

Gr. 1: Heinzelreiter/
Kurschl*Software Engineering*

Übungen zu SWK5/WEA5 im WS 2018/19

Gr. 2: Pimminger/
KurschlName Simon GUSCHLBAUERAufwand in h 100Gr. 3: Sklenitzka/
NadschlägerAbgegeben am 20.01.2019

Punkte _____

korr. _____

WeatherTracer – wetr ['wetər]

Ein System zum Visualisieren von lokalen Wetterdaten

Es existieren zahlreiche Web-Seiten und mobile Anwendungen, die uns Wettervorhersagen bis weit in die Zukunft anbieten. Nicht immer sind diese Prognosen seriös, da sie auf relativ wenigen Wetterstationen basieren (und eine langfristige Prognose mit großen Unsicherheiten verbunden ist). Aufgrund der geographischen Struktur Österreichs weist das Wetter in unserem Land teilweise große regionale Unterschiede auf. Das ist ein weiterer Faktor, der eine Wetterprognose in Österreich sehr schwierig macht.

Die Basis für jede Wettervorhersage sind Daten von einem möglichst dichten Netz an Wetterstationen. Nicht nur offizielle Stellen wie die Zentralanstalt für Meteorologie und Geodynamik betreiben Wetterstationen, auch eine zunehmende Anzahl von Privatpersonen verfügt über einfache Funkstationen, die über einen Netzwerkanschluss Wetterdaten liefern können.

Im Rahmen dieser Projektarbeit soll eine Anwendung entwickelt werden, welche Daten von vielen Wetterstationen übernehmen kann, diese statistisch auswertet und die Ergebnisse grafisch ansprechend visualisiert. Diese Daten bilden zwar die Basis für eine Wetterprognose, allerdings ist diese Funktionalität nicht Bestandteil dieser Projektarbeit.

Funktionale Anforderungen

Im Rahmen dieser Projektarbeit ist das Softwaresystem zur Verwaltung der Stammdaten von Wetterstationen und der von diesen Stationen gesendeten Messdaten zu entwickeln. Die Stations- und Wetterdaten sind in einer zentralen Datenbank zu speichern, Wetterdaten müssen statistisch ausgewertet und visualisiert werden. Das Softwaresystem besteht aus folgenden Komponenten:

- *Wetr.Server* ist die zentrale Komponente zur Datenverwaltung. *Wetr.Server* verfügt auch über eine flexible und effiziente Abfragekomponente, welche Möglichkeiten zur Datenfilterung und Datenkumulation bietet.
- *Wetr.WebService* exportiert die gesamte Funktionalität von *Wetr.Server* in Form eines Web-Dienstes, der über zwei Endpunkte verfügt. Über einen Endpunkt kann *Wetr.Web* mit dem Server kommunizieren. Über den zweiten Endpunkt können Wetterdaten in das System eingespeist werden.
- *Wetr.Cockpit* dient zur Darstellung der aktuellen Wetterlage und zur Visualisierung der längerfristigen Entwicklung von Wetterdaten. Mit entsprechenden Rechten können die Stammdaten der Wetterstationen gewartet werden (hinzufügen, aktualisieren, löschen).
- *Wetr.Web*: Der Web-Client stellt ebenfalls die aktuelle Wetterlage dar, bietet Möglichkeiten zur Visualisierung von historischen Wetterdaten und hat eingeschränkte Funktionalität zur Administration von Wetterstationen.
- *Wetr.Simulator*: Mit dieser Komponente können automatisiert Wetterdaten generiert und an *Wetr.Server* übergeben werden.

(a) Datenmodell

Im Folgenden werden die wesentlichen Attribute der wichtigsten Entitäten des Systems definiert. Aus der Anforderungsspezifikation können sich zusätzliche Entitäten und weitere Eigenschaften der angeführten Entitäten ergeben, die entsprechend zu ergänzen sind.

- (a.1) *Station*: Für jede Wetterstation sind der Name, der Stationstyp (Fabrikat u. dgl.), die Adresse sowie deren Geokoordinaten (Längen- und Breitengrad) zu speichern. Die Postleitzahl definiert die Zuordnung der Station zu einer Gemeinde und damit auch die Zuordnung zu einem Bezirk und einem Bundesland.
- (a.2) *Community/District/Province*: In diesen Entitäten sind die Stammdaten der Gemeinden, Bezirke und Bundesländer gespeichert. Auf diese Daten wird nur lesend zugegriffen. Die Administration dieser Daten ist nicht Teil dieses Projekts.
- (a.3) *User*: Diese Entität dient zur Verwaltung der Zugangsdaten jener BenutzerInnen, welche die Stammdaten einer Station verändern dürfen.
- (a.4) *Measurement*: Eine Messung enthält unterschiedliche Wetterdaten, wie Lufttemperatur, Luftdruck, Regenmenge, Luftfeuchtigkeit, Windgeschwindigkeit, Windrichtung. Jede Messung enthält einen Messwert und einen Zeitstempel. Messattributen ist eine Einheit (Grad Celsius, km/h etc.) zuzuordnen. Es ist davon auszugehen, dass Wetterdaten unregelmäßig und mit unterschiedlicher Frequenz gemeldet werden.


(b) Wetr.Server

Die Serverkomponente stellt die zentrale Funktionalität für die Clients zur Verfügung.

