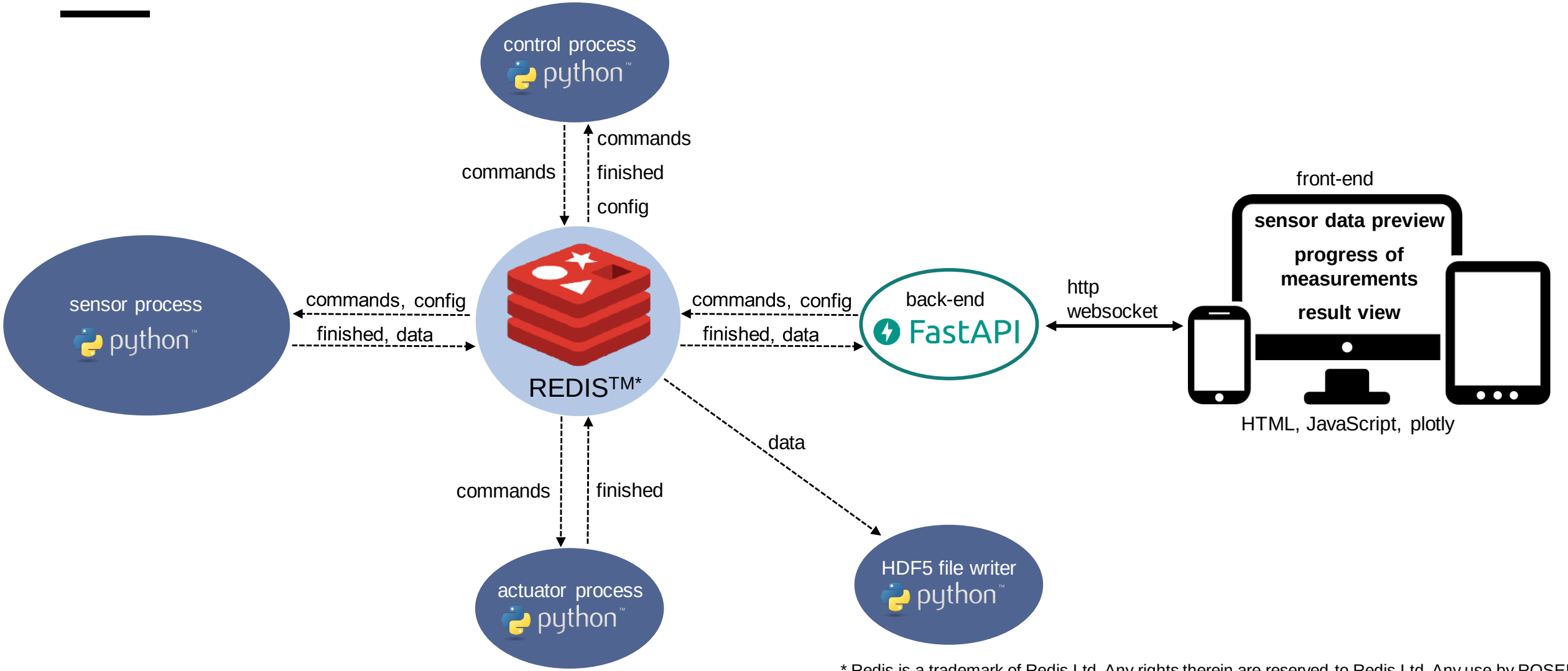
WEB-BASED LIVE VISUALISATION OF SENSOR DATA

