

FH HAGENBERG

PROJEKTARBEIT

---

# Weather Tracer - Dokumentation

---

*Autor:*

Daniel ENGLISCH

*Übungsleiter:*

Stefan NADSCHLÄGER

19. Januar 2019

Angular Application



# Inhaltsverzeichnis

|          |                            |          |
|----------|----------------------------|----------|
| <b>1</b> | <b>Inbetriebnahme</b>      | <b>2</b> |
| <b>2</b> | <b>Aufbau</b>              | <b>3</b> |
| <b>3</b> | <b>Details: ApiService</b> | <b>4</b> |
| <b>4</b> | <b>Komponenten</b>         | <b>6</b> |
| 4.1      | App . . . . .              | 6        |
| 4.2      | Login . . . . .            | 6        |
| 4.3      | Home . . . . .             | 7        |
| 4.4      | StationCard . . . . .      | 9        |
| 4.5      | EditStation . . . . .      | 10       |
| 4.6      | AddStation . . . . .       | 11       |
| 4.7      | Search . . . . .           | 11       |
| 4.8      | Query . . . . .            | 12       |
| 4.9      | Dashboard . . . . .        | 16       |
| 4.10     | DashboardCard . . . . .    | 17       |
| 4.11     | Settings . . . . .         | 17       |

# 1 Inbetriebnahme

Um das Wetr-Frontend in Betrieb zu nehmen, muss die aktuellste Version von GitHub<sup>1</sup> heruntergeladen werden. Hierfür müssen zuvor folgende Pakete installiert werden:

- Yarn<sup>2</sup>
- windows-build-tools (*yarn add g windows-build-tools*)
- karma-cli (*yarn add g karma-cli*)
- angular-cli (*yarn add g @angular/cli@6.1.3*)

Natürlich müssen die in der *package.json* Datei spezifizierten Pakete installiert werden mit *yarn install --check-files*. Um die Applikation zu bauen und bereitzustellen muss im Projektverzeichnis der Befehl *ng server --port 80* ausgeführt werden.

Danach kann die Applikation unter *http://localhost/* aufgerufen werden. Beachten Sie, dass für die fehlerfreie Verwendung die Wetr-RestAPI unter dem Port 5000 bereits laufen muss. Falls diese API unter einer anderen *Host:Port*-Kombination erreichbar ist, kann dies in der Datei *src/app/services/api.servive.ts* bei der Konstanten 'apiString' angepasst werden.

---

<sup>1</sup><https://github.com/DanielEnglisch/wetr-frontend>

<sup>2</sup><https://yarnpkg.com/en/docs/install>

## 2 Aufbau

Im Projekt existieren folgende **Services**:

- **ApiService** - Zuständig für die Kommunikation mit der Wetr-RestAPI.
- **SettingsService** - Für das Speichern und Lesen der Benutzereinstellungen im *local-Storage* verantwortlich.

Weiters existieren folgende **Komponenten** welche in Abschnitt 3 genauer beschrieben werden:

- **App** - Hauptkomponente die unter anderem die Navigationsleiste beinhaltet.
- **Login** - Zum Eingeben der Zugangsdaten zur Wetr-Plattform.
- **Home** - Zeigt eine Übersicht über die eigenen Stationen an.
- **StationCard** - Wiederverwendbare Komponente die eine Station repräsentiert.
- **EditStation** - Formular zum Editieren einer Station.
- **AddStation** - Formular zum Hinzufügen einer Station.
- **Search** - Zum Suchen von Stationen in Communities.
- **Query** - Um Daten einer Station abzufragen und neue Messwerte einzufügen.
- **Dashboard** - Zum anzeigen von favorisierten Stationsabfragen pro Benutzer.
- **DashboardCard** - Wiederverwendbare Komponente zum grafischen Darstellen einer gespeicherten Abfrage.
- **Settings** - Lokale Seite zum Ändern von Einstellungen.

**Verwendete externe Pakete:**

*Angular Material, Bootstrap 4, FlashMessages, hightCharts, ng-datetime-picker, fontawesome-icons, ...*

### 3 Details: ApiService

Der *ApiService* wurde nicht mit *Swagger* generiert, sondern selbst geschrieben, um ein besseres Verständnis für die Funktionsweise solcher einer Architektur zu erlangen.

Beim erstmaligen Starten der Applikation wird überprüft, ob sich ein Token im *localStorage* befindet. Falls nicht, wird der Benutzer aufgefordert sich einzuloggen. Beim Einloggen wird ein normaler Post-Request zur API gesendet und bei Erfolg der Token für abgesicherte Routen zurückgegeben. Dieser wird im *localStorage* gespeichert.

Für Anfragen auf gesicherte Routen wird ein Art Wrapper für Get/Post/Put/Delete-Requests verwendet, der als Auth-Header den vorher erwähnten Token mitsendet. Für Anfragen, die dies nicht benötigen, werden die normalen von der Angular-Klasse *HttpClient* zur Verfügung gestellten Methoden verwendet.

