SVEUČILIŠTE U SPLITU

FAKULTET ELEKTROTEHNIKE, STROJARSTVA I BRODOGRADNJE

IZVJEŠTAJ PROJEKTA

KVALITETA ZRAKA

Martina Batinić i Ana Kuten

Split, srpanj 2020.

SADRŽAJ

1. UVOD

U današnje vrijeme sve češće se spominju razne klimatske promjene i za većinu njih je glavni krivac čovjek. Zagađenje zraka različitim ispušnim plinovima jedan je od glavnih problema. U ovom radu projektiran je jednostavan sustav za ispitivanje kvalitete zraka, konkretno razine ugljikovog dioksida (CO2) u zraku.

Bežične senzorske mreže (WSN) spadaju u kategoriju *ad hoc* mreža bez teške infrastrukture uspostavljenih za neku posebnu svrhu. Kod takvih mreža fokus je na interakciji mreže s čovjekovim okruženjem, a ne sa samim čovjekom. WSN sustavi su građeni od tzv. čvorova (*engl. nodes*), koji međusobno komuniciraju slanjem i primanjem podataka. Podaci putuju od čvora do čvora i obrađuju se. Izvorište podataka (*eng. data source*) „mjeri“ okolinu i šalje prikupljene/obrađene podatke nekom odredištu. Tipično je opremljeno različitom vrstom detektora. Odredište podataka (*eng. data sink*) očekuje i prima podatke iz WSN i može biti dio mreže ili vanjski element kao npr. PDA ili bazna stanica. Aktuator upravlja nekakvim uređajem.

Iako se za realizaciju ovog projekta kao napajanje koristi USB port, u realnim uvjetima se za napajanje koristi baterija. Najveći izazov kod ovakvih mreža predstavlja smanjenje potrošnje cijelog sustava zbog ograničenog kapaciteta baterije. Zbog toga je potrebno senzor staviti u "*sleepmode*" kad je god to moguće te ga periodički paliti kako bi mogao izmjeriti trenutno stanje okruženja [1].

MQ135 senzor očitava vrijednost CO2 u zraku te se ti podaci prikupljaju na Arduino UNO mikrokontroleru koji služi kao predajnik. Putem radio nRF24L01 modula ostvarena je radio-komunikacija. Podaci se šalju Arduino UNO mikrokontroleru koji služi kao prijemnik. Putem MQTT brokera podaci se šalju na server, a na temelju prikupljenih podataka u Grafani je kreiran graf koji prikazuje vrijednosti CO2.

2. KORIŠTENA OPREMA

2.1. Arduino UNO

Arduino je elektronička prototipna platforma namjenjena kreiranju elektroničkih projekata. Sastoji se od hardware dijela koji je zapravo fizički elektronički programabilni strujni krug (poznat kao i mikrokontroler) i software dijela koji se naziva IDE (*eng. Integrated Development Environment*) kojega se pokreće na računalu i iz njega programira i upravlja samom pločicom.

Samo programiranje pločica ne zahtijeva dodatan dio hardwarea (koji se zove programator) kao za ostale mikrokontrolerske sustave, nego je dovoljan USB kabel koji se može povezati sa svakim računalnom, bez obzira na operacijski sustav. Arduino se može programirati iz Windowsa, Mac-a, Linuxa, Androida... Programiranje pločice se obavlja u prilagođenoj verziji C++ programskog jezika [2].

Arduino Uno je mikrokontroler pločica koja se temelji na ATmega328 mikročipu. Na slici 2.1. prikazane su tehničke specifikacije pločice.

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current for I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328)
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz

Slika 2.1. Tehničke specifikacije Arduino UNO pločice [3]

Opće funkcije pin-ova:

LED: Postoji ugrađena LED dioda koju pokreće digitalni pin 13.

VIN: Ulazni napon na ploču kada se koristi vanjski izvor napajanja.

5V: Ovaj pin izvodi reguliranih 5V od regulatora na ploči. Ploča se može napajati iz utičnice istosmjerne struje (7 - 20V), USB priključka (5V) ili VIN pina na ploči (7-20V).

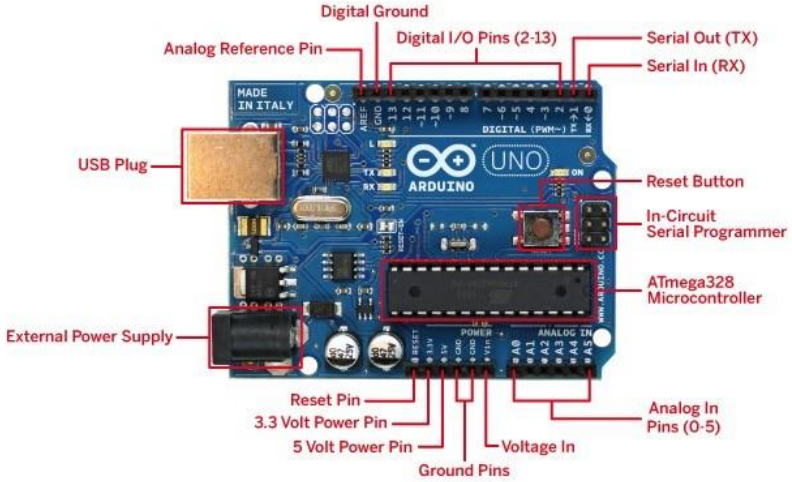
3V3: Napajanje od 3,3 volta koje generira ugrađeni regulator. Maksimalna jakost struje je 50 mA.