Jannis Lübbe · PyCon DE · bcc Berlin Congress Center · 12 April 2022

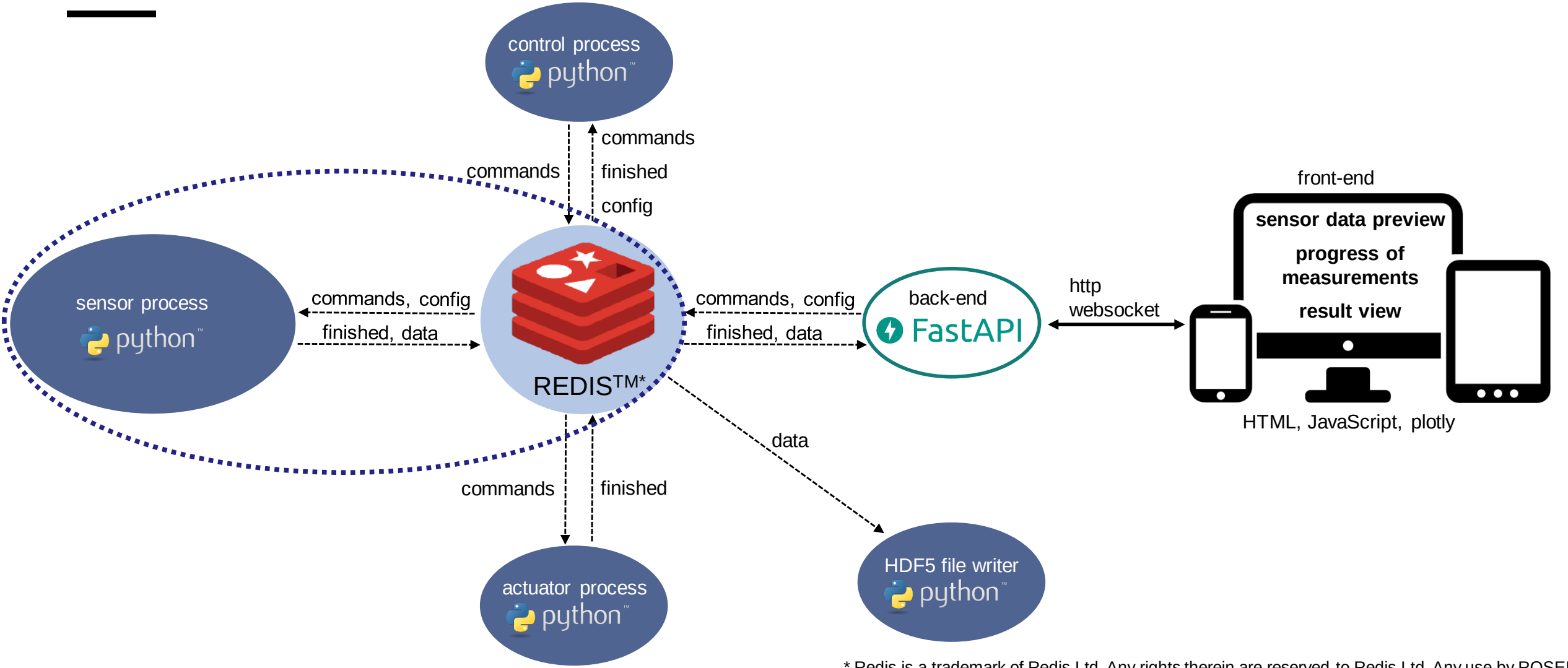
ROSEN
empowered by technology

LAB SETUP



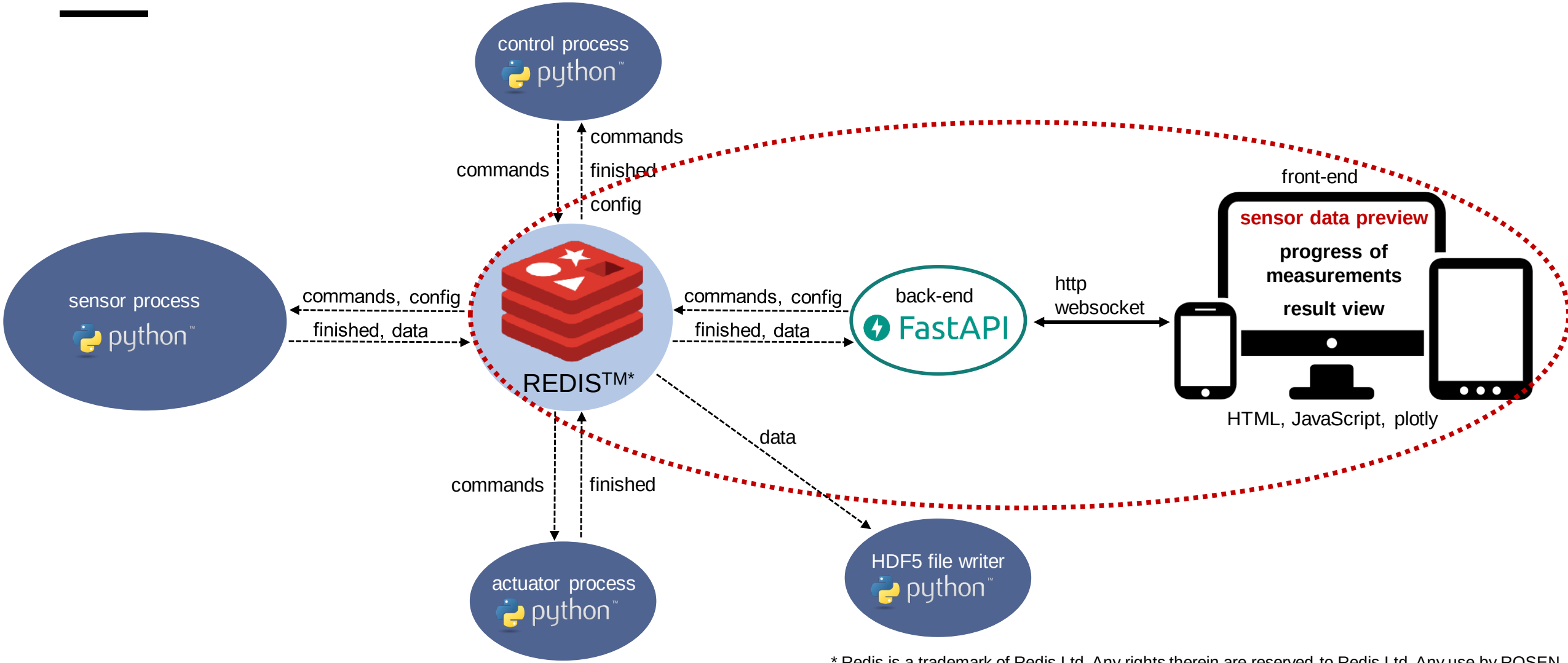
* Redis is a trademark of Redis Ltd. Any rights therein are reserved to Redis Ltd. Any use by ROSEN is for referential purposes only and does not indicate any sponsorship, endorsement or affiliation between Redis and ROSEN.

