Wettermonitor

Inhalt

[1. Projektbeschrieb 1](#_Toc29465242)

[2. Hardware 1](#_Toc29465243)

[3. Programmstruktur 1](#_Toc29465244)

[4. Module 2](#_Toc29465245)

[4.1. Main.py 2](#_Toc29465246)

[4.2. Config.py 2](#_Toc29465247)

[4.3. DB\_Fill.py 2](#_Toc29465248)

[4.4. Import\_Data\_API.py 3](#_Toc29465249)

[4.5. Prediction.py 3](#_Toc29465250)

[4.6. Vis.py 3](#_Toc29465251)

[5. Anwendungsspezifisch 3](#_Toc29465252)

[5.1. Erforderte Libraries 3](#_Toc29465253)

[5.2. Installation auf Raspberry Pi 4](#_Toc29465254)

[5.3. Anbindung an eine neue Datenbank 5](#_Toc29465255)

# Projektbeschrieb

Der Segelclub Zürichsee beauftragt Sie, für die Clubmitglieder einen Wettermonitor zu bauen. Der Wettermonitor wird neben den Bootsliegeplätzen über einen Bildschirm Wetterdaten für Segler und andere Wassersportler auf hilfreiche Weise darstellen, sodass Segler sie unter anderem zur Planung eines Segeltörns nutzen können.

Ausserdem sollen Vergleichsdaten aus der Vergangenheit dargestellt werden. Können aus diesen Rückschlüssen auf mögliche Wetter-Entwicklungen gezogen werden, soll das ebenfalls angezeigt werden.

Als Hardware für den Wettermonitor steht ihnen ein Raspberry Pi Computer mit Monitor zur Verfügung. Die Wetterdaten holen Sie sich fortlaufend von einem Webservice der Seepolizei Zürich und speichern sie in einer Datenbank auf dem Raspberry Pi. Die Software soll hauptsächlich in Python programmiert werden.

Wir gehen davon aus, dass später einmal andere Datenquellen angeschlossen werden sollen oder Erfahrungen von Nutzern über die Zeit hinweg zu Änderungswünschen führen. Solche Veränderungen an der Software müssen dann auch Ihnen fremde Programmierer machen können.

# Hardware

Die Hardware besteht aus einem Raspberry Pi und einem passenden Monitor mit einem kapazitiven Touchscreen. Ein Raspberry ist ein kleiner Computer. Genaueres kann hier nachgelesen werden: <https://de.wikipedia.org/wiki/Raspberry_Pi>

# Programmstruktur

Ein Bild, das Screenshot enthält.

Automatisch generierte Beschreibung

Abbildung : Aufbau des Programms

Die Programmstruktur ist wie auf Abbildung 1 abgebildet aufgeteilt. Es werden Daten von einer API, in unserem Fall von der Webseite der Seepolizei, geholt und in eine lokale InfluxDB geladen.

Danach können die Daten von der lokalen InfluxDB abgefragt und verwendet werden. Die Datentransformation ist hierbei ein wesentlicher Bestandteil, welcher auf die einzelnen Module aufgeteilt wurde. Die Prognose wurde ebenfalls auf ein Modul aufgeteilt. Beide Bestandteile werden im Schritt der Visualisierung aufbereitet und in einem Graphischen User Interface angezeigt.

# Module

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung : Aufbau der Module

Der Modulaufbau ist wie in der Abbildung oben aufgebaut. Dabei fungiert das «Main.py» Modul als Hauptfile, von welchem alle anderen Instanzen initialisiert werden.

Im «Config.py» File sind alle veränderlichen Variablen hinterlegt, die bei der Abfrage und der Verbindung einer neuen Datenbank angepasst werden müssen.

Auf die Module wird im nächsten Kapitel näher eingegangen.

## Main.py

Dieses File steuert ist im eigentlichen Sinne das Hauptprogramm und ruft alle untergeordneten Funktionen auf, die mit diesem File verknüpft sind, die für den Betrieb des Wettermonitors nötig sind. Dieses Modul ist ebenfalls zuständig für einen reibungslosen Ablauf auf dem Raspberry Pi

## Config.py

Das «Config» Modul dient als Ablage der Variablen, die bei Gebrauch aus diesem File importiert werden und genutzt werden können.

Da es für uns wichtig ist, alle diese Variablen an einem Ort zu haben, haben wir für diesen Zweck ein einzelnes Modul dafür erstellt. Dieses Modul macht auch das Verbinden zu einer neuen Datenbank einfacher und die gesetzten Parameter in den anderen Modulen müssen nicht mehr gewechselt werden.

## DB\_Fill.py

Dieses Modul beherbergt alles, was man mit der Datenbank in Verbindung bringen kann, ausser der Datenabfrage. Es hat darin Funktionen, die dazu dienen eine Verbindung zur Datenbank herzustellen, die Datenbank zu leeren und historische, sowie auch aktuelle Wetterdaten zu laden, die von der API abgerufen wurden.

## Import\_Data\_API.py

In diesem Modul sind relevante Bestandteile zur Abfrage der Daten vorhanden. Darin befinden sich vor allem Funktionen, welche später bei der Datenvisualisierung eine wichtige Rolle spielen. Ein Beispiel dafür ist die Abfrage der letzten Daten, welche in die Datenbank geladen wurden.

## Prediction.py

Die Vorhersage bezieht sich auf die Lufttemperatur und wurde anhand eines statistischen Tests implementiert, dem Kolmogorov-Smirnoff Test[[1]](#footnote-1). Dabei nimmt man Referenzdaten aus einem ganzen Jahr und bündelt sie in Tage. Dabei wird der vorherige Tag vom jetzigen Zeitraum mit all den Referenzdaten von einem Jahr verglihen. Der Folgetag von der Verteilung, der die kleinste Abweichung vom aktuellen Zeitraum hat, entspricht sozusagen der Temperatur vom nächsten Tag.

## Vis.py

Als Visualisationswerkzeug wird ein Plotly Dashboard [[2]](#footnote-2)verwendet. In diesem Plotly Dashboard werden links die Winddaten von beiden Wetterstationen angezeigt. Darunter fallen Windgeschwindigkeit, Windkraft und Windböen, welche für einen Zeitraum von einer Woche angezeigt werden. Auf der rechten Seite werden Lufttemperatur, Luftdruck, Taupunkt, Luftfeuchtigkeit und Wassertemperatur angezeigt

, Windrichtung, sowie Daten über Temperatur, Wassertemperatur usw. Siehe Abbildung:

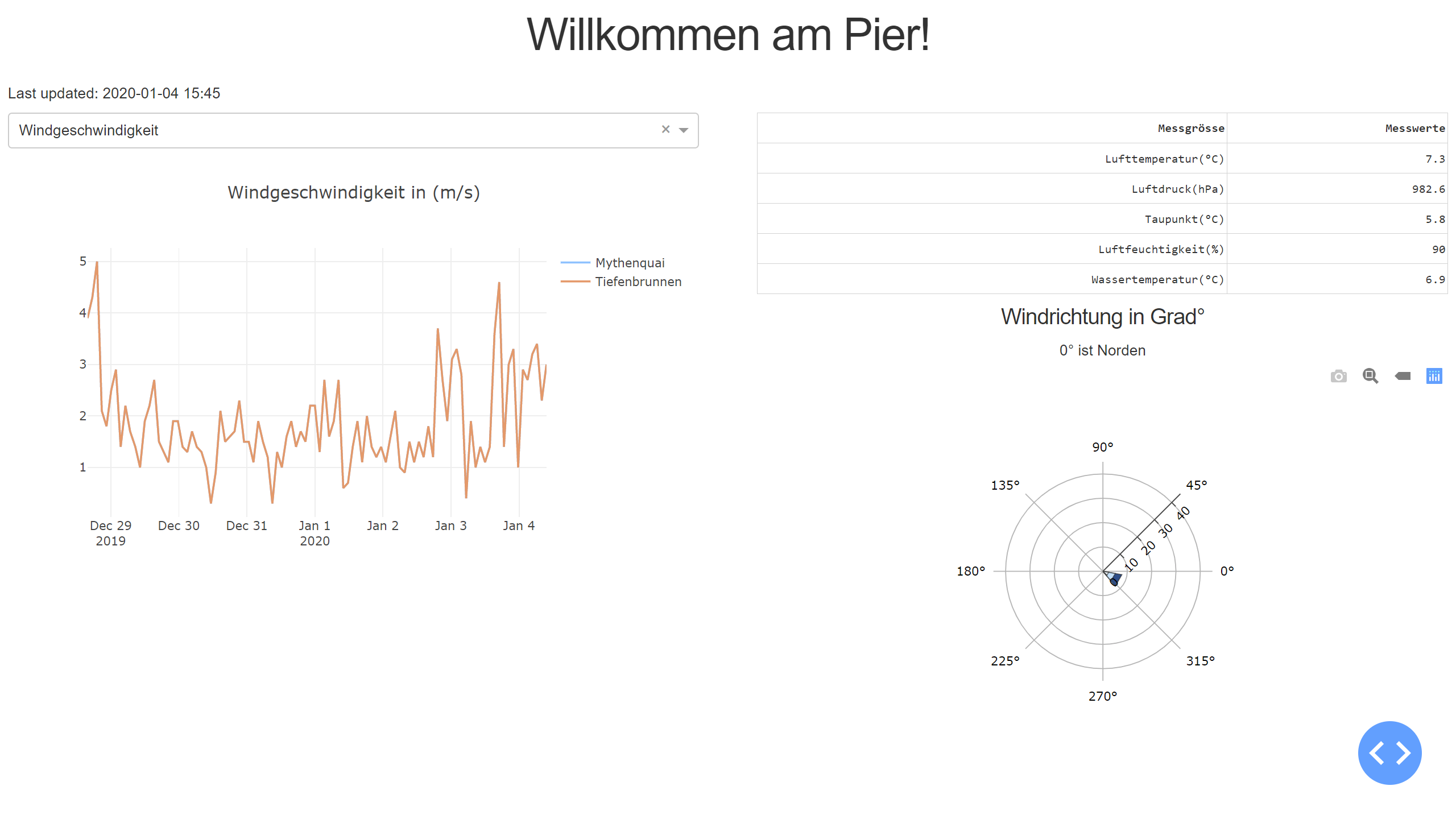


Abbildung : Plotly Dashboard Wettermonitor

# Anwendungsspezifisch

## Erforderte Libraries

Erforderte Packages:

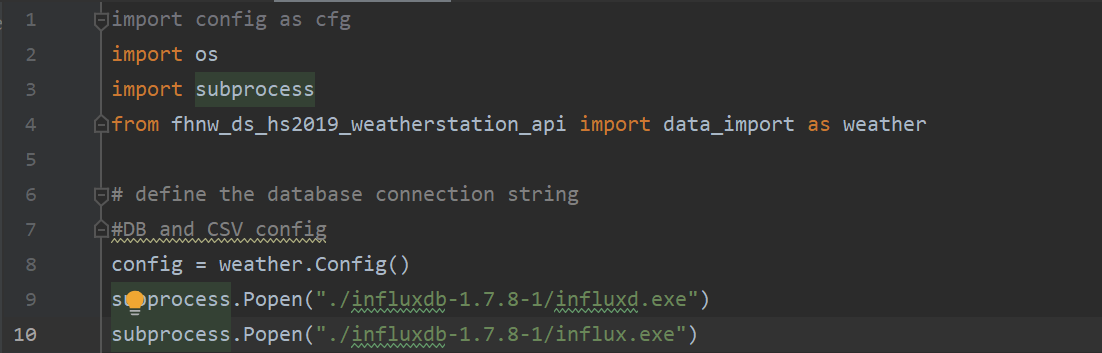
* datetime
* influxdb
* fhnw\_ds\_hs2019\_weatherstation\_api
* pandas
* numpy
* scipy
* dash
* plotly.graph\_objects

## Startup und in Betriebnahme des Raspberry Pi’s

1. Neues Terminal öffnen:CMD: cd Desktop/wettermonitor/Coding
2. Programm Main.py starten: python3 Main.py
3. Warten bis Prozess beendet.
4. Neues Terminal öffnen: CMD: cd Desktop/wettermonitor/Coding
5. Programm Vis.py starten: CMD: python3 Vis.py
6. Webseite mit der gegebenen Localhost-Adresse aufrufen. Die Adresse wird in der Ausgabe im Terminal angezeigt. (127.0.0.1:8050)
7. Dashboard sollte nun angezeigt werden mit den aktuellen Wetterdaten.
8. Vollbildmodus mit Taste F11 starten.

## Neu-Installation auf Raspberry Pi

1. Starten des Raspberry Terminals
2. Updaten des Raspberry Pi’s:
   1. CMD: sudo apt update
3. Clonen des Repositories in festgelegtes Directory mit dem Befehl.
   1. CMD: cd Desktop
   2. CMD: git clone <https://gitlab.fhnw.ch/simon.staehli/wettermonitor.git>
4. Alle obengenannten Packages müssen im Python 3 installiert und lauffähig sein. Die Packages können mit dem «pip3 install» Befehl installiert werden.
   1. CMD: pip3 install «Package Name»
5. Installieren der InfluxDB für Linux Nutzer.
   1. CMD: sudo apt-get install influxdb
6. Kontrollieren der Benutzereinstellungen über Befehle unten. Falls die Rechte noch nicht hinzugefügt wurden, müssen sie noch geändert werden:
   1. CMD: cd Desktop/wettermonitor/Coding
   2. CMD: ls -l influxdb-1.7.8-1
7. Berechtigungen zum Ausführen der Datei hinzufügen für den Benutzer:
   1. CMD: Cd Desktop/wettermonitor/Coding/influxdb-1.7.8-1
   2. CMD: chmod u+x influx.exe
   3. CMD: chmod u+x influxd.exe
8. Als nächstes müssen noch Anpassungen in dem Programm gemacht werden, da man nicht die gleichen Befehle laufen lassen kann, wie im Windows.
   1. Öffnen des Moduls: DB\_Fill.py



* 1. Auskommentieren der Codezeilen 2, 9, 10, 12 und 14

1. Nun kann das Programm gestartet werden:
   1. Neues Terminal öffnen:CMD: cd Desktop/wettermonitor/Coding
   2. Programm Main.py starten: python3 Main.py
   3. Warten bis Prozess beendet.
   4. Neues Terminal öffnen: CMD: cd Desktop/wettermonitor/Coding
   5. Programm Vis.py starten: python 3 Vis.py
   6. Webseite mit der gegebenen Localhost-Adresse aufrufen. Die Adresse wird in der Ausgabe im Terminal angezeigt. (127.0.0.1:8050)
2. Dashboard sollte nun angezeigt werden mit den aktuellen Wetterdaten.

## Anbindung an eine neue Datenbank

Um den Raspberry an eine Datenbank anzubinden sind einige Schritte erfolgreich, die das erfolgreiche Anbinden möglich machen. Es wird per Default bereits eine angepasst SELECT Funktion im Modul «Import\_Data\_API.py» mitgeliefert. Nachfolgend sind die erforderten Schritte aufgelistet:

1. Config.py File öffnen und folgende Anpassungen vornehmen:



* 1. Im kommentierten Bereich «Erweiterte Datenquelle» alle vorhandenen Variablen anpassen mit denen von der neuen Datenquelle. «ext\_datasource» muss auf True gesetzt werden.
  2. Um die Implementation einer erweiterten Datenquelle zu vereinfachen wurden per se bereits zwei Funktionen im Modul «Import\_Data\_API.py» bereitgestellt, die benutzt werden können, die Daten abzufragen.
  3. Die Funktionen heissen:
     1. Select\_timedelta\_ext: Die Funktion benötigt als Argumente, anders als bei der normalen Funktion, noch den neuen Client als Eingabe und die Stations.
     2. Get\_latest\_data\_ext: Als Argumente dienen auch wieder der Client und die Station.

1. <https://de.wikipedia.org/wiki/Kolmogorow-Smirnow-Test> [↑](#footnote-ref-1)
2. <https://plot.ly/dash/> [↑](#footnote-ref-2)