- (b.1) *Stammdatenverwaltung*: Die Serverkomponente muss eine geeignete Schnittstelle zur Verwaltung der Wetterstationen (hinzufügen, löschen, ändern) anbieten.
- (b.2) *Verwaltung der Messdaten*: Die Wetterdaten sind die wesentlichen Bewegungsdaten des Systems. Es ist eine Funktion zum Hinzufügen eines neuen Messdatensatzes vorzusehen. Messdaten sind immer mit einer Wetterstation verbunden.
- (b.3) *Abfragen*: Die Abfragekomponente ist das Kernmodul des Servers. Das Abfragemodul soll Möglichkeiten zur Datenkumulation und zur Filterung von Wetterdaten bieten:
 - Filtern nach Wetterstation bzw. Region: Eine Region ist durch einen Punkt und einen Radius definiert, kann aber auch eine Gemeinde, ein Bezirk oder ein Bundesland sein.
 - Kumulation von Daten für bestimmte Zeitintervalle (Stunde, Tag, Woche, Monat): Datenkumulation kann Minimum-, Maximum-, Durchschnittsbildung aber auch Aufsummieren der Messwerte bedeuten.

Hier einige Beispiele für Abfragen, welche diese Komponente unterstützen soll:

- Was war die Durchschnitts-/Minimal-/Maximal-Temperatur bei einer bestimmten Wetterstation/in einer Region gruppiert nach Tagen/Wochen/Monaten?
- Was war die tages-/wochen-/monatsweise kumulierte Niederschlagsmenge bei einer bestimmten Wetterstation?
- Was war die durchschnittliche Niederschlagsmenge in einer bestimmten Region gruppiert nach Tagen/Wochen/Monaten?

- (b.4)  *Messdatenanalyse*: Es soll eine einfache Auswertungskomponente geschaffen werden, mit der in regelmäßigen Abständen Warnungen über aktuell auftretende Wetterphänomene über Twitter gegeben werden können. Darunter fallen z.B. Starkregen, Sturm oder gefrierender Regen. Die Wetterwarnung sollte, neben dem auftretenden Phänomen selbst, auch unbedingt die Region beinhalten.

(c) Wetr.WebService

Die für *Wetr.Web* und *Wetr.Simulator* erforderliche Funktionalität ist in Form eines REST-basierten Web-Service zu exportieren. Der Zugriff auf das Web-Service muss nicht abgesichert werden.

(d) Wetr.Simulator

Der Wettersimulator hat die Aufgabe, die Messdaten, die im Realbetrieb von den Wetterstationen geliefert werden, zufallszahlengesteuert zu generieren und über die dafür vorgesehene REST-Schnittstelle in das System einzuspeisen. Der Simulator kann Messwerte in einem vorgegebenen Zeitbereich, der auch in der Vergangenheit liegen kann, simulieren und besitzt eine Zeitrafferfunktion zur beschleunigten Generierung von Daten. Sie können den Simulator dafür verwenden, die Robustheit und Leistungsfähigkeit der Implementierung zu demonstrieren.



(d.1) *Hinzufügen/Entfernen einer Wetterstation*: Aus der Liste der verfügbaren Wetterstationen muss eine Station ausgewählt und zum Simulator hinzugefügt werden können. Da der Simulator mehrere Wetterstationen unterstützen muss, muss dieser Vorgang beliebig oft wiederholbar sein. Auch das Löschen einer Wetterstation ist vorzusehen.

(d.2) *Simulationseinstellungen*: Für einen Simulationsdurchgang muss die simulierte Wetterstation, das Simulationsdatum und die -uhrzeit (Beginn und Ende) sowie die Simulationsgeschwindigkeit (z. B. halbe Geschwindigkeit, reale Geschwindigkeit, 10-fache Geschwindigkeit) festgelegt werden. Verwenden Sie ansprechende Steuerelemente (Combobox, Datumsauswahl, Slider, etc.).

(d.3) *Konfiguration der Messfolgen*: Eine Messfolge ist eine Menge von zusammenhängenden Messwerten über einen gewissen Zeitraum (z. B. Verlauf der Temperatur bei einer bestimmten Wetterstation über einen Tag hinweg). Pro Wetterstation sind mehrere Messfolgen zu simulieren. Für jede Messfolgen müssen folgende Parameter eingestellt werden können:

- Messwert, der simuliert werden soll (z. B. Temperatur, Luftdruck etc.)
- Frequenz, in der neue Daten generiert werden sollen (z. B. alle 10 s)
- Wertebereich (von/bis) für die erzeugten Werte
- Strategie zur Verteilung
 - linearer Anstieg/Abstieg zwischen *von* und *bis*
 - zufälliger Wert zwischen *von* und *bis*
 - mindestens eine frei wählbare zusätzliche Strategie, die einen beliebigen Messwert besonders gut simulieren kann (z. B. Temperaturverteilung über den Tag, ansteigende und abnehmende Regenmenge, etc.)


Generieren Sie möglichst realistische Messfolgen. Beachten Sie, dass ein Messwert nicht vollkommen unabhängig vom vorausgehende Wert ist. Die Temperatur wird beispielsweise in einem kurzen Zeitraum nicht völlig zufällige Werte in einem bestimmten Zahlenbereich annehmen.

