Università degli studi di Salerno

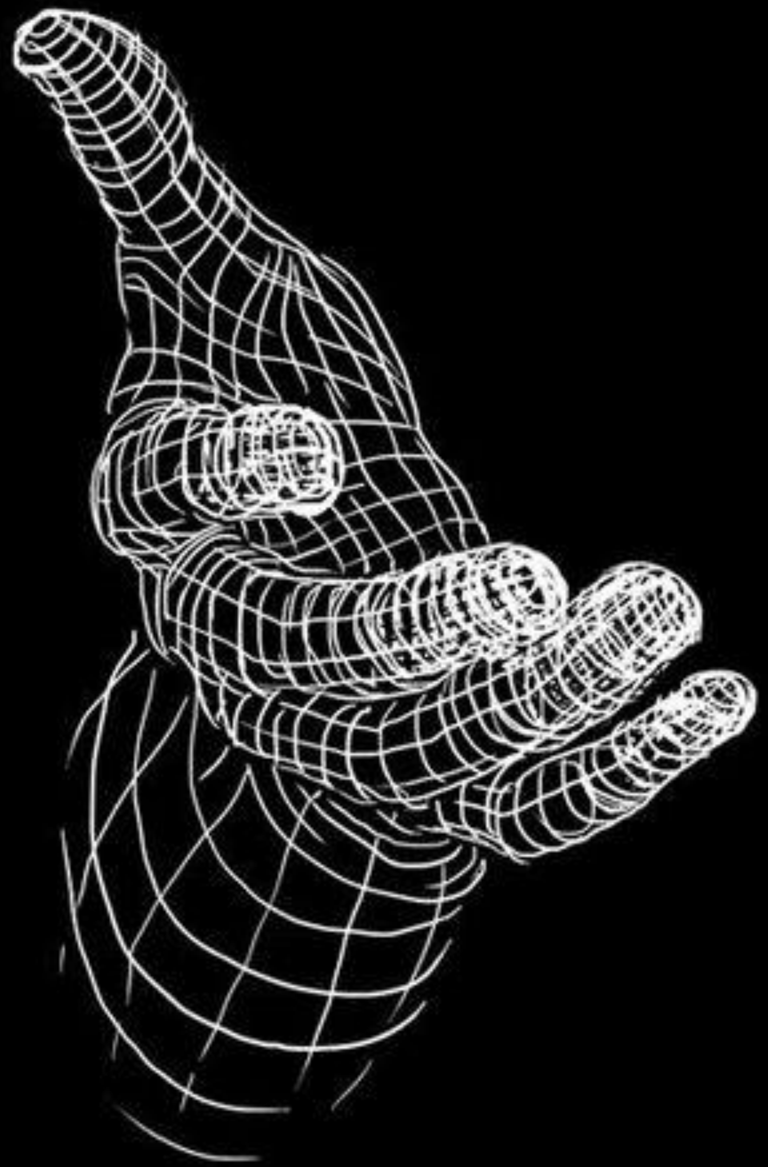
AUGMENTED REALITY DRONES SWARM

Presented by
- Andrea Bucci
- Amedeo Leone
- Lorenzo Sorrentino

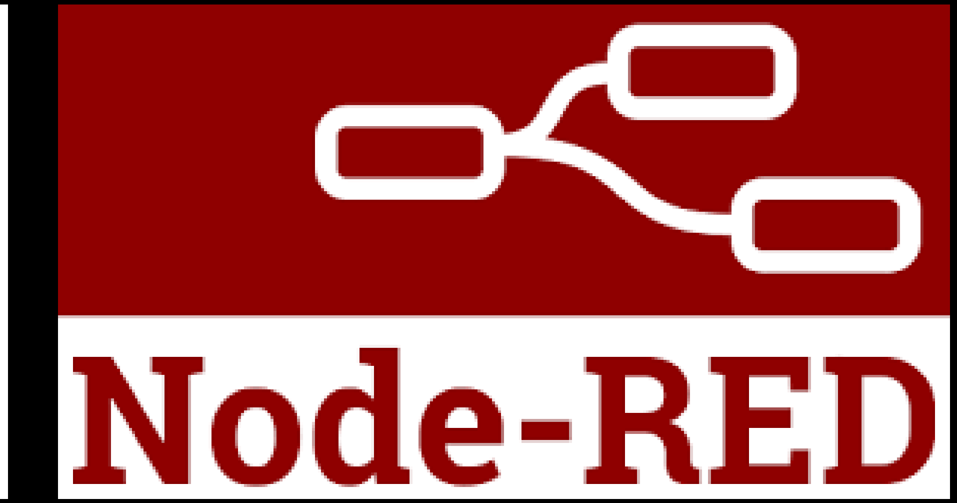
INTRODUCTION

Our university project aims to integrate advanced technologies for the control and monitoring of a swarm of DJI Tello drones. Using the Meta Quest headset and its controllers, we have implemented an intuitive system that allows two drones to be piloted in a synchronized and interactive way, taking advantage of augmented reality.