LAB SETUP



* Redis is a trademark of Redis Ltd. Any rights therein are reserved to Redis Ltd. Any use by ROSEN is for referential purposes only and does not indicate any sponsorship, endorsement or affiliation between Redis and ROSEN.

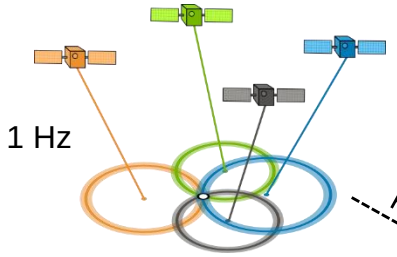
LAB SETUP



* Redis is a trademark of Redis Ltd. Any rights therein are reserved to Redis Ltd. Any use by ROSEN is for referential purposes only and does not indicate any sponsorship, endorsement or affiliation between Redis and ROSEN.

HOBBY PROJECT: FLIGHT DATA RECORDER

ROSEN
empowered by technology



1 Hz

GPS

position, speed, time

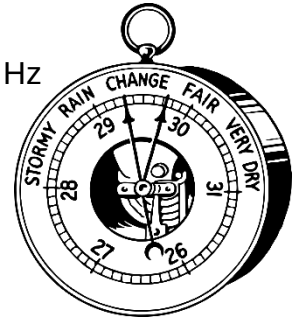
merge process
python

sensor data

combined sensor data

storage of combined sensor data

12.5 Hz



barometer