- (d.4) *Steuern der Simulation:* Die Simulation kann gestartet und gestoppt werden. Dadurch wird mit der Generierung von Messwerten begonnen bzw. diese wieder angehalten. Die Benutzeroberfläche muss während der Simulation responsiv sein, um jederzeit eine Simulation stoppen zu können.
- (d.5)  *Ändern der Simulationsgeschwindigkeit:* Die Simulationsgeschwindigkeit kann während einer laufenden Simulation verändert werden und wirkt sich unmittelbar aus.
- (d.6) *Darstellung der Wetterstationen:* Die aktuellen Messwerte der Wetterstationen sowie der Messdatenverlauf muss grafisch ansprechend aufbereitet werden. Als Orientierungshilfe können reale Wetterstationen dienen. Die simulierten Wetterstationen sollen optisch klar voneinander getrennt werden (z. B. Aufteilung in Reiter o. Ä.).
- (d.7)  *Lastsimulation:* Im Simulator kann die Generierung von Messwerten für eine größere Anzahl von Wetterstationen ausgelöst werden. Die resultierenden Messwertfolgen müssen nicht grafisch dargestellt werden. Welche Parameter Sie für diesen Anwendungsfall konfigurierbar machen, ist Ihnen überlassen.

(e) Wetr.Cockpit

Wetr.Cockpit ist ein Client, der vor allem von Spezialisten (Meteorologen) genutzt wird. Diese Komponente hat zwei wesentliche Funktionalitäten: die Bearbeitung von Wetterstationen und die Analyse von Wetterdaten.

- (e.1) *Verwaltung von Wetterstationen:* Es können neue Wetterstationen erfasst, editiert und gelöscht werden (siehe (b.1)). Das Löschen einer Wetterstation soll nur möglich sein, wenn für diese Wetterstation noch keine Messdaten vorhanden sind.
- (e.2) *Wetteranalysen:* So wie in (b.3) beschrieben, sollen auf Basis der erfassten Wetterdaten Abfragen durchgeführt werden können. Die BenutzerIn muss die dafür benötigten Abfrageparameter, wie Filterbedingungen, Parameter zur Gruppierung der Daten, in einer übersichtlichen Art und Weise eingeben können. Orientieren Sie sich an der in (b.3) beschriebenen Funktionalität.

 Das Ergebnis einer Abfrage muss in grafischer Form angezeigt werden. Die Verwendung einer bestehenden Chart-Komponente wird empfohlen.

Einerteams können sich auf eine tabellarische Darstellung der Daten beschränken.


Es ist besonders darauf zu achten, dass während der Durchführung einer Abfrage – die längere Zeit in Anspruch nehmen kann – die Benutzeroberfläche nicht blockiert ist.

(f) Wetr.Web

Mithilfe dieses Klienten können sich BenutzerInnen einen Überblick über die aktuelle Wettersituation verschaffen. Wichtig ist u. a., dass BenutzerInnen eine Präferenzliste führen können, in die sie Wetterstationen eintragen und so die Wetterentwicklung an mehreren Orten verfolgen können. Grundsätzlich ist bei der Benutzerschnittstelle auf eine einfache Benutzbarkeit und eine optisch ansprechende Umsetzung zu achten. Als ein gutes Beispiel sei auf <https://www.wunderground.com/> hingewiesen.

- (f.1) *Öffentliche Suche:* BenutzerInnen sollen nach Wetterdaten für einen bestimmten Ort suchen können. Die Ergebnisse sollen sowohl grafisch (z.B. mit Hilfe von Charts), als auch tabellarisch angezeigt werden. Achten Sie auf eine möglichst übersichtliche Darstellung. Erlauben Sie den BenutzerInnen, Details (bspw. Stationstyp) zu Wetterstationen ein- und ausblenden zu können.
- (f.2) *Visualisierung.* Zeigen Sie die Wetterdaten in unterschiedlichen Varianten an, beispielsweise stündliche Wetterdaten, Tageszusammenfassung mit niedrigstem und höchstem Wert und stündliche Temperaturkurve. Zeigen Sie auch Statistiken zu den einzelnen Messdaten an (grafisch und tabellarisch). Erlauben Sie dem Benutzer diese anzupassen (Zeitbereich, Ort, etc.).
- (f.3) *Login:* Ein Benutzer soll die Möglichkeit haben, sich einzuloggen. Ein besonderer Bonus hier wäre die Verwendung von OAuth und OpenID.
- (f.4) *Benutzerdefinierte Ansicht / Verwalten von Präferenzen:* Die BenutzerInnen sollen sich ihr eigenes „Wetterdashboard“ zusammenstellen können. Die Präferenzen (z.B. Temperatureinheit, Wetterstationen) sollen über die Benutzersession hinaus im Browser (Local Storage) verwaltet/gespeichert werden. Die BenutzerInnen können die anzuzeigenden Wetterstationen auf Basis einer Suchfunktion selbst bestimmen. Geben Sie den BenutzerInnen auch die Möglichkeit, die Art der Anzeige (z.B. stündliche Wetterdaten, Tageszusammenfassung, etc.) zu konfigurieren und speichern Sie diese Einstellungen, sodass beim nächsten Login das benutzerdefinierte Dashboard wieder angezeigt wird.
- (f.5) *Verwalten eigener Wetterstationen und Eingabe eigener Wetterdaten:* Schaffen Sie eine Möglichkeit, dass registrierte BenutzerInnen eigene Wetterstationen verwalten und Wetterdaten eingeben können. Das soll allerdings nur authentifizierten Benutzern erlaubt sein. Wir wollen damit sicherstellen, dass nur hochwertige Daten in die Datenbank gelangen.

Ergebnisse

Die in der Anforderungsdefinition festgelegten Funktionen sind in Einer- oder Zweierteams zu realisieren. Anforderungen, die mit dem Symbol  gekennzeichnet sind, müssen nur von Zweierteams ausgearbeitet werden. Die WEA5 zuzuordnende Funktionalität ist in Form von Einzelarbeiten umzusetzen.