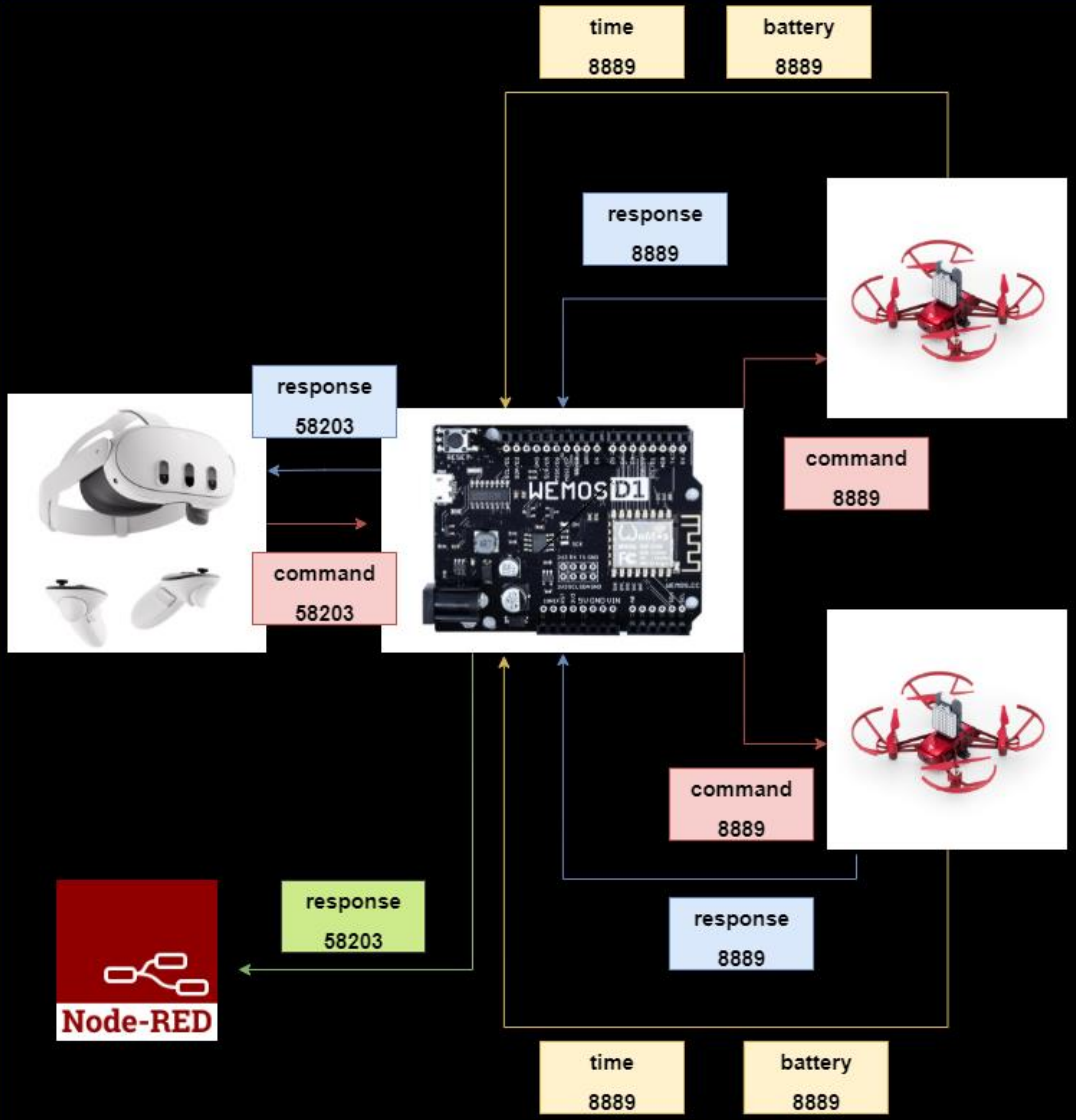
HARDWARE COMPONENTS



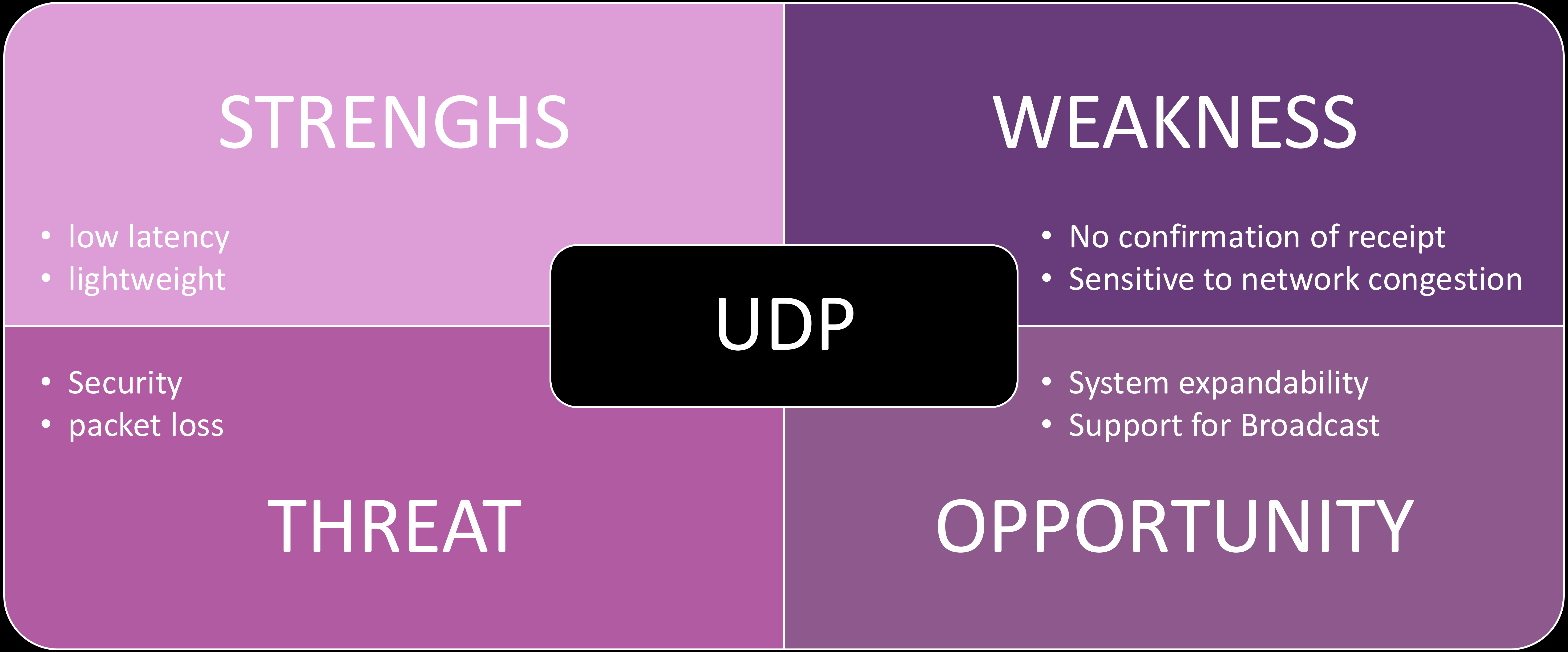
SOFTWARE



ARCHITECTURE DIAGRAM



SWOT ANALYSIS



STRENGHS

- low latency
- lightweight

WEAKNESS

