

Dokumentation

Inhalt

Projektplanung	2
Projektidee.....	2
Aufgabenverteilung.....	2
Geschäftsprozess:	2
Test der Luftqualität im Rahmen der Filterentwicklung.....	2
Implementierung	5
AirPollution Service	5
Aufbau und Funktionsweise	5
Performance.....	6
City-Convert-WebService	7
Graphische Darstellung	12
Abbildungsverzeichnis	16

Projektplanung

Projektidee

Wir haben schon zeitig begonnen zu überlegen, welches Thema wir nutzen sollen, sind aber auf kein wirkliches Ergebnis gekommen. Schließlich sind wir übereingekommen, die frei nutzbare Air -Pollution-API von OpenWeather zu nutzen. Diese war leicht verständlich und gut dokumentiert. Zudem war sie kostenfrei und notfalls konnte man einen zweiten Account erstellen, sollte der Key gesperrt werden.

Die konkrete Planung der Umsetzung erfolgte erst einiges später. Dabei haben wir uns darauf geeinigt, eine Forschergruppe zu sein, welche Pilotprojekte gestartet hat für Filteranlagen bei Biomasse-Kraftwerken und die jetzt die Luftverschmutzung überprüfen möchten.

Um dies zu erreichen, wollten wir einen webservice und eine Graphische Auswertung entwickeln. Die Daten sollten von der Open-Weather-API kommen.

Aufgabenverteilung

Die Aufgabenverteilung in der Gruppe war folgende:

Justin Unger	Maik Blümel	Maximilian Hausknecht
Entwicklung des CityConvert Webservice	Entwicklung des AirPollution Webservice	Implementierung der Graphischen Oberfläche
Dokumentation der Projektidee	Dockerimages erstellen und in Porttainer hochladen	Einbinden der Webservices in die Graphische Oberfläche und einbinden von Google-Maps und Google-Diagrams in die Graphische Oberfläche
Dokumentation der Implementierung des CityConvert Webservices	Dokumentation der Implementierung des AirPollution Webservice	Dokumentation der Implementierung der Graphischen Oberfläche
Entwicklung des Geschäftsprozess (auch zusammen mit dem BPNM Diagramm)	Entwicklung des Geschäftsprozess (auch zusammen mit dem BPNM Diagramm)	Entwicklung des Geschäftsprozess (auch zusammen mit dem BPNM Diagramm)

Der Aufwand war in etwa gleich und diese Tabelle stellt auch nur die Verantwortlichkeiten.

Geschäftsprozess:

Test der Luftqualität im Rahmen der Filterentwicklung

Das am Ende dieses Kapitels dargestellt BPMN-Diagramm zeigt, die der Test der Luftqualität abgelaufen ist. Gestartet wird der Prozess, indem die Labor-Tests positive, verlässliche Ergebnisse geliefert haben. Wenn diese Ergebnisse von einer Zweiten unabhängigen Forschergruppe ebenfalls bestätigt wurden, wird das Pilotprojekt gestartet und die Filter in 3 Biogasanlagen eingebaut. Der erhöhte Aufwand durch die Kontrollgruppe wird dadurch gerechtfertigt, dass die Einbauten der Filter teuer sind. Dies erhöht den Druck erfolgreiche Ergebnisse zu liefern, sodass man nur sehr vielversprechende Filter einbauen möchte. Die Herstellung der Filter erfolgt im

Labor und der Einbau erfolgt durch eine Baufirma der Wahl. Es sollte dazu ein Ausschreiben geben, da es keine drängenden Zeitpläne in dieser Entwicklung gibt und man das verlässlichste und günstigste Angebot nutzen möchte. Wenn der Einbau abgeschlossen ist, beginnt die Überprüfung der Luftqualität. Es werden Daten direkt an unserem Standort erhoben und verglichen mit den Daten der Wetterstationen in der Umgebung. Zudem wird die Entwicklung der Luftqualität bei umliegenden Wetterstationen aufgezeichnet und ausgewertet. Diese Entwicklung wird von uns überwacht und solange keine Verschlechterung der Luftqualität feststellbar ist, laufen die Messungen 3 Jahre. Nach diesen 3 Jahren, werden die Daten veröffentlicht und sollten nach Möglichkeit von anderen Filterentwicklern und Wissenschaftlern ausgewertet werden. Wenn die Ergebnisse besser sind als die der alten Filter, werden die neuen Filter die alten Filter baldmöglichst ersetzen.

Im Falle das die Daten sagen, dass unser Filter schlechter ist als der alte Filter, dann wird dieser Filterentwurf verworfen und die Entwicklung eines neuen Filters würde anlaufen. Dies sollte durch die intensive Laborprüfung aber nicht passieren, da ansonsten 3 Jahre und sehr viel Geld für nichts verwendet wurden.