GND: uzemljenje.

IOREF: Ovaj pin na ploči pruža referentnu vrijednost napona s kojom mikrokontroler radi.

RESET: uobičajeno se koristi za dodavanje gumba za resetiranje.

Svaki od 14 digitalnih pinova i 6 analognih pinova se može koristiti kao ulaz ili izlaz, pod softverskom kontrolom, koristeći funkcije pinMode (), digitalWrite () i digitalRead (). Svaki pin može osigurati ili primiti 20 mA kao preporučeno radno stanje i ima unutarnji otpornik (po defaultu isključen) od 20-50 kΩ. Ne smije se prekoračiti maksimalno 40 mA na bilo kojem I / O pinu kako bi se izbjeglo trajno oštećenje mikrokontrolera. Uno ima 6 analognih ulaza s oznakom A0 do A5; svaki daje 10 bita razlučivosti (tj. 1024 različite vrijednosti) [4]. Na slici 2.2. označeni su glavni dijelovi Arduino UNO pločice.



Slika 2.2. Izgled Arduino UNO pločice [5]

2.2. Nordic nRF24L01+

Nordic nRF24L01+ je primopredajnik kojega karakterizira niska potrošnja (*eng. Ultra Low Power - ULP*). Maksimalna brzina komunikacije podataka koja se može postići je do 2 Mbps na frekvenciji od 2,4 GHz ISM (*eng. Industrial, scientific and medical*). Ovi primopredajnici koriste 2,4 GHz nelicencirani pojas poput mnogih WiFi routera, Bluetooth, nekih bežičnih telefona itd; čiji je raspon od 2,400 do 2,525 GHz. Širina nRF24L01 kanala je 1 MHz što ukupno omogućava komunikaciju na 125 nepreklapajućih kanala (0 .. 124). Potrošnja nRF24L01+ radio modula prilikom transmisije/primanja paketa je u aktivnom modu manja od 14mA, dok je modu spavanja manja od 1 uA, što ga uz napajanje od 1,9 do 3,6V čini idealnim za realizaciju primopredajnika male potrošnje korištenjem AA/AAA baterija. Za realizaciju ukupnog radio sustava sa nRF24L01+ primopredajnikom, potreban je i mikrokontroler (npr. Arduino) te još neke pasivne komponente.

Svojstva:

Frekvencijski pojas 2,4 GHz,

126 radio kanala,

Brzina transmisije: 250kbps, 1 and 2Mbps,

GFSK modulacija, pojasne širine od 1 or 2MHz,

Transmiter: 11.3mA na 0dBm izlazne snage,

Napajanje: od 1,9 V do 3,6 V,

Mala veličina: 15mm*29mm,

Mala potrošnja snage,

900nA sleep stanje,

11.3mA Radio TX na 0dBm,

13.3mA Radio RX na 2Mbps,

Izlazna snaga 0, -6, -12, i -18dBm,

Osjetljivost -94dBm RX pri 250 Kbps,

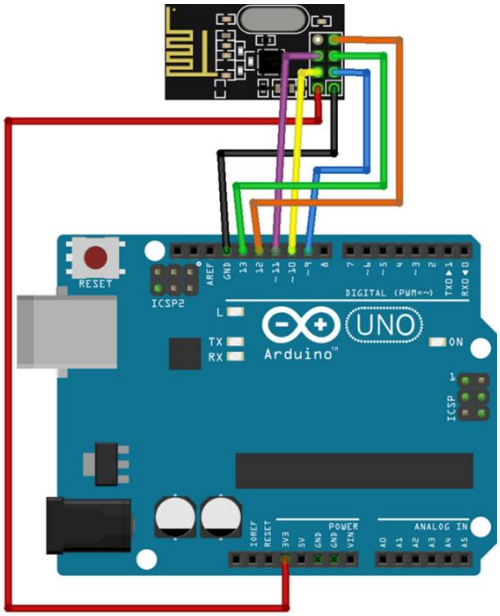
Osjetljivost -82dBm RX pri 2 Mbps,

Osjetljivost -85dBm RX pri 1 Mbps.

Na slici 2.3. prikazan je Nordic nRF24L01+ primopredajnik.