- No confirmation of receipt
- Sensitive to network congestion

UDP

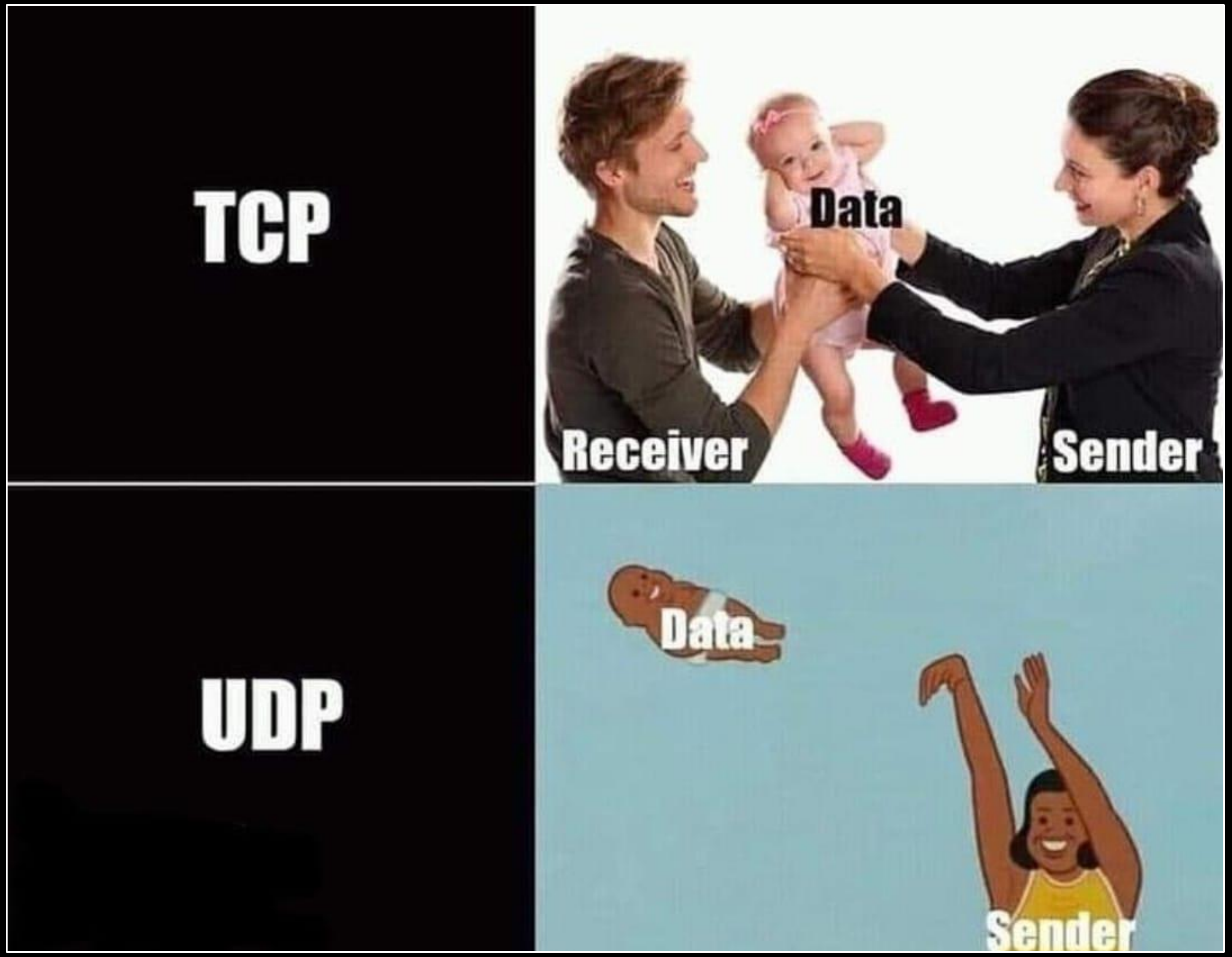
- Security
- packet loss

THREAT

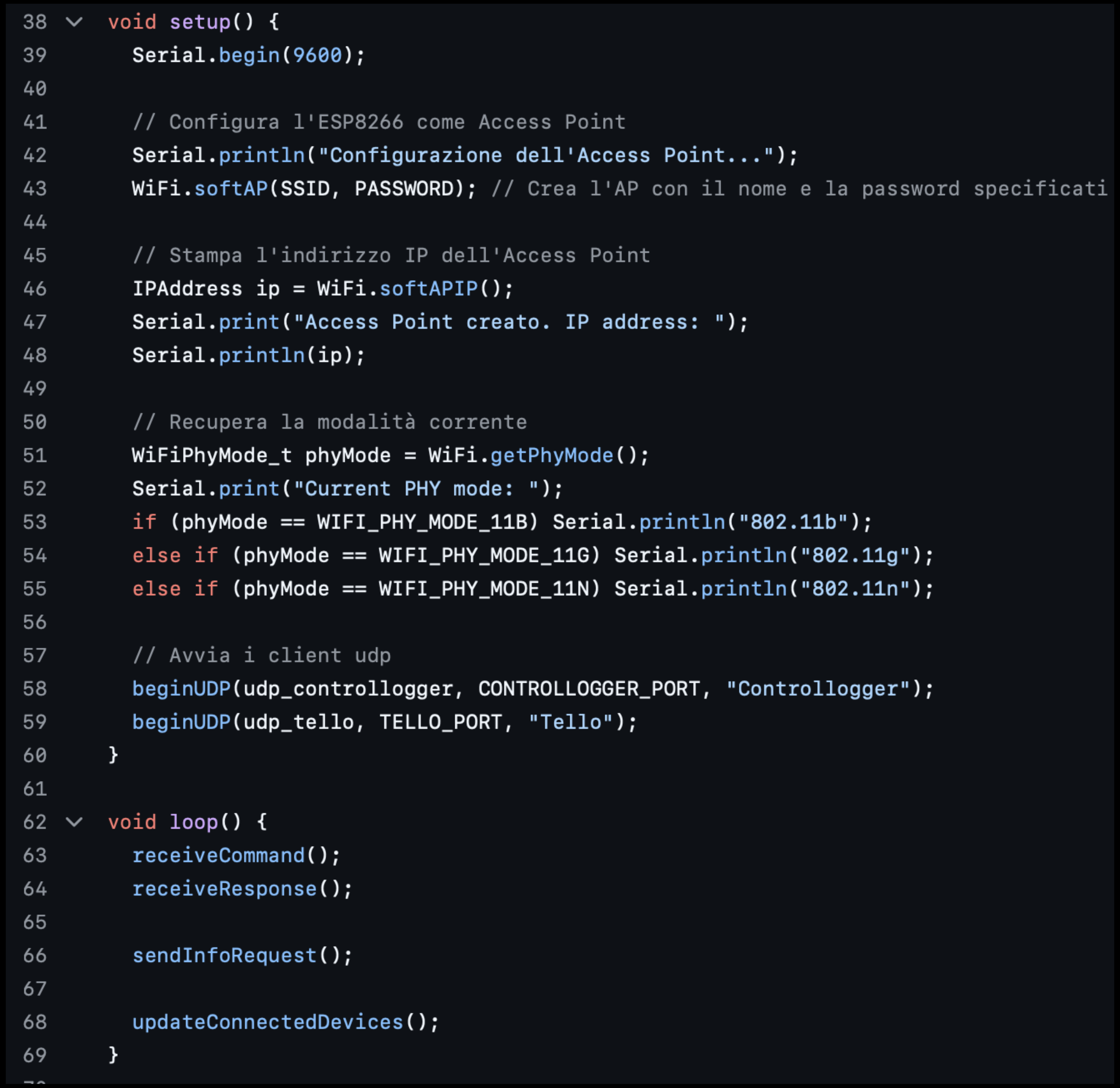
- System expandability
- Support for Broadcast