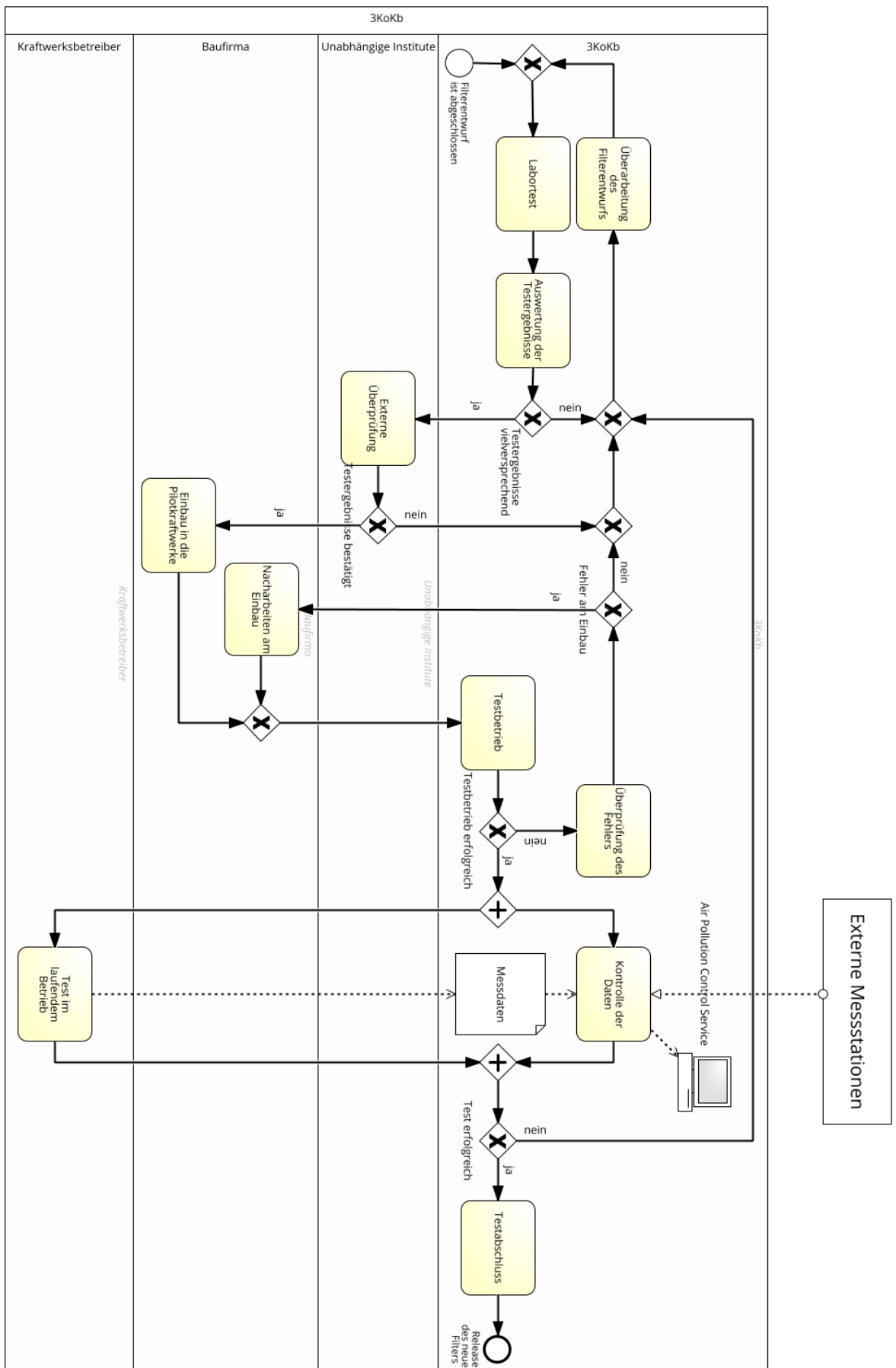



Abbildung 1 BPMN-Diagramm

Implementierung

AirPollution Service

Aufbau und Funktionsweise

Der AirPollution Service stellt die zur Überprüfung der Luftqualität nötigen Daten bereit. Dazu greift er bei Eingabe eines Städtenamens auf den City-Convert-Service zurück, um die nötigen Koordinaten zu ermitteln. Die gewünschten Messwerte werden von OpenWeatherMap bereitgestellt. Dabei können entweder die aktuellen Werte oder Werte für eine bestimmte Zeitspanne abgefragt werden. Die Zeitspanne muss in Form von Start- und Enddatum als Unixzeit angegeben werden. Aus den zurückgemeldeten Daten werden die Messwerte extrahiert und als JSONArray zurückgegeben. Die genaue Struktur wird in dargestellt.

The image shows a JSON structure for 'PollutionData'. It is a nested object with three main properties: 'main', 'components', and 'dt'. The 'main' property contains an 'aqi' (integer) and a description 'Air Quality Index'. The 'components' property is an object containing various pollutant names and their concentrations as floats. The 'dt' property is a number representing the date in Unix format. The pollutants listed are co, no, no2, o3, so2, pm2_5, pm10, and nh3.