Die Entwicklung der Projektarbeit ist auf GitHub durchzuführen. Alle Teams bekommen ein Repository zugewiesen, auf welches das Team und die Übungsleiter Zugriff haben.

Erstellen Sie für alle Dokumente und Quelltext-Dateien eine übersichtliche Verzeichnisstruktur und packen Sie diese in eine ZIP-Datei. Dieses Archiv stellen Sie Ihrem Übungsleiter auf der E-Learning-Plattform in den Kursen zur SWK5-Übung bzw. zu WEA5 zur Verfügung. Achten Sie darauf, dass Sie nur die relevanten (keine generierten) Dateien in das Archiv geben. Große Archive mit unnötigen Dateien vergeuden Speicherplatz und können daher zu Punkteabzügen führen. Große Binärdateien, wie Komponenten von Drittherstellern oder umfangreiche Datenbanken können Sie in Ihrem GitHub-Repository ablegen. Vermerken Sie dies aber in der Dokumentation Ihres Projekts.

Konzentrieren Sie sich bei der Dokumentation dieser Projektarbeit auf die Darstellung der Architektur und des Designs Ihrer Anwendung. Verwenden Sie dazu gängige Darstellungsformen wie ER- und UML-Diagramme (Use-Case-, Klassen- und Sequenzdiagramme). Zeigen Sie anhand von zwei aussagekräftigen Anwendungsfällen u. a. die Interaktionen der verschiedenen Systemkomponenten mithilfe eines Sequenzdiagramms – auch über Systemgrenzen hinweg (WEA5 und SWK5). Versuchen Sie diesen Teil der Arbeit besonders übersichtlich zu gestalten. Die Dokumente sind ausschließlich in Form von PDF-Dateien abzugeben.

1 Installation

1.1 Voraussetzungen

- NPM
- Angular CLI

1.2 Verwendete Technologien

- Bootstrap
- Material Design for Bootstrap
- Font Awesome

1.3 Inbetriebnahme

Um das Projekt in Betrieb zu nehmen muss zunächst *npm install* durchgeführt werden um die benötigten Pakete zu installieren. Danach kann mit *ng serve* die Angular Anwendung gestartet werden. Standardmäßig ist die Anwendung dann unter **http://localhost:4200/** erreichbar.

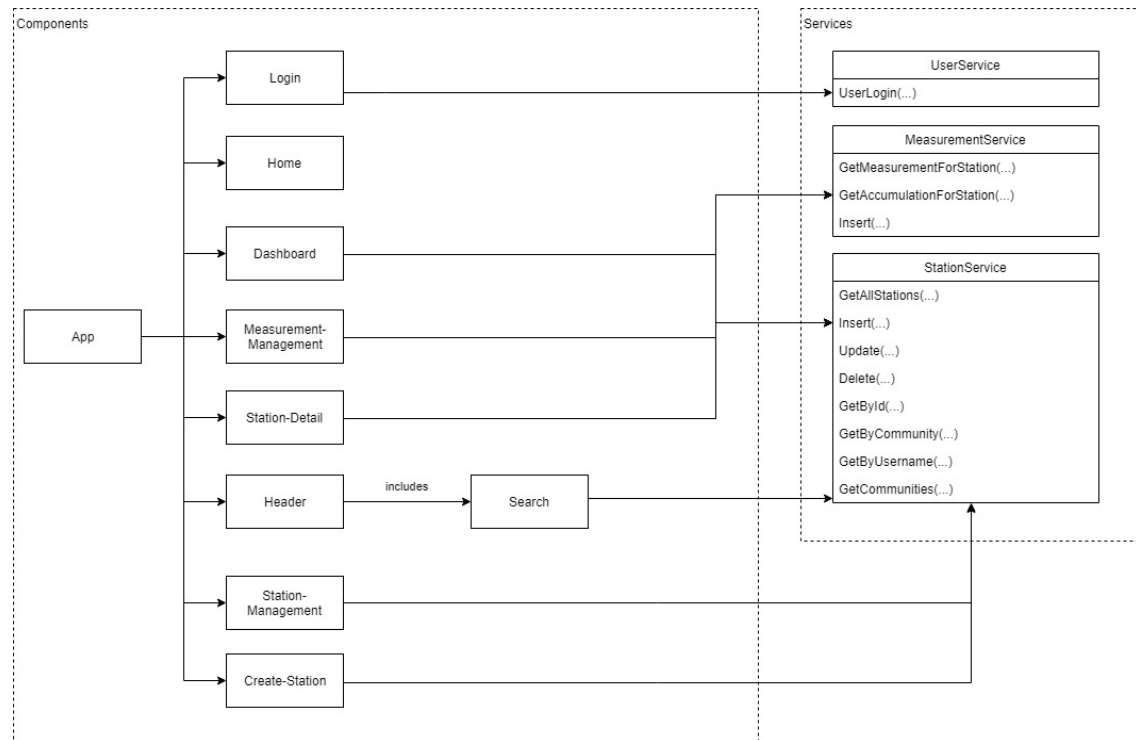
Die Adresse des zu verwendenden REST-Services kann in der Datei *api-configuration.ts* festgelegt werden. Standardmäßig ist dort **http://localhost:54405** hinterlegt.

Achtung

Um den Service ordnungsgemäß zu verwenden muss eine entsprechender REST-Service eingetragen und verfügbar sein.