Falls der Token abgelaufen ist, wird der Status Code 401 empfangen und zur Login-Seite umgeleitet, um nach erneutem Einloggen einen neuen Token zu erhalten.

Zuständigkeiten der *ApiService* Klasse:

- **Statische Daten**

- *loadStaticData()* - Einmaliges Anfordern von statischen Daten wie *StationTypes* oder *Communities*.
- *revolve...()* - Auflösen einer Id zu der Bezeichnung.
- *get...()* - Rückgabe des angeforderten Arrays.

- **Login**

- *login()* - Anfordern eines Tokens.
- *getEmail()* - Liefert die E-Mail-Adresse des eingeloggten Benutzers.
- *loggedIn()* - Ob gerade ein Benutzer eingeloggt ist.
- *logout()* - Löschen des Tokens.

- **Dashboard<sup>3</sup>**

- *getDashboardQueries()* - Laden der im *localStorage* sich befindenden Dashboard-Queries pro Benutzer.
- *addQueryToDashboard()* - Hinzufügen neuer Dashboard-Queries für den aktuellen Benutzer.
- *removeQueryToDashboard()* - Löschen eines Dashboard-Queries für den aktuellen Benutzer.

---

<sup>3</sup>Für jeden eingeloggten Benutzer individuell.

- **Local Storage**

- *loadLocalStorage()* - Laden der für diese Applikation relevanten localStorage-Daten.
- *saveLocalStorage()* - Speichern der für diese Applikation relevanten localStorage-Daten.

- **Http Communication**

- *Jwt[VERB]()* - Authentifizierte Anfrage an die RestAPI.
- *[VERB]()* - Anonyme Anfrage an die RestAPI.

- **Stations**

- *getMyStations()* - Liefert die Stationen des eingeloggten Benutzers.
- *getStation(id)* - Liefert die Station mit der eingebenden Id.
- *getStationsForCommunity()* - Liefert alle Stationen für die eine Community.
- *deleteStation()* - Löscht eine Station des eingeloggten Benutzers.
- *editStatoin()* - Bearbeitet eine Station des eingeloggten Benutzers.
- *addStation()* - Fügt eine neue Station für den eingeloggten Benutzer hinzu.
- *queryStation()* - Fragt Messdaten für die gewünschte Station ab.
- *addMeasurement()* - Fügt einer Station neue Messdaten hinzu.

## 4 Komponenten

### 4.1 App

Diese Hauptkomponente wird als Template für alle anderen Seiten verwendet. Sie beinhaltet die Navbar mit der auf alle anderen Komponenten dieser Applikation navigiert werden kann. Im eingeloggten Zustand werden alle möglichen Navigationsbutton angezeigt (Abbildung 1). Beim ausgeloggten Zustand wird nur der Button zur öffentlichen Suche und zum Einloggen angezeigt (Abbildung 2).

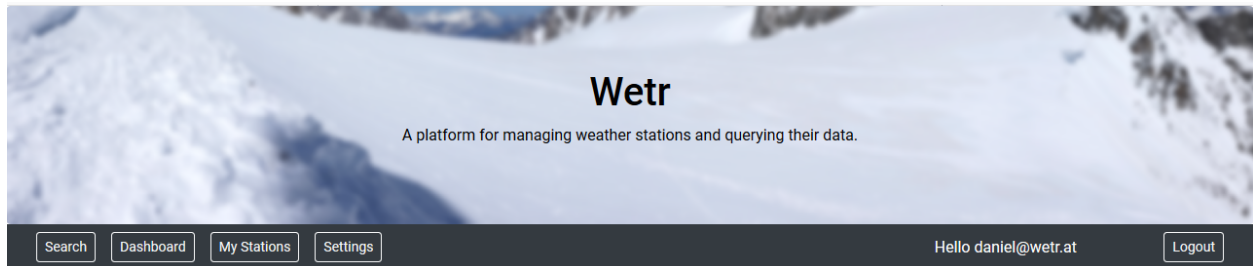


Abbildung 1: Gleichbleibender Inhalt der Seite mit Navbar im eingeloggten Zustand.