PollutionData ▾ {	
main	main ▾ {
	aqi integer
	Air Quality Index
	}
components	Components ▾ {
	co number(\$float)
	concentration of carbon monoxide
	no number(\$float)
	concentration of nitrogen monoxide
	no2 number(\$float)
	concentration of nitrogen dioxide
	o3 number(\$float)
	concentration of ozone
	so2 number(\$float)
	concentration of sulphur dioxide
	pm2_5 number(\$float)
	concentration of fine particles matter
	pm10 number(\$float)
	concentration of coarse particulate matter
	nh3 number(\$float)
	concentration of ammonia
	}
dt	number
	date in unix format
}	

Abbildung 2 Struktur der Rückgabe einer Abfrage an den AirPollution Service

Die Anzahl der Elemente ist dabei direkt abhängig von der Größe des gewählten Zeitraums, da für jede volle Stunde ein Satz an Messdaten übermittelt wird. Bei jedem übergebenen Datensatz ist auch das Datum enthalten, um eine Zuordnung zu ermöglichen.

Der AirPollution Service kann über die in **Fehler! Verweisquelle konnte nicht gefunden werden.** gezeigten Endpunkte angesprochen werden.

AirPollutionService

GET	/actualPollutionIn	get the actual pollutiondata by cityname	-
GET	/HistoryPollutionIn	get the history pollutiondata by cityname	
GET	/actualPollutionCoord	get the actual pollutiondata by longitude and latitude	
GET	/HistoryPollutionCoord	get the history pollutiondata by longitude and latitude	

Abbildung 3 Endpunkte des AirPollution Service

Dabei müssen für die aktuellen Messdaten entweder der Ort (actualPollutionIn) oder die Koordinaten (actualPollutionCoord) angegeben werden. Für eine Abfrage vergangener Daten (HistoryPollutionIn oder HistoryPollutionCoord) muss zusätzlich noch ein Zeitraum (startDate/endDate) in Unixzeit angegeben werden. Nachdem eine Überprüfung der eingegebenen Daten durch den City-Convert-Service erfolgt ist, wird eine der beiden in **Fehler! Verweisquelle konnte nicht gefunden werden.** gezeigten Funktionen aufgerufen und die Messwerte von OpenWeatherMap abgefragt. Anschließend werden die Daten in der in **gezeigten** Struktur übermittelt.

```
@Produces(MediaType.APPLICATION_JSON)
public JSONArray getHistoryPollutionData(float lat, float lon, int start, int end) {
    //get Data from OpenWeatherMap
    final Response response = hwt.queryParam("lat",lat).queryParam("lon",lon).queryParam("start",start).queryParam("end",end).queryParam("appid", APIKEY).request(MediaType.APPLICATION_JSON).get();
    final JSONObject jsonObject = Json.createReader(response.readEntity(InputStream.class)).readObject();
    final JSONArray componentData = jsonObject.getJSONArray("list");
    if(response != null){
        response.close(); //close connection
    }
    //return pollutiondata
    return componentData;
}

@Produces(MediaType.APPLICATION_JSON)
public JSONArray getPollutionData(float lat, float lon) {
    //get Data from OpenWeatherMap
    final Response response = hwt.queryParam("lat",lat).queryParam("lon",lon).queryParam("appid", APIKEY).request(MediaType.APPLICATION_JSON).get();
    final JSONObject jsonObject = Json.createReader(response.readEntity(InputStream.class)).readObject();
    final JSONArray componentData = jsonObject.getJSONArray("list");
    if(response != null){
        response.close(); //close connection
    }
    //return pollutiondata
    return componentData;
}
```

Abbildung 4 Abfrage der Messwerte von OpenWeatherMap

Performance

Um die Performance zu testen wurden mit Hilfe von Postman jeweils 100 Abfragen an die Endpunkte gesendet. Dabei wurden für die einzelnen Endpunkte folgende mittlere Antwortzeiten ermittelt:

- /actualPollutionIn: 882ms
- /HistoryPollutionIn: 769ms
- /actualPollutionCoord: 176ms
- /HistoryPollutionCoord: 176ms

Der deutliche Unterschied in der mittleren Antwortzeit zwischen den Endpunkten, die auf Koordinaten zurückgreifen und den Endpunkten, die als Parameter einen Städtenamen erhalten, wird darauf zurückgeführt, dass der City-Convert-Webservice schneller eine Rückgabe liefert. Es wird vermutet, dass dies durch die unterschiedliche Anzahl an Einträgen für Städtenamen und Koordinaten verursacht wird.

City-Convert-WebService

Der City-Convert-Service soll sicherstellen, dass die Daten korrekt aufbereitet werden und somit der Air-Pollution-Service korrekt arbeiten kann. Ebenfalls greift das Frontend darauf zu, um zu einer Stadt die Koordinaten oder anders herum zu bekommen.

Die Daten werden als JSON-Objekt übergeben.