Slika 2.3. Nordic nRF24L01+ primopredajnik

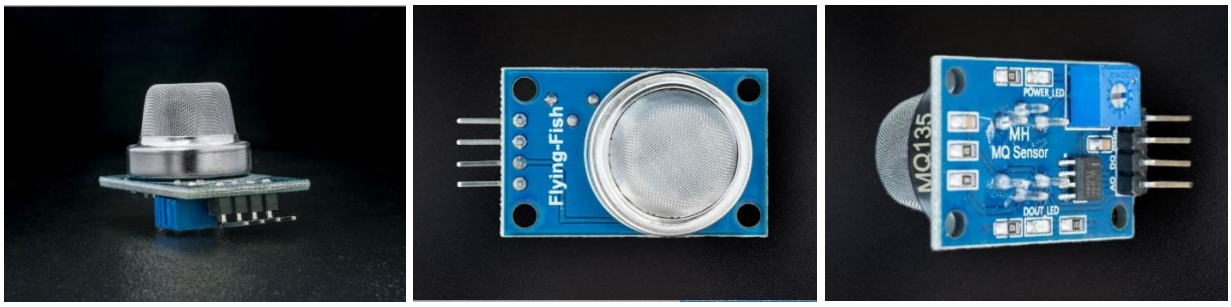
Prilikom slanja paketa svi uređaji koji slušaju na tom istom radio kanalu će primiti poruku. Transmiter prilikom slanja dodaje adresu primatelja u poruku (tzv. pipe) te prijemnik s druge strane ignorira sve poruke koje nemaju tu adresu. Ukupna veličina adrese je 5 byte-ova (40 bitova). Nordic nRF24L01+ modul može istovremeno slušati poruke sa 6 pipe-ova. Ti pipe-ovi omogućavaju nRF24 istovremeno slušanje poruka sa 6 različitih uređaja koje imaju 6 različitih (jedinstvenih) adresa. Slika 2.4. prikazuje shemu spajanja nRF24L01 modula s Arduino UNO pločicom.



Slika 2.4. Shema spajanja nRF24L01 modula s Arduinom

2.3 MQ-135 senzor plinova

Senzor plinova MQ135 registrira prisutnost brojnih plinova u zraku koji uvjetuju općenitu kvalitetu zraka. Najbolje registrira dok je sam alkohol u tekućem stanju, ali sasvim dobro mjeri i koncentraciju tvari u plinovitom stanju. Potrebno mu je kratko vrijeme zagrijavanja kako bi ispravno radio. Slika 2.5. prikazuje izgled MQ135 senzora.



Slika 2.5. MQ-135 senzor plinova

• Detektira: amonijak, sulfide, pare benzena, dim, druge opasne plinove.

• Dimenzije: 32mm x 22mm x 27mm.

• Napon: 5V.

• Digitalni(HIGH/LOW) i analogni(0V-5V) izlaz.

• S LM393 komparatorom.

• Dolazi s zalemljenim 4-pinskim pod kutem muškim headerom**.** [6]

Ovisno o vrsti plina i koncentraciji potrebno je podešavati osjetljivost senzora.

Karakteristike senzora:

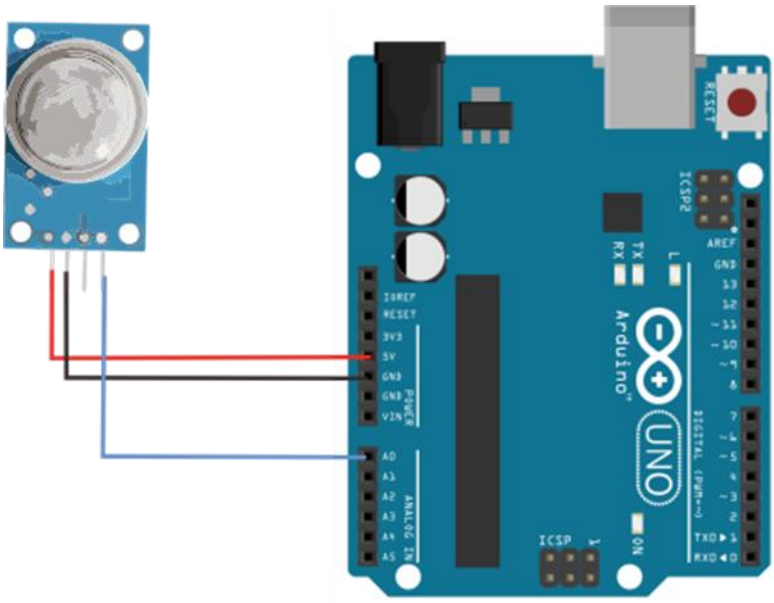
širok opseg detekcije,

brz odziv i velika osjetljivost,

stabilan i dug vijek trajanja,

jednostavan pogonski krug. [7]

Na slici 2.6. prikazana je shema spajanja MQ135 senzora s Arduino UNO pločicom.