pressure

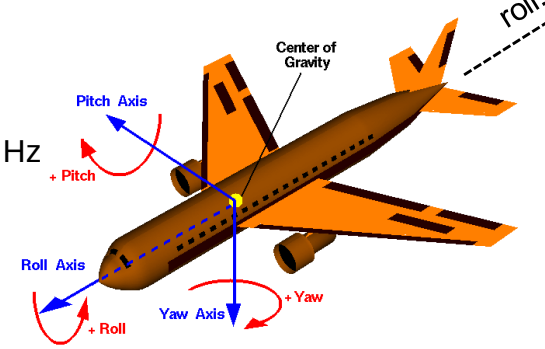
REDIS™*

review / export

FastAPI

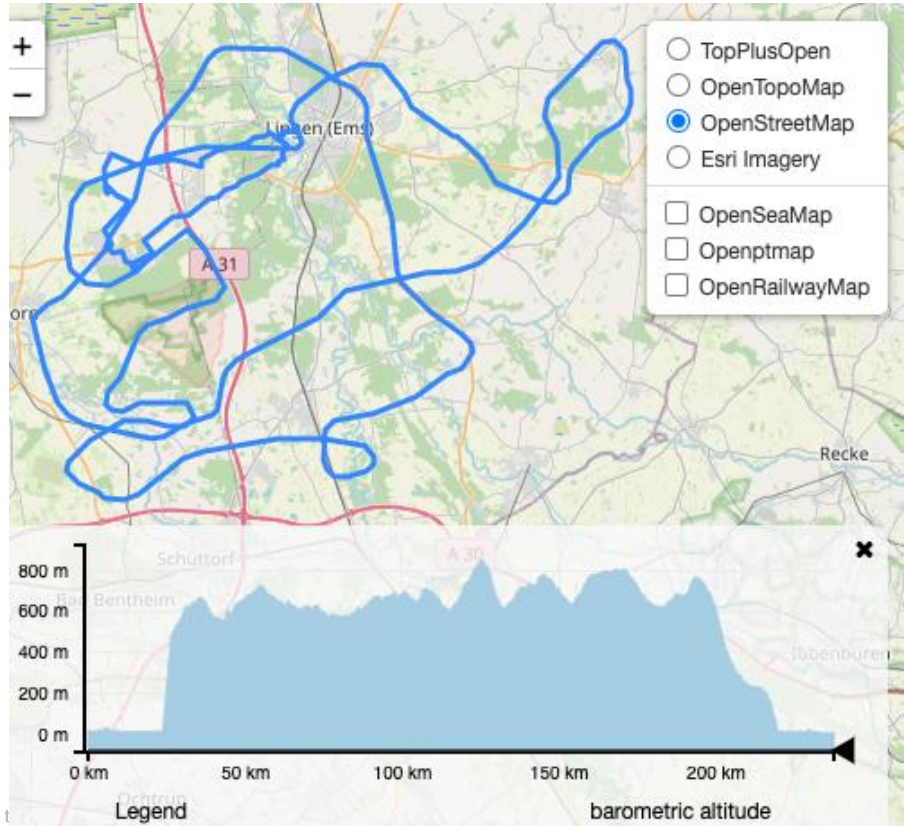
roll, pitch, yaw

25 Hz

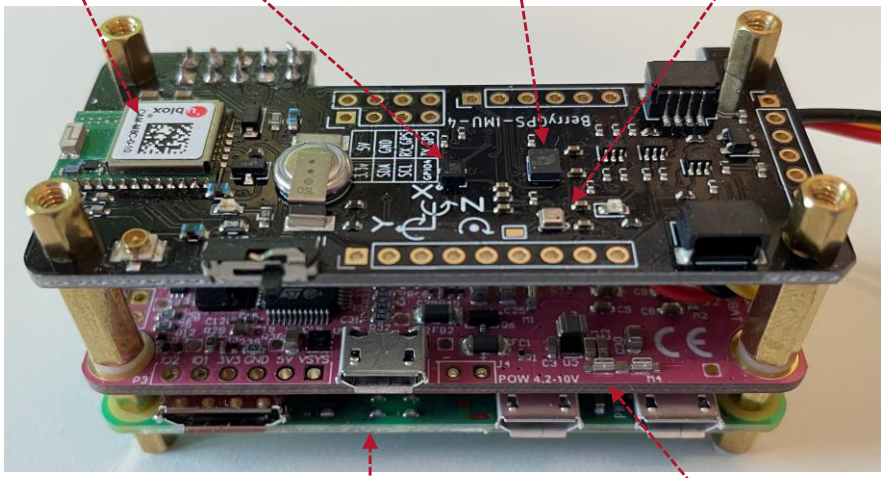


accelerometer, gyro, magnetometer

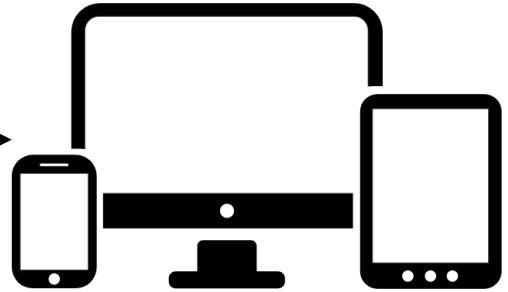
https://github.com/jaluebbe/GPSTracker



TODAY'S SETUP



mobile WiFi hotspot

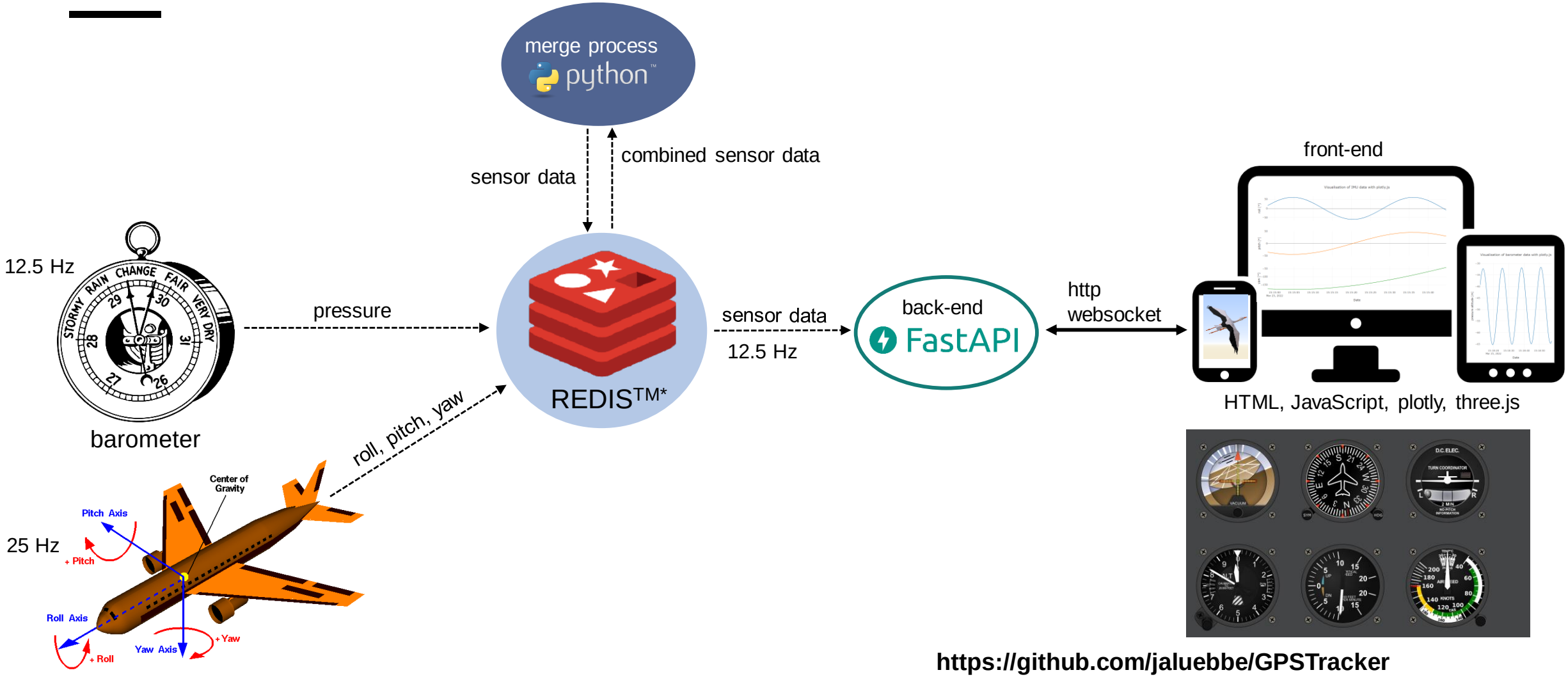


my notebook and your devices

(WiFi credentials are published later)

https://github.com/jaluebbe/GPSTracker

ADDING SENSOR DATA LIVE VISUALISATION



https://github.com/jaluebbe/GPSTracker

https://github.com/jaluebbe/FastAPIWebSocketExample

PUBLISHING DATA TO A REDIS CHANNEL

ROSEN
empowered by technology

```
import time, redis, json
from my_sensors import Barometer