Die Übergebenen Daten sind in der folgenden Struktur:

```
{
    „id“: „XX“,
    „cityName“: „XX“,
    „country“: „XX“,
    „coord“:
    {
        „lat“: „XX“,
        „lon“: „XX“,
    }
}
```

Diese enthält alle wichtigen Daten, um eine Stadt zu identifizieren und mit den Daten weiter zu arbeiten. Die IDs stimmen auch mit den IDs von Open-Weather-API überein.

Um diese Daten zu erhalten, nutzt der City-Convert-Service eine Datei, welche Open-Weather zur Verfügung gestellt hat, in welcher alle Städte aufgelistet sind, aus welchen Open-Weather Daten erhält. Dies sind 200 000 Einträge.

Die Daten darin sind wie in Abbildung 5 Informationen des CityConvert WebServices gespeichert.

Die Endpunkte sind folgende:

- (1) :port/cityconvert/city?cityName=name
- (2) :port/cityconvert/coord?lat=X&lon=Y

Der Endpunkt (1) gibt alle Informationen zu einer Stadt mit dem eingegebenen Namen zurück. Da aber der Name nicht eindeutig ist, kann es passieren, dass man die „falschen“ Informationen bekommt.

Um dies zu vermeiden haben wir den zweiten Endpunkt hinzugefügt. Dieser bietet die Möglichkeit mithilfe Längen- und Breitengrade in eine Stadt umzuwandeln. Dabei wird folgende Formel verwendet, um die nächste Stadt zu den eingegebenen Längen- und Breitengraden zu ermitteln. Dabei ist V1 ein Vektor aus den eingegebenen Koordinaten und V2 ein Vektor aus den gerade ausgelesenen Koordinaten. S soll der Abstand sein:

$$V1 = \begin{pmatrix} x1 \\ y1 \end{pmatrix}$$

$$V2 = \begin{pmatrix} x2 \\ y2 \end{pmatrix}$$

$$S = \sqrt{(x1 - x2)^2 + (y1 - y2)^2}$$

Es wird dann die Stadt genutzt, welche am nächste liegt.

Sollte es zu einem Fehler kommen, werden als ID, Längen- und Breitengrad auf -1 gesetzt und der name und country werden auf die Fehler-Informationen gesetzt. Auf die ID und die Koordinaten kann getestet werden. Für Debug-Zwecke sind die Fehlerinformationen wichtig.

```
[
{
  "id": 833,
  "name": "Heşâr-e Sefîd",
  "state": "",
  "country": "IR",
  "coord": {
    "lon": 47.159401,
    "lat": 34.330502
  }
},
{
  "id": 2960,
  "name": "'Ayn Ḥalâqîm",
  "state": "",
  "country": "SY",
  "coord": {
    "lon": 36.321911,
    "lat": 34.940079
  }
},
]
```

Abbildung 5 Informationen des CityConvert Webservices

Die Endpunkte sind folgende:

Servers

▾

CityConvert ▾

GET **/city** get citydata by cityname

GET **/coord** get citydata by coordinates

Schemas ▾

city ▾ {

id	number(\$long)
cityName	string(\$text)
country	string
coord	> {...}

}

Abbildung 6

Der Endpunkt „/city“ gibt alle Informationen zu einer Stadt mit dem eingegebenen Namen zurück. Da aber der Name nicht eindeutig ist, kann es passieren, dass man die „falschen“ Informationen bekommt.