Slika 2.6. Shema spajanja Arduina i senzora za plin

3. MQTT

MQTT je protokol za komunikaciju razvijen u IBMu. Radi na principu publish/subscribe te je kreiran je kao jednostavan protokol za upotrebu. Dizajniran je kako bi se minimizirala propusnost mreže i zahtjevi resursa uređaja. Jedna od bitnih značajki protokola je osiguranje pouzdanosti i određenog stupnja isporuke poruke. Najviše se koristi u IoT (eng. *Internet of Things* – Internet stvari) uređajima i M2M (eng. *Machine to Machine*) komunikaciji gdje je potreban protokol koji ne povećava potrošnju energije. Vrlo je brz u prijenosu podataka. Minimizira pakete podataka, troši malo energije i štedi bateriju povezanog uređaja.

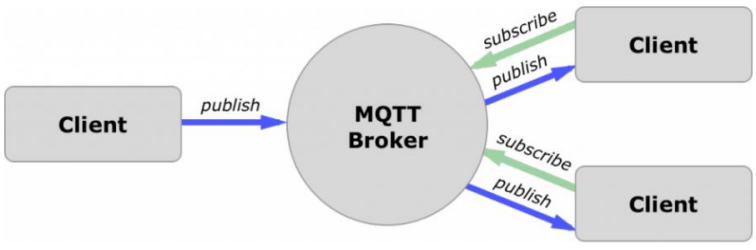
Ovakav model razdvaja klijenta koji šalje poruku (publish) od klijenta koji prima poruku (subscribe). Publisher i subscriber nikada izravno ne komuniciraju.Veza između njih se upravlja brokerom. Broker filtrira sve dolazne poruke i pravilno ih distribuira preplatnicima.

MQTT koristi topic (temu) kako bi odredio koja poruka ide kojem klijentu (pretplatniku). Tema je hijerarhijski strukturiran niz koji se može koristiti za filtriranje i usmjeravanje poruka. Sastoji se od jedne ili više razina. Publisher definira neku temu i objavljuje poruku pod tom temom. Subscriber se preplaćuje na tu temu i prima tu poruku. Uloga brokera je da preuzme poruku i dostavi je subscriberu

Quality of Service (QoS) je bitan faktor MQTT protokola. On definira garanciju isporuke za određenu poruku. Postoje 3 razine QoS u MQTT-u:

1. Najviše odjednom (0)
2. Barem jednom (1)
3. Točno jednom (2). [8]

Na slici 3.1. prikazana je arhitektura MQTT protokola.



Slika 3.1. MQTT protokol [9]

4. DOCKER

Kontejner (*eng. Container*) predstavlja jedinicu softvera koja pakira cjelokupan kod, aplikaciju, potrebne biblioteke i komponente o kojima aplikacija ovisi. U kontejner se praktički pakira sve ono što je potrebno za pokretanje neke aplikacije. Na taj način se kontejner može prenositi i pokretati bilo gdje, neovisno o distribuciji operacijskog sustava i infrastrukturi na kojoj se pokreće. Docker je alat dizajniran kako bi korisniku olakšao stvaranje, implementaciju i pokretanje aplikacija pomoću kontejnera.

Multi-container Docker aplikacija upotrebljava sljedeće servise:

Node-RED - flow-based razvojni alat za vizualno programiranje (host port: 1880),

InfluxDB - time series baza podataka (host port: 8086),

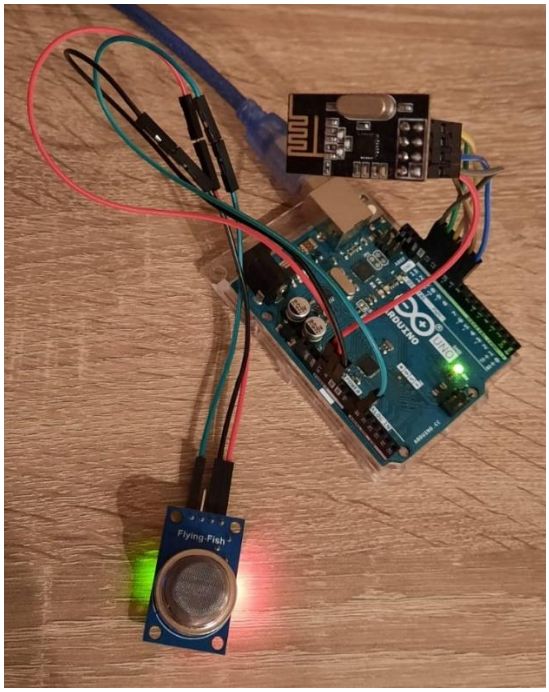
Chronograf - admin korisničko sučelje za InfluxDB (host port: 127.0.0.1:8888),

Grafana - vizualizacijsko korisničko sučelje za InfluxDB (host port: 3000). [10]

5. PRAKTIČNA REALIZACIJA PROJEKTA

5.1. Transmitter

Arduino UNO pločica, nRF24L01 modul i MQ135 senzor zajedno čine transmitter. Preko senzora se prikupljaju odabrani podaci iz okoline, zatim se preko radio odašiljača ti podaci šalju receiveru. Senzor i radio odašiljač se na Arduino spajaju prema ranije prikazanim shemama, a sklop u stvarnim uvjetima je prikazan na slici 5.1.



