Industriestandardformat für die Darstellung einer Vielzahl von Objekttypen, einschließlich Punkt, Mehrpunkt, Linienstreifen, Mehrfachstreifen, Polygon, Multipolygon, Geometrieerfassung, Merkmal und Merkmalssammlung.“*

Zusammenfassen ist GeoJSON ein Datenstruktur, die auf JavaScript Objekt Notation (JSON) basiert ist, um die Geofence-Objekte wie Punkte oder Polygon und die Eigenschaften von diesen Objekten zu speichern. Ein Beispiel von GeoJSON ist:

```
{
  "type": "Feature",
  "geometry": {
    "type": "Point",
    "coordinates": [125.6, 10.1]
  },
  "properties": {
    "name": "Dinagat Islands"
  }
}
```

Ein erwähnenswerter Punkt ist, dass in GeoJSON Breiten- und Längengrade in der Reihenfolge Längengrad und Breitengrad angeordnet sind. Zum Beispiel beträgt in den obigen GeoJSON-Daten der Längengrad 125,6 und der Breitengrad 10,1. Bei Tile38 Platform ist außerdem zu beachten, dass ein Geofence-Objekt in Tile38 mit einer eindeutigen KeyID und einem Field Name definiert ist.

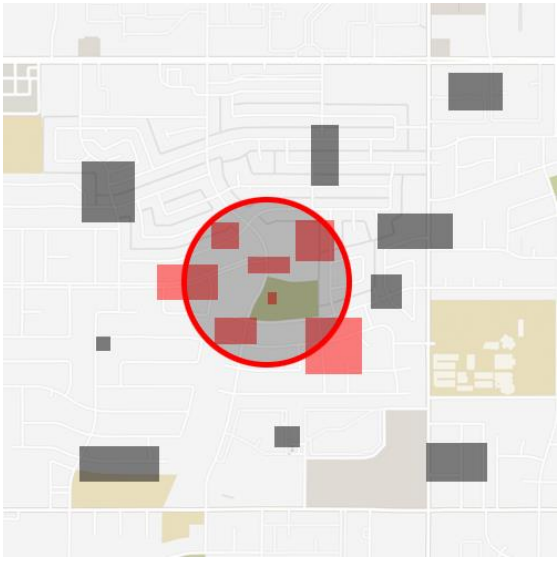

Abbildung 9 Ein Beispiel von Tile38 Plattform [7].

4 Node-RED Plattform:

Laut der Node-RED Definition von Node-RED Webseiten:

> *„Node-RED is a programming tool for wiring together hardware devices, APIs and online services in new and interesting ways. It provides a browser-based editor that makes it easy to wire together flows using the wide range of nodes in the palette that can be deployed to its runtime in a single-click."* [8]
>
> *„Node-RED ist ein Programmiertool zum Verkabeln von Hardwaregeräten, APIs und Onlinediensten auf neue und interessante Weise. Es bietet einen browserbasierten Editor, mit den Flow mithilfe der Vielzahl von Knoten in der Palette, die mit einem einzigen Klick zur Laufzeit bereitgestellt werden können, einfach miteinander verbunden werden können."*

Zusammenfassend ist Node-RED eine weit verbreitete Plattform im IoT für die schnelle Gestaltung von Benutzeroberflächen.

Abbildung 10 Ein Beispiel von Node-RED [9].

Im Rahmen dieses Projekts ist Node-RED ein schnelles Tool zum Erstellen von Benutzeroberflächen für Geofence.

5 HTTP Request:

5.1 Einführung:

Laut der HTTP Definition von Wikipedia:

> *„Das Hypertext Transfer Protocol (HTTP, englisch für Hypertext-Übertragungsprotokoll) ist ein zustandsloses Protokoll zur Übertragung von Daten auf der Anwendungsschicht über ein Rechnernetz. Es wird hauptsächlich eingesetzt, um Webseiten (Hypertext-Dokumente) aus dem World Wide Web (WWW) in einen Webbrowser zu laden. Es ist jedoch nicht prinzipiell darauf beschränkt und auch als allgemeines Dateiübertragungsprotokoll sehr verbreitet."* [10]

Zusammenfassend ist HTTP ein Protokoll, das in der Anwendungsschicht des OSI-Modells liegt, um Webseiten (Hypertext-Dokumente) aus dem World Wide Web (WWW) in einen Webbrowser zu laden.

OSI model

Layer	Name	Example protocols
7	Application Layer	HTTP, FTP, DNS, SNMP, Telnet
6	Presentation Layer	SSL, TLS
5	Session Layer	NetBIOS, PPTP
4	Transport Layer	TCP, UDP
3	Network Layer	IP, ARP, ICMP, IPSec
2	Data Link Layer	PPP, ATM, Ethernet
1	Physical Layer	Ethernet, USB, Bluetooth, IEEE802.11

Abbildung 11 HTTP Protokoll in der Anwendungsschicht von OSI-Modell. [11]

5.2 Wie HTTP Request funktioniert:

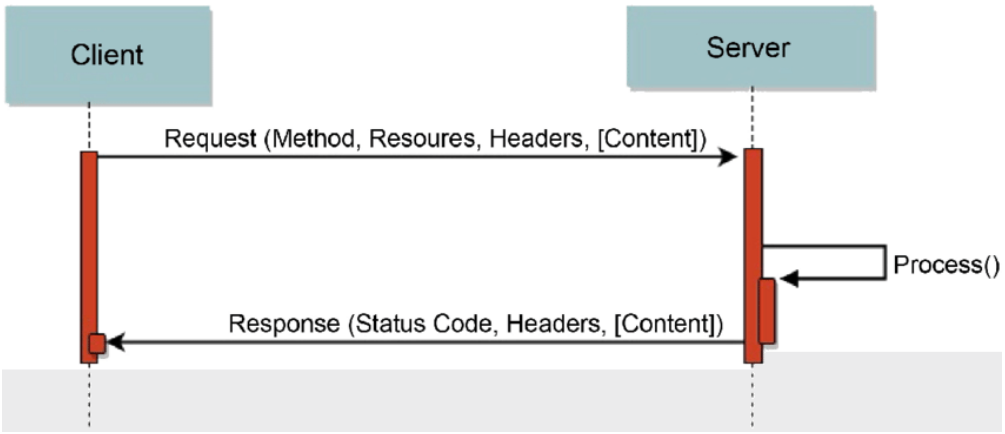

Abbildung 12 Wie HTTP Request funktioniert. [12]

Ein HTTP Request ist eine Aktion, die für eine Ressource ausgeführt wird, die durch eine bestimmte Request-URL identifiziert wird. Es gibt verschiedene HTTP Request Methode, denen jedoch jeweils ein bestimmter Zweck zugewiesen ist. HTTP Request fungiert als zwischengeschaltete Transportmethode zwischen einem Client / einer Anwendung und einem Server. Der Client sendet ein HTTP Request an den Server. Nach der Internalisierung der Nachricht sendet der Server eine Antwort zurück. Die Antwort enthält Statusinformationen zum Request.

Die HTTP Request Methode umfassen acht grundlegende Methode:

GET
: GET wird verwendet, um Daten von einer angegebenen Ressource auf einem Server abzurufen und anzufordern. GET ist eine der beliebtesten HTTP Request Techniken. Die GET-Methode wird verwendet, um alle Informationen abzurufen, die durch die Request-URL identifiziert werden.

POST
: Eine weitere beliebte HTTP Request Methode ist POST. In der Webkommunikation werden POST Request verwendet, um Daten an einen Server zu senden, um eine Ressource zu erstellen oder zu aktualisieren. Die mit der POST Request Methode an den Server gesendeten Informationen werden im Anforderungshauptteil der HTTP Request archiviert. Die HTTP Request Methode POST wird häufig verwendet, um benutzergenerierte Daten an einen Server zu senden. Ein Beispiel ist, wenn ein Benutzer ein Profilfoto hochlädt.

HEAD
: Die HEAD Technik fordert eine Reaktion an, die der GET Request ähnlich ist, jedoch keinen Nachrichtentext in der Antwort enthält. Die HEAD Request Methode ist nützlich, um Metadaten wiederherzustellen, die gemäß den Headern geschrieben wurden, ohne den gesamten Inhalt zu übertragen. Die Technik wird häufig verwendet, wenn Hypertext-Links auf Zugänglichkeit, Gültigkeit und aktuelle Änderungen getestet werden.

PUT
: Die PUT Request Methode ist ähnlich wie POST Request Methode, da es zum Senden von Daten an den Server zum Erstellen oder Aktualisieren einer Ressource verwendet wird. Der Unterschied zwischen den beiden besteht darin, dass PUT Request idempotent sind. Dies bedeutet, dass die Ergebnisse immer gleich sind, wenn dieselben PUT Request mehrmals aufgerufen werden.

DELETE
: Die DELETE Request Methode wird verwendet, um Ressourcen zu löschen, die durch eine bestimmte URL angegeben sind. Durch eine DELETE Request wird die Zielressource entfernt.

TRACE
: Die TRACE Request Methode werden verwendet, um einen Remote-Loopback-Test für Anwendungen auf dem Pfad zur Zielressource aufzurufen. Mit der TRACE Methode können Clients anzeigen, welche Nachricht am anderen Ende der Anforderungskette empfangen wird, damit sie die Informationen für Test- oder Diagnosefunktionen verwenden können.

CONNECT

Die CONNECT Request wird vom Client verwendet, um eine Netzwerkverbindung zu einem Webserver über ein bestimmtes HTTP herzustellen. Ein gutes Beispiel ist das SSL-Tunneling. Kurz gesagt, die CONNECT Request erstellt einen Tunnel zum Server, der durch eine bestimmte URL identifiziert wird.

PATCH

Eine PATCH Request ist ähnlich wie POST und PUT. Der Hauptzweck besteht jedoch darin, teilweise Änderungen an der Ressource vorzunehmen. Und genau wie eine POST Request ist auch die PATCH Request nicht idempotent. Im Gegensatz zu POST und PUT, die eine vollständige Benutzerentität erforderlich ist, darf man bei PATCH Request nur den aktualisierten Benutzernamen senden.

Im Rahmen dieser Projektarbeit wurden die GET und POST Request Methode verwendet, um die neue Geofence-Objekte zu definieren und um die Informationen von Tile38 Server abzurufen.

```
/** Case 1: Set position for a polygon **/
@POST("SET {keyID} {fieldName} OBJECT {\"type\":\"Polygon\",\"coordinates\":[[{longitudeLatitude}]]}")
Call<SetPositionForArea> defineNewArea(@Path("keyID") String keyID, @Path("fieldName") String name,
        @Path("longitudeLatitude") String longitudeLatitude);

/** Case 2: Set position for rectangle **/
@POST("SET {keyID} {fieldName} BOUNDS {southwestLatitude} {southwestLongitude} {northeastLatitude} {northeastLongitude}")
Call<SetPositionForRectangle> defineNewRectangle(@Path("keyID") String keyID, @Path("fieldName") String name,
        @Path("southwestLatitude") double southwestLatitude, @Path("southwestLongitude") double southwestLongitude,
        @Path("northeastLatitude") double northeastLatitude, @Path("northeastLongitude") double northeastLongitude);

/** Case 3: Set position for a point **/
@POST("SET {keyID} {fieldName} POINT {latitude} {longitude}")
Call<SetPositionForPoint> defineNewPoint(@Path("keyID") String keyID, @Path("fieldName") String name,
        @Path("latitude") double latitude, @Path("longitude") double longitude);

/** Case 4: Get all defined objects **/
@GET("KEYS *")
Call<GetKey> getAllKey();

/** Case 5: Check if object exists in area by passing parameter **/
@GET("TEST POINT {latitude} {longitude} WITHIN GET {keyID} {fieldName}")
Call<Area> objectExistInAreaByPassingParameter(@Path("latitude") double latitude,
        @Path("longitude") double longitude, @Path("keyID") String keyID, @Path("fieldName") String fieldName);

/** Case 6: Remove all objects using same keyID **/
@POST("DROP {keyID}")
Call<DropAllObject> removeAllObjectFromKeyID(@Path("keyID") String keyID);
```

Abbildung 13 GET und POST Request im Quellecode von Java App.