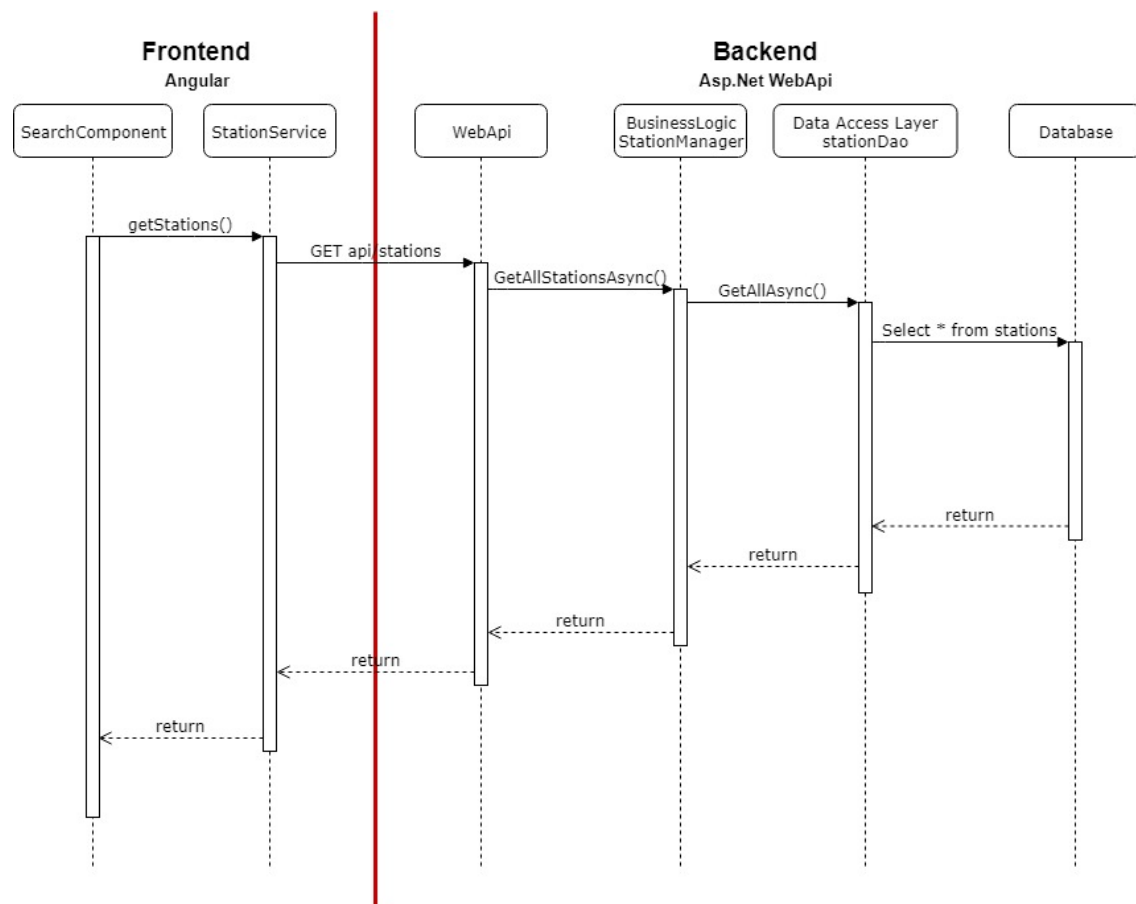
2 Architektur

2.1 Komponentendiagramm



2.2 Sequenzdiagramm

Dieses Sequenzdiagramm zeigt den Aufbau der Architektur am Beispiel der Stationssuche.



3 Services

Die Services und Modelklassen wurden automatisch aus dem vom REST-Service zur Verfügung gestellten swagger.json generiert. Deswegen ist die Namensgebung und Codestruktur nicht optimal.

3.1 measurement-service

- `MeasurementGetMeasurementForStation(...)`

Führt eine einfache Abfrage der Messungen für eine Station durch

- `MeasurementGetAccumulationForStation(...)`

Führt eine akkumulierte Abfrage der Messungen für eine Station durch

- `MeasurementInsert(...)`

Gibt eine neu erstellte Messungen an den REST-Service weiter.

3.2 user-service

- `UserLogin(params: UserLoginParams)`

Nimmt Benutzernamen und Passwort entgegen und gibt diese an den REST-Service weiter.