r = redis.Redis()
sensor = Barometer()

while True:
    data = sensor.get_sensor_data()
    r.publish("barometer", json.dumps(data))
    time.sleep(0.08)
```

```
{
    "sensor": “my_barometer"
    "timestamp": 1648123202.19,
    "pressure": 100682.2,
    "temperature": 5.1,
}
```

https://github.com/jaluebbe/GPSTracker

CONSUMING DATA FROM A REDIS CHANNEL

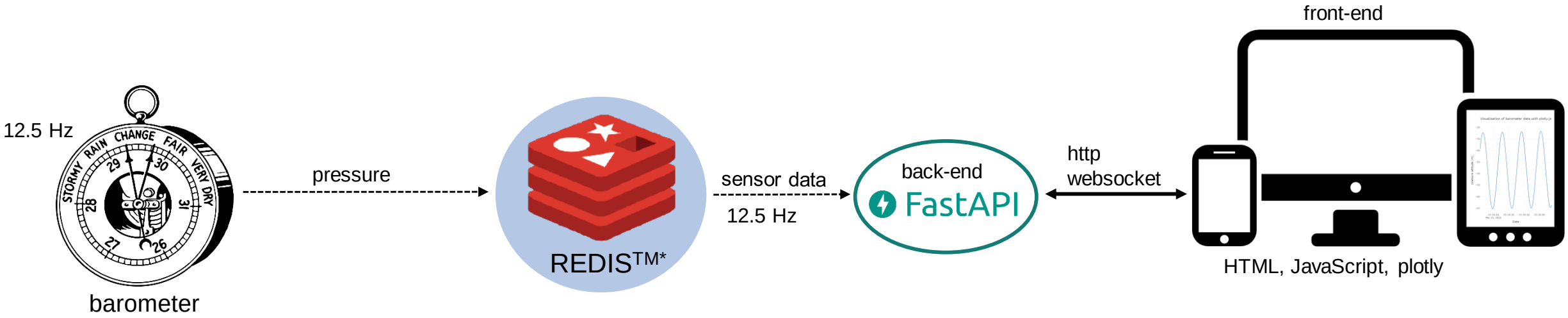
ROSEN
empowered by technology

```
import redis, json

r = redis.Redis(decode_responses=True)
pubsub = r.pubsub(ignore_subscribe_messages=False)
pubsub.subscribe("barometer")

for item in pubsub.listen():
    barometer_data = json.loads(item["data"])
```

```
{
    "sensor": “my_barometer"
    "timestamp": 1648123202.19,
    "pressure": 100682.2,
    "temperature": 5.1,
}
```