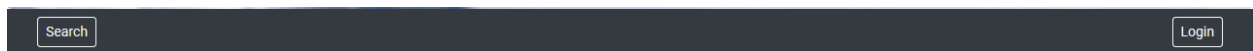
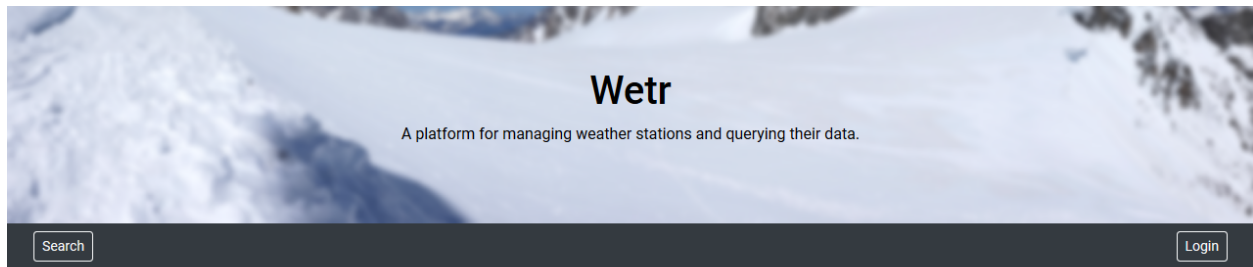


Abbildung 2: Navbar im ausgeloggtem Zustand.

### 4.2 Login

Die Login-Seite beinhaltet zwei Felder zum Eingaben der Benutzerdaten, welche live validiert werden. Nur wenn die Daten valide sind, kann der Login-Button gedrückt werden (Abbildung 3). Während der API-Abfrage wird ein Ladeindikator angezeigt. Solch ein Indikator wird in der gesamten Applikation bei ledenden Elementen angezeigt und wird dahien nicht länger erwähnt. Bei falschen Zugangsdaten wird ein entsprechender Fehler angezeigt (Abbildung 4).



## Login

Email

Password

Login

Abbildung 3: Die Login-Seite.

## Login

Invalid credentials!

Email  
falsch@wetr.at

Password  
●●●●

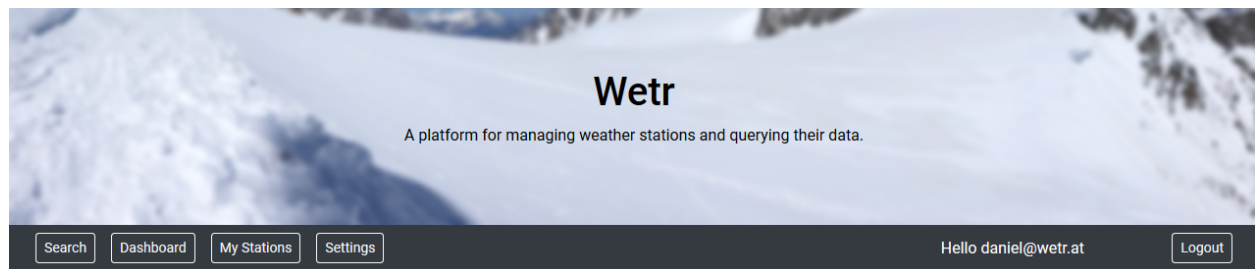
Login

Abbildung 4: Login-Seite mit falschen Zugangsdaten.

### 4.3 Home

Nach dem erfolgreichen Einloggen werden die Stationen des eingeloggten Benutzers angezeigt (Abbildung 5). Es ist hier möglich Details zu den angezeigten Stationen auszu-blenden und neue Stationen anzulegen (siehe Abschnitt 4.6). Die Details zu den angezeigten StationCards wird in Abschnitt 4.4 beschrieben.





## My stations

Hide Details: ☐












| Oma Opa Station   | Dachstation   | Garten  |
|---|---|---|
| <i>Community:</i> Achner  | <i>Community:</i> Dellach im Drautal  | <i>Community:</i> Dellach im Drautal  |
| <i>Type:</i> VAMES  | <i>Type:</i> TAWES  | <i>Type:</i> TAWES  |
|    |    |    |

Abbildung 5: Übersicht über die Stationen des eingeloggten Benutzers.

## 4.4 StationCard

Diese Komponente repräsentiert die Anzeige einer Station. Es gibt vier verschiedene Ansichten der StationCards:

- Ansicht: MyStations, detailreich - Abbildung 6
- Ansicht: MyStations - Abbildung 7
- Ansicht: Search, detailreich - Abbildung 8
- Ansicht: Search - Abbildung 9

Die öffentliche suche wird in Abschnitt 4.7 beschrieben.

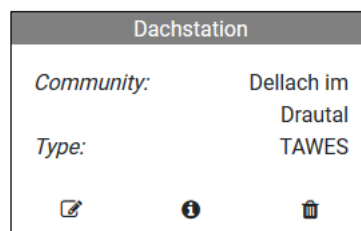


Abbildung 6: StationCard in der Detailansicht mit der Möglichkeit zum Abfragen, Bearbeiten und Löschen.

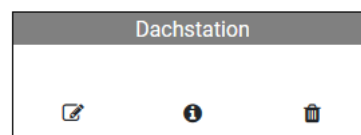


Abbildung 7: StationCard in der reduzierten Ansicht mit der Möglichkeit zum Abfragen, Bearbeiten und Löschen.