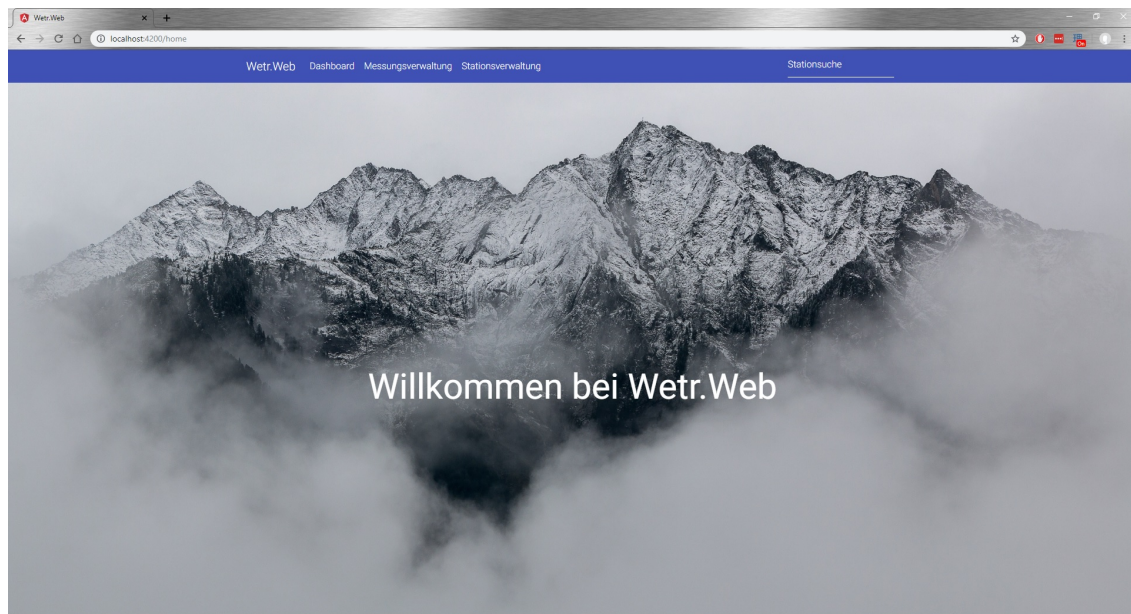
3.3 station-service

- `StationGetAllStations(...)`
Fragt alle Stationen ab.
- `StationInsert(...)`
Fügt eine neue Station ein
- `StationUpdate(...)`
Ändert eine bestehende Station
- `StationDelete(...)`
Löscht eine bestehende Station
- `StationGetById(...)`
Fragt Station nach Stations-Id ab
- `StationGetByCommunity(...)`
Fragt Stationen nach Gemeinde ab
- `StationGetByUsername(...)`
Fragt Stationen nach Ersteller ab
- `StationGetCommunities(...)`
Fragt alle Gemeinden ab

4 Komponenten

4.1 Home

Diese Komponente stellt den Einstiegspunkt in die Anwendung dar. Von hier aus sind alle wichtigen Bereiche der Anwendung erreichbar. Außerdem wird nach man nach dem Ausloggen wieder auf diesen Bereich weitergeleitet.

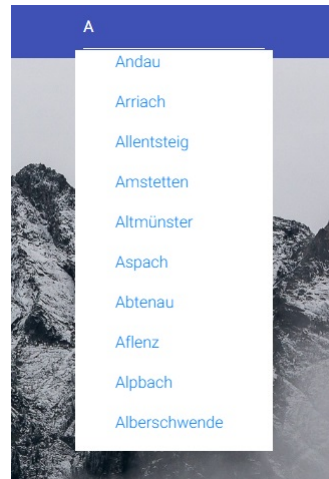


4.2 Header

Die Header-Komponenten ist immer sichtbar und beinhaltet die Navigation sowie die öffentliche Stationssuche.

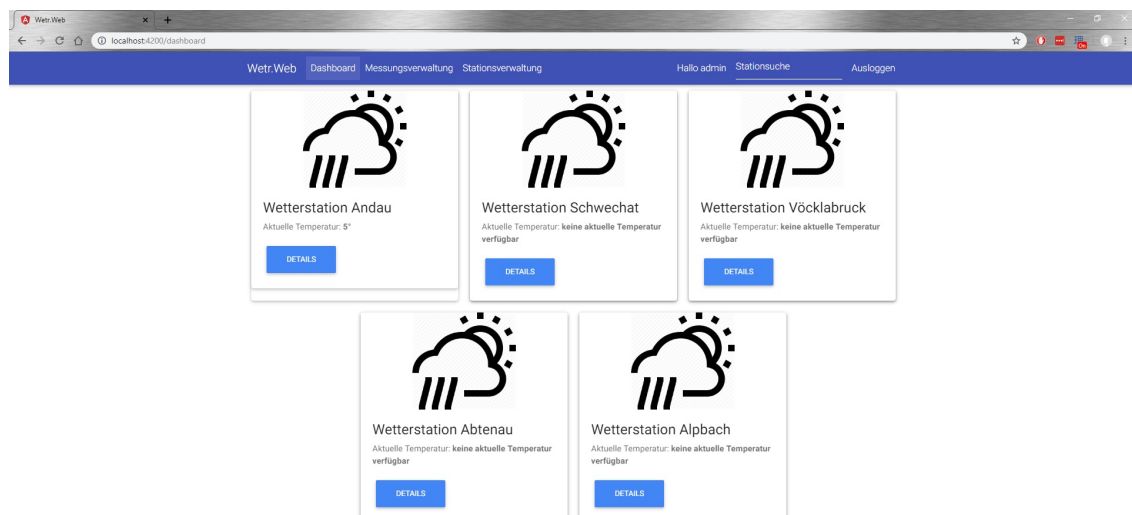
4.3 Search

Die Search-Komponente ist in die Header-Komponente integriert und kann von jedem Benutzer verwendet werden um nach Stationen zu suchen und die Detailseiten der Stationen aufzurufen.