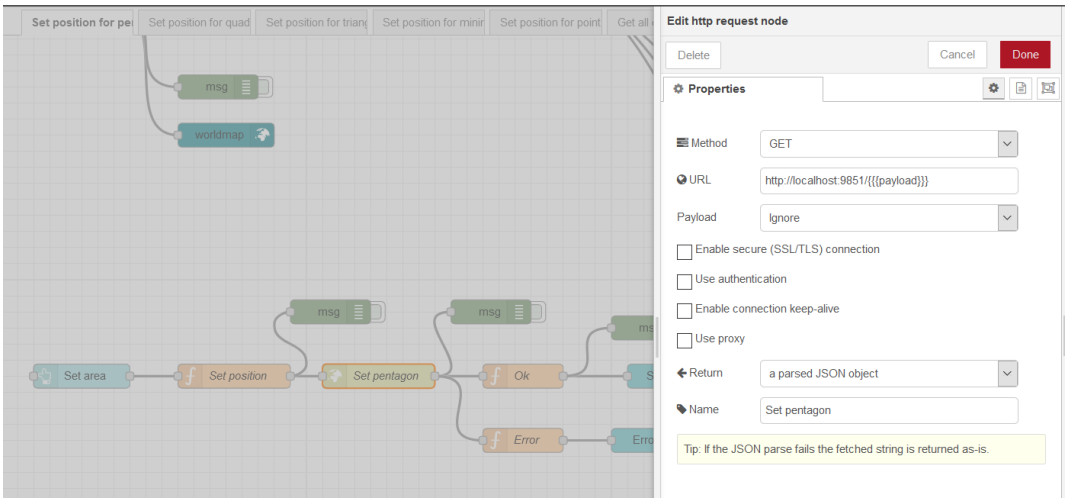

Abbildung 14 GET Request im Quellecode von Node-RED.

6 Applikation:

In dieser Projektarbeit werden zwei Anwendungen auf Java- und Node-RED Plattformen vorgestellt, die beide auf der Open Source Software Plattform Tile38 basieren.

6.1 Java Applikation:

Diese Java Applikation wird verwendet, um einige der Funktionen von Geofence sowie die Open Source-Plattform Tile38 zu demonstrieren. Die hier verwendeten Funktionen umfassen das Definieren eines Polygons, das Definieren eines minimalen Begrenzungsrechtecks, das Definieren eines Standorts, die Anzeigen alle definierten Objekte, das Überprüfen, ob sich das Objekt in den definierten Bereichen oder Standort befindet, das Entfernen aller Objekte basierend auf der KeyID und das Entfernen ein bestimmtes Objekt, die Änderung der KeyID, die Änderung den Server auf schreibgeschützt, die Pingen den Server an und der Abruf die aktuellen Serverinformationen. In jeder Funktion gibt es Beispiele zu demonstrieren.

```
INFO  [11:56:37.808] [displayMenu    ] [org.ict.geofencing.Main              ] [Main.java          ] [  113] - Tile38 Menu:
INFO  [11:56:37.816] [displayMenu    ] [org.ict.geofencing.Main              ] [Main.java          ] [  114] - 1. Set position for polygon
INFO  [11:56:37.817] [displayMenu    ] [org.ict.geofencing.Main              ] [Main.java          ] [  115] - 2. Set position for minimum bounding rectangle
INFO  [11:56:37.817] [displayMenu    ] [org.ict.geofencing.Main              ] [Main.java          ] [  116] - 3. Set position for point
INFO  [11:56:37.818] [displayMenu    ] [org.ict.geofencing.Main              ] [Main.java          ] [  117] - 4. Get all defined objects
INFO  [11:56:37.819] [displayMenu    ] [org.ict.geofencing.Main              ] [Main.java          ] [  118] - 5. Find out if object in area
INFO  [11:56:37.819] [displayMenu    ] [org.ict.geofencing.Main              ] [Main.java          ] [  119] - 6. Remove all objects using same keyID
INFO  [11:56:37.820] [displayMenu    ] [org.ict.geofencing.Main              ] [Main.java          ] [  120] - 7. Remove a specified object
INFO  [11:56:37.821] [displayMenu    ] [org.ict.geofencing.Main              ] [Main.java          ] [  121] - 8. Rename a keyID
INFO  [11:56:37.821] [displayMenu    ] [org.ict.geofencing.Main              ] [Main.java          ] [  122] - 9. Set server to read only mode
INFO  [11:56:37.822] [displayMenu    ] [org.ict.geofencing.Main              ] [Main.java          ] [  123] - 10. Ping to server
INFO  [11:56:37.822] [displayMenu    ] [org.ict.geofencing.Main              ] [Main.java          ] [  124] - 11. Get server information
INFO  [11:56:37.823] [main           ] [org.ict.geofencing.Main              ] [Main.java          ] [ 1085] - Enter your choice:
```

Abbildung 15 Funktionen der Java Applikation.

6.1.1 Definieren ein Bereich:

Hier wird ein Bereich rund um den FH Dortmund Standort Sonnenstraße als ein Geofence-Objekt in Form von Polygon mit vier Punkten definiert. Das Geofence-Objekt hat die KeyID *fhDortmund* und Field Name *campusSonnenstr*.

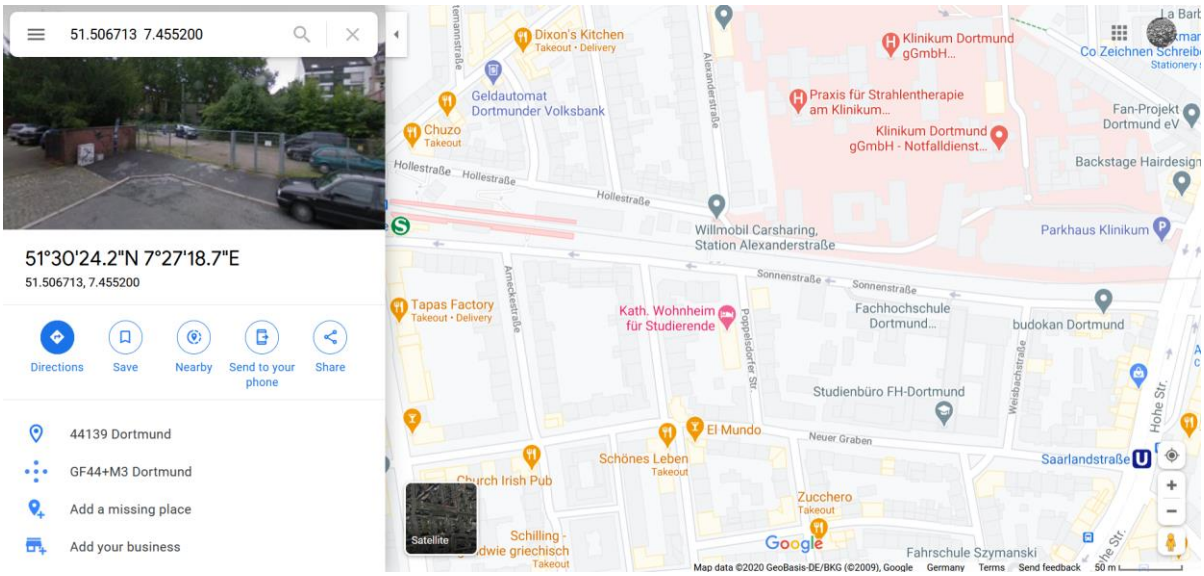

Abbildung 16 Eine Koordinate an der FH Dortmund Standort Sonnenstraße.

Punkt 1	Punkt 2	Punkt 3	Punkt 4
51,506713 - 7,455200	51,506512 - 7,458390	51,505421 - 7,458165	51,505634 - 7,455344

Tabelle 1 Koordinaten (Breitengrad/Längengrad) vier polygonförmige Eckpunkte des FH Dortmund Standort Sonnenstraße.

```
INFO  [11:57:06.047] [setPositionForPolygon] [org.ict.geofencing.Main                ] [Main.java          ] [  317] - Set position for a area...
INFO  [11:57:06.048] [setPositionForPolygon] [org.ict.geofencing.Main                ] [Main.java          ] [  318] - Enter keyID:
fhDortmund
INFO  [11:57:11.612] [setPositionForPolygon] [org.ict.geofencing.Main                ] [Main.java          ] [  324] - Enter field name:
campusSonnenstr
INFO  [11:57:26.117] [setPositionForAreaGetNumberCorner] [org.ict.geofencing.Main                ] [Main.java          ] [  256] - How many corner in area:
4
INFO  [11:57:32.206] [setPositionForAreaEnterLongitudeLatitude] [org.ict.geofencing.Main                ] [Main.java          ] [  195] - Enter latitude of 1 corner
51.506713
INFO  [11:57:44.984] [setPositionForAreaEnterLongitudeLatitude] [org.ict.geofencing.Main                ] [Main.java          ] [  197] - Enter longitude of 1 corne
7.455200
INFO  [11:57:49.710] [setPositionForAreaEnterLongitudeLatitude] [org.ict.geofencing.Main                ] [Main.java          ] [  195] - Enter latitude of 2 corner
51.506512
INFO  [11:57:54.529] [setPositionForAreaEnterLongitudeLatitude] [org.ict.geofencing.Main                ] [Main.java          ] [  197] - Enter longitude of 2 corne
7.458390
INFO  [11:57:57.726] [setPositionForAreaEnterLongitudeLatitude] [org.ict.geofencing.Main                ] [Main.java          ] [  195] - Enter latitude of 3 corner
51.505421
INFO  [11:58:02.072] [setPositionForAreaEnterLongitudeLatitude] [org.ict.geofencing.Main                ] [Main.java          ] [  197] - Enter longitude of 3 corne
7.458165
INFO  [11:58:05.616] [setPositionForAreaEnterLongitudeLatitude] [org.ict.geofencing.Main                ] [Main.java          ] [  195] - Enter latitude of 4 corner
51.505634
INFO  [11:58:08.983] [setPositionForAreaEnterLongitudeLatitude] [org.ict.geofencing.Main                ] [Main.java          ] [  197] - Enter longitude of 4 corne
7.455344
```

Abbildung 17 Eingabe Informationen von vier oben genannten Koordinaten der FH Dortmund.

Das Ergebnis nach Eingabe des oben genannten Geofence-Objekts:

```
sitionForAreaDisplayResponse] [org.ict.geofencing.Main                ] [Main.java          ] [  166] - Successful set position for area fhDortmund campusSonnenstr
```

Abbildung 18 Ergebnis nach Eingabe des oben genannten Geofence-Objekts.

6.1.2 Definieren ein minimales Begrenzungsrechteck:

In diesem Fall wird ein minimales Begrenzungsrechteck nur mit den Koordinaten von zwei Punkten definiert, nämlich südwestlicher Breite, südwestlicher Länge, nordöstlicher Breite, nordöstlicher Länge.

Hier wird das Gebäude A der FH Dortmund Standort Sonnenstraße mit zwei Punkten definiert:

	Breitengrad	Längengrad
südwestliche	51,505600	7,457404
nordöstliche	51,506397	7,458230

Tabelle 2 Koordinate von südwestlichen und nordöstlichen Punkten des Gebäude A der FH Dortmund Standort Sonnenstraße

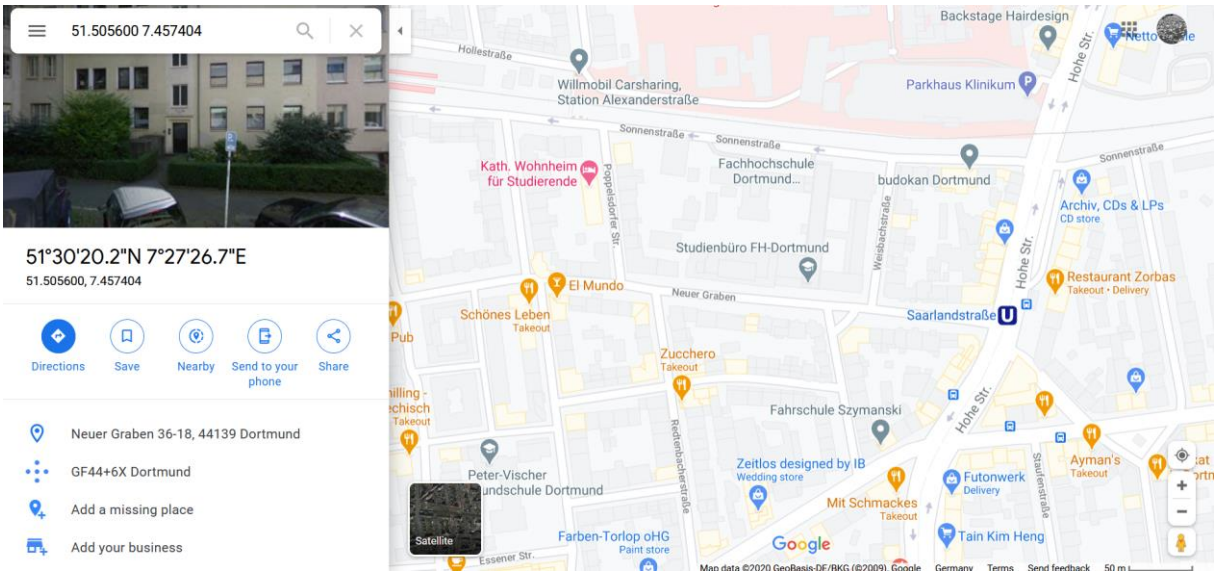

Abbildung 19 Eine Koordinate von einem Punkt an der FH Dortmund.

```
INFO  [11:59:06.353] [setPositionForMinimumBoundingRectangle] [org.ict.geofencing.Main                ] [Main.java          ] [  408] - Set position for a area...
INFO  [11:59:06.354] [setPositionForMinimumBoundingRectangle] [org.ict.geofencing.Main                ] [Main.java          ] [  409] - Enter keyID:
fhDortmund
INFO  [11:59:16.190] [setPositionForMinimumBoundingRectangle] [org.ict.geofencing.Main                ] [Main.java          ] [  415] - Enter field name:
buildingA
INFO  [11:59:18.723] [setPositionForRectangleInputLatitudeLongitude] [org.ict.geofencing.Main                ] [Main.java          ] [  369] - Enter southwest latit
51.505600
INFO  [12:02:54.115] [setPositionForRectangleInputLatitudeLongitude] [org.ict.geofencing.Main                ] [Main.java          ] [  371] - Enter southwest longi
7.457404
INFO  [12:03:02.971] [setPositionForRectangleInputLatitudeLongitude] [org.ict.geofencing.Main                ] [Main.java          ] [  373] - Enter northeast latit
51.506397
INFO  [12:03:06.812] [setPositionForRectangleInputLatitudeLongitude] [org.ict.geofencing.Main                ] [Main.java          ] [  375] - Enter northeast longi
7.458230
```

Abbildung 20 Eingabe Informationen von zwei oben genannten Koordinaten der FH Dortmund.

Das Ergebnis nach Eingabe des oben genannten Geofence-Objekts:

```
ctangleInputLatitudeLongitude] [org.ict.geofencing.Main                ] [Main.java          ] [  385] - Successful set position for Rectangle fhDortmund buildingA
```

Abbildung 21 Ergebnis nach Eingabe des oben genannten Geofence-Objekts.

6.1.3 Definieren einen Standort:

Ähnlich wie oben wird ein Ort als Geofence-Objekt mit der KeyID *fhDortmund* und Field Name *sonnenstr* sowie einem Breitengrad von 51,50665 und einem Längengrad von 7,45563 definiert.

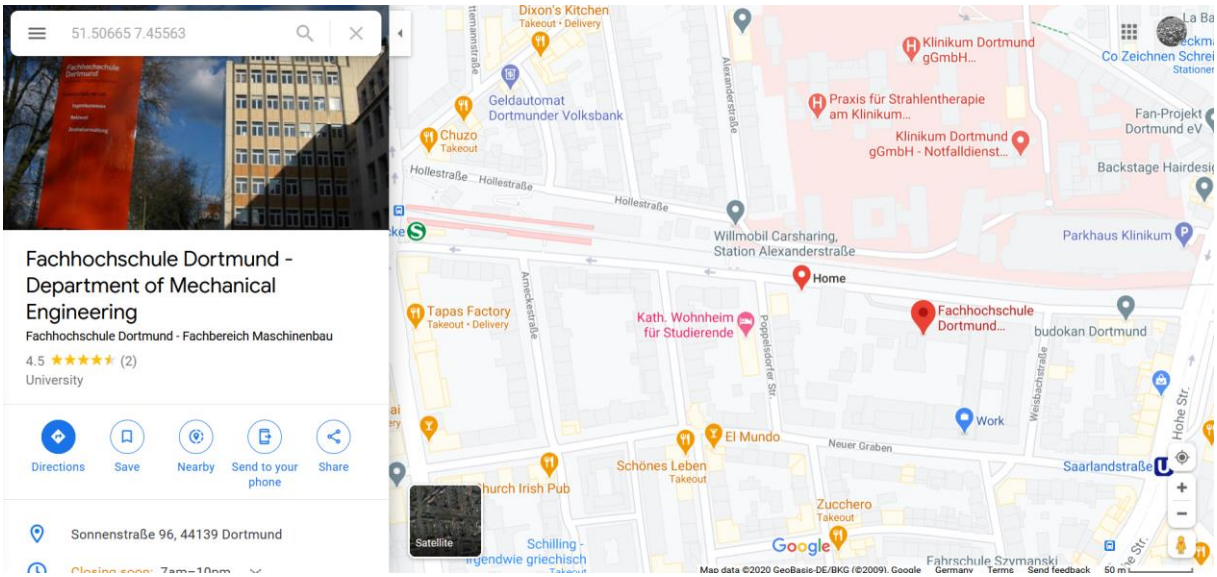

Abbildung 22 Koordinate von FH Dortmund Standort Sonnenstraße.

```
INFO  [12:04:44.452] [setPositionForPoint] [org.ict.geofencing.Main                ] [Main.java          ] [  495] - Set position for a point...
INFO  [12:04:44.452] [setPositionForPoint] [org.ict.geofencing.Main                ] [Main.java          ] [  496] - Enter keyID:
fhDortmund
INFO  [12:04:48.738] [setPositionForPoint] [org.ict.geofencing.Main                ] [Main.java          ] [  502] - Enter field name:
sonnenstr
INFO  [12:04:55.302] [setPositionForPointEnterLongitudeLatitude] [org.ict.geofencing.Main                ] [Main.java          ] [  460] - Enter latitude:
51.50665
INFO  [12:05:08.220] [setPositionForPointEnterLongitudeLatitude] [org.ict.geofencing.Main                ] [Main.java          ] [  462] - Enter longitude:
7.45563
```

Abbildung 23 Eingabe Informationen von oben genannten Koordinaten der FH Dortmund.

Das Ergebnis nach Eingabe des oben genannten Geofence-Objekts:

```
onForPointEnterLongitudeLatitude] [org.ict.geofencing.Main                ] [Main.java          ] [  471] - Successful set position for point fhDortmund sonnenstr
```

Abbildung 24 Ergebnis nach Eingabe des oben genannten Geofence-Objekts.

6.1.4 Anzeigen alle definierten Objekte:

In diesem Fall werden alle definierten Geofence-Objekte angezeigt.

```
[  651] - Key: Dortmund
[  652] - ID: Westpark
[  667] - Type: Pentagon
[  675] - Coordinates (longitude/latitude): [[[7.447133,51.511131],[7.449371,51.512299],[7.450092,51.511535],[7.45002,51.507829],[7.45002,51.507829],[7.447133,51.511
[  678] - --------------------------------
[  651] - Key: fhDortmund
[  652] - ID: buildingA
[  663] - Type: Quadrangle
[  675] - Coordinates (longitude/latitude): [[[7.457404,51.5056],[7.45823,51.5056],[7.45823,51.506397],[7.457404,51.506397],[7.457404,51.5056]]]
[  678] - --------------------------------
[  651] - Key: fhDortmund
[  652] - ID: campusSonnenstr
[  663] - Type: Quadrangle
[  675] - Coordinates (longitude/latitude): [[[7.4552,51.506713],[7.45839,51.506512],[7.458165,51.505421],[7.458165,51.505421],[7.4552,51.506713]]]
[  678] - --------------------------------
[  651] - Key: fhDortmund
[  652] - ID: sonnenstr
[  655] - Type: Point
[  675] - Coordinates (longitude/latitude): [7.45563,51.50665]
[  678] - --------------------------------
[  689] - Found 4 objects with 2 keys in database.
[  691] - 1 are points.
[  692] - 0 are triangles.
[  693] - 2 are quadrangles.
[  694] - 1 are pentagons.
[  695] - 0 are polygons more than >= 5 cornes.
```

Abbildung 25 Alle definierten Geofence-Objekte.

6.1.5 Überprüfen eine Position in definierten Bereichen oder Standort:

Eine Position in der FH Dortmund Standort Sonnenstraße mit einem Breitengrad von 51,506001 und einem Längengrad von 7,457747 wurde zufällig zum Testen ausgewählt. Diese Position kann eine Person darstellen, die sich in diesem Bereich befindet.