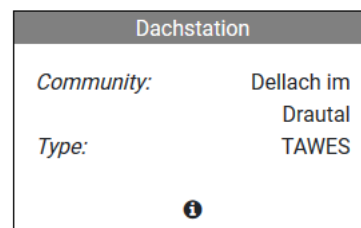


Abbildung 8: StationCard in der Detailansicht mit der Möglichkeit zum Abfragen.

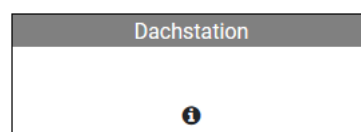
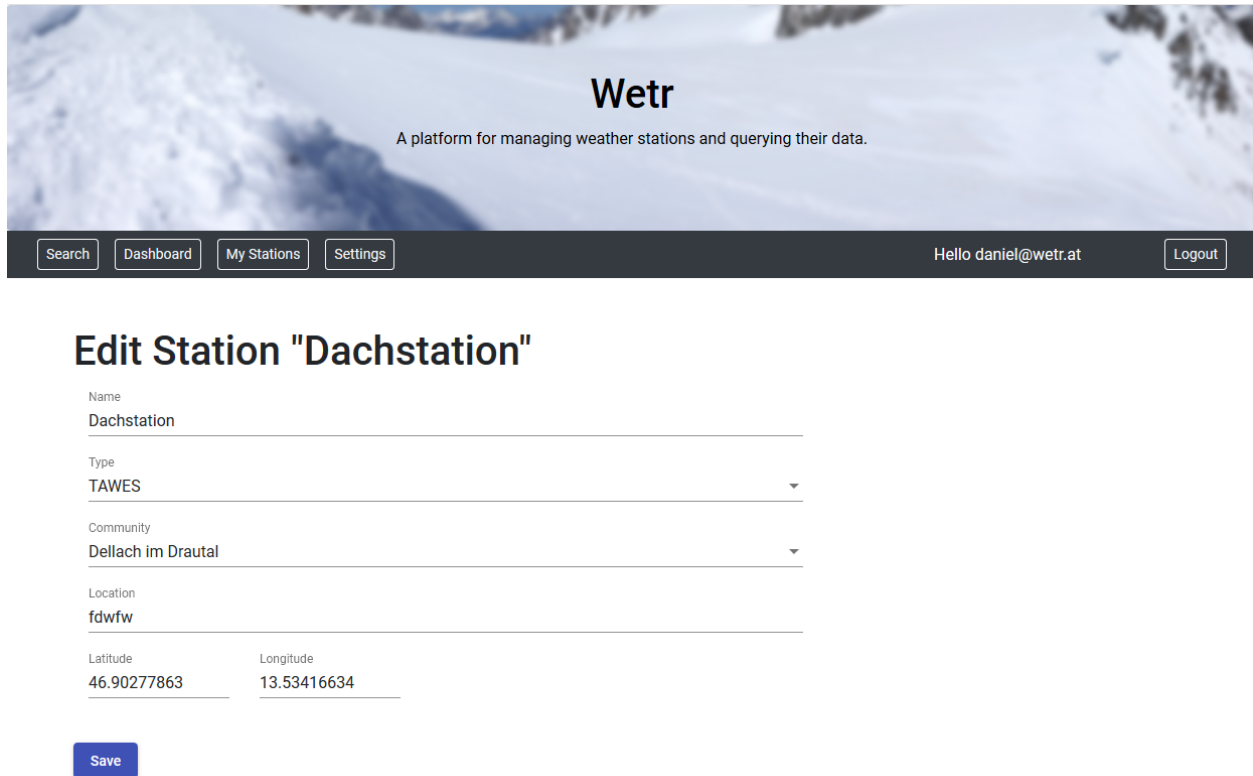


Abbildung 9: StationCard in der reduzierten Ansicht mit der Möglichkeit zum Abfragen.

## 4.5 EditStation

Ansicht zum Editieren von Stationen (Abbildung 10). Es können nur eigene Stationen editiert werden. Das Formular kann nur abgeschickt werden, wenn es valide ist. Beim Erfolgreichen Ändern der Daten bzw. bei Fehlern wird eine entsprechende FlashMessage angezeigt (Abbildung 11). Dies ist im gesamten System gleich und wird daher nicht länger durch Screenshots gezeigt.