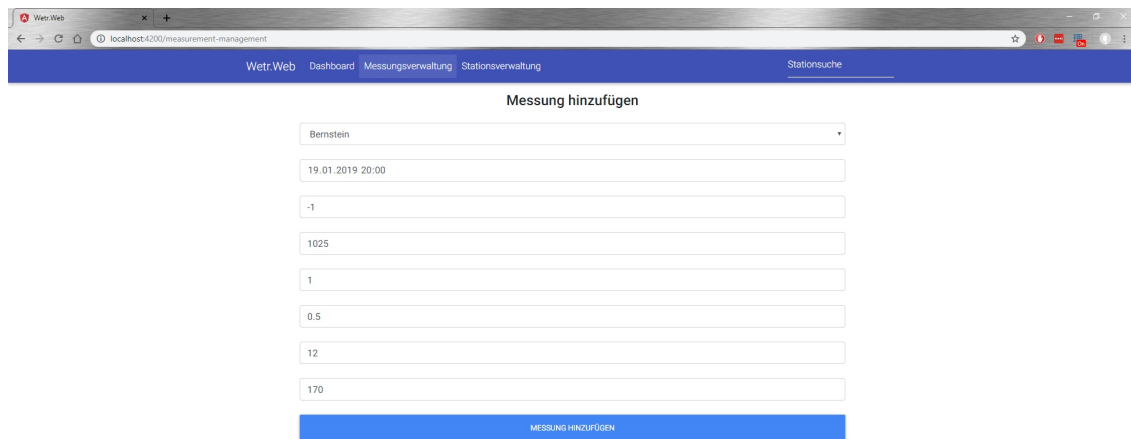
4.4 Dashboard

Die Dashboard-Komponente ist nur für eingeloggte Benutzer verfügbar und zeigt die aktuelle Temperatur sowie Links zu den vorher favorisierten Stationen des Benutzers an.



4.5 Measurement-Management

Diese Komponente dient dazu um neue Messungen anzulegen. Sollten die Angaben nicht korrekt sein, weist eine Meldung darauf hin und der Button kann nicht gedrückt werden. Nach erfolgreichem Absenden gibt eine weitere Hinweismeldung dem Benutzer Feedback über den Status der Operation.



The screenshot shows a web browser window with the URL `localhost:4200/measurement-management`. The page has a blue header with navigation links: `Wetr.Web`, `Dashboard`, `Messungsverwaltung`, `Stationsverwaltung`, and `Stationsuche`. The main content area is titled `Messung hinzufügen` and contains a form with the following fields:

- `Bernstein` (dropdown menu)
- `19.01.2019 20:00` (text input)
- `-1` (text input)
- `1025` (text input)
- `1` (text input)
- `0.5` (text input)
- `12` (text input)
- `170` (text input)

At the bottom of the form is a blue button labeled `MESSUNG HINZUFÜGEN`.

Messung hinzufügen

Messung hinzugefügt

4.6 Station-Management

Diese Komponente dient dazu um bestehende Stationen zu ändern oder zu löschen. Außerdem kann man von hier aus zur Seite „Station anlegen“ navigieren.

The screenshot shows a web browser window with the URL `localhost:4200/station-management`. The page title is "Stationsverwaltung". At the top, there is a navigation bar with links: "Wetr.Web", "Dashboard", "Messungsverwaltung", "Stationsverwaltung", "Hallo admin", "Stationsuche", and "Ausloggen". Below the navigation bar, there are two buttons: "STATION LÖSCHEN" (red) and "STATION ANLEGEN" (green). The main content area is divided into two sections. On the left, there is a list of stations with a dropdown menu. The list includes: Andau, Bernstein, Bruckneudorf, Güssing, Lutzmannsburg, Mattenbourg, Neusiedl Am See, Wörterberg, Arnach, Bad Bleiberg, Dellach, Feistritz Ob Bleiburg, Ferlach, Friesach, Friesach, Gmünd, Kotschach-Mauthen, Mallnitz, Obervellach, Preitenegg, Allertsteig, Amstetten, Bärnkopf, Berndorf, Brunn Am Gebirge, Gänsemdorf, Gumpoldskirchen, Horn, Klausen-Leopoldsdorf, and Langenlois. On the right, there is a form for editing a station. The form has the following fields: "Stationsname:" (text input), "Stationstyp:" (dropdown menu with "Typ 1" selected), "Stationsmarke:" (text input with "Marke 1" selected), "Standort:" (text input with "17,03333282 | 47,77249908" entered), "Straße:" (text input), and "Gemeinde:" (text input with "Andau" selected). At the bottom of the form, there is a blue button labeled "ÄNDERUNG SPEICHERN".