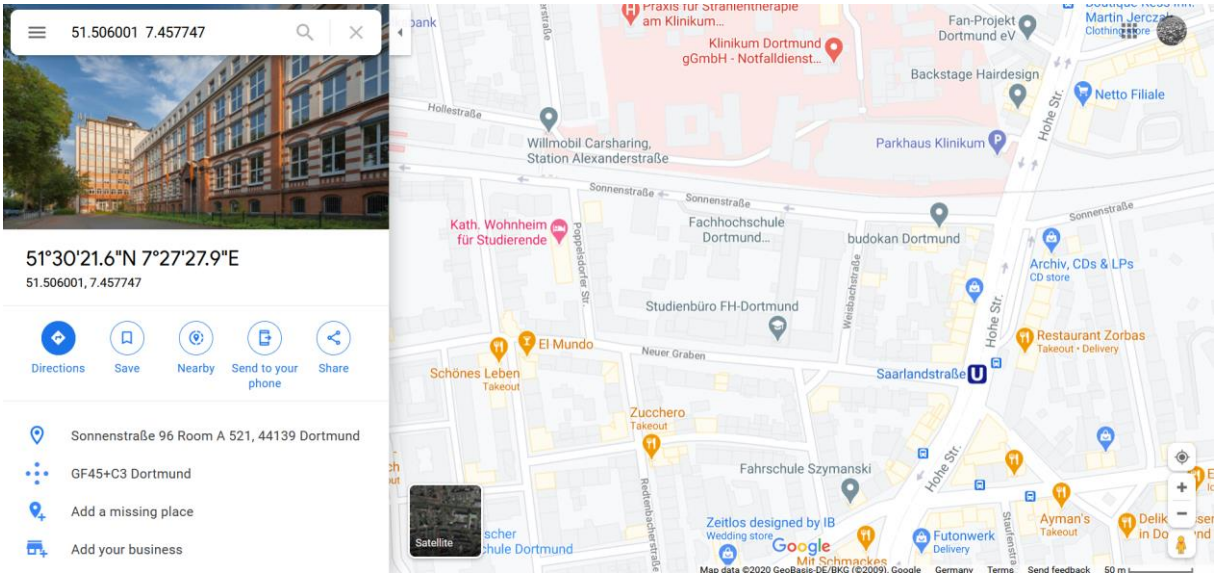

Abbildung 26 Eine zufällige ausgewählte Position in der FH Dortmund.

```
INFO  [12:10:50.861] [checkObjectExistInArea] [org.ict.geofencing.Main              ] [Main.java          ] [  786] - Check if object exists in area...
INFO  [12:10:50.862] [checkObjectExistInArea] [org.ict.geofencing.Main              ] [Main.java          ] [  787] - Enter latitude:
51.506001
INFO  [12:10:55.915] [checkObjectExistInArea] [org.ict.geofencing.Main              ] [Main.java          ] [  789] - Enter longitude:
7.457747
```

Abbildung 27 Eingabe Informationen der zufälligen ausgewählten Position in der FH Dortmund.

Das Ergebnis nach Eingabe der oben genannten Position:

```
[findAreaOfObject] [org.ict.geofencing.Main              ] [Main.java          ] [  754] - Object was found in 2 following defined area [keyID, fieldName].
[findAreaOfObject] [org.ict.geofencing.Main              ] [Main.java          ] [  758] - [fhDortmund,buildingA]
[findAreaOfObject] [org.ict.geofencing.Main              ] [Main.java          ] [  758] - [fhDortmund,campusSonnenstr]
```

Abbildung 28 Ergebnis nach Eingabe der oben genannten Position.

Dieser Ort oder diese Person befindet sich sowohl in dem Geofence-Objekt mit der KeyID *fhDortmund* und dem Field Name *campusSonnenstr* als auch in dem Geofence-Objekt mit der KeyID *fhDortmund* und dem Field Name *buildingA*.

6.1.6 Entfernen aller Objekte basierend auf der KeyID:

In diesem Fall werden alle Geofence-Objekte gelöscht, die mit derselben KeyID definiert wurden. Hier wurden alle Geofence-Objekte mit der KeyID *Dortmund* gelöscht

```
INFO  [12:32:29.634] [removeAllObjectsWithKey] [org.ict.geofencing.Main               ] [Main.java          ] [  816] - Remove all objects from keyID...
INFO  [12:32:29.635] [removeAllObjectsWithKey] [org.ict.geofencing.Main               ] [Main.java          ] [  817] - Enter keyID:
Dortmund
```

Abbildung 29 Eingabe die KeyID, die gelöscht werden soll.

```
[removeAllObjectsWithKey] [org.ict.geofencing.Main               ] [Main.java          ] [  827] - Remove all objects with key ID: Dortmund successful.
```

Abbildung 30 Ergebnis nach der Eingabe der KeyID.

```
[  651] - Key: fhDortmund
[  652] - ID: buildingA
[  663] - Type: Quadrangle
[  675] - Coordinates (longitude/latitude): [[[7.457404,51.5056],[7.45823,51.5056],[7.45823,51.506397],[7.457404,51.506397],[7.457404,51.5056]]]
[  678] - --------------------------------
[  651] - Key: fhDortmund
[  652] - ID: campusSonnenstr
[  663] - Type: Quadrangle
[  675] - Coordinates (longitude/latitude): [[[7.4552,51.506713],[7.45839,51.506512],[7.458165,51.505421],[7.458165,51.505421],[7.4552,51.506713]]]
[  678] - --------------------------------
[  651] - Key: fhDortmund
[  652] - ID: sonnenstr
[  655] - Type: Point
[  675] - Coordinates (longitude/latitude): [7.45563,51.50665]
[  678] - --------------------------------
[  689] - Found 3 objects with 1 keys in database.
[  691] - 1 are points.
[  692] - 0 are triangles.
[  693] - 2 are quadrangles.
[  694] - 0 are pentagons.
[  695] - 0 are polygons more than >= 5 cornes.
```

Abbildung 31 Überprüfen nochmals die Geofence-Objekte.

6.1.7 Entfernen ein bestimmtes Objekt:

In diesem Fall wird ein Geofence-Objekt mit der spezifischen KeyID und dem Field Name entfernt. Hier wurde ein Geofence-Objekt mit der KeyID *fhDortmund* und dem Field Name *sonnenstr* entfernt.

```
INFO  [12:33:58.616] [removeSpecifiedObject] [org.ict.geofencing.Main               ] [Main.java          ] [  851] - Remove a specified object...
INFO  [12:33:58.617] [removeSpecifiedObject] [org.ict.geofencing.Main               ] [Main.java          ] [  852] - Enter keyID:
fhDortmund
INFO  [12:34:05.111] [removeSpecifiedObject] [org.ict.geofencing.Main               ] [Main.java          ] [  854] - Enter fieldName:
sonnenstr
```

Abbildung 32 Eingabe KeyID und Field Name von Objekt, das gelöscht werden soll.

```
[org.ict.geofencing.Main               ] [Main.java          ] [  867] - Remove successful object with keyID: fhDortmund, fieldName: sonnenstr
```

Abbildung 33 Ergebnis nach der Eingabe KeyID und Field Name zum Löschen Objekt.

```
[  651] - Key: fhDortmund
[  652] - ID: buildingA
[  663] - Type: Quadrangle
[  675] - Coordinates (longitude/latitude): [[[7.457404,51.5056],[7.45823,51.5056],[7.45823,51.506397],[7.457404,51.506397],[7.457404,51.5056]]]
[  678] - --------------------------------
[  651] - Key: fhDortmund
[  652] - ID: campusSonnenstr
[  663] - Type: Quadrangle
[  675] - Coordinates (longitude/latitude): [[[7.4552,51.506713],[7.45839,51.506512],[7.458165,51.505421],[7.458165,51.505421],[7.4552,51.506713]]]
[  678] - --------------------------------
[  689] - Found 2 objects with 1 keys in database.
[  691] - 0 are points.
[  692] - 0 are triangles.
[  693] - 2 are quadrangles.
[  694] - 0 are pentagons.
[  695] - 0 are polygons more than >= 5 cornes.
```

Abbildung 34 Überprüfen nochmals die Geofence-Objekte nach dem Löschen.

6.1.8 Änderung die KeyID:

Hier kann die vordefinierte KeyID geändert werden. Im folgenden Fall wird die KeyID von fhDortmund in FHDO geändert.

```
INFO  [12:35:34.901] [renameKeyID    ] [org.ict.geofencing.Main               ] [Main.java          ] [  923] - Rename key...
INFO  [12:35:34.901] [renameKeyID    ] [org.ict.geofencing.Main               ] [Main.java          ] [  924] - Enter old keyID:
fhDortmund
INFO  [12:35:37.138] [renameKeyID    ] [org.ict.geofencing.Main               ] [Main.java          ] [  932] - Enter new keyID:
FHDO
```

Abbildung 35 Eingabe alte und neue KeyID von Objekt, das geändert werden soll.

```
[renameKeyID    ] [org.ict.geofencing.Main               ] [Main.java          ] [  942] - Rename key fhDortmund to new key name FHDO successful.
```

Abbildung 36 Ergebnis nach der Eingabe von KeyID und Field Name zur Änderung Objekt.

```
[  651] - Key: FHDO
[  652] - ID: buildingA
[  663] - Type: Quadrangle
[  675] - Coordinates (longitude/latitude): [[[7.457404,51.5056],[7.45823,51.5056],[7.45823,51.506397],[7.457404,51.506397],[7.457404,51.5056]]]
[  678] - ---------------------------------
[  651] - Key: FHDO
[  652] - ID: campusSonnenstr
[  663] - Type: Quadrangle
[  675] - Coordinates (longitude/latitude): [[[7.4552,51.506713],[7.45839,51.506512],[7.458165,51.505421],[7.458165,51.505421],[7.4552,51.506713]]]
[  678] - ---------------------------------
[  689] - Found 2 objects with 1 keys in database.
[  691] - 0 are points.
[  692] - 0 are triangles.
[  693] - 2 are quadrangles.
[  694] - 0 are pentagons.
[  695] - 0 are polygons more than >= 5 cornes.
```

Abbildung 37 Überprüfen nochmals die Geofence-Objekte nach der Änderung.