https://github.com/jaluebbe/GPSTracker

BAROMETER DEMO



https://github.com/jaluebbe/GPSTracker

https://github.com/jaluebbe/FastAPIWebSocketExample

HOSTING WEBSOCKETS WITH FASTAPI

ROSEN
empowered by technology

```
from fastapi import FastAPI, WebSocket
from fastapi.staticfiles import StaticFiles
import aioredis, asyncio

r = aioredis.Redis(decode_responses=True)
app = FastAPI()
app.mount("/static", StaticFiles(directory="static"), name="static")


# continued on next slide
```

https://github.com/jaluebbe/FastAPIWebSocketExample

HOSTING WEBSOCKETS WITH FASTAPI

ROSEN
empowered by technology

```
@app.websocket("/ws/imu_pressure")
async def websocket_endpoint(websocket: WebSocket):
    await websocket.accept()
    pubsub = redis_connection.pubsub(ignore_subscribe_messages=True)
    await pubsub.subscribe("imu_pressure")

    while True:
        try:
            message = await pubsub.get_message()
            if message is not None:
                await websocket.send_text(message["data"])
            await asyncio.sleep(0.01)
        except asyncio.TimeoutError:
            pass
```

https://github.com/jaluebbe/FastAPIWebSocketExample

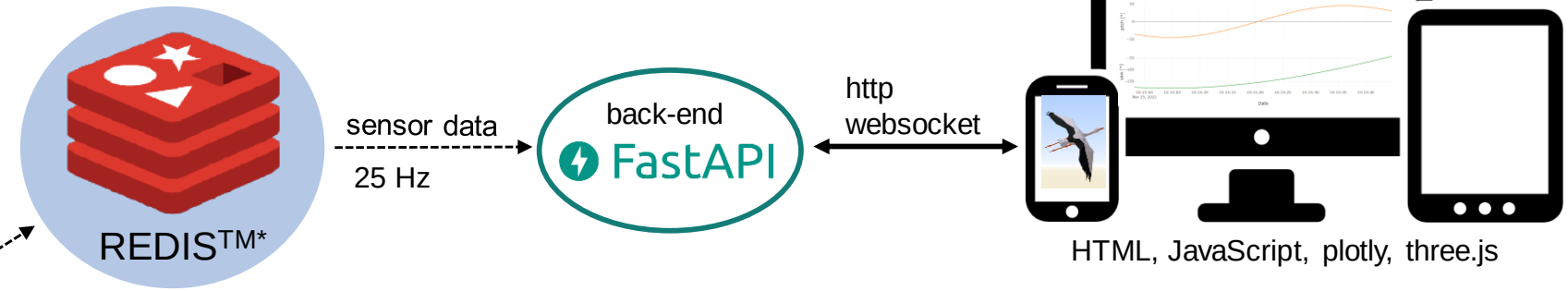
CONSUMING WEBSOCKETS WITH JAVASCRIPT

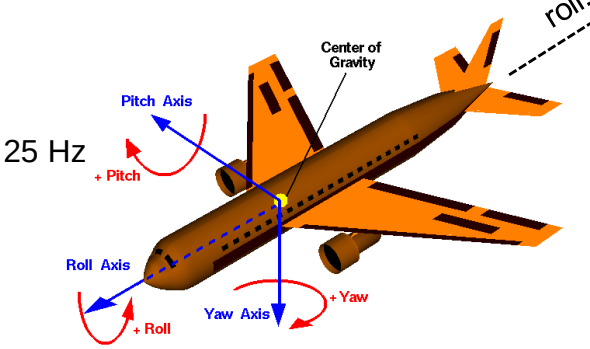
ROSEN
empowered by technology

```
var url = "ws://" + window.location.host + "/ws/imu_pressure“;
var ws = new WebSocket(url);

ws.onmessage = function(event) {
    let message = JSON.parse(event.data);
    visualiseData(message);
}
```