OPPORTUNITY

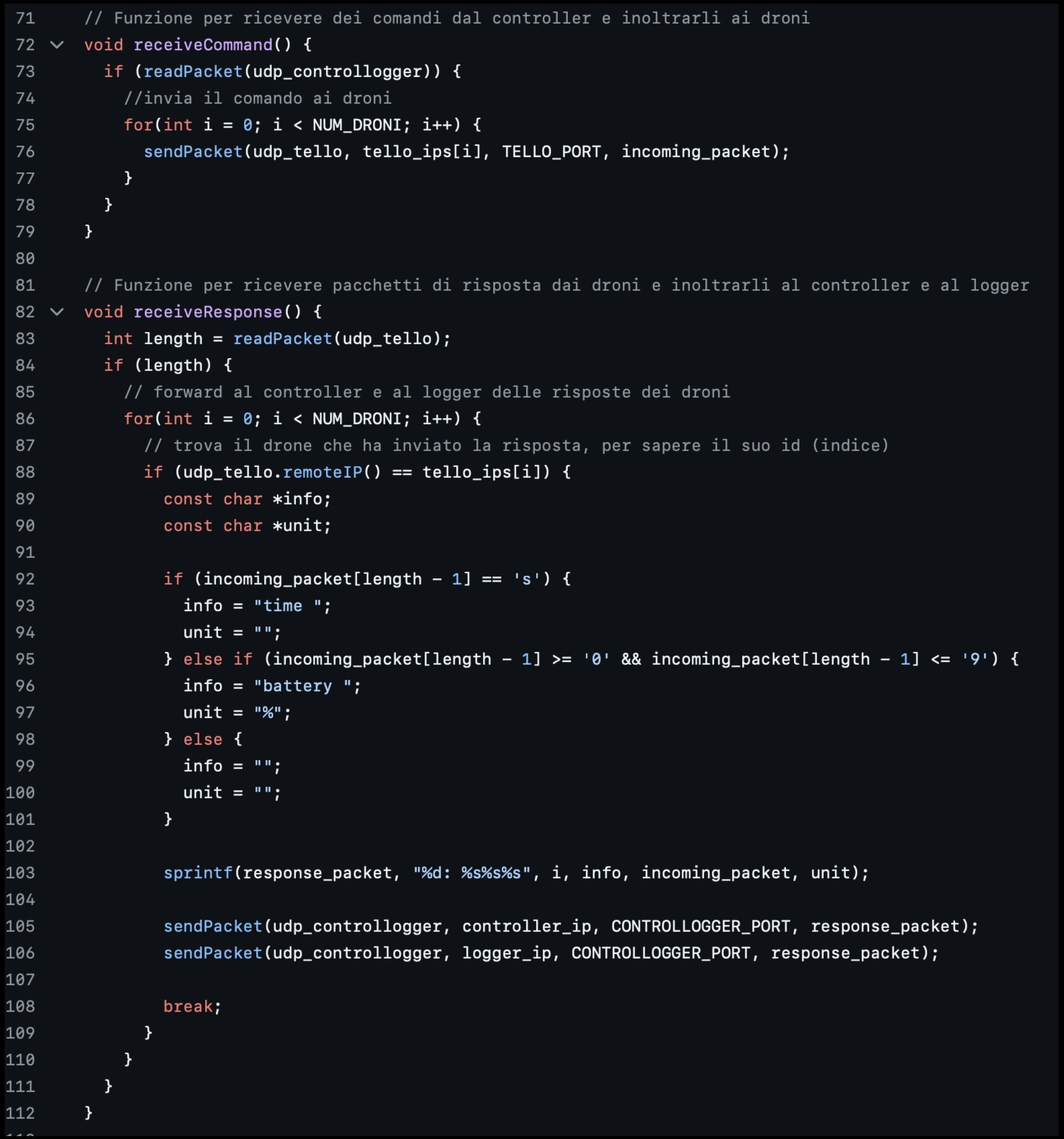
UDP



ESP8266



```
void setup() {
  Serial.begin(9600);

  // Configura l'ESP8266 come Access Point
  Serial.println("Configurazione dell'Access Point...");
  WiFi.softAP(SSID, PASSWORD); // Crea l'AP con il nome e la password specificati

  // Stampa l'indirizzo IP dell'Access Point
  IPAddress ip = WiFi.softAPIP();
  Serial.print("Access Point creato. IP address: ");
  Serial.println(ip);

  // Recupera la modalità corrente
  WiFiPhyMode_t phyMode = WiFi.getPhyMode();
  Serial.print("Current PHY mode: ");
  if (phyMode == WIFI_PHY_MODE_11B) Serial.println("802.11b");
  else if (phyMode == WIFI_PHY_MODE_11G) Serial.println("802.11g");
  else if (phyMode == WIFI_PHY_MODE_11N) Serial.println("802.11n");

  // Avvia i client udp
  beginUDP(udp_controllogger, CONTROLLOGGER_PORT, "Controllogger");
  beginUDP(udp_tello, TELLO_PORT, "Tello");
}

void loop() {
  receiveCommand();
  receiveResponse();

  sendInfoRequest();

  updateConnectedDevices();
}
```

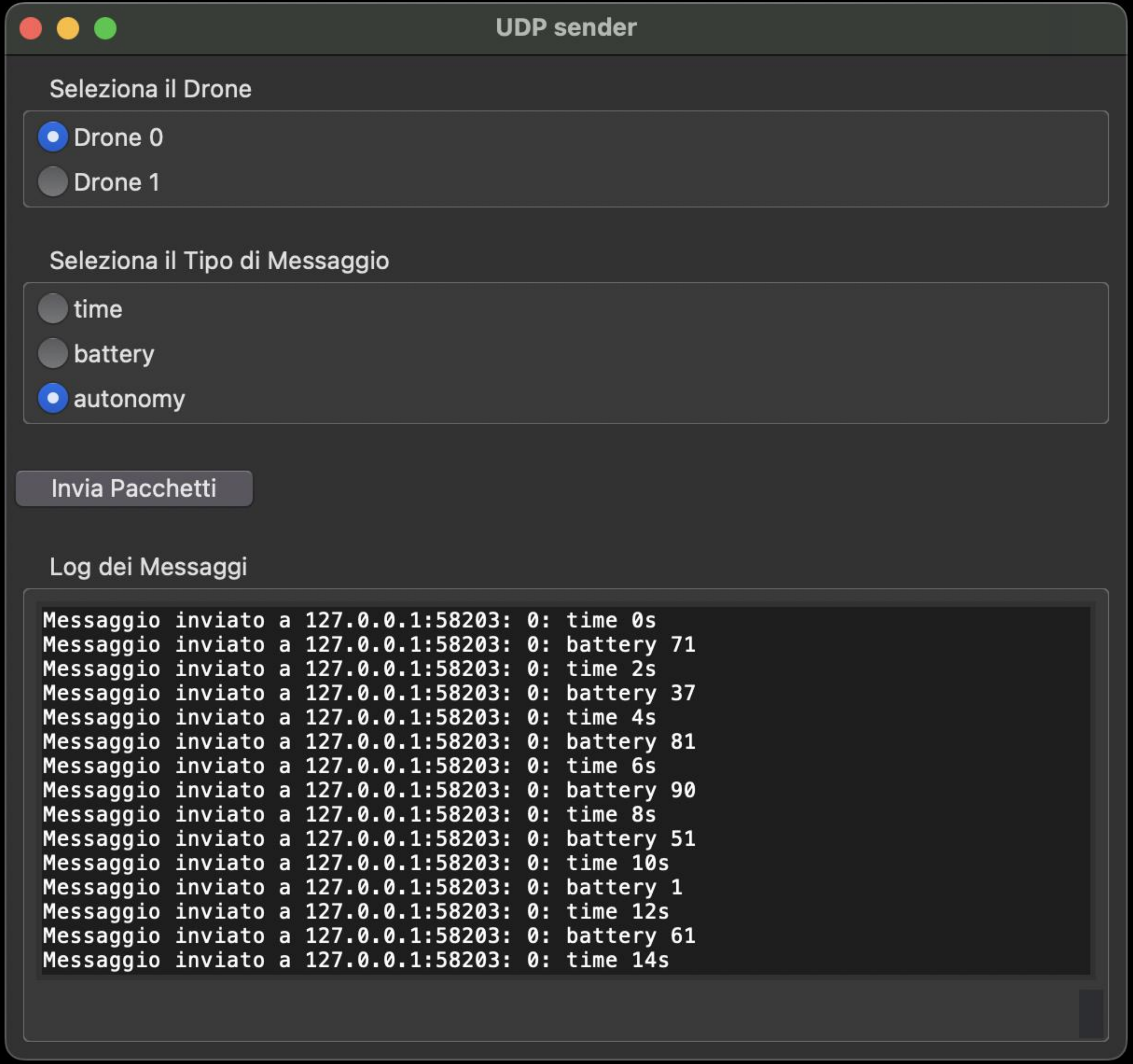


```
// Funzione per ricevere dei comandi dal controller e inoltrarli ai droni
void receiveCommand() {
  if (readPacket(udp_controllogger)) {
    //invia il comando ai droni
    for(int i = 0; i < NUM_DRONI; i++) {
      sendPacket(udp_tello, tello_ips[i], TELLO_PORT, incoming_packet);
    }
  }
}

// Funzione per ricevere pacchetti di risposta dai droni e inoltrarli al controller e al logger
void receiveResponse() {
  int length = readPacket(udp_tello);
  if (length) {
    // forward al controller e al logger delle risposte dei droni
    for(int i = 0; i < NUM_DRONI; i++) {
      // trova il drone che ha inviato la risposta, per sapere il suo id (indice)
      if (udp_tello.remoteIP() == tello_ips[i]) {
        const char *info;
        const char *unit;

        if (incoming_packet[length - 1] == 's') {
          info = "time ";
          unit = "";
        } else if (incoming_packet[length - 1] >= '0' && incoming_packet[length - 1] <= '9') {
          info = "battery ";
          unit = "%";
        } else {
          info = "";
          unit = "";
        }

        sprintf(response_packet, "%d: %s%s%s", i, info, incoming_packet, unit);

        sendPacket(udp_controllogger, controller_ip, CONTROLLOGGER_PORT, response_packet);
        sendPacket(udp_controllogger, logger_ip, CONTROLLOGGER_PORT, response_packet);

        break;
      }
    }
  }
}
```

PYTHON

Why use python?

-Simple way to mock sending udp packets



UDP sender

Seleziona il Drone

● Drone 0
○ Drone 1

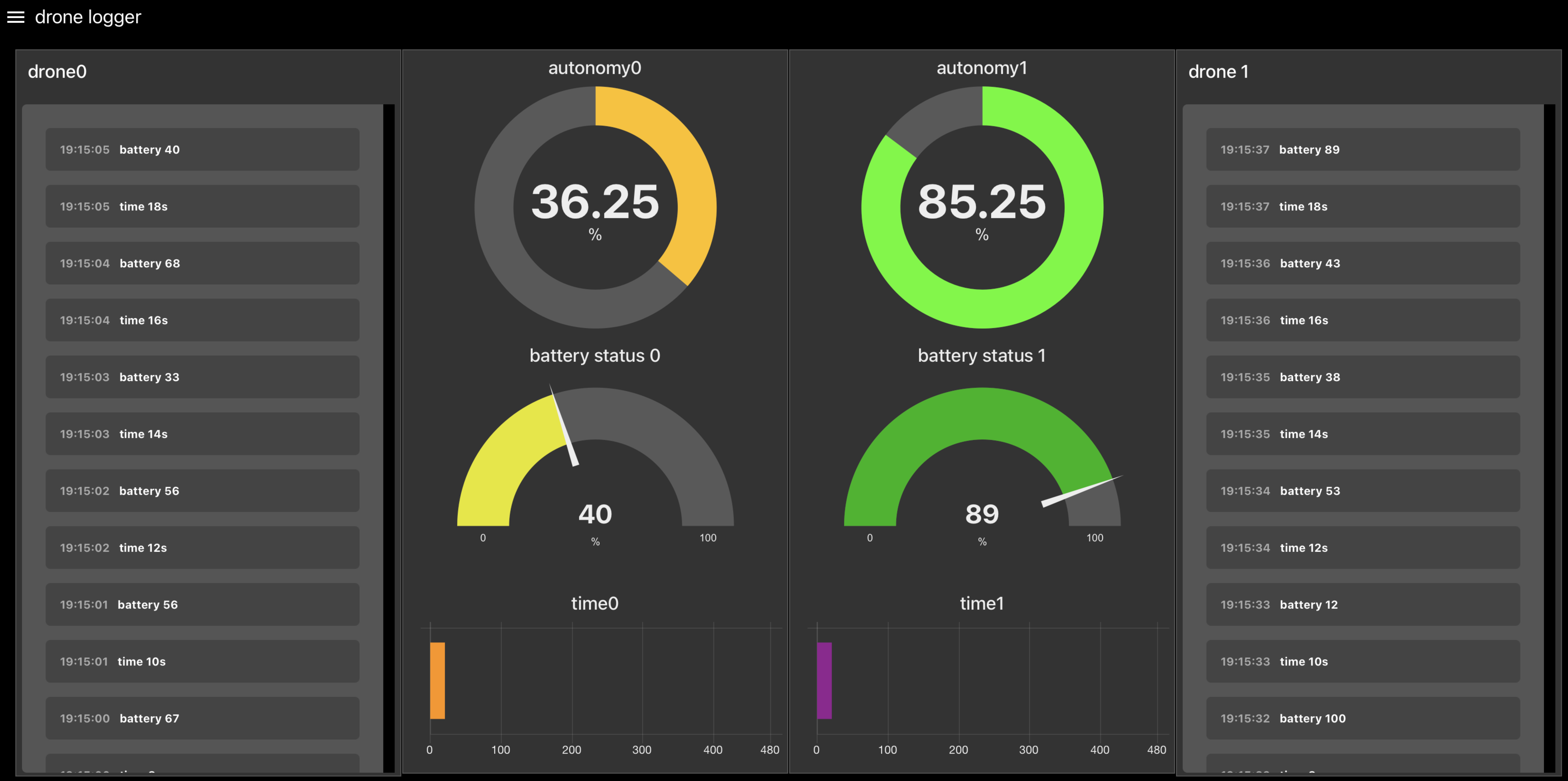
Seleziona il Tipo di Messaggio

○ time
○ battery
● autonomy

Log dei Messaggi

```
Messaggio inviato a 127.0.0.1:58203: 0: time 0s
Messaggio inviato a 127.0.0.1:58203: 0: battery 71
Messaggio inviato a 127.0.0.1:58203: 0: time 2s
Messaggio inviato a 127.0.0.1:58203: 0: battery 37
Messaggio inviato a 127.0.0.1:58203: 0: time 4s
Messaggio inviato a 127.0.0.1:58203: 0: battery 81
Messaggio inviato a 127.0.0.1:58203: 0: time 6s
Messaggio inviato a 127.0.0.1:58203: 0: battery 90
Messaggio inviato a 127.0.0.1:58203: 0: time 8s
Messaggio inviato a 127.0.0.1:58203: 0: battery 51
Messaggio inviato a 127.0.0.1:58203: 0: time 10s
Messaggio inviato a 127.0.0.1:58203: 0: battery 1
Messaggio inviato a 127.0.0.1:58203: 0: time 12s
Messaggio inviato a 127.0.0.1:58203: 0: battery 61
Messaggio inviato a 127.0.0.1:58203: 0: time 14s
```

NODE-RED



drone logger

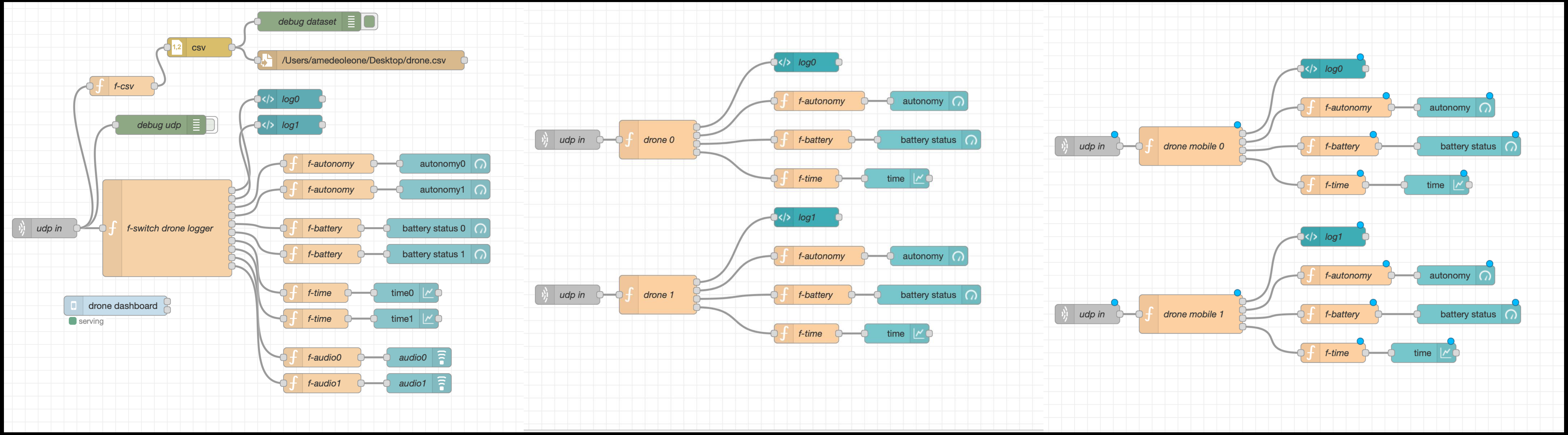
drone0

```
19:15:05  battery 40
19:15:05  time 18s
19:15:04  battery 68
19:15:04  time 16s
19:15:03  battery 33
19:15:03  time 14s
19:15:02  battery 56
19:15:02  time 12s
19:15:01  battery 56
19:15:01  time 10s
19:15:00  battery 67
```

drone 1

```
19:15:37  battery 89
19:15:37  time 18s
19:15:36  battery 43
19:15:36  time 16s
19:15:35  battery 38
19:15:35  time 14s
19:15:34  battery 53
19:15:34  time 12s
19:15:33  battery 12
19:15:33  time 10s
19:15:32  battery 100
```