6.1.9 Änderung den Server auf schreibgeschützt:

Hier kann der schreibgeschützte Modus des Servers ein- oder ausgeschaltet werden.

```
INFO  [12:36:50.287] [setReadMode    ] [org.ict.geofencing.Main               ] [Main.java          ] [  966] - Set server read only mode...
Nov 14, 2020 12:36:50 PM okhttp3.internal.platform.Platform log
INFO: --> GET http://localhost:9851/SERVER http/1.1
Nov 14, 2020 12:36:50 PM okhttp3.internal.platform.Platform log
INFO: --> END GET
Nov 14, 2020 12:36:50 PM okhttp3.internal.platform.Platform log
INFO: <-- 200 OK http://localhost:9851/SERVER (9ms)
Nov 14, 2020 12:36:50 PM okhttp3.internal.platform.Platform log
INFO: Connection: close
Nov 14, 2020 12:36:50 PM okhttp3.internal.platform.Platform log
INFO: Content-Length: 414
Nov 14, 2020 12:36:50 PM okhttp3.internal.platform.Platform log
INFO: Content-Type: application/json; charset=utf-8
Nov 14, 2020 12:36:50 PM okhttp3.internal.platform.Platform log
INFO:
Nov 14, 2020 12:36:50 PM okhttp3.internal.platform.Platform log
INFO: {"ok":true,"stats":{"aof_size":7832,"avg_item_size":375293,"cpus":8,"heap_released":60104704,"heap_size":2627056,"http_transport":true,"id":"58bb31daa2b6

Nov 14, 2020 12:36:50 PM okhttp3.internal.platform.Platform log
INFO: <-- END HTTP (414-byte body)
INFO  [12:36:50.302] [setReadMode    ] [org.ict.geofencing.Main               ] [Main.java          ] [  991] - Read only is off.
INFO  [12:36:50.303] [setReadMode    ] [org.ict.geofencing.Main               ] [Main.java          ] [  992] - Set read mode? Enter yes for on, no for off:
yes
```

Abbildung 38 Änderung schreibgeschützte Modus des Servers.

```
INFO  [12:36:53.176] [setReadMode    ] [org.ict.geofencing.Main               ] [Main.java          ] [ 1001] - Set successful read only mode on.
```

Abbildung 39 Ergebnis nach der Änderung schreibgeschützte Modus des Servers.

6.1.10 Pingen den Server an:

In diesem Fall kann die Ping-Zeit zum Server übergeprüft werden.

```
INFO  [12:38:15.817] [pingToServer   ] [org.ict.geofencing.Main               ] [Main.java          ] [ 1024] - Ping to server...
Nov 14, 2020 12:38:15 PM okhttp3.internal.platform.Platform log
INFO: --> POST http://localhost:9851/PING http/1.1
Nov 14, 2020 12:38:15 PM okhttp3.internal.platform.Platform log
INFO: Content-Length: 0
Nov 14, 2020 12:38:15 PM okhttp3.internal.platform.Platform log
INFO:
Nov 14, 2020 12:38:15 PM okhttp3.internal.platform.Platform log
INFO:
Nov 14, 2020 12:38:15 PM okhttp3.internal.platform.Platform log
INFO: --> END POST (0-byte body)
Nov 14, 2020 12:38:15 PM okhttp3.internal.platform.Platform log
INFO: <-- 200 OK http://localhost:9851/PING (5ms)
Nov 14, 2020 12:38:15 PM okhttp3.internal.platform.Platform log
INFO: Connection: close
Nov 14, 2020 12:38:15 PM okhttp3.internal.platform.Platform log
INFO: Content-Length: 45
Nov 14, 2020 12:38:15 PM okhttp3.internal.platform.Platform log
INFO: Content-Type: application/json; charset=utf-8
Nov 14, 2020 12:38:15 PM okhttp3.internal.platform.Platform log
INFO:
Nov 14, 2020 12:38:15 PM okhttp3.internal.platform.Platform log
INFO: {"ok":true,"ping":"pong","elapsed":"500ns"}

Nov 14, 2020 12:38:15 PM okhttp3.internal.platform.Platform log
INFO: <-- END HTTP (45-byte body)
INFO  [12:38:15.827] [pingToServer   ] [org.ict.geofencing.Main               ] [Main.java          ] [ 1029] - Server is reachable.
INFO  [12:38:15.827] [pingToServer   ] [org.ict.geofencing.Main               ] [Main.java          ] [ 1030] - Return from server: pong
INFO  [12:38:15.828] [pingToServer   ] [org.ict.geofencing.Main               ] [Main.java          ] [ 1031] - Time to return packet: 500ns
```

Abbildung 40 Pingen den Server an.

6.1.11 Abruf die aktuellen Serverinformationen:

Hier werden einige Informationen zum Tile38-Server angezeigt, z. B. Anzahl der Geofence-Objekte, Anzahl der Schlüssel-IDs, der schreibgeschützte Modus des Servers, Version von Tile38.

```
INFO  [12:38:50.160] [checkServerStatus] [org.ict.geofencing.Main              ] [Main.java          ] [ 1050] - Check server status...
Nov 14, 2020 12:38:50 PM okhttp3.internal.platform.Platform log
INFO: --> GET http://localhost:9851/SERVER http/1.1
Nov 14, 2020 12:38:50 PM okhttp3.internal.platform.Platform log
INFO: --> END GET
Nov 14, 2020 12:38:50 PM okhttp3.internal.platform.Platform log
INFO: <-- 200 OK http://localhost:9851/SERVER (4ms)
Nov 14, 2020 12:38:50 PM okhttp3.internal.platform.Platform log
INFO: Connection: close
Nov 14, 2020 12:38:50 PM okhttp3.internal.platform.Platform log
INFO: Content-Length: 412
Nov 14, 2020 12:38:50 PM okhttp3.internal.platform.Platform log
INFO: Content-Type: application/json; charset=utf-8
Nov 14, 2020 12:38:50 PM okhttp3.internal.platform.Platform log
INFO:
Nov 14, 2020 12:38:50 PM okhttp3.internal.platform.Platform log
INFO: {"ok":true,"stats":{"aof_size":7832,"avg_item_size":375396,"cpus":8,"heap_released":60186624,"heap_size":2627776,"http_transport":true,"id":"58bb31daa2b652b8822

Nov 14, 2020 12:38:50 PM okhttp3.internal.platform.Platform log
INFO: <-- END HTTP (412-byte body)
INFO  [12:38:50.167] [checkServerStatus] [org.ict.geofencing.Main              ] [Main.java          ] [ 1055] - Tile38 is on.
INFO  [12:38:50.168] [checkServerStatus] [org.ict.geofencing.Main              ] [Main.java          ] [ 1056] - Number of objects in server: 2
INFO  [12:38:50.168] [checkServerStatus] [org.ict.geofencing.Main              ] [Main.java          ] [ 1057] - Number of keys in server: 1
INFO  [12:38:50.168] [checkServerStatus] [org.ict.geofencing.Main              ] [Main.java          ] [ 1059] - Read only is on.
INFO  [12:38:50.168] [checkServerStatus] [org.ict.geofencing.Main              ] [Main.java          ] [ 1063] - Version of Tile38: 1.22.4
```

Abbildung 41 Abruf die aktuellen Serverinformationen.

6.2 Node-RED Applikation:

Die Funktionen dieser Node-RED App sind ähnlich wie die Java App, verfügen jedoch über eine zusätzliche Benutzeroberfläche.

6.2.1 Definieren ein Bereich:

Ein polygonförmiger Bereich mit vielen Eckpunkten kann in Bereiche mit Formen wie Fünfeck, Viereck oder Dreieck unterteilt werden. Wie im folgenden Beispiel ist die städtische Kliniken Dortmund ein Polygon mit sechs Eckpunkten und kann in ein Fünfeck und ein Dreieck unterteilt werden.

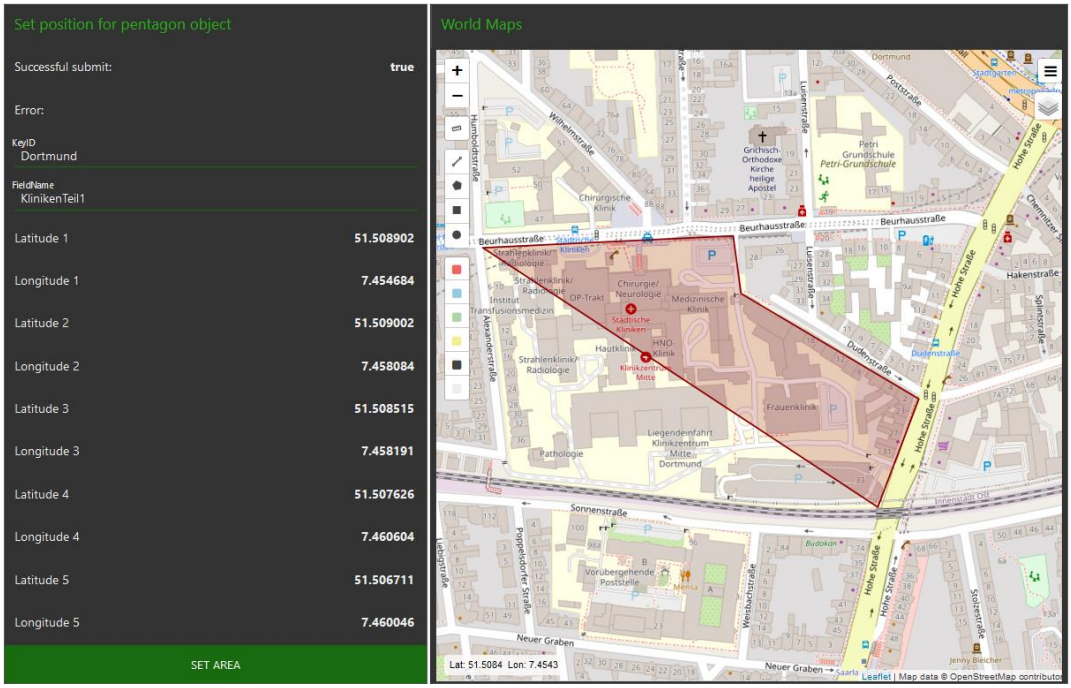

Abbildung 42 Definieren ein Fünfeck.

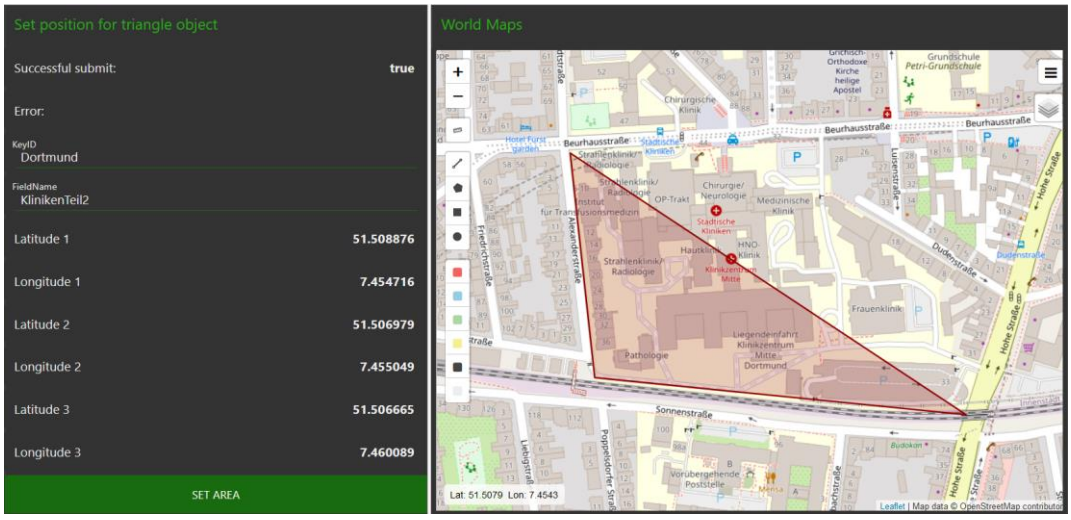

Abbildung 43 Definieren ein Dreieck.