Slika 5.1. Transmitter

Programski kod za transmitter je podijeljen u dvije biblioteke, za čitanje senzora i za slanje/primanje podataka. Postoji više vrsta MQ senzora i svaki je osjetljiv na određene plinove. Da bi se radilo s njima potrebno je instalirati TroykaMQ.h biblioteku, unutar koje se nalaze različiti primjeri kodova. U ovom projektu odabrano je mjerenje razine CO2 u zraku pomoću MQ135 senzora, za kojeg se kod također može naći u instaliranoj biblioteci i prilično je jednostavan. Senzor očitava vrijednost CO2 u zraku u određenim vremenskim intervalima te se ta vrijednost ispisuje. Mjerna jedinica za kvalitetu zraka je ppm (*eng: parts per million*), upotrebljava se za izražavanje koncentracije u relativnim proporcijama i bezdimenzionalna je veličina. U zatvorenoj prostoriji

vrijednost mora biti između 350 ppm i 1000 ppm. Da bi se osigurala potrebna ušteda energije koriste se funkcije RF_powerDown i RF_powerUp te State Machine.

```
void RADIO::RF_powerUp() {
  radio.powerUp();
}

void RADIO::RF_powerDown() {
  radio.powerDown();
}
```

```
typedef enum
{
    READ_SERIAL,
    READ_SENSORS,
    RADIO_TX,
    SLEEP,
}
```

Typedef enumeracijom prvo su stvorena četiri stanja kojima je postignuto da je Arduino isključen dok senzor ne očita vrijednost CO2. State Machine:

```
void loop() {
  switch (state)
  {
  case READ_SERIAL:
    state = READ_SENSORS;
    break;
  case READ_SENSORS:
    dataToSend.air = sensor.readCO2();
    delay(100);
    state = RADIO_TX;
    break;
  case RADIO_TX:
    Serial.println(F("slanje"));
    Serial.println(dataToSend.air);
    delay(100);
    radioNRF.RF_send(dataToSend);
    state = SLEEP;
    break;
  case SLEEP:
    delay(50);
    LowPower.powerDown(SLEEP_8S, ADC_OFF, BOD_OFF);
    delay(50);
    state = READ_SERIAL;
    break;
  default:
    break;
  }
}
```

5.2. Receiver

Arduino i nRF24L01 radio odašiljač povezani kao na ranije prikazanoj shemi čine receiver koji prima podatke poslane od strane transmittera. Sklop u realnim uvjetima je prikazan na slici 5.2.

Slika 5.2. Receiver

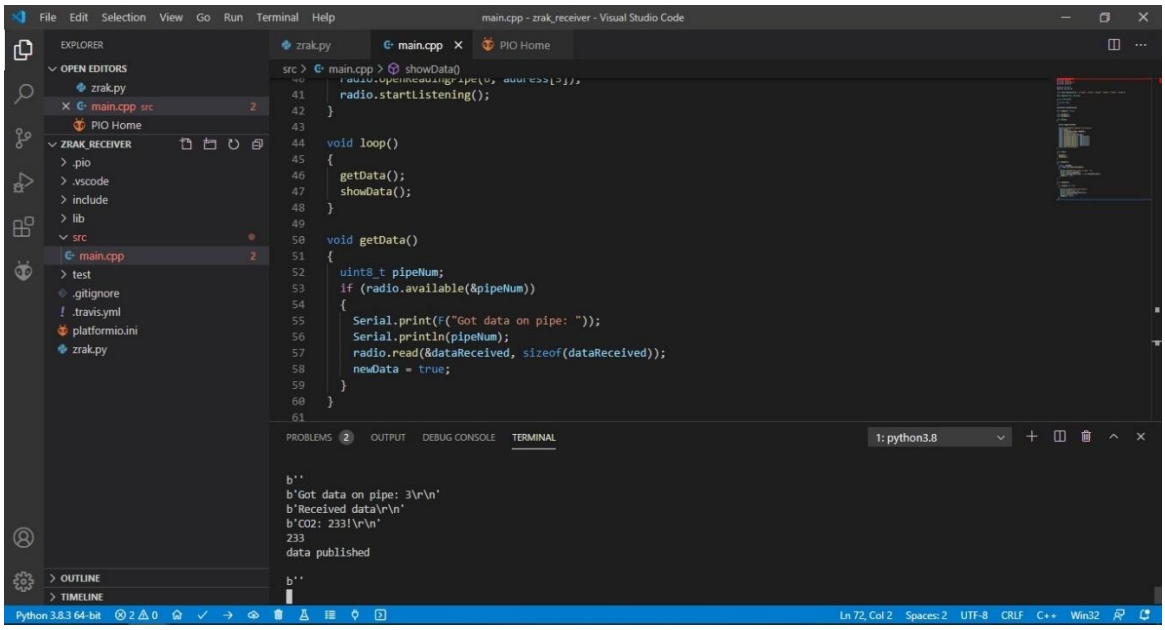
Programski kod za receiver se sastoji od .cpp file-a za čitanje primljenih podataka i od Python skripte putem koje se ti podaci šalju na Mqtt server te se na taj način uspostavlja serijska komunikacija. Unutar Python skripte se navode podaci o brokeru, klijentu, temi, qos-u te se piše petlja za slanje/ispis podataka na serveru. Dijelovi korištene Python skripte prikazani su u nastavku:

```
import serial
import paho.mqtt.client as mqtt
BROKER = "mqtt.eclipse.org"
CLIENTID = "MQTTExample"
TOPIC = "A507/sensors/mbatin/kvaliteta_zraka"
COMPORT = "COM4"
QOS = 1
```

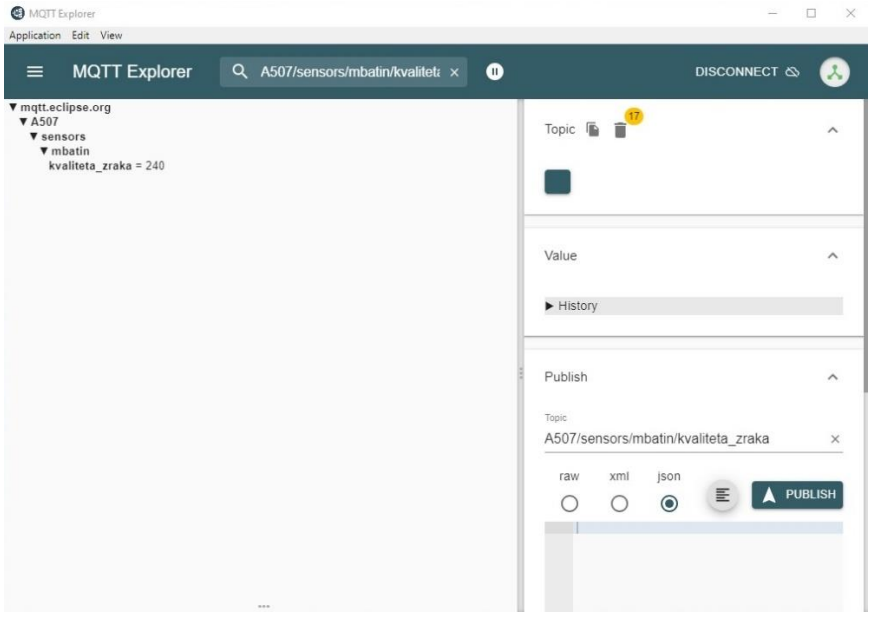
```
import time
flag_connected = 0

def on_connect(client, userdata, flags, rc):
    logging.debug("Connected result code "+str(rc))
    client.loop_stop()
def on_disconnect(client, userdata, rc):
    logging.debug("DisConnected result code "+str(rc))
    client.loop_stop()
def on_publish(client,userdata,result):
    print("data published \n")
    pass

mqttc = mqtt.Client(CLIENTID)
mqttc.on_connect = on_connect
mqttc.on_disconnect = on_disconnect
mqttc.on_publish = on_publish
mqttc.connect(BROKER)
mqttc.loop_start()

ser = serial.Serial(COMPORT, 115200, timeout=5)
while True:
    message = ser.readline()
    print (message)
    if b'CO2:' in message:
        string, air = message.split(b' ')
        air, rest = air.split(b'!')
        print(air.decode('utf-8'))
        mqttc.publish(TOPIC, payload=air.decode('utf-
8'), qos=QOS, retain=False)
    time.sleep(0.01)
mqttc.disconnect()
time.sleep(1)
```

5.3. Prikaz dobivenih rezultata

Prvi korak je pokretanje transmittera u Visual Studio Code-u nakon čega on počinje očitavanje i slanje vrijednosti CO2. Zatim se kod receivera pokrene Python skripta i u terminalu se mogu vidjeti vrijednosti CO2 u zraku, kao što je prikazano na slici 5.3. Dobivene vrijednosti se šalju i na MQTT server.

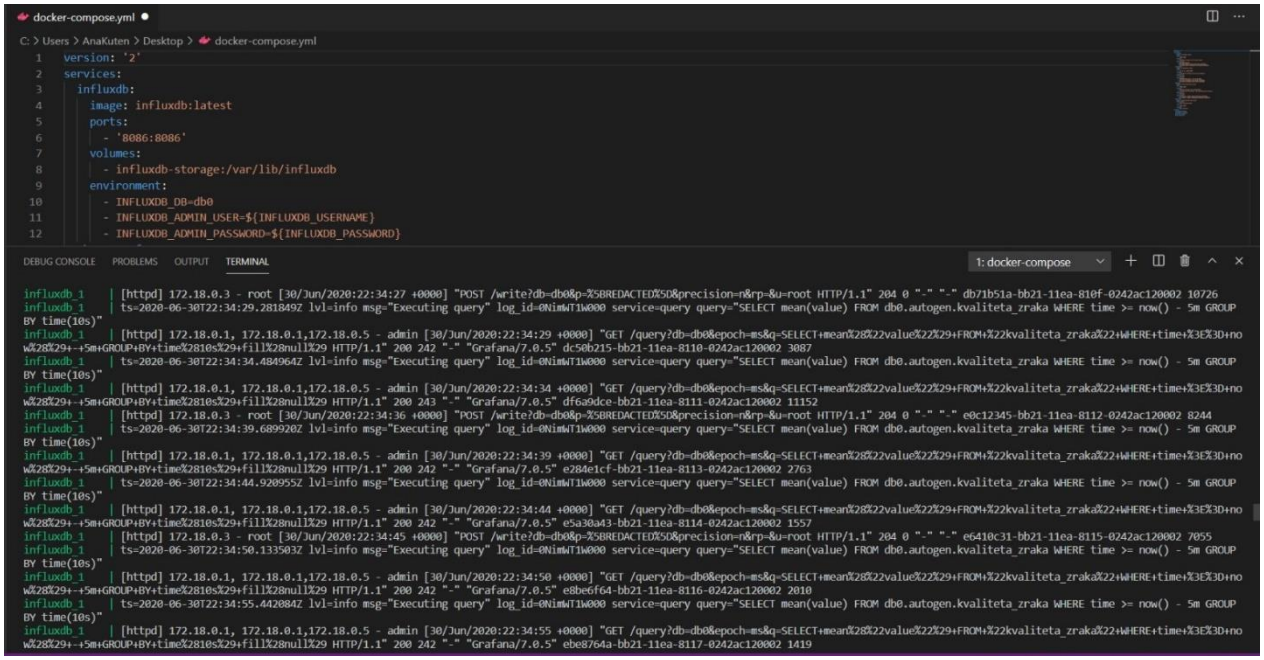


Slika 5.3. Primanje podataka

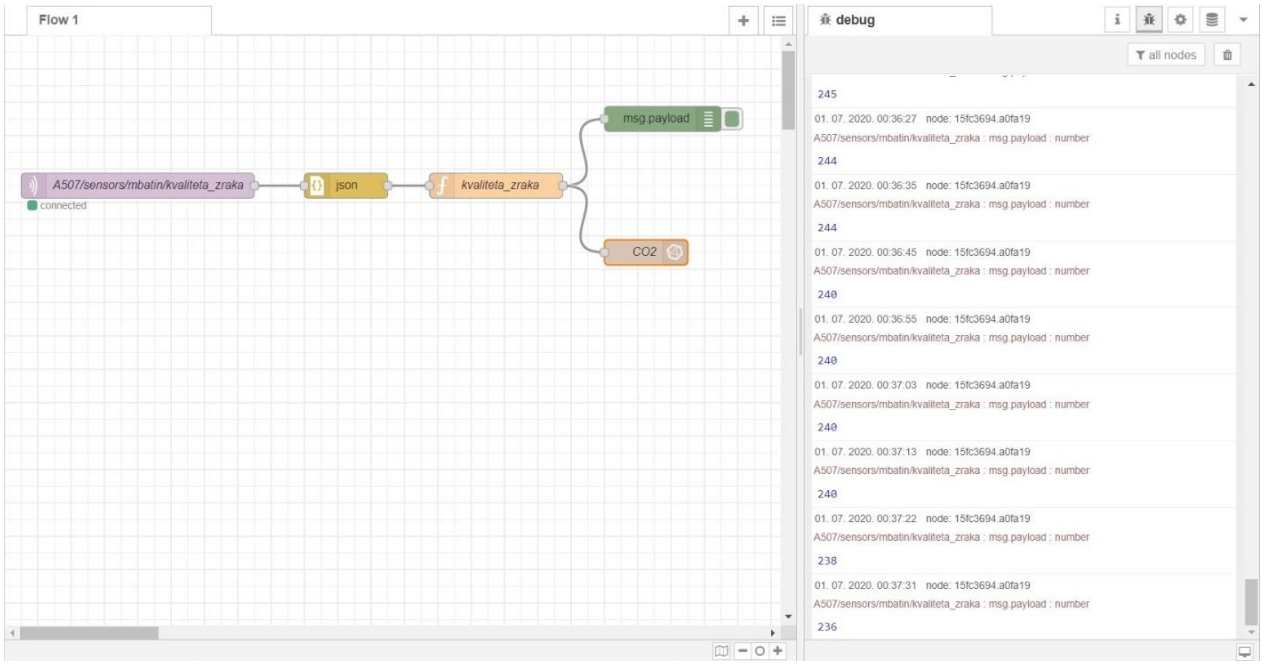


Slika 5.4. Prikaz podataka na MQTT-u

Sljedeće slike prikazuju način korištenja funkcionalnosti Docker kontejnera kako bi se pokrenule aplikacije za prikupljanje podataka, pohranu u bazu te vizualizaciju u obliku grafova.



Slika 5.5. Pokretanje docker-compose.yml datoteke



Slika 5.6. Node-RED flowchart

Na slici 5.6. može se uočiti da se Node-RED spaja na MQTT broker te se pretplaćuje na određenu temu. Podaci koji stižu sa brokera se zapisuju u InfluxDB bazu podataka db0.

Slika 5.7. Grafički prikaz dobivenih rezultata

Može se uočiti skok vrijednosti u jednom trenutku. Taj skok dobiven je nakon kratkog puhanja u senzor, a zatim vrijednost CO2 postupno pada. Graf je u ovom slučaju postavljen tako da ispisuje vrijednosti svako 5 sekundi.

Iz rezultata dobivenih u ovom projektu možemo zaključiti da senzor MQ135 ima dobar omjer cijene i kvalitete. Prije prvog korištenja potrebna je kratka kalibracija nakon koje je senzor spreman za upotrebu. Senzoru je također potrebno određeno vrijeme zagrijavanja tijekom kojeg se ispisuju krive vrijednosti (u ovom slučaju vrijednosti C02 su bile oko 5000), ali nakon što se zagrije prikazuje točne rezultate.

LITERATURA

[1] Čagalj, M.: „Bežične senzorske mreže: Uvod u senzorske mreže“, predavanje 1

[2] https://e-radionica.com/hr/blog/2015/10/08/sto-je-arduino-i-croduino/

[3]https://www.google.com/search?q=arduino+uno+tehnical+specifications&tbm=isch&ved=2ahUKEwjy0vbbiqLqAhVXwioKHfiNCfcQ2-cCegQIABAA&oq=arduino+uno+tehnical+specifications&gs_lcp=CgNpbWcQAzoCCAA6BAgAEB46BAgAEBM6BggAEB4QEzoGCAAQCBAeUM-qB1j18Adg8fIHaAFwAHgAgAH1AYgBuxuSAQYwLjI0LjGYAQCgAQGqAQtnd3Mtd2l6LWltZw&sclient=img&ei=SUf3XrKKENeEqwH4m6a4Dw&bih=597&biw=1242&rlz=1C1CHZL_hrHR710HR710

[4] https://en.wikipedia.org/wiki/Arduino_Uno

[5]https://www.google.com/imgres?imgurl=https%3A%2F%2Fi0.wp.com%2Fschemobotics.com%2Fwp-content%2Fuploads%2F2018%2F11%2Farduino-uno-r3-pic3.jpg%3Ffit%3D629%252C400&imgrefurl=http%3A%2F%2Fschemobotics.com%2Fproduct%2Farduino-uno-rev3%2F&docid=l02yGAZ4jLCunM&tbnid=FrEdlthzkHlCLM&vet=1&w=629&h=400&itg=1&bih=597&biw=1242&ved=2ahUKEwjWu76EkaLqAhVK-yoKHdEZAoIQxiAoAnoECAEQHQ&iact=c&ictx=1

[6] https://e-radionica.com/hr/mq135-senzor-plinova.html

[7] https://e-radionica.com/productdata/SNS-MQ135.pdf

[8] https://github.com/toperkov/WiSe-2019-20/blob/master/instructions/lab-6.md

[9]https://1sheeld.com/mqtt-protocol/?__cf_chl_jschl_tk__=7f6dc785942edfb6bf5c2d653a800babc36f8e20-1593479606-0-AdKQy0JVFLE2vTM6cbXlYRqfxEVaCgeGFdwMo5DAlKHW9cJrWpYW3QXxD9vFnrG8Co2enQWOnQw9lGu3pB7BaHH_pGmjRQZakgOn1lhvXnuu0Rgbsn_HovGGk8Vq6q2pZti5r5qL-SODfGE5Y1ze-ZlzWBGqXWCWbqZmbNaWFFgvuzDQr0x2ixosat3fa_NzP16W8MaMUUf0XRkmOf0-3HQ-84ROPmfjxstFy0-0_cFa9IHhVvq0tKWSYRtYY6NXeob54LEEwCLCLp9pH81PtdHgCTcLvmEUYGAQSwsHq8qg

[10] https://github.com/toperkov/WiSe-2019-20/blob/master/instructions/lab-8.md