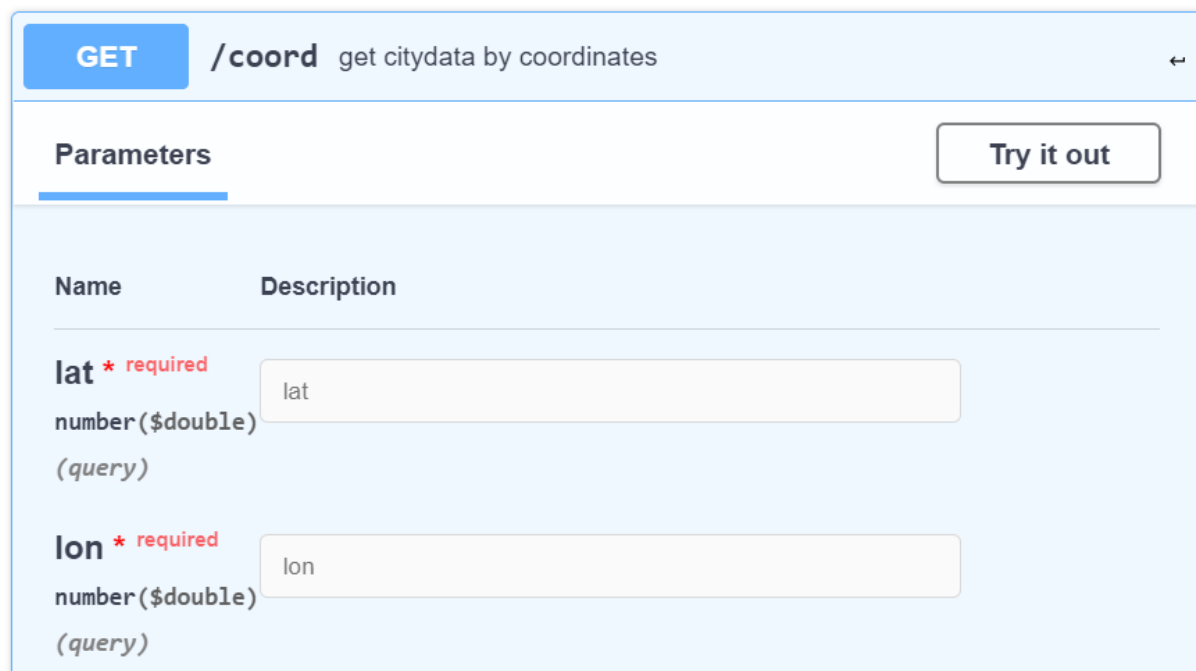


Swagger UI for the **GET /city** endpoint, labeled "get citydata by cityname". The interface includes a "Parameters" section with a "Try it out" button. The parameter table shows:

Name	Description
cityName * required string (query)	Name of the city

A text input field for the query parameter is shown with the placeholder text "cityName - Name of the city".

Abbildung 7



Swagger UI for the **GET /coord** endpoint, labeled "get citydata by coordinates". The interface includes a "Parameters" section with a "Try it out" button. The parameter table shows:

Name	Description
lat * required number(\$double) (query)	lat
lon * required number(\$double) (query)	lon

Text input fields for the query parameters "lat" and "lon" are shown.

Abbildung 8

Sollte es zu einem Fehler kommen, werden als ID, Längen- und Breitengrad auf -1 gesetzt und der name und country werden auf die Fehler-Informationen gesetzt. Auf die ID und die Koordinaten kann getestet werden. Für Debug-Zwecke sind die Fehlerinformationen wichtig.

Die Performance ist eher schlecht für einen Webservice. Folgende Werte sind bei 100 Tests rausgekommen:

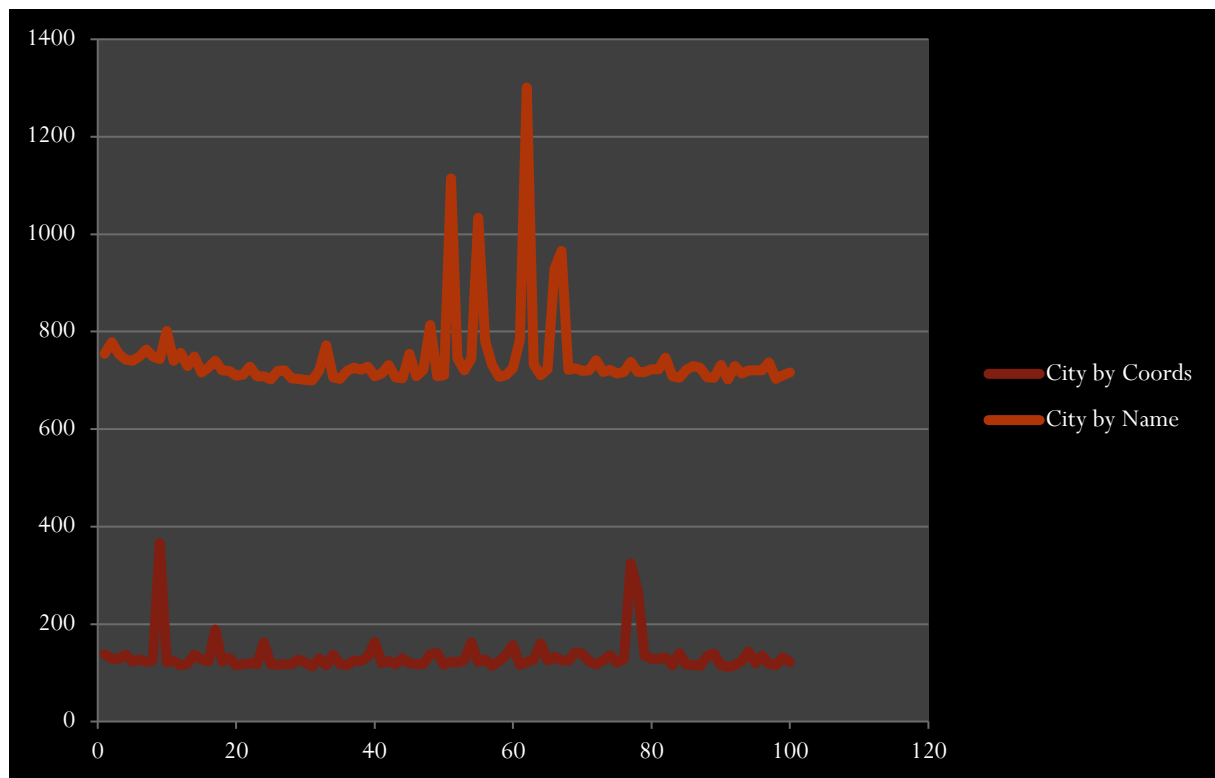


Abbildung 9

Um dies zu vermeiden haben wir den zweiten Endpunkt hinzugefügt. Dieser bietet die Möglichkeit mithilfe Längen- und Breitengrade in eine Stadt umzuwandeln. Dabei wird folgende Formel verwendet, um die nächste Stadt zu den eingegebenen Längen- und Breitengraden zu ermitteln. Dabei ist V1 ein Vektor aus den eingegebenen Koordinaten und V2 ein Vektor aus den gerade ausgelesenen Koordinaten. S soll der Abstand sein:

$$V1 = \begin{pmatrix} x1 \\ y1 \end{pmatrix}$$

$$V2 = \begin{pmatrix} x2 \\ y2 \end{pmatrix}$$

$$S = \sqrt{(x1 - x2)^2 + (y1 - y2)^2}$$

Es wird dann die Stadt genutzt, welche am nächste liegt.

Durchschnitt		
Maximum	1301	366
Minimum	700	112
Standartabweichung	82,9991349	35,8135952

Man sieht deutlich, dass die Anfrage über den Namen mehr Zeit benötigt. Das liegt daran, dass die verwendete Datei sehr groß ist, da sie Städte Weltweit berücksichtigt.

Der Endpunkt der Koordinaten ist schneller, da in seiner Datei nur die Städte von Deutschland berücksichtigt werden.

Für unsere Anwendung ist eine solche Verzögerung aber unbedeutend, da es keine Hochgeschwindigkeitsanwendung ist.

Sollte man aber die Laufzeit verbessern wollen, könnte man entweder eine Datenbank benutzen, um die Informationen zu erhalten, die Datei aufteilen und parallel durchsuchen oder man schaltet vor die weltweite Datei die Datei für Deutschland. Die letzte Variante hätte aber das Risiko, dass Städte außerhalb von Deutschland deutlich langsamer gefunden werden.

Graphische Darstellung

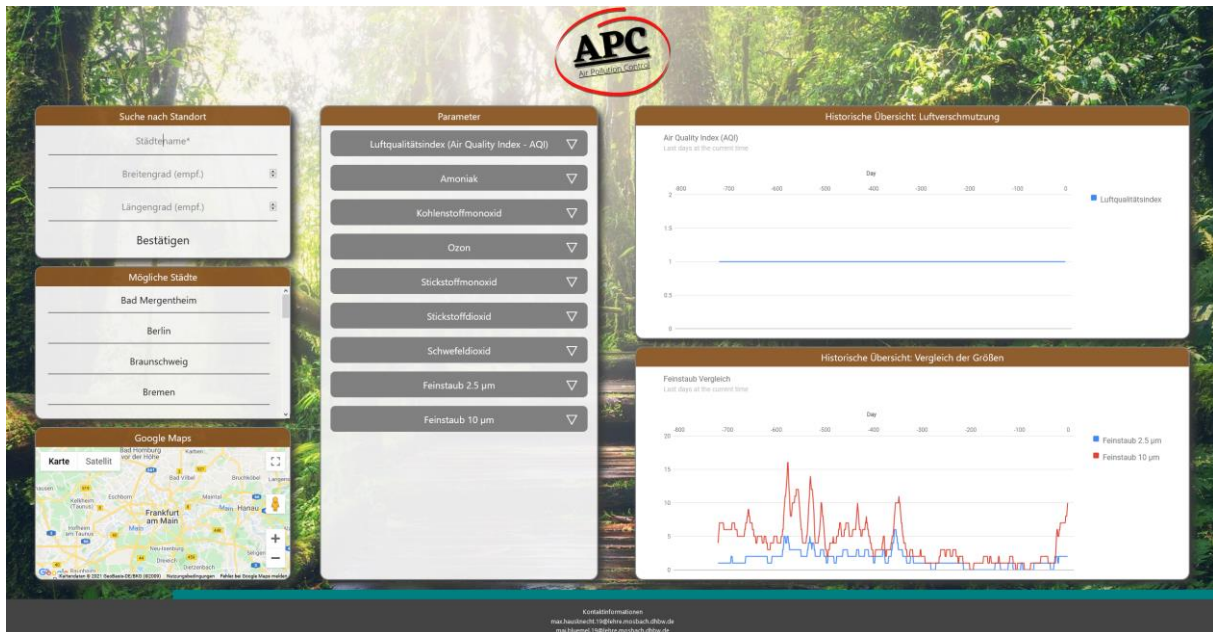


Abbildung 10 GUI der Webseite

Das Frontend ist recht einfach gehalten und es wurde versucht, den Blick auf wenige, wichtige Objekte zu richten.

Als erstes erkennt man mittig, oben platziert unser Logo „APC – Air Pollution Control“. Dieses ist eher schlicht und auf die Wortmarke konzentriert.