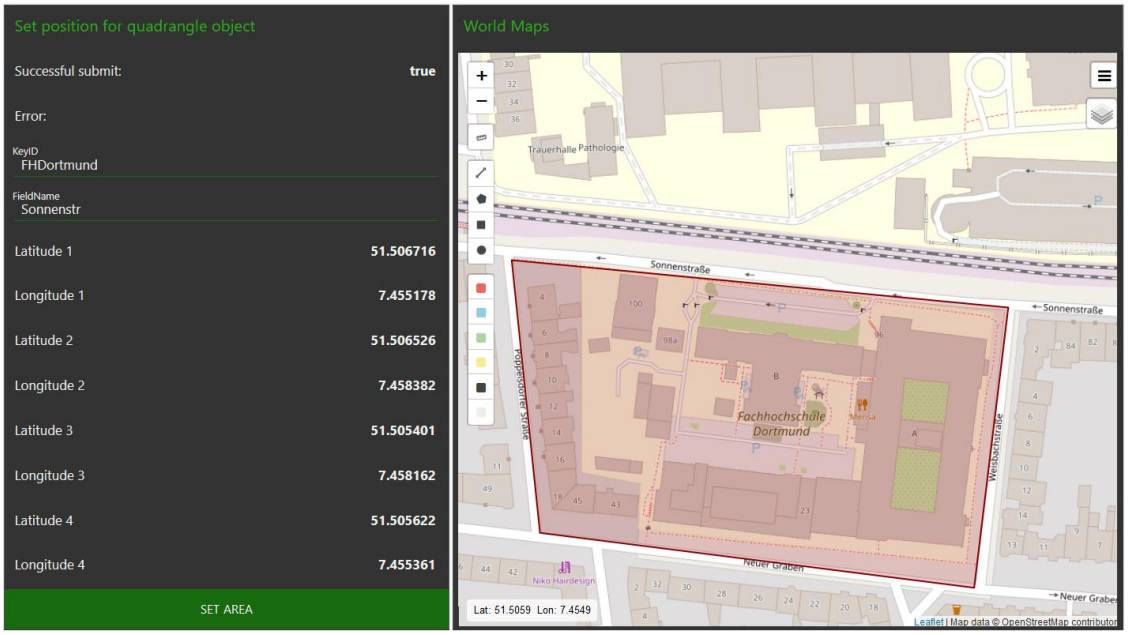

Abbildung 44 Definieren ein Viereck.

6.2.2 Definieren ein minimales Begrenzungsrechteck:

Das Gebäude A von FH Dortmund Standort Sonnenstraße wird als Rechteck Geofence-Objekt mit zwei Koordinaten definiert.

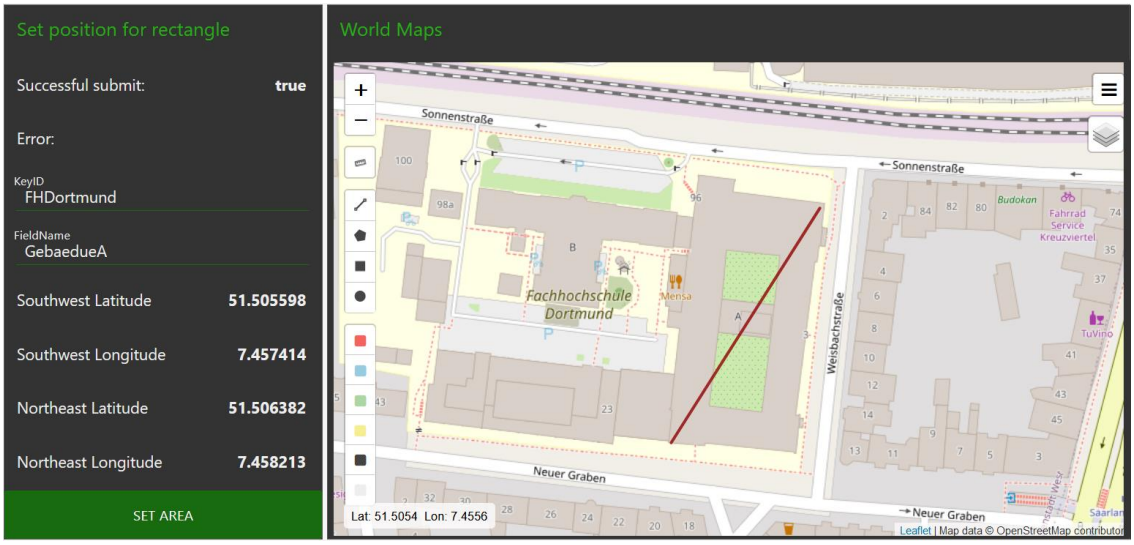

Abbildung 45 Definieren ein minimales Begrenzungsrechteck.

Dies kann verwendet werden, um schnell ein Rechteck mit Koordinaten von nur 2 Punkten zu definieren.

6.2.3 Definieren einen Standort:

In diesem Fall wird ein Punkt im Gebäude A als Geofence-Objekt mit einem Breitengrad von 51,506017 und einem Längengrad von 7,457770 definiert.

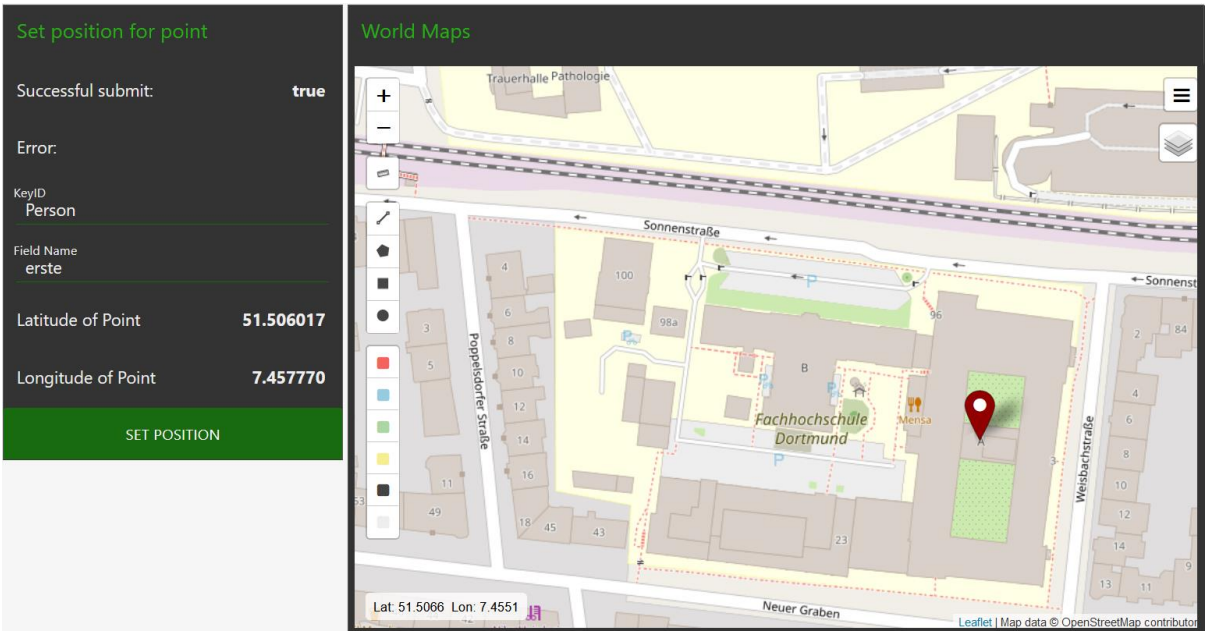

Abbildung 46 Definieren einen Standort in der FH Dortmund Campus Sonnenstraße.

Ähnlich werden zwei weitere Punkte in der städtischen Klinken Dortmund und im anderen Bereich näher von FH Dortmund definiert.

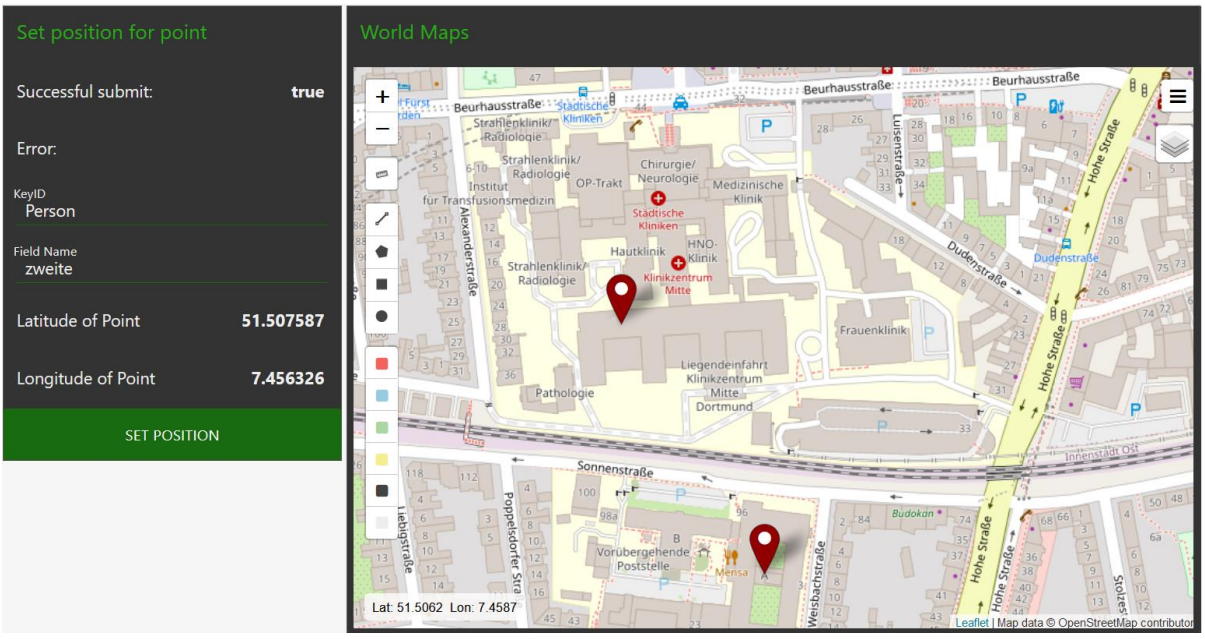

Abbildung 47 Definieren einen Standort in der städtischen Klinken Dortmund.