**Wetr**  
A platform for managing weather stations and querying their data.

Search Dashboard My Stations Settings Hello daniel@wetr.at Logout

### Edit Station "Dachstation"

Name  
Dachstation

Type  
TAWES

Community  
Dellach im Drautal

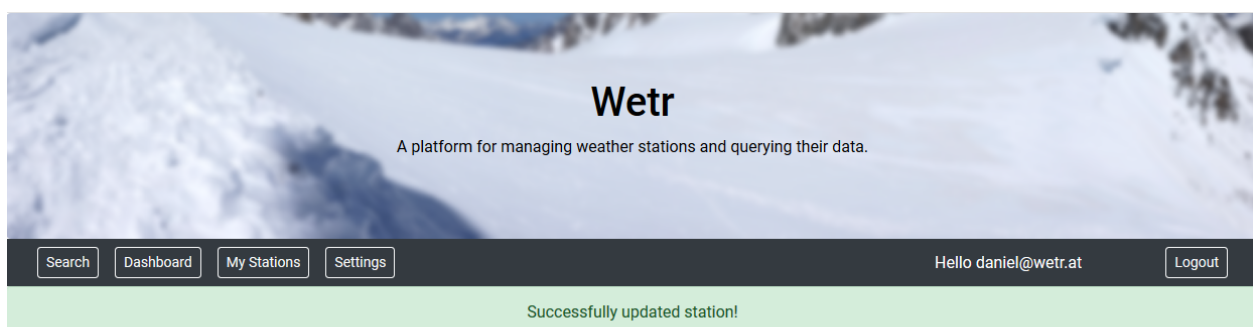
Location  
fdwfw

Latitude  
46.90277863

Longitude  
13.53416634

Save

Abbildung 10: Formular zum Bearbeiten von Stationsdaten.



**Wetr**  
A platform for managing weather stations and querying their data.

Search Dashboard My Stations Settings Hello daniel@wetr.at Logout

### Edit Station "Dachstation"

Name  
Dachstation

Successfully updated station!

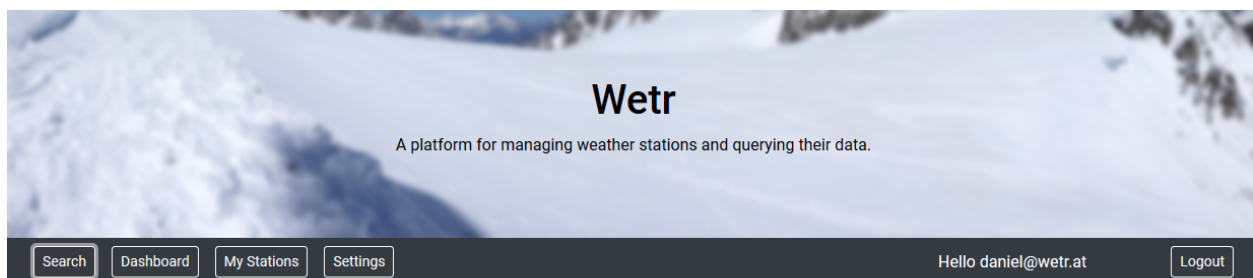
Abbildung 11: Anzeige von FlashMessages in dieser Applikation.

## 4.6 AddStation

Die Funktionsweise und das Aussehen sind sehr ähnlich zum Editieren einer Station (Abschnitt 4.5), deshalb werden hier keine Screenshots gezeigt.

## 4.7 Search

Auf dieser Seite kann nach Stationen gesucht werden (Abbildung 12). Die Eingabe der Community wird durch autocompletion unterstützt (Abbildung 13). Die Ergebnisse der Suche werden mit StationCard-Komponenten angezeigt (Abbildung 14). Hierbei können die Details zu einer Station wieder versteckt werden.



### Search stations:

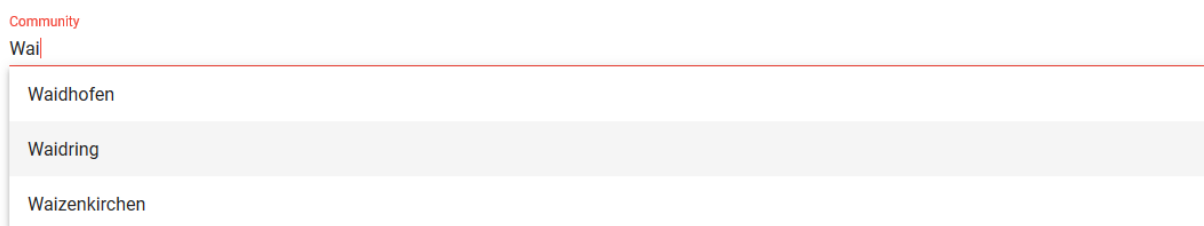
Community

Stations:

Hide Details: ☐

There are no stations.

Abbildung 12: Übersicht der öffentlichen Suche ohne Ergebnisse.



Community

Wai

Waidhofen

Waidring

Waizenkirchen

Abbildung 13: Darstellung der AutoCompletion bei Communities.