https://github.com/jaluebbe/FastAPIWebSocketExample

ORIENTATION SENSOR DEMO





accelerometer, gyro, magnetometer

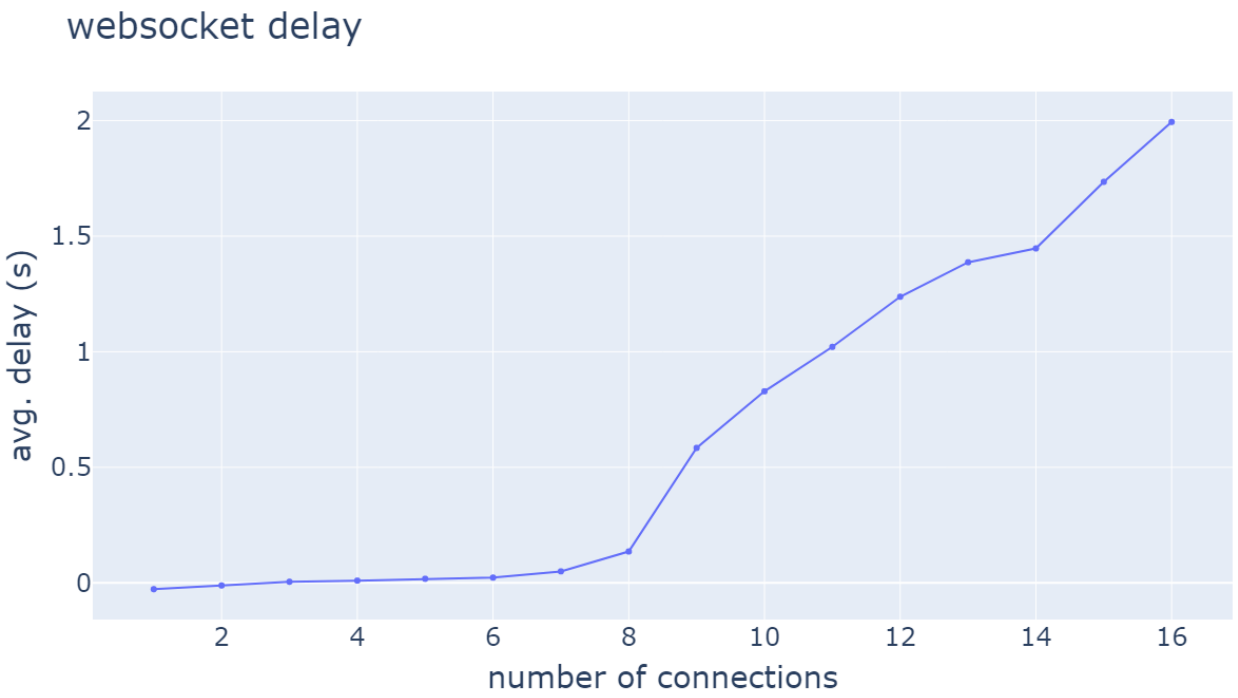
https://github.com/jaluebbe/GPSTracker

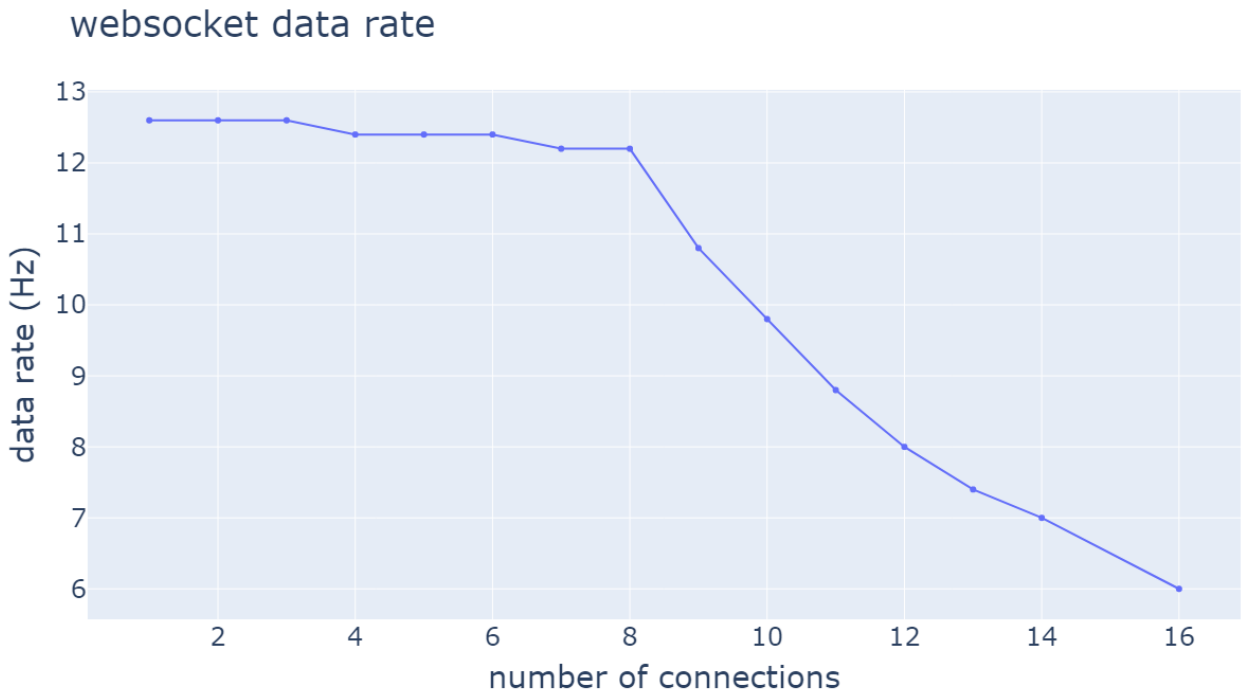
https://github.com/jaluebbe/FastAPIWebSocketExample