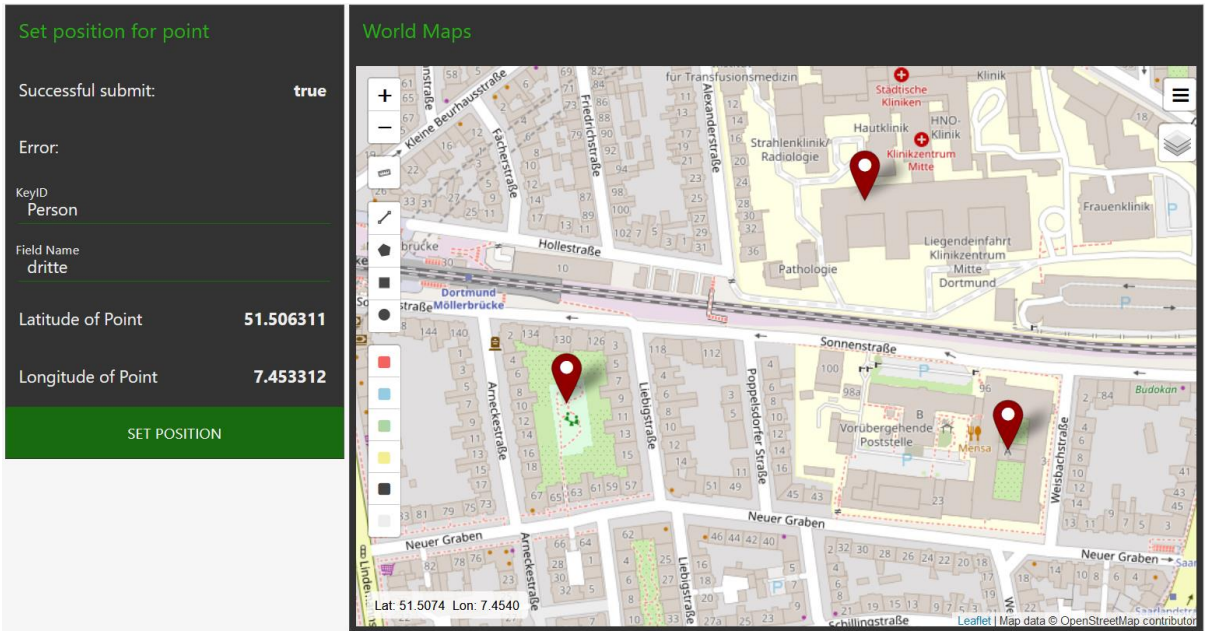

Abbildung 48 Definieren einen Standort im anderen Bereich näher von FH Dortmund

6.2.4 Anzeigen alle definierten Objekte:

Ähnlich wie in der Java App werden alle definierten Objekte auf dem Server angezeigt.

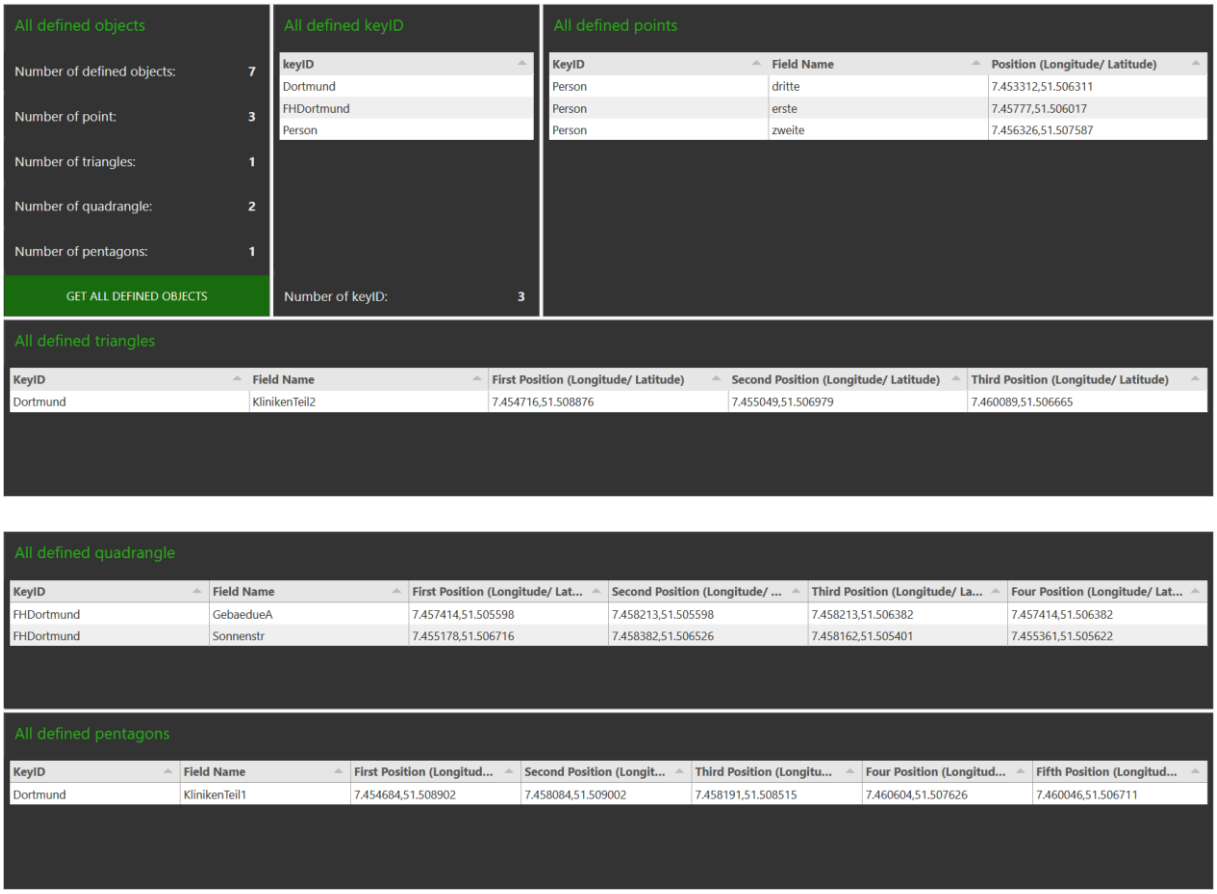

Abbildung 49 Alle definierten Objekte

6.2.5 Überprüfen Position in einem definierten Bereich oder Standort:

Hier ist eine Position in der FH Dortmund Standort Sonnenstr als Beispiel zu nennen. Nach dem Überprüfen im Node-RED befindet sich diese Position in der FH Dortmund Standort Sonnenstraße und exakter im Gebäude A. Die beide Bereiche wurden gerade oben als Geofence-Objekte im Node-RED definiert.

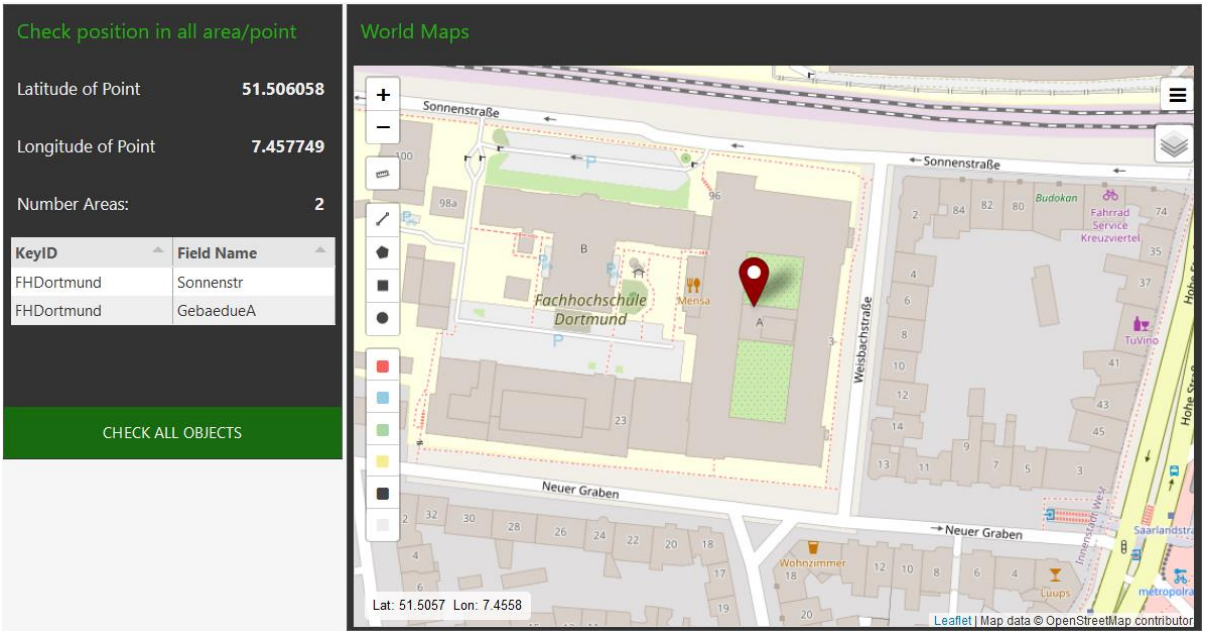

Abbildung 50 Überprüfen Position in einem Gebäude in der FH Dortmund Standort Sonnenstraße.

6.2.6 Alle definierten Objekte im Kreis:

Alle Geofence-Objekte in einem Kreis, dessen Mittelpunkt den Breitengrad 51.507728 und den Längengrad 7.457153 sowie den Radius 96,6 Meter hat, werden angezeigt.

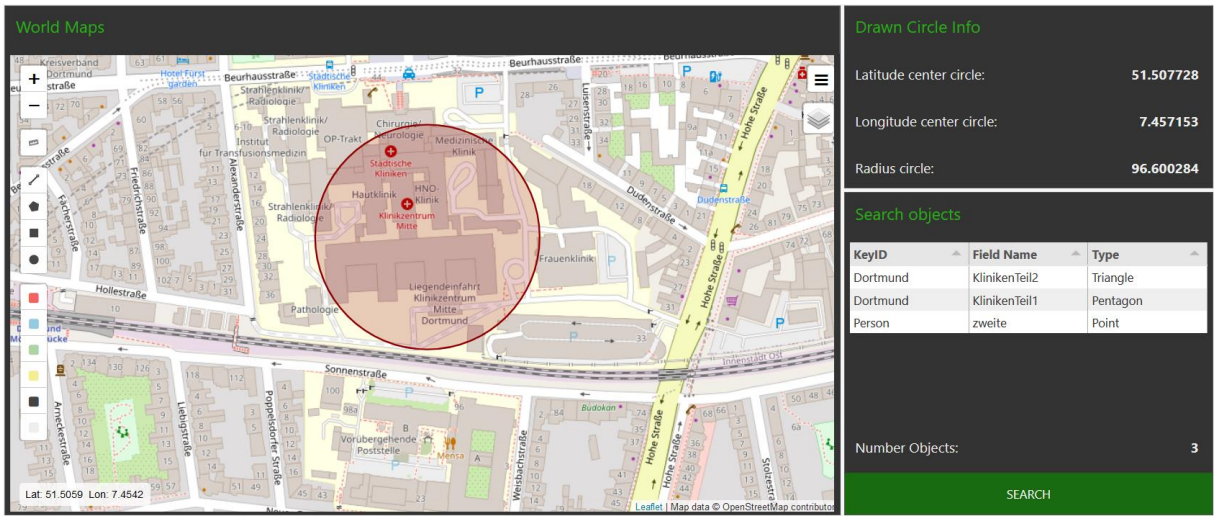

Abbildung 51 Alle definierten Geofence-Objekte im Kreis.

6.2.7 Entfernen aller Objekte basierend auf der KeyID:

Ähnlich wie bei der Java App kann auch das Objekt der Node-Red App mit der KeyID entfernt werden.

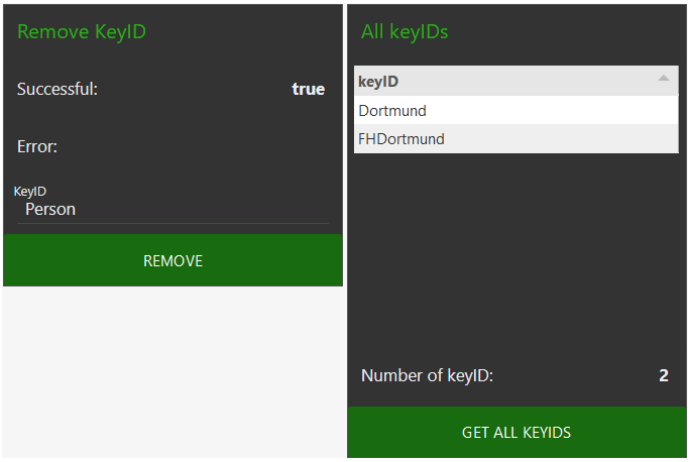

Abbildung 52 Entfernen aller Objekte basierend auf der KeyID.

6.2.8 Entfernen ein bestimmtes Objekt:

Ähnlich wie in der Java-App kann ein Geofence-Objekt in der Node-Red-App auch mit einer bestimmten KeyID und einem bestimmten Field Name gelöscht werden.

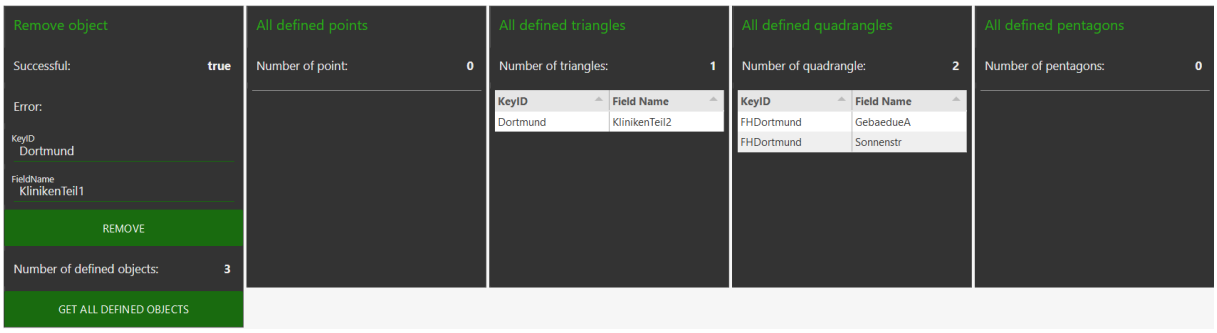

Abbildung 53 Entfernen ein bestimmtes Objekt.

6.2.9 Änderung die KeyID:

In dieser Node-Red App kann nicht nur das Objekt gelöscht, sondern auch die KeyID geändert werden.

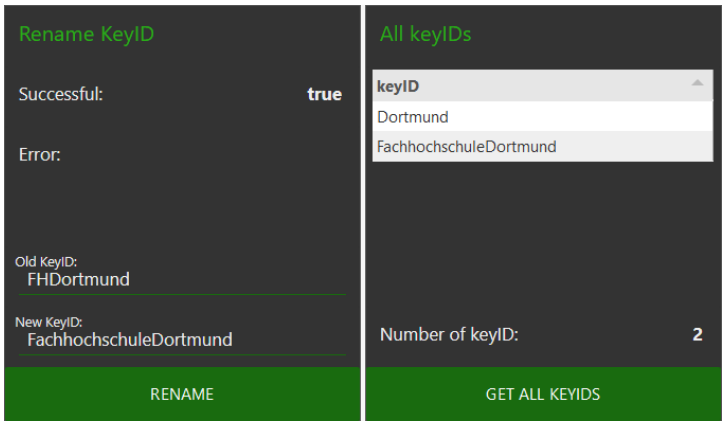

Abbildung 54 Änderung die KeyID

6.2.10 Änderung den Server auf schreibgeschützt:

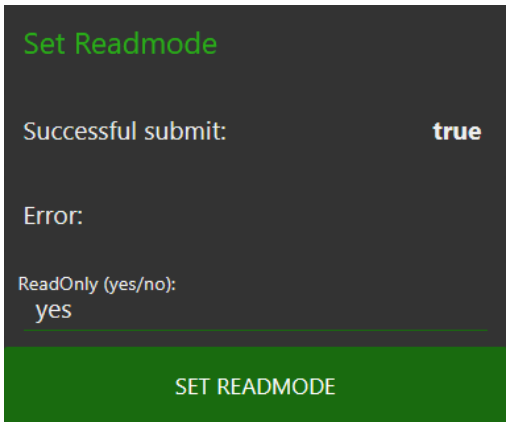

Abbildung 55 Änderung den Server auf schreibgeschützt.

6.2.11 Pingen den Server an:

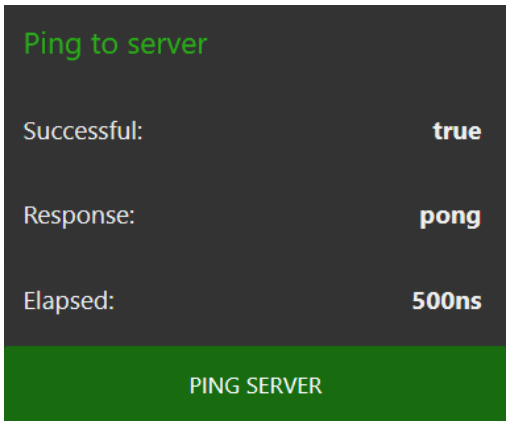

Abbildung 56 Pingen den Server an.

6.2.12 Abruf die aktuellen Serverinformationen:

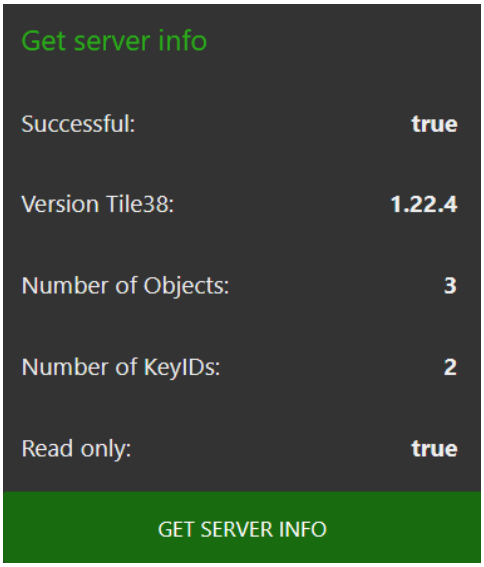

Abbildung 57 Abruf die aktuellen Serverinformationen.

7 Fazit und Ausblick

Heutzutage sind IoT-Anwendungen ein wesentlicher Bestandteil des Alltags geworden. Eine solche Anwendung ist Geofence, eine Anwendung, die in einer Vielzahl von Bereichen eingesetzt werden kann, von Smart Homes über Shopping-Display-Werbung bis hin zu Bereichsmanagement und Menschen.

Um Geofence anwenden zu können, muss zunächst ein System zum Definieren und Verwalten von Objekten erstellt werden, damit die aktuelle Position des Objekts leicht erkannt werden kann. Tile38 Plattform ist eine Open Source-Anwendung mit dem Ziel, ein solches Geofence-System zu erstellen. Alle Objekte in Tile38 können mit dem GeoJSON-Datentyp, der nun in Datenbanksystemen und gängigen Anwendungen sehr verbreitet ist, verwaltet werden.

Mit der HTTP-Request können Geofence-Objekte problemlos bearbeitet und überprüft werden. Seitdem wird die Verwaltung und Verwendung von Tile38 durch HTTP-Request auf Plattformen und Programmiersprachen wie C ++, Java, Python, Node-RED unkomplizierter.

In diesem Projekt wurden einige Funktionen der Tile38-Plattform über Java- und Node-RED-Anwendungen vorgestellt. Diese Anwendungen können zunächst intern von Entwicklern verwendet werden, um beispielsweise Geofence-Objekte zu erstellen, den Standort von Geofence-Objekten zu überprüfen und Geofence-Server zu testen.

Mit dem Entwicklungspotential von Geofence und der Tile38-Plattform sind die oben beschriebenen Funktionen nicht nur für Entwickler für den internen Gebrauch gedacht, sondern können sich auch zu ausgereifteren Anwendungen für Benutzer entwickeln.

8 Literaturverzeichnis

[1] „What is IOT?,“ [Online]. Unter:

https://www.educba.com/what-is-iot/. [Zugegriffen am 15.11.2020]

[2] Dr. Ovidiu Vermesan SINTEF, Norway, Dr. Peter Friess EU, Belgium, "Internet of Things: Converging Technologies for Smart Environments and Integrated Ecosystems", ISBN: 978-87-92982-96-4(E-Book), River Publishers, 2013.

[3] „Best IoT Applications,“ [Online]. Unter:

http://www.techgeekbuzz.com/iot-applications/. [Zugegriffen am 10.11.2020]

[4] „How to use geofencing in push notifications,“ [Online]. Unter:

http://blog.apps-builder.com/how-to-use-geofencing-push-notifications/. [Zugegriffen am 16.11.2020]

[5] „Intelligent Car Parking Management,“ [Online]. Unter:

https://www.infsoft.com/use-cases/intelligent-car-parking-management. [Zugegriffen am 5.11.2020]

[6] „QR code access control reader with TCP/IP,“ [Online]. Unter:

https://eurocloud.en.ecplaza.net/products/qr-code-access-control-reader-with_4340355. [Zugegriffen am 5.11.2020]

[7] „Tile38 Webseite,“ [Online]. Unter:

https://github.com/tidwall/tile38. [Zugegriffen am 18.10.2020]

[8] „NodeRed Webseite,“ [Online]. Unter:

https://nodered.org/. [Zugegriffen am 18.10.2020]

[9] „Weatherlink Dashboard,“ [Online]. Unter:

https://github.com/MatsA/Davis-Weatherlink-dashboard. [Zugegriffen am 14.10.2020]

[10] „Hypertext Transfer Protocol,“ [Online]. Unter:

https://de.wikipedia.org/wiki/Hypertext_Transfer_Protocol. [Zugegriffen am 14.10.2020]

[11] „OSI model,“ [Online]. Unter:

https://infosys.beckhoff.com/english.php?content=../content/1033/tf6310_tc3_tcpip/9007199338987915.html&id=. [Zugegriffen am 14.10.2020]

[12] Rozik, A.S., Tolba, A.S. and El-Dosuky, M.A., „Design and Implementation of the Sense Egypt Platform for Real-Time Analysis of IoT Data Streams,“ in *Advances in Internet of Things*, ISSN Online: 2161-6825, Scientific Research Publishing Inc., 2016, s. 69.

9 Anhang

- Hausarbeit dieses Projekts als PDF.
- Anleitung für Java und Node-RED Applikation als PDF.
- Java Applikation (Ordner: geofencing_java).
- Node-RED Applikation ([Iuk Projekt-Anh Tu Nguyen]Node-RED Flows.json).