**Interfaces / Protocols:**

Device1  
(Feather)

Device2  
(Feather)

Device3  
(Feather)

Master  
(RaspberryPI)

Server  
(RaspberryPI)

Oracle Cloud  
(Datenbank)

BLE

SSL-

Sockets

http

SCD30

I2C

Light Sensor

Analog

SCD30

I2C

Light Sensor

Analog

SCD30

I2C

Light Sensor

Analog

Magnet Sensor

Digital

Python notebook

**Protokolle:**

* I2C:
  + https://www.mikrocontroller.net/articles/I%C2%B2C
* BLE:
  + Custom:
    - Master -> DeviceX:
      * Command als string («measure\_request»)
    - DeviceX -> Master:
      * Messung als json
        + Json:

{  
 'sensorNameX’:  
 {‘messungX’:’value, ‘messungY’:’value’, …},  
  
 ‘sensorNameY’:  
 {‘messungX’:’value’, ‘messungY’:’value’, …},  
  
 …  
}

* SSL-Sockets:
  + Custom:
    - Master -> Server:
      * Messungen als string mit Format: «data~’messung1’;’messung2’;…»
        + Json:

{  
 «timeStamp»:’timestamp of measurements’,  
  
 «data»:{‘devNameX’:’jsonOfDev.’, ‘devNameX’:’jsonOfDev.’, …}  
}

* + - Server -> Master:
      * Antwort als string («failed» / «confirmed») je nach dem ob die daten erfolgreich empfangen und verarbeitet wurden
* http:
  + rest API:
    - https://developer.oracle.com/databases/oracle-rest-data-services-for-python-developers.html

**Dataflow vom Beginn einer Messung bis zur Persistierung:**

1. Device1 – 3 beginnen zu Advertisen
2. Master versucht die Advertisements der Devices zu finden
3. Master speichert jetzige Zeit
4. Master versucht sich mit einem Device dessen Advertisement gefunden wurde (und noch nicht versucht wurde zu verbinden) zu verbinden
5. Master sendet Befehl: «measure\_request»
6. Device empfängt Befehl und beginnt die Angeschlossenen Sensoren auszulesen
7. Device Device konvertiert die Resultate/Fehlermeldung(en) der Messung zu einem Json und sendet diese zurück an den Master
8. Master speichert die Messergebnisse / Fehlermeldungen und schliesst die Verbindung
9. Schritte 4 bis werden nun solange wiederholt, bis mit allen Devices (dessen Advertizement gefunden wurde)  
   wenigstens ein Verbindungsversuch stattgefunden hat
10. Master konvertiert nun Messresultate / Fehlermeldungen mit dem zuvor gespeicherten Zeitstempel zu einem Json
11. Master versucht sich nun mit dem Server zu verbinden
12. Server akzeptiert die Verbindung
13. Master sendet alle vorhandenen Messungen (neue und cached) an Server
14. Server empfängt daten und versucht diese zu verarbeiten und speichert die in der Datenbank
15. Bei erfolgreicher Verarbeitung sendet der Server «confirmed» an den Master zurück (ansonnsten wird «failed» gesendet)
16. Master empfängt Antwort von Server
17. Falls die Verbindung zum Server während der Transaktion fehlschlägt (Verbindungsabbruch oder «failed» empfangen) oder diese erst gar nicht aufgebaut werden kann, speichert der Master die neue Messung im «cache» file