Programe folosite

Android Studio

In zilele noastre cea mai rapidă cale catre infomație este internetul, iar cel mai comun și la îndemână dispozitiv cu acces la internet este telefonul. Prin acesta sunt oamenii sunt cel mai rapid notificați despre diferite articole, postari, sau chiar stiri si date care le-ar putea salva viața la un moment dat. Din cauza acestor factori am ales ca aplicația folosită pentru a monitoriza cantitatea de monoxid de carbon dintr-o incapere sa fie o aplicație de tip mobile. Prin urmare unul dintre programele folosite la crearea aplicatiei este Android Studio.

Android Studio este un IDE (Integrated Developement Kit) special folosit pentru aplicațiile mobile care rulează cu un sisteme de operare de tip Android. Acesta suport diferite limbaje de programare precum Java sau C++.

A screenshot of a computer

Description automatically generated

Android Studio foloseste ca și sistem de build tipul Gradle, in care se pot adauga diferite dependințe in funcție de nevoile programatorului.

Gradle este o unealta automata pentru crearea aplicatiilor, acesta castigandu-și popuaritatea prin sustinerea aplicatilor care sunt scrise in limbaje de programare precum Java sau C++. Totodata, Gradle permite construirea si testearea produselor software.

A screenshot of a computer

Description automatically generated

In imaginea de mai sus se pot observa dependințele pe care le-am adaugat pentru a afisa diferite grafice sau pentru a trimite mail-uri. Mai multe detalii se alfta in capitolul Dependinte.

O alta caracteristica pe care o Android Studio o prezinta este cea de emulare. Android Studio ne pune la dispozitie un intreg emulator pentru a emula diferite dispozitive care au caracteristici specifice nevoilor.

A screenshot of a computer

Description automatically generated with medium confidence

Caracteristici precum tipul dispozitivului, rezolutia acestuia, dimensiunea ecranului, densitatea pixelilor sau imagini de sistem. Imaginile de sistem reprezinta o implementare Android, care este open source, si care poate fi rulata pe mai multe sisteme de tip Android. Imaginea ajuta la testarea, rezolvarea problemelor de compatibilitate și simularea unui mediu in care se pot rula aplicatiile care sunt create in Android Studio.

A screenshot of a computer

Description automatically generated

Android Studio este cel mai potrivit IDE pentr cerințele date si pentru diversitatea pe care o are in dezvoltarea aplicațiilor.

Arduino IDE

Arduino IDE este un editor de text pentru Arduino. In Arduino IDE se pot scrie asa numitele sketches, sau schite. Aceste schițe, sunt verificate si compilate, iar daca tot este in regula acestea sunt incarcate pe diferite placuțe de dezvlotare.

A screenshot of a computer

Description automatically generated with medium confidence

Arduino IDE dispune si de un monitor serial in care se afiseaza toate datele transmise pe port la un anumit baud rate(numarul de schimbari ce le are un semnal intr-o secunda).

Arduino IDE foloseste ca limbaj de programare o varianta de C++, care are cateva metode si funcții adiționale. Datorita faptului ca am ales o placa de dezvoltare Arduino UNO, Arduino IDE, este indispensabil pentru implementarea functionalitaților necesare.

ThingSpeak

ThingSpeak, este un soft open source care permite conectarea dispozitivelor care au acces la internet. Acesta permite vizualizarea și analizarea datelor live primite de la diferiti senzori. ThingSpeak poate primi date de la orice dispozitiv conectat la internet prin folosirea unor REST APIs.(REST APIs sunt detaliate la capitolul REST API)

Un alt instrument foarte folositor care este inclus in ThingSpeak, este MATLAB. In ThingSpeak se poate scrie, genera sau rula cod MATLAB, care ajuta la vizualizarea si analiza datelor.

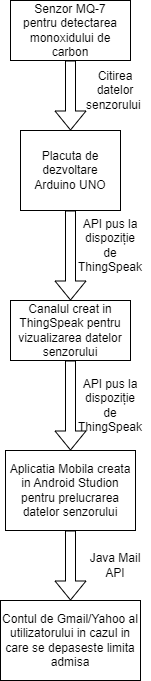
A picture containing text, screenshot, line, plot

Description automatically generated

In imaginea \*NUMARUL IMAGINII\* sunt afisate datele care au fost inregistrate de catre senzorul de monoxid de carbon. Acestea sunt transmise in timp real de la senzor si sunt salvate si expuse pe grafic.

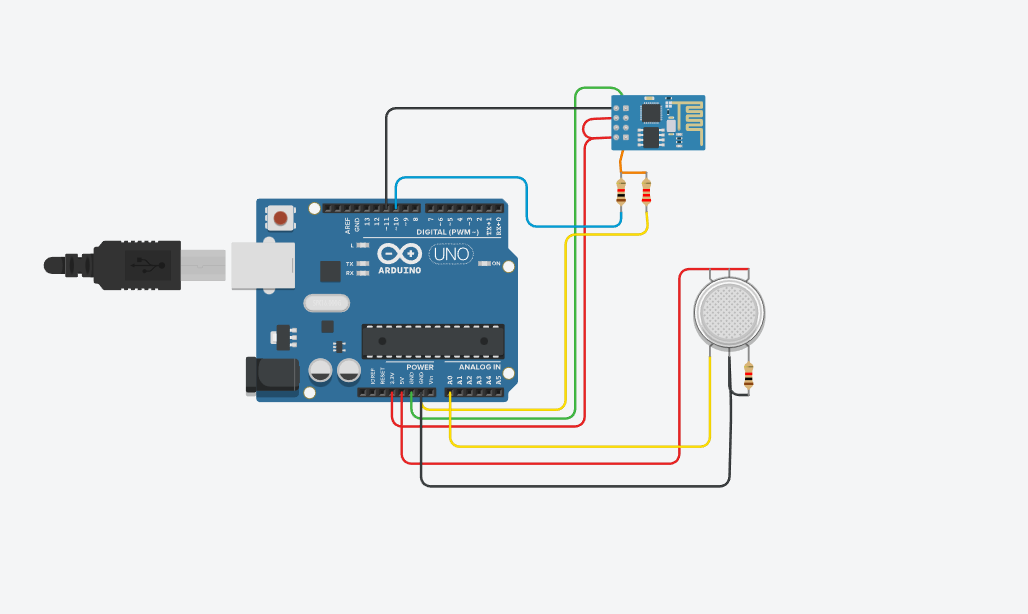
Diagrama aplicației

Pentru a aveam o vizualizare mai usoara si pentru a vedea cum sunt transmise datele de la senzorul de monoxid de carbon catre aplicatia creata in Android Studio, am creat o diagrama in care este expus modul si direcția in care sunt transmise datele. Funcționalitatea aplicației se bazeaza pe transmiterea valorilor inregistrate de catre senzorul de monoxid de carbon, catre aplicatia mobila.



Senzorul MQ-7 măsoara cantiteatea de monoxid de carbon din aerul încăperii, dupa care acesta o transmite pe portul analogic al placutei, dupa care placuta se conecteaza la internet printr-un modul ESPX8266 pentru a putea face un request catre API-ul pus la dispozitie de catre ThingSpeak. In ThingSpeak s-a creat un canal in prealabil si s-a generat o cheie care este necesara pentru folosirea API-ului. Dupa ce se configureaza modului ESPX8266 se face se trasmite valoarea senzorului catre canalul de pe ThingSpeak unde se adauga in grafic. Mai departe aplicatia mobila foloseste alt API pentru a prelua datele de pe canalul din ThingSpeak si pentru a le prelucra. Aplicatia verifica daca valoarea masurata de catre senzor este mai mare decat o limita setata la lansarea aplicatiei sau, o limita setata de catre utilizator direct din interfața. Daca valoarea masurata de senzor este mai mare decat limita impusa de catre utilizator, se va trimite un email catre adresa de mail care este setata in fereastra de log-in a aplicatiei. Procesul se repeta la ficare 3 minute, sau la un intreval de timp stabilit de catre utilizator. Totodata utilizatorul are la indemana un istoric al mail-urilor, pentru a verifica si un grafic cu datele in timp real colectate de la senzor.

Diagrama Arduino



In imaginea “Numarul IMAGINII” este realizat montajul senzorului de detectare al monoxidului de carbon si modulul care permite conectarea placutei la internet.

Senzor monxid de Carbon MQ-7

Senzorul de monoxid de carbon MQ-7 este un senzor care are poate fi folosit la o multitudine de aplicații, de la aplicații care consta in monitorizarea locuințelor sau spațiilor in care pot fi degajate cantitati de monoxid de carbon pana la aplicații mai vaste de uz industrial. Este un senzor care nu are costuri mari de producție si care este usor de folosit.



Senzorul functioneaza pe baza unui filament de dioxid de staniu(SnO2). Dioxidul de staniu are o conductivitate mai mica cand in aer nu se afla alte gaze. Conductivitatea creste pe masura ce cantitatea de monoxid de carbon din aer creste, astfel putand fi masurată. Senzorul poate masura cantitați intre 10ppm(parti per million) pana la 10000 ppm. Senzorul functioneaza cel mai bine dup ace s-a incalzit, deoarece gazele care se absorb la temperaturi mai mici de functionare se disipa.

Interfața:

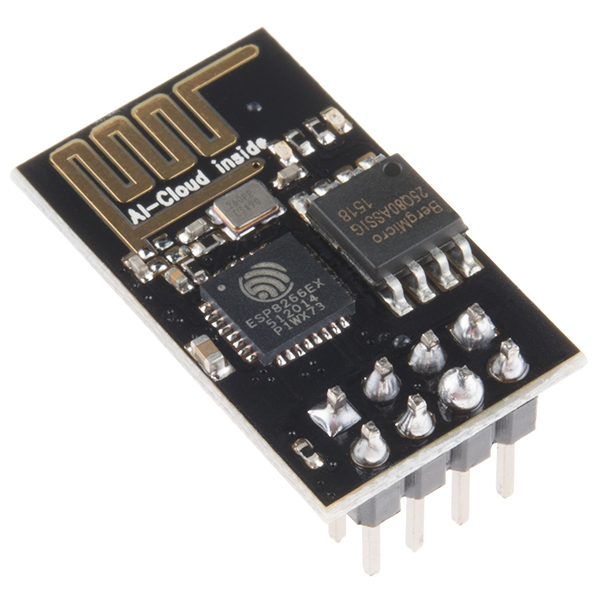
Senzorul dispune de 4 pini, dupa cum se poate vedea si in tabelul “NUMAR TABEL”:

|  |  |  |
| --- | --- | --- |
| **Numar Pin** | **Simbol** | **Descriere** |
| 1 | A0 | Iesire Analogica |
| 2 | D0 | Iesire Digitala |
| 3 | GND | Ground(0V) |
| 4 | VCC | Sursa 2.5-5V |

Senzorul dispune de multe avantaje precum sesitivtate ridicata, deoarece poate masura schimbari minore ale cantitații de monoxid de carbon din aer, are un interval de masurare foarte mare, ideal si pentru aplicații de uz industrial, are o durata de viata ridicată, si nu este costisitor din punct de vedere financiar.

Modulