SISTEM DE MONITORIZARE A CALITATII AERULUI

Autor: Mihai-Octavian Coposescu

Coordonator: Prof. Dr. Ing. Zoltan Francisc BARUCH

Continutul prezentarii

- Contextul proiectului
- Obiectivele proiectului
- Solutia aleasa
- Proiectarea sistemului
- Implementarea solutiei
- Teste si rezultate
- Concluzii
- Bibliografie

Contextul Proiectului

Cum ne afecteaza calitatea aerului?

90% din timp il petrecem in spatii inchise...

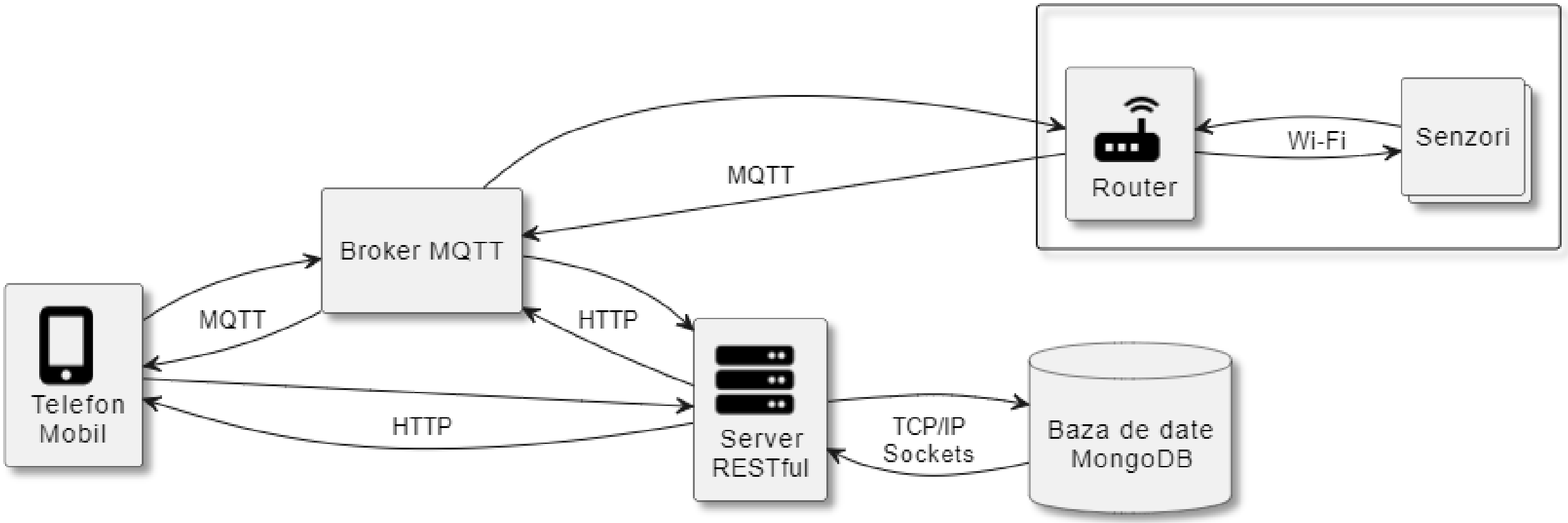
Obiectivele proiectului

- Masurarea parametrilor cheie de calitate a aerului.
- Citirea parametrilor in timp real.
- Citirea parametrilor istorici.
- Oferirea unei interfete intuitive si usor de utilizat.

Solutia aleasa

- Un sistem de monitorizare a calitatii aerului avand caracteristicile:
  - Senzori pentru esantionarea parametrilor de calitate a aerului:
    - Temperatura, Umiditate, VOC, PM1.0, PM2.5, PM4.0, PM10.0, TPS
  - Conexiune Wi-Fi in banda de frecventa 2.4GHz
  - Broker MQTT pentru datele in timp real
  - Baza de date MongoDB pentru datele istorice
  - Server RESTful pentru accesul la baza de date
  - Aplicatie Android reprezentand interfata cu utilizatorul

Proiectarea sistemului 1



Proiectarea sistemului 2

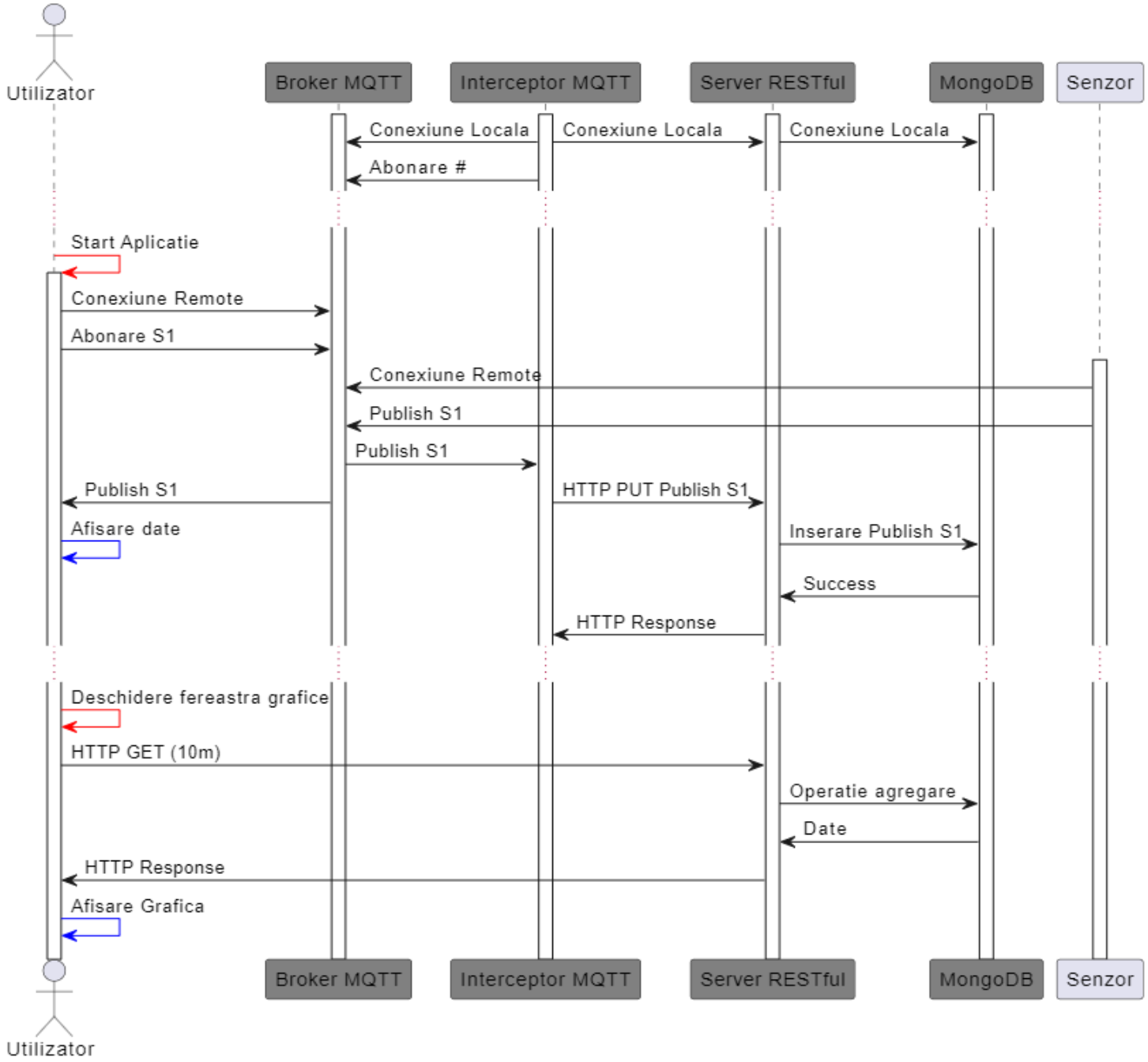


Diagrama de secventa

Proiectarea sistemului 3

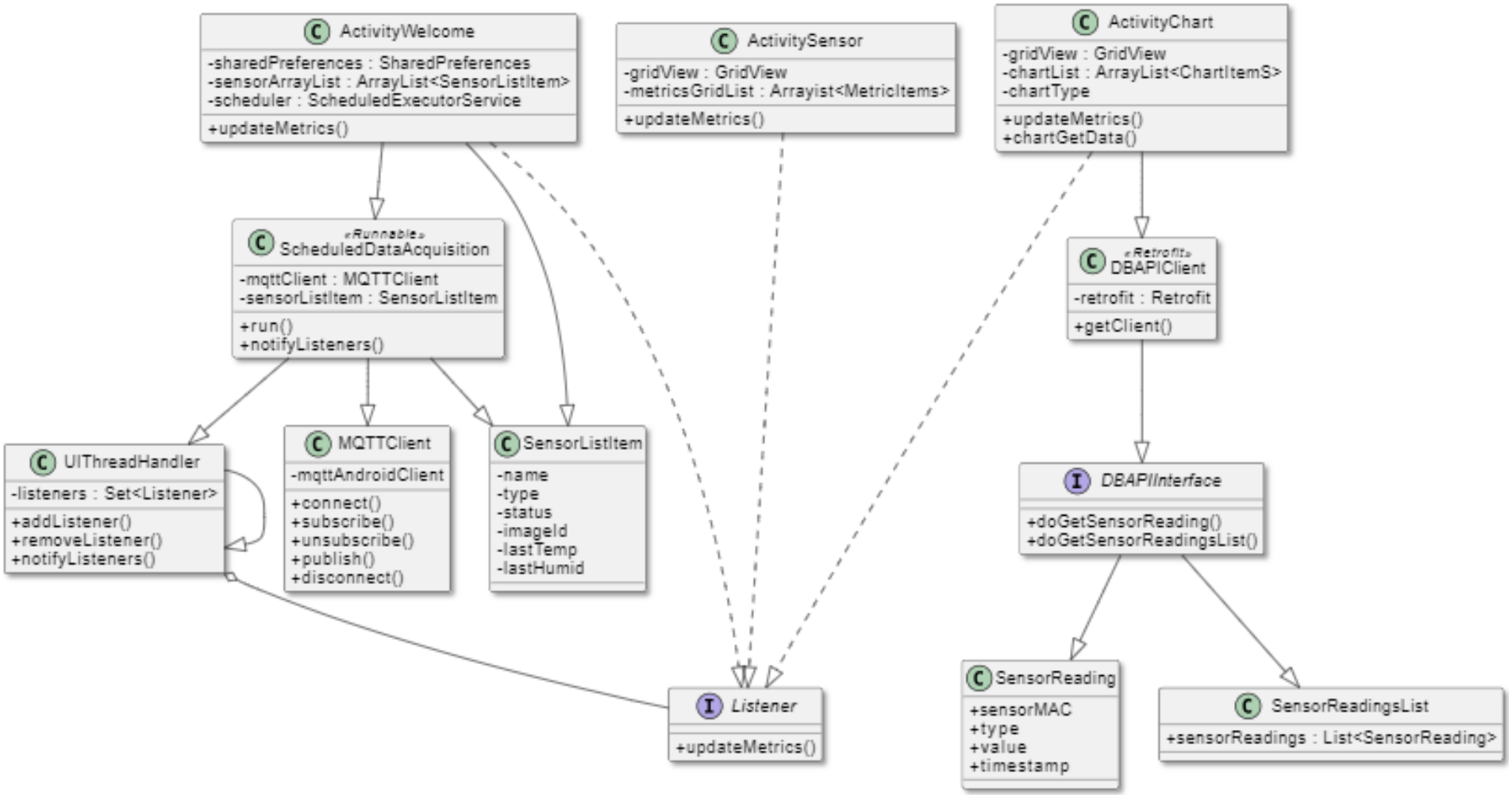
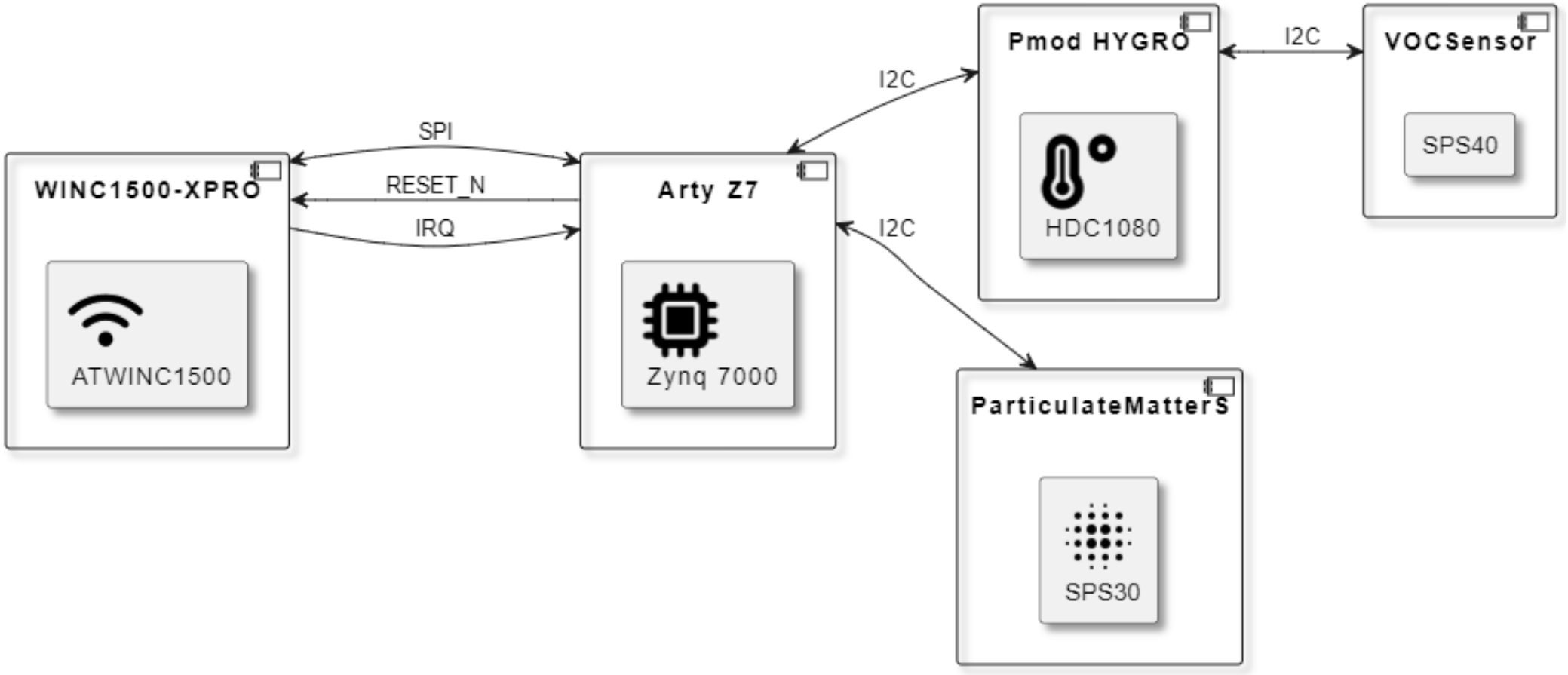


Diagrama de clase a aplicatiei Android

Implementarea solutiei 1

- Modulul senzor:
  - Placa de dezvoltare ArtyZ7
  - Senzori pentru esantionarea parametrilor de calitate a aerului:
    - Texas Instruments HDC1080 pentru temperatura si umiditate
    - Sensirion SPS30 pentru particulele in suspensie
    - Sensirion SGP40 pentru compusii organici volatili
  - Controller de retea ATWINC1500
  - Esantionarea periodica a parametrilor
  - Transmiterea parametrilor catre Brokerul MQTT

Implementarea solutiei 2



Arhitectura modulului senzor

Implementarea solutiei 3

- Broker MQTT:
  - Am utilizat Brokerul MQTT Mosquitto
  - Exemplu topic: “readings/F8F005ADB2A9/airQ1”
  - Subscriber wildcard pentru interceptarea mesajelor de la senzori
    - “readings/#”
  - Transmiterea mesajelor interceptate catre serverul RESTful
  - Salvarea mesajelor in baza de date


```
{
  "Temperature" : 29.468,
  "Humidity" : 59.301,
  "PM1.0" : 30.597,
  "PM2.5" : 32.356,
  "PM4.0" : 32.356,
  "PM10.0" : 32.356,
  "TPS" : 446,
  "VOCIndex" : 316
}
```


Formatul datelor

Implementarea solutiei 4

- Baza de date:
  - Am utilizat baza de date MongoDB
  - Colectii de tip serii in timp
  - Datele sunt salvate in documente
  - O colectie este formata din mai multe documente
  - MongoDB structureaza automat datele in Buckets


```
{
  timestamp: ISODate('2024-08-16T13:28:41.849Z'),
  metadata: { sensorMAC: 'F8F005ADB2A9', type: 'airQl' },
  'PM2.5': 19.96,
  _id: ObjectId('66bf5409878dda90726c34a2'),
  'PM1.0': 18.875,
  VOCIndex: 100,
  TPS: 418,
  'PM4.0': 19.96,
  Humidity: 56.689,
  Temperature: 29.75,
  'PM10.0': 19.96
}
```


Exemplu document din baza de date

Implementarea solutiei 5

- Serverul RESTful:
  - Am utilizat biblioteca Flask
  - Definire rute de intrare in server (adrese URl) (Rute)
  - Validarea datelor care intra si care ies din server (Controller)
  - Definire resurse specifice fiecarei adrese URL (Model)
  - Formatare date raspuns (View)
  - Marcaj temporal UTC

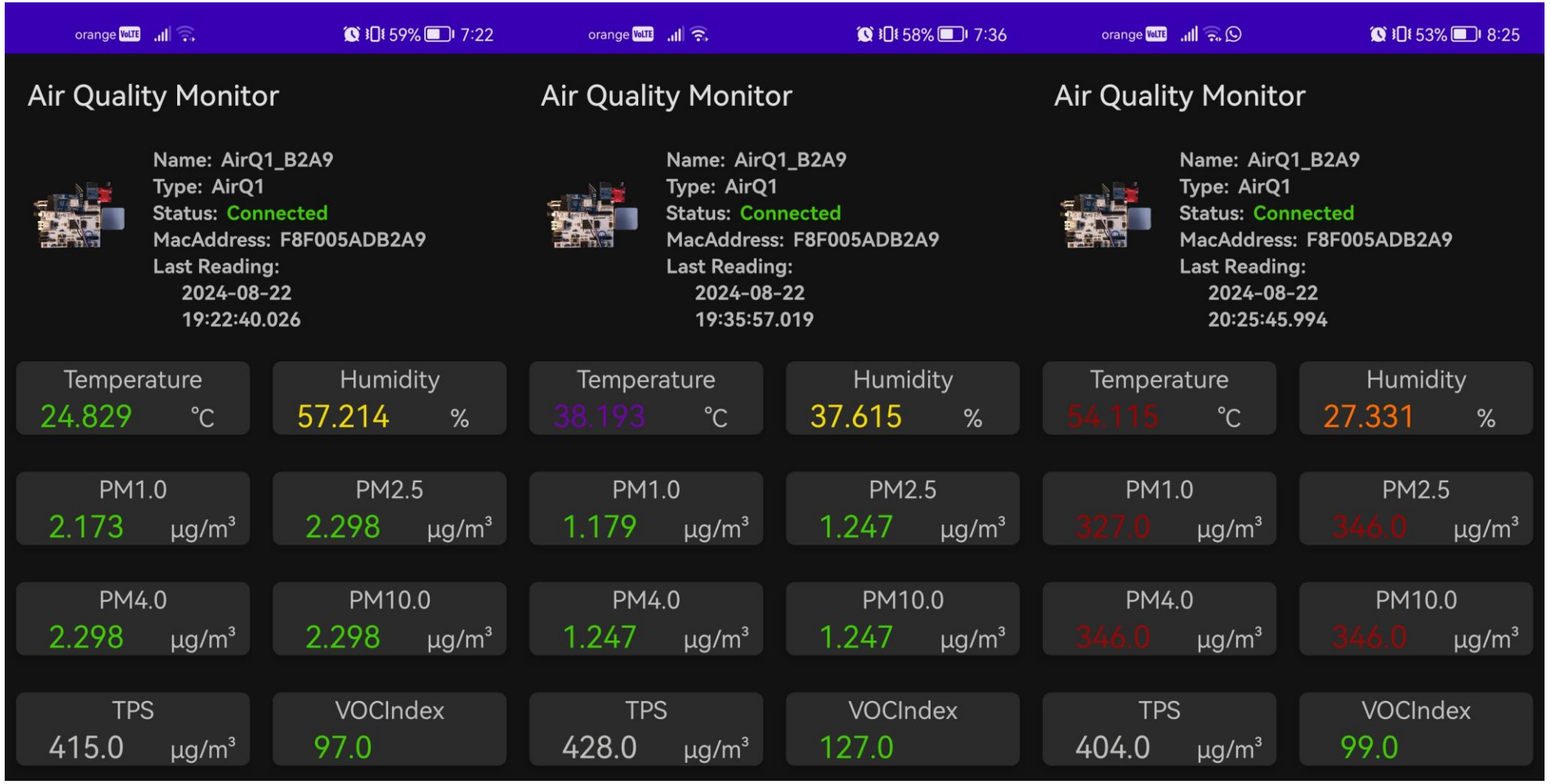
Implementarea solutiei 6

- Aplicatie Android:
  - Pagina principala prezinta o lista a senzorilor accesati
  - Selectarea unui senzor deschide pagina de vizualizare a parametrilor
  - Selectarea oricarui parametru din lista deschide pagina de grafice
  - Mentine o conexiune MQTT cu Brokerul
  - Realizeaza cereri HTTP pentru afisarea graficelor
  - Datele senzorilor instalati sunt salvate in memoria telefonului

Teste si rezultate 1

- Variatia mediului la care sunt expusi senzorii
  - Variatia temperaturii
  - Variatia particulelor in suspensie
- Masurarea consumului senzorului
  - 150 mA/h
- Scalabilitatea sistemului
  - 20 de senzori simulati

Teste si rezultate 2



Parametrii pe durata testului de variatie a temperaturii

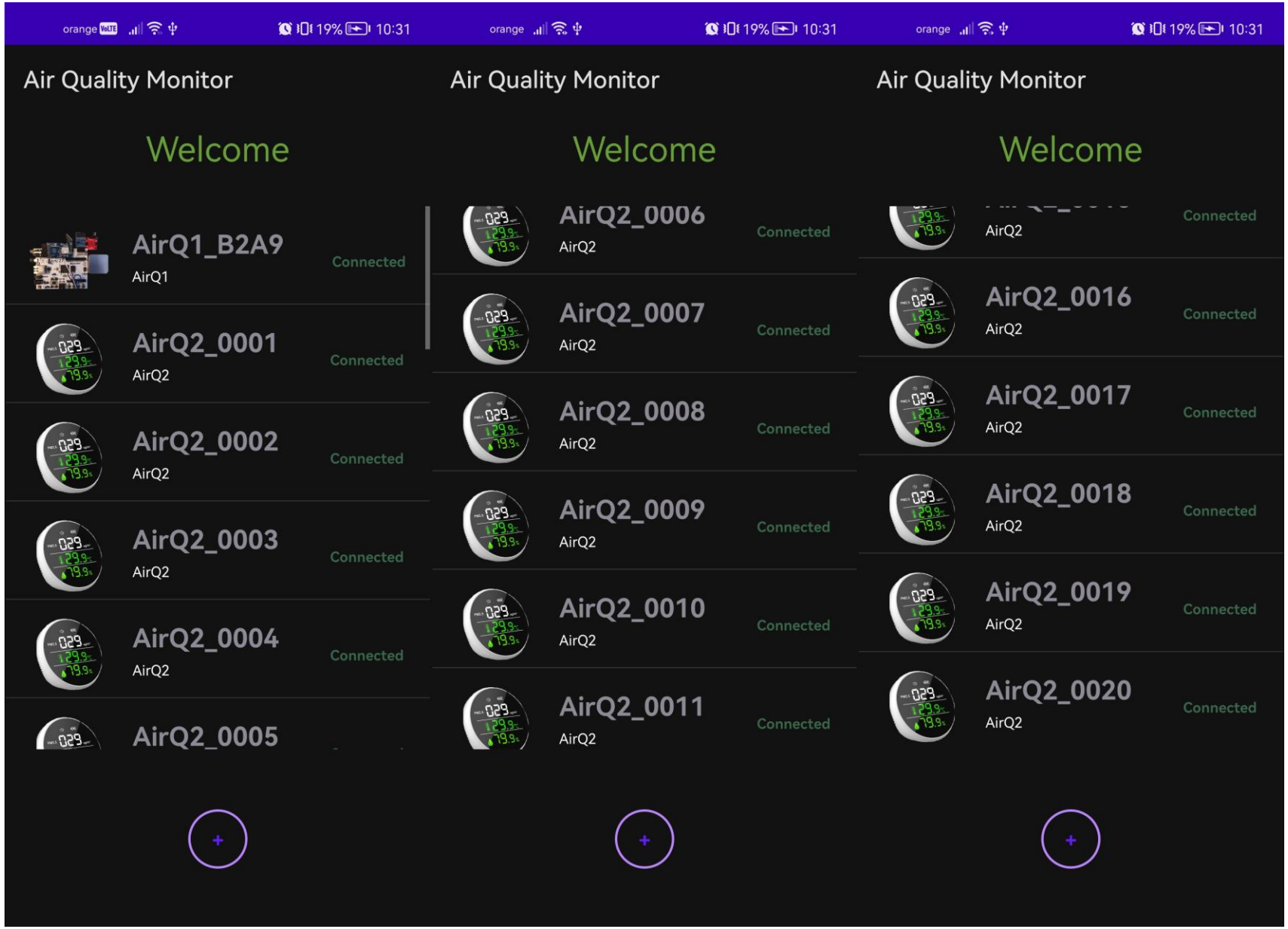
Teste si rezultate 3

Graficele la finalul testului de variatie a temperaturii

Teste si rezultate 4

- Latenta sistemului
  - Retea de 21 de senzori
  - T_timp_real = 1.624 secunde
  - T_istoric_10m = 134 milisecunde
  - T_istoric_6h = 634 milisecunde
  - T_istoric_1d = 1.556 secunde
  - T_db = 1.379 secunde

Teste si rezultate 5



Capturi de ecran din timpul testului de scalabilitate

Concluzii

- Contributii personale:
  - Integrarea si imbunatatirea driverului Paho MQTT embedded
  - Integrarea controllerului de retea ATWINC1500 cu procesorul Zynq 7000
  - Structurarea datelor in baza de date pentru performante optime
  - Raportarea a 2 parametrii de calitate a aerului mai putin uzuali
    - PM4.0 si TPS

Bibliografie

- W. Stallings, Data and Computer Communications, 8th ed. Pearson Education,2007.
- M. Grinberg, Flask Web Development, Developing Web Applications with Python,2nd ed. O'Reilly Media, 2018.
- MQTT Version 3.1.1, OASIS Standard, 29 October 2014. [Online]. Available:http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html
- P. Done, Practical MongoDB Aggregations. Packt, 2023.

Multumesc pentru atentie!