NODE-RED DIAGRAM

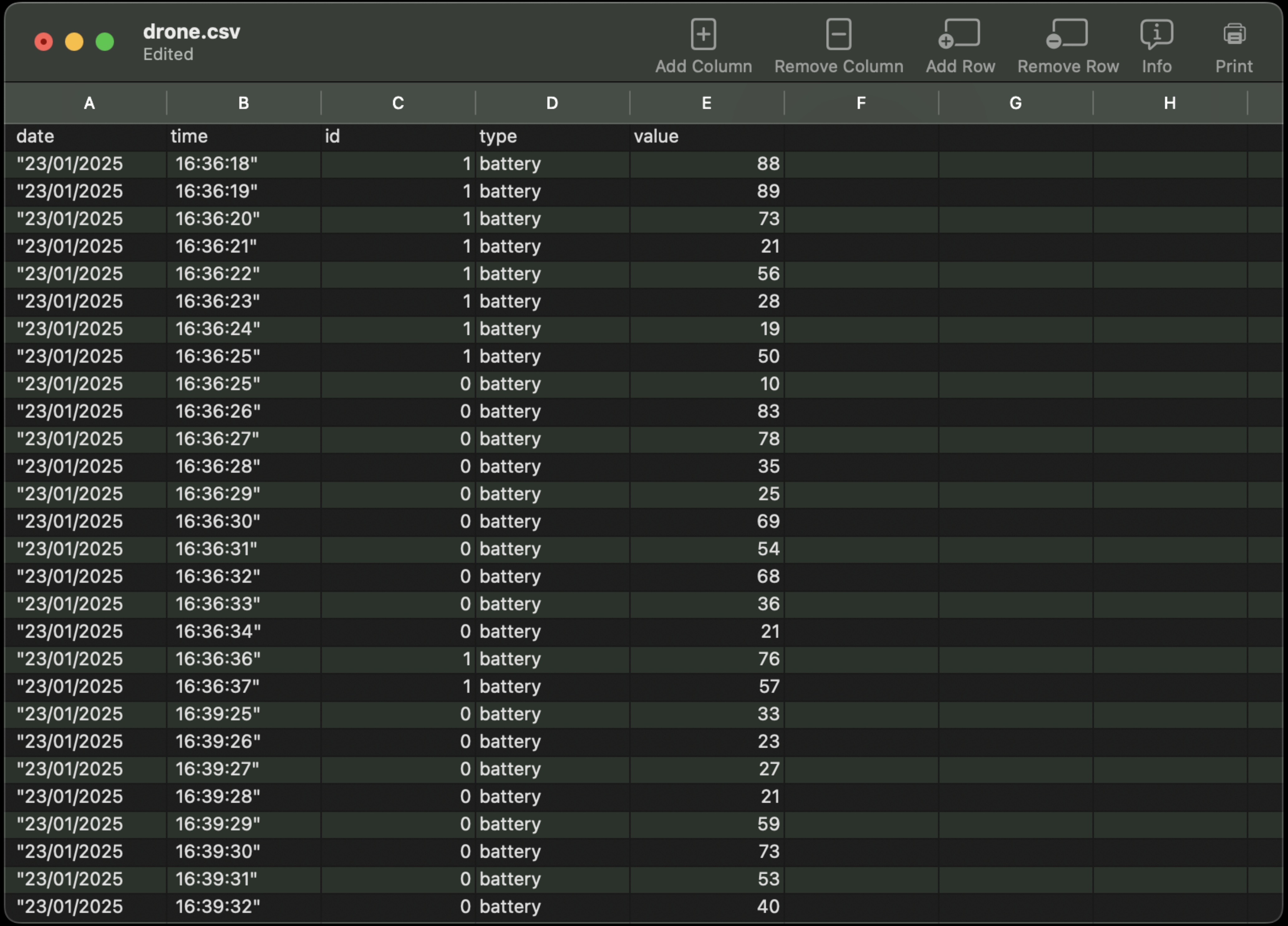


DATASET

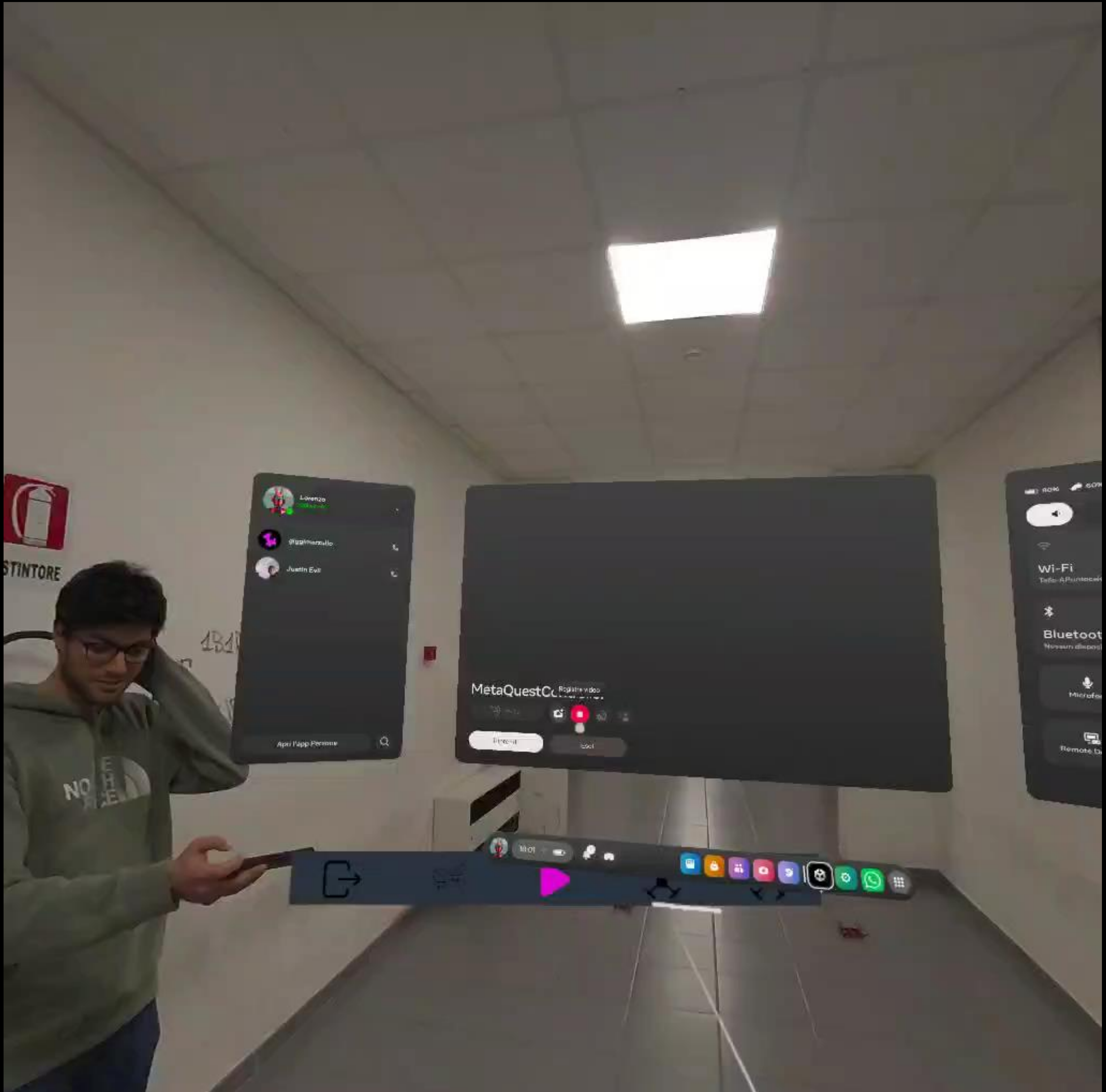
Why save data?