## Search stations:

Community  
Waidring

Stations:

Hide Details: ☐


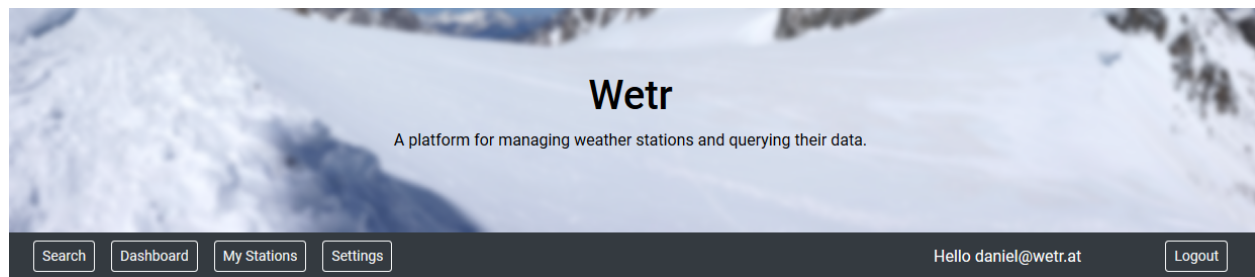
| LOFERER ALM   |          |
|---|----------|
| Community:  | Waidring |
| Type:   | TAWES    |
|  |          |

Abbildung 14: Resultate einer öffentlichen Suche.

### 4.8 Query

Auf dieser Seite können einerseits Abfragen zu Messdaten gemacht werden bzw. diese angelegt werden (Abbildung 15). Das linke Formular beinhaltet zur Auswahl der Start und Enddatums der Abfrage einen Datpicker (Abbildung 16). Die Dropdowns sind mit den möglichen Abfragekriterien bestückt. Das Ergebnis der Abfrage wird unten in einem Chart (Abbildung 17) und in tabellarischer Form (Abbildung 18) dargestellt. Beim Einfügen von Messdaten wird zur Eingabe der Timestamp ein DateTime Picker verwendet (Abbildung 19). Es können Abfragen zum Dashboard hinzugefügt werden, wobei diese als Enddatum das heutige Datum haben müssen um beispielsweise eine Abfrage zu ermöglichen, die die Durchschnittstemperatur der letzten zwei Wochen anzeigt.



## Query Station "LOFERER ALM" in Waidring

Start date \*  
1/12/2019

End date \*  
1/19/2019

Type \*  
Lufttemperatur

Reduction Type \*  
Average

Grouping Type \*  
Day

Send Query
Add to dashboard

### Add Measurement

1/15/2019, 11:56 AM

Type \*  
Regenmenge

Value  
10

Add Measurement

Abbildung 15: Formular zum Abfragen und Hinzufügen von Messdaten.

Start date \*  
1/12/2019

End date \*  
1/19/2019

JAN 2019

S M T W T F S

JAN

1 2 3 4 5

6 7 8 9 10 11 12

13 14 15 16 17 18 19

20 21 22 23 24 25 26

27 28 29 30 31

Abbildung 16: DatePicker für Start und Enddatum bei Abfragen.

## Query results:

Chart:

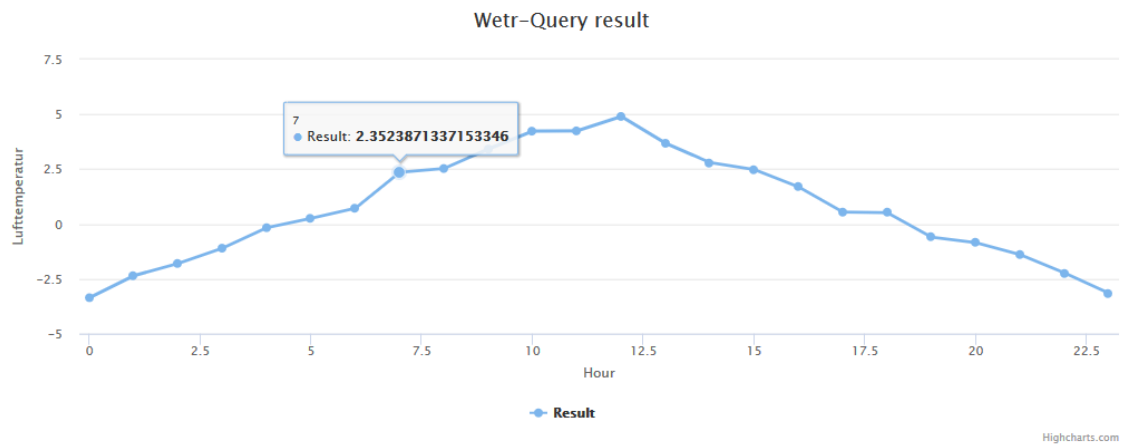


Abbildung 17: Chart zum Darstellen eines Abfrageergebnisses.

Table:

| Time    | Value               |
|---------|---------------------|
| Hour 0  | -3.3603586302120587 |
| Hour 1  | -2.3573773022944775 |
| Hour 2  | -1.7998732623682911 |
| Hour 3  | -1.103709819815761  |
| Hour 4  | -0.1712686244309918 |
| Hour 5  | 0.2560347097439895  |
| Hour 6  | 0.716863389848118   |
| Hour 7  | 2.3523871337153346  |
| Hour 8  | 2.5263690329687813  |
| Hour 9  | 3.412709455048936   |
| Hour 10 | 4.2301032736109345  |
| Hour 11 | 4.24094568111379    |

Abbildung 18: Tabelle zum Darstellen eines Abfrageergebnisses.

The image shows a DateTimePicker interface. At the top, there is a navigation bar with left and right arrows, the text "Jan 2019" with a dropdown arrow, and another right arrow. Below this is a calendar grid with days of the week (Sun to Sat) as headers. The dates are arranged in a standard calendar format. The date "19" is highlighted with a circle. Below the calendar is a time picker section with two input boxes, one for hours ("11") and one for minutes ("56"), separated by a colon. Each box has up and down arrows for selection. At the bottom, there are two buttons: "Cancel" and "Set".

| Sun | Mon | Tue | Wed | Thu | Fri | Sat |
|-----|-----|-----|-----|-----|-----|-----|
| 30  | 31  | 1   | 2   | 3   | 4   | 5   |
| 6   | 7   | 8   | 9   | 10  | 11  | 12  |
| 13  | 14  | 15  | 16  | 17  | 18  | 19  |
| 20  | 21  | 22  | 23  | 24  | 25  | 26  |
| 27  | 28  | 29  | 30  | 31  | 1   | 2   |
| 3   | 4   | 5   | 6   | 7   | 8   | 9   |

11 : 56

Cancel Set

Abbildung 19: DateTimePicker zum auswählen einer Timestamp für das Hinzufügen von Messwerten.



## 4.9 Dashboard

Im Dashboard werden favorisierte Abfragen gespeichert und in Form von Dashboard-Cards angezeigt (Abbildung 20). Diese werden pro Benutzer im localStorage gespeichert. Die genauer Funktionsweise wird in Abschnitt 4.10) genauer beschrieben.