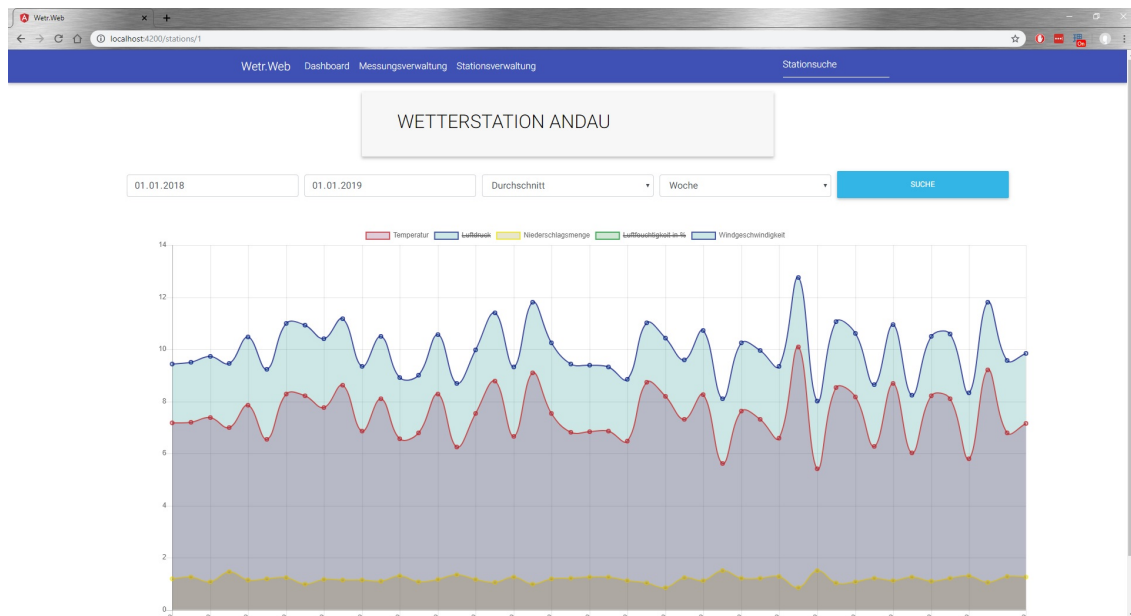
4.7 Create-Station

Diese Komponente dient dazu um neue Stationen anzulegen.

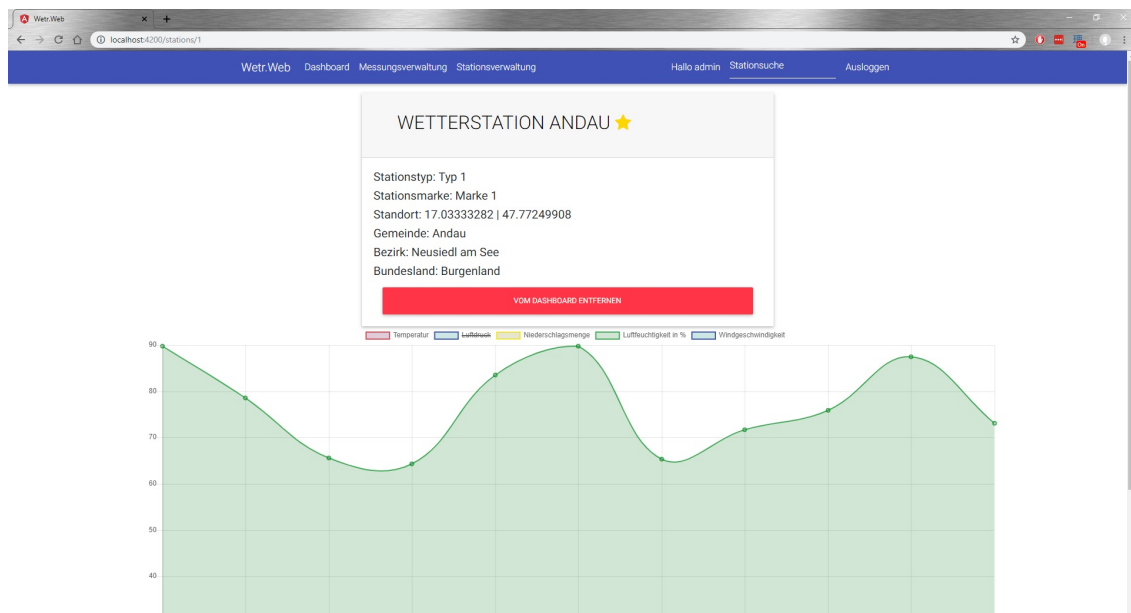
The screenshot shows a web browser window with the URL `localhost:4200/create-station`. The page title is "Station anlegen". At the top, there is a navigation bar with links: "Wetr.Web", "Dashboard", "Messungsverwaltung", "Stationsverwaltung", "Hallo admin", "Stationsuche", and "Ausloggen". Below the navigation bar, there is a form for creating a new station. The form has the following fields: "Stationsname" (text input), "Straße" (text input), "Eisenstadt" (dropdown menu), "Breitengrad" (text input), and "Längengrad" (text input). At the bottom of the form, there is a blue button labeled "STATION HINZUFÜGEN".

4.8 Station-Details

Zeigt alle Daten zu einer Station an sowie Messungsdaten zu dieser Station.

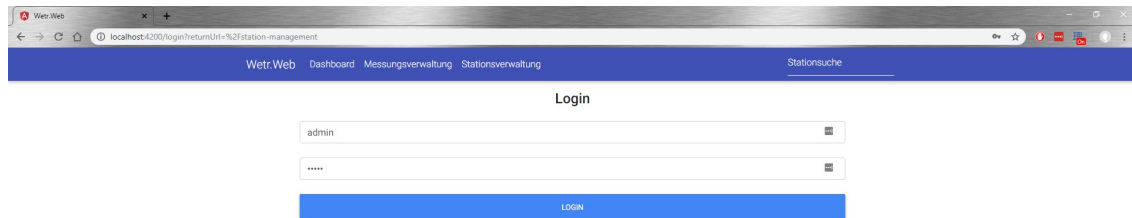


Hier gibt es außerdem die Möglichkeit für eingeloggte Benutzer eine Station zum Dashboard hinzuzufügen



4.9 Login

Auf diese Komponente wird der Benutzer verwiesen wenn er Bereiche der Anwendung aufruft auf die nur eingeloggte Benutzer Zugriff haben.



The screenshot shows a web browser window with the address bar displaying 'localhost:4200/login?returnUrl=%2Fstation-management'. The page has a blue header bar with navigation links: 'Wetr.Web', 'Dashboard', 'Messungsverwaltung', 'Stationsverwaltung', and 'Stationsuche'. The main content area is titled 'Login' and contains two input fields: the first is labeled 'admin' and the second is masked with '*****'. Below these fields is a blue button labeled 'Login'.