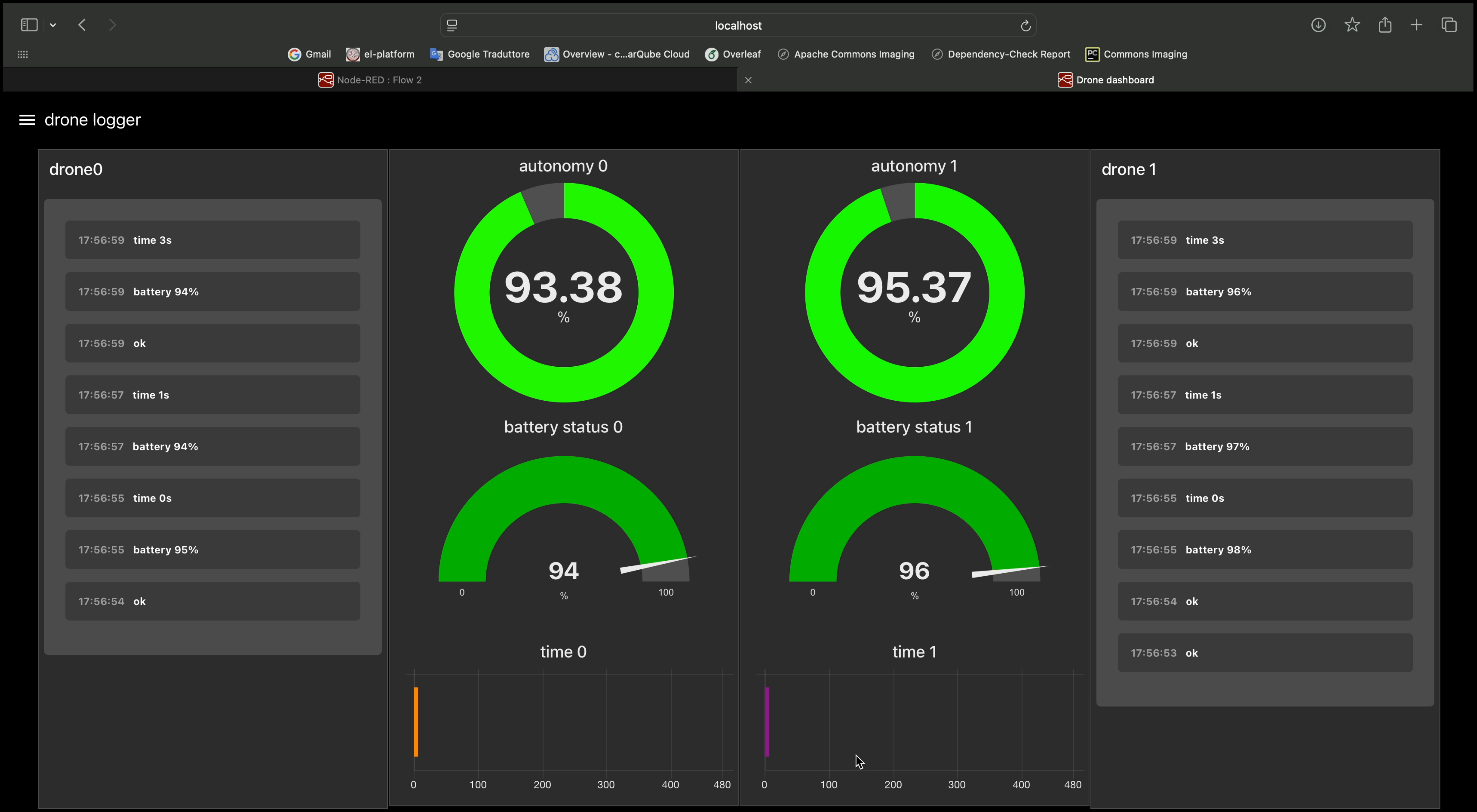
-Operations History

-Data Analysis



date	time	id	type	value
"23/01/2025	16:36:18"	1	battery	88
"23/01/2025	16:36:19"	1	battery	89
"23/01/2025	16:36:20"	1	battery	73
"23/01/2025	16:36:21"	1	battery	21
"23/01/2025	16:36:22"	1	battery	56
"23/01/2025	16:36:23"	1	battery	28
"23/01/2025	16:36:24"	1	battery	19
"23/01/2025	16:36:25"	1	battery	50
"23/01/2025	16:36:25"	0	battery	10
"23/01/2025	16:36:26"	0	battery	83
"23/01/2025	16:36:27"	0	battery	78
"23/01/2025	16:36:28"	0	battery	35
"23/01/2025	16:36:29"	0	battery	25
"23/01/2025	16:36:30"	0	battery	69
"23/01/2025	16:36:31"	0	battery	54
"23/01/2025	16:36:32"	0	battery	68
"23/01/2025	16:36:33"	0	battery	36
"23/01/2025	16:36:34"	0	battery	21
"23/01/2025	16:36:36"	1	battery	76
"23/01/2025	16:36:37"	1	battery	57
"23/01/2025	16:39:25"	0	battery	33
"23/01/2025	16:39:26"	0	battery	23
"23/01/2025	16:39:27"	0	battery	27
"23/01/2025	16:39:28"	0	battery	21
"23/01/2025	16:39:29"	0	battery	59
"23/01/2025	16:39:30"	0	battery	73
"23/01/2025	16:39:31"	0	battery	53
"23/01/2025	16:39:32"	0	battery	40


STINTORE
Lorenzo
Justin Evil
Apri l'app Persone
MetaQuestC
Registra video
Wi-Fi
Bluetoot
Microfo



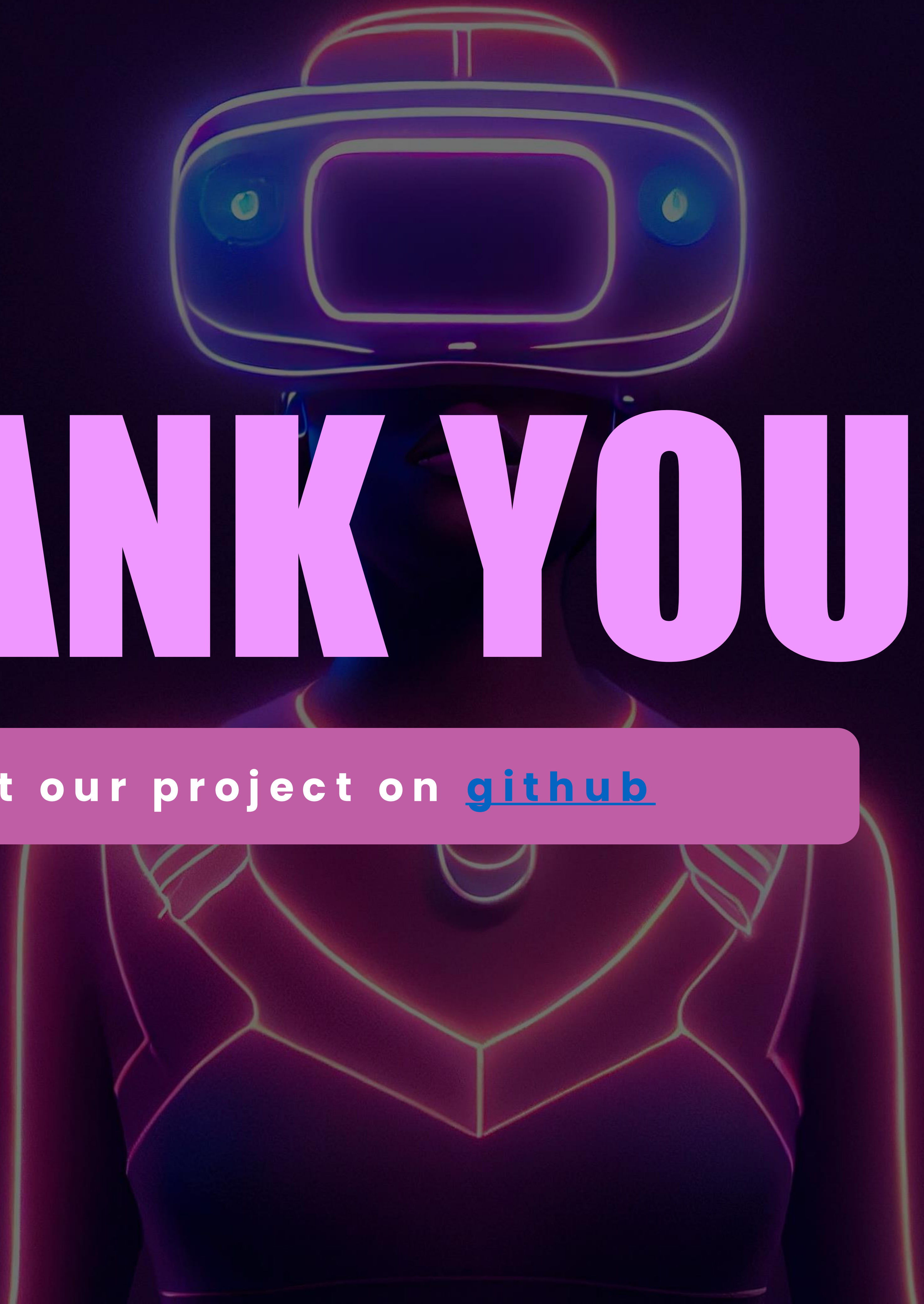
drone logger

drone0

17:56:59 time 3s
17:56:59 battery 94%
17:56:59 ok
17:56:57 time 1s
17:56:57 battery 94%
17:56:55 time 0s
17:56:55 battery 95%
17:56:54 ok

drone 1

17:56:59 time 3s
17:56:59 battery 96%
17:56:59 ok
17:56:57 time 1s
17:56:57 battery 97%
17:56:55 time 0s
17:56:55 battery 98%
17:56:54 ok
17:56:53 ok



THANK YOU

Visit our project on github