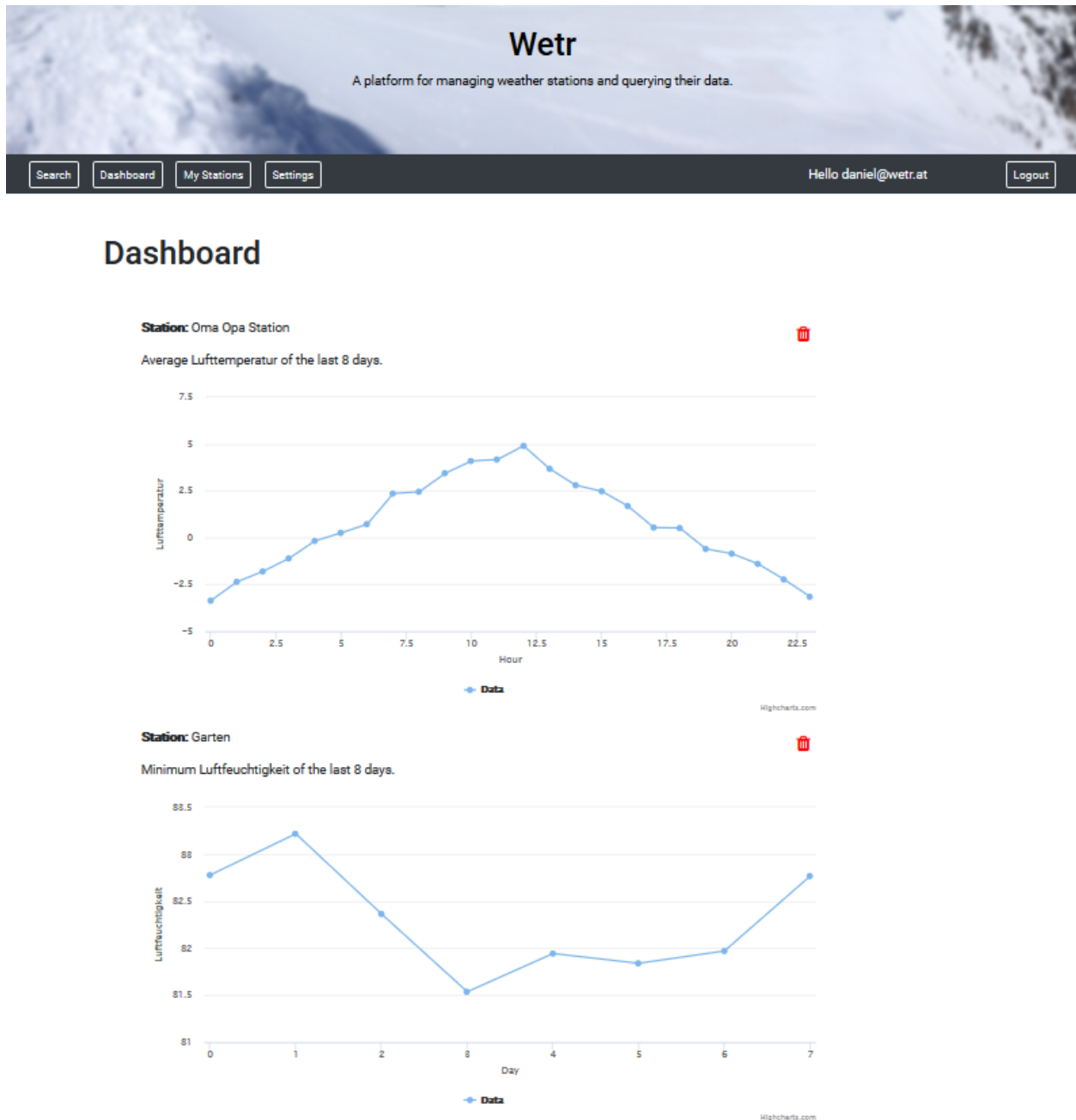


Abbildung 20: Individuelles Dashboard für den eingeloggtten Benutzer.

## 4.10 DashboardCard

Die DashboardCards können wieder gelöscht werden und zeigen neben den eigentlichen Daten (welche jedes mal neu abgefragt werden) eine kurze Beschreibung an, aus der der Kontext der Daten ersichtlich gemacht werden soll (Abbildung 21). Es können nur Abfragen gespeichert werden die Daten in den letzten X Tagen anzeigen. Somit werden Abfragen deren Zeitraum bereits abgeschlossen ist nicht unterstützt. Das Format der Beschreibung lautet: `< AggragationType > < MeasurementType > of the last X days`.

**Station:** Oma Opa Station



Average Lufttemperatur of the last 8 days.

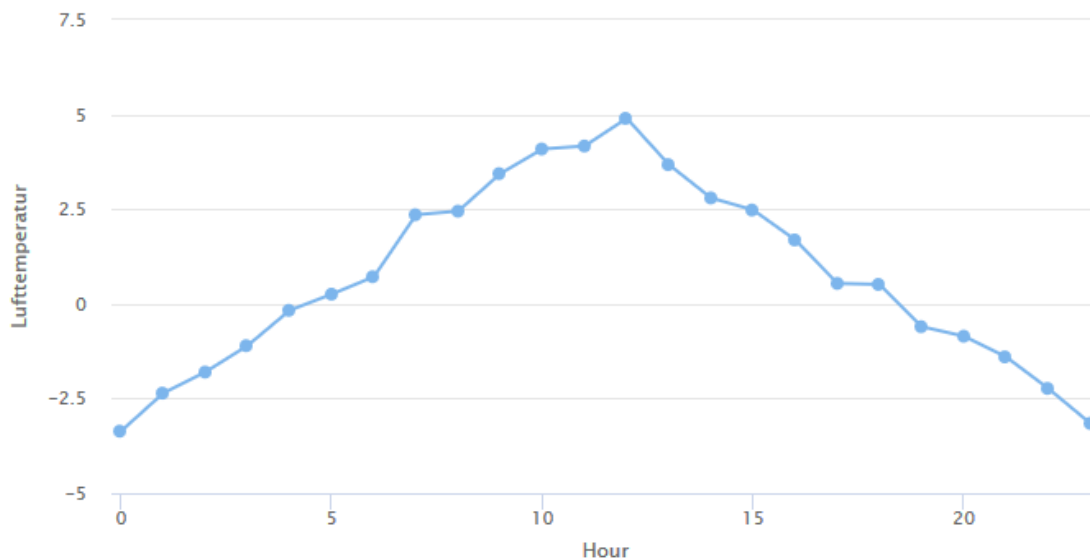
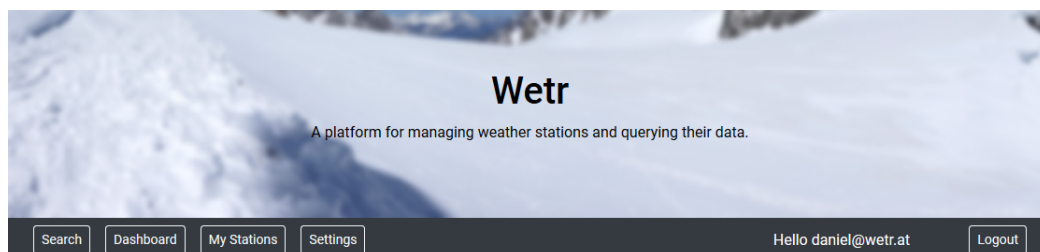


Abbildung 21: Detailansicht einer DashboardCard.

## 4.11 Settings

Diese Seite beinhaltet allgemeine Einstellungen, welche im localStorage gespeichert werden (Abbildung 22). Beispielfhaft gibt es die Einstellung die Temperatureinheit in Fahrenheit zu ändern. Dies wirkt sich auf alle Anzeigen von Temperaturabfragen aus.



### Settings

☒ Use Fahrenheit as the temperature unit.

Abbildung 22: Settings-Seite.