Abbildung 6 Logo

Auch das Hintergrundbild fällt ins Auge und soll zum einen unsere Verbindung zur Natur bzw. der Umwelt zeigen, zum anderen dient es als farblicher Akzent zu den milchig weißen, mit Inhalt gefüllten „Containern“.

Um hier einen passenden, kontrastreichen, aber auch angenehmen Übergang zu schaffen, stehen die Überschriften auf einem leicht durchsichtigen Braun. Dieses Braun stellt auch unsere Verbindung zwischen Hintergrundbild (Grün-Braun des Waldes) und Logo (rote Umrandung) dar.

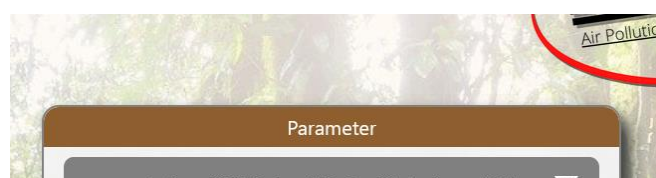


Abbildung 11 Farblicher Übergang

Für die Auswahl des Standortes stehen dem Nutzer auf der linken Seite entweder ein Formular mit der Eingabemöglichkeit des Städtenamens, des Längen- und des Breitengrades, sowie eine Schnellauswahl verschiedener Städte zur Verfügung. Zur Kontrolle, dass auch wirklich der richtige Standort ausgewählt wurde, wird am unteren linken Rand eine Google Maps Karte gezeigt.

The screenshot displays a user interface for location selection. It consists of three main sections:

- Suche nach Standort**: A search form with fields for 'Städtename*' (city name), 'Breitengrad (empf.)' (latitude), and 'Längengrad (empf.)' (longitude), followed by a 'Bestätigen' (confirm) button.
- Mögliche Städte**: A scrollable list of suggested cities, including 'Bad Mergentheim', 'Berlin', 'Braunschweig', and 'Bremen'.
- Google Maps**: A map view centered on Frankfurt am Main, showing surrounding areas like Karkheim, Eschborn, and Main.

Abbildung 12 Darstellung des Eingabebereichs

Um die wichtigsten Daten nochmal hervorzuheben, befinden sich am rechten Rand zwei Diagramme. Im oberen Diagramm kann man den historischen Verlauf der Luftqualität betrachten und im unteren Diagramm kann man die Entwicklung der Feinstaubkonzentration in der Luft in Abhängigkeit von der Zeit erkennen. Wichtig war es uns, dass man auf einen Blick die Luftqualität in der Umgebung erkennen kann und auch sieht, welcher Zusammenhang zwischen der Luftqualität und der Feinstaubkonzentration herrscht.

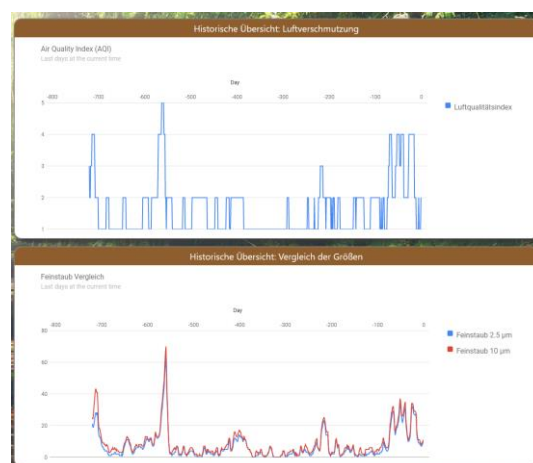


Abbildung 13 Darstellung der Diagramme

Die Diagramme werden, wie die Google-Maps-Map auch, über eine Google API aufgerufen. Dies geschieht via JavaScript im Code:

```

<script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"></script>
<script type="text/javascript">
  google.charts.load('current', {'packages':['line']});
  google.charts.setOnLoadCallback(drawChart);

  function drawChart() {

    var data = new google.visualization.DataTable();
    var x = "<?php
      foreach($histResponse as $stuff){
        echo $stuff['main']['aqi'].\"-\"; } ?>";
    x = x.split("-");
    data.addColumn('number', 'Day');
    data.addColumn('number', 'Luftqualitätsindex');

    var y = (x.length-1);
    var xz = -y;
    var z = -y;
    while(y>0){
      data.addRow([
        [(xz-z), parseInt(x[y])]
      ]);
      z++;
      y=y-1;
    }

    var options = {
      chart: {
        title: 'Air Quality Index (AQI)',
        subtitle: 'Last days at the current time'
      },
      axes: {
        x: {
          0: {side: 'top'}
        }
      }
    };

    var chart = new google.charts.Line(document.getElementById('topChart'));
    chart.draw(data, google.charts.Line.convertOptions(options));
  }
</script>

```

Abbildung 14 Beispielhafter Aufruf eines Diagrammes

Weitere Bestandteile der Luft, welche zur Verschmutzung beitragen, sind unter dem Feld „Parameter“ gelistet. Diese weiteren Parameter sind aufklappbar und beinhalten allgemeine Informationen, Informationen über die Gesundheits- und Umweltschädlichkeit, sowie über den aktuellen Wert am ausgewählten Standort. Das Aufklappbare „Menü“ dient hier der Übersichtlichkeit, da wir den Nutzer der Webseite nicht mit nicht benötigten Informationen „überfluten“ möchten.



Abbildung 15 Aufklappbare Parameter

Zuletzt geben wir am unteren Rand der Seite die Kontaktinformationen, das Copyright und auch das Impressum an. Um den Seitenabschluss zu symbolisieren wurde hier auf ein dunkles Grau als Hintergrundfarbe gesetzt. Der

Übergang zum Hintergrundbild und dem Rest der Seite stellt ein dunkler, türkiser Streifen dar. Dieser wurde über einen Schatten des „Footer“-Elements realisiert.



Abbildung 12 Footer

Impressum

Angaben gemäß § 5 TMG

Maximilian Hausknecht
Im Burgstädtle 11
6
74821 Mosbach

Kontakt

Telefon: [-]
E-Mail: hausknechtmax01@gmail.com

Haftung für Inhalte

Als Diensteanbieter sind wir gemäß § 7 Abs.1 TMG für eigene Inhalte auf diesen Seiten nach den allgemeinen Gesetzen verantwortlich. Nach §§ 8 bis 10 TMG sind wir als Diensteanbieter jedoch nicht verpflichtet, übermittelte oder gespeicherte fremde Informationen zu überwachen oder nach Umständen zu forschen, die auf eine rechtswidrige Tätigkeit hinweisen.

Verpflichtungen zur Entfernung oder Sperrung der Nutzung von Informationen nach den allgemeinen Gesetzen bleiben hiervon unberührt. Eine diesbezügliche Haftung ist jedoch erst ab dem Zeitpunkt der Kenntnis einer konkreten Rechtsverletzung möglich. Bei Bekanntwerden von entsprechenden Rechtsverletzungen werden wir diese Inhalte umgehend entfernen.

Haftung für Links

Unser Angebot enthält Links zu externen Websites Dritter, auf deren Inhalte wir keinen Einfluss haben. Deshalb können wir für diese fremden Inhalte auch keine Gewähr übernehmen. Für die Inhalte der verlinkten Seiten ist stets der jeweilige Anbieter oder Betreiber der Seiten verantwortlich. Die verlinkten Seiten wurden zum Zeitpunkt der Verlinkung auf mögliche Rechtsverstöße überprüft. Rechtswidrige Inhalte waren zum Zeitpunkt der Verlinkung nicht erkennbar.

Eine permanente inhaltliche Kontrolle der verlinkten Seiten ist jedoch ohne konkrete Anhaltspunkte einer Rechtsverletzung nicht zumutbar. Bei Bekanntwerden von Rechtsverletzungen werden wir derartige Links umgehend entfernen.

Urheberrecht

Die durch die Seitenbetreiber erstellten Inhalte und Werke auf diesen Seiten unterliegen dem deutschen Urheberrecht. Die Vervielfältigung, Bearbeitung, Verbreitung und jede Art der Verwertung außerhalb der Grenzen des Urheberrechtes bedürfen der schriftlichen Zustimmung des jeweiligen Autors bzw. Erstellers. Downloads und Kopien dieser Seite sind nur für den privaten, nicht kommerziellen Gebrauch gestattet.

Soweit die Inhalte auf dieser Seite nicht vom Betreiber erstellt wurden, werden die Urheberrechte Dritter beachtet. Insbesondere werden Inhalte Dritter als solche gekennzeichnet. Sollten Sie trotzdem auf eine Urheberrechtsverletzung aufmerksam werden, bitten wir um einen entsprechenden Hinweis. Bei Bekanntwerden von Rechtsverletzungen werden wir derartige Inhalte umgehend entfernen.

Abbildung 16 Verwendetes Impressum

Abbildungsverzeichnis

Abbildung 1 BPMN-Diagramm	4
Abbildung 2 Struktur der Rückgabe einer Abfrage an den AirPollution Service	5
Abbildung 3 Endpunkte des AirPollution Service	6
Abbildung 4 Abfrage der Messwerte von OpenWeatherMap	6
Abbildung 5 Informationen des CityConvert WebServices.....	8
Abbildung 6	8
Abbildung 7	9
Abbildung 8	9
Abbildung 9	10
Abbildung 10 GUI der Webseite	12
Abbildung 11 Farblicher Übergang	12
Abbildung 12 Darstellung des Eingabebereichs	13
Abbildung 13 Darstellung der Diagramme	13
Abbildung 14 Beispielhafter Aufruf eines Diagrammes	14
Abbildung 15 Aufklappbare Parameter.....	14
Abbildung 16 Verwendetes Impressum	15