PERFORMANCE

Raspberry Pi Zero W (single core)

data rate 12.5 Hz



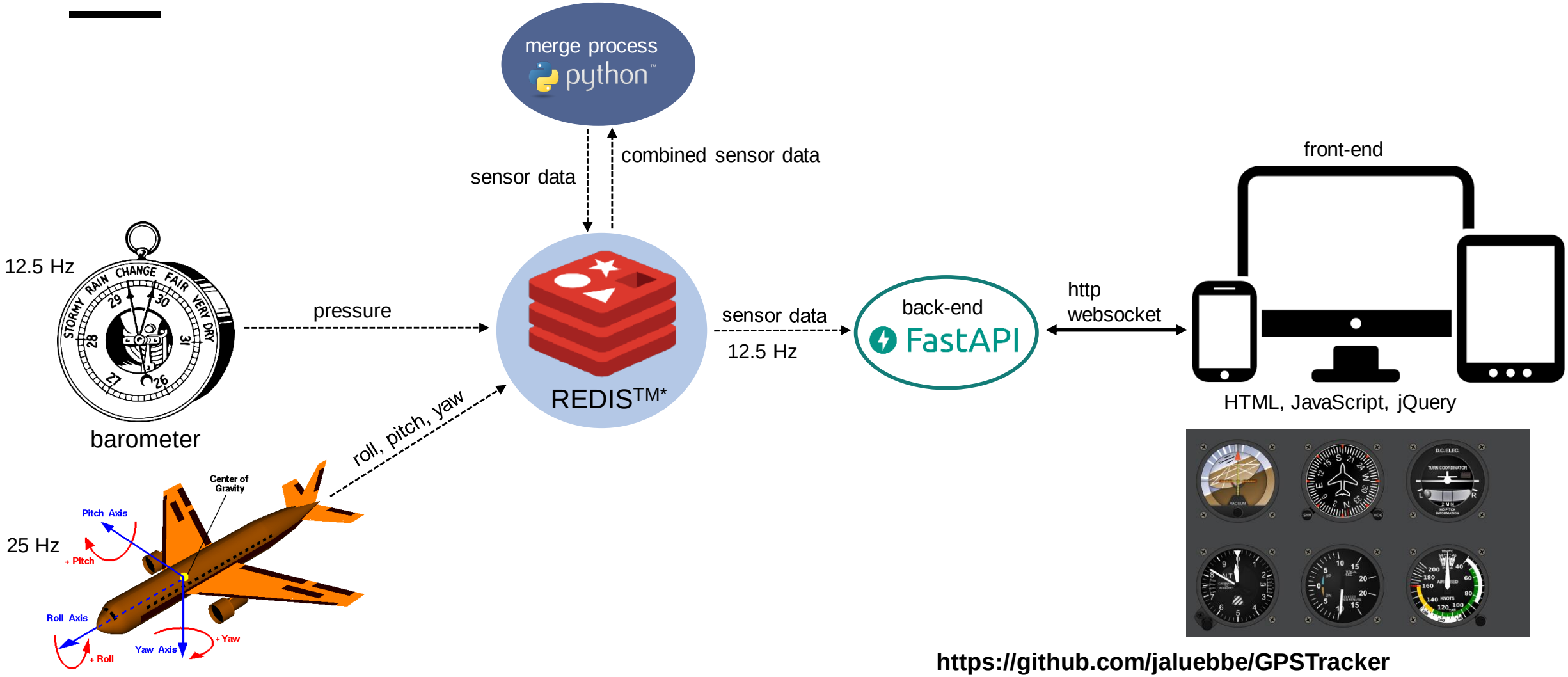


System clock difference on host and client may cause negative delays.

https://github.com/jaluebbe/GPSTracker

https://github.com/jaluebbe/FastAPIWebSocketExample

FLIGHT INSTRUMENTS DEMO



https://github.com/jaluebbe/GPSTracker

https://github.com/jaluebbe/FastAPIWebSocketExample

INTERACTIVE EXAMPLE

ROSEN
empowered by technology

Join with your mobile device:

WIFI: ROSEN_DEMO
password: websockets

Bird with single core Raspberry Pi zero:
http://192.168.4.12:8080

Quad core Raspberry Pi zero 2:
http://192.168.4.11:8080

THANK YOU FOR JOINING THIS PRESENTATION.

ROSEN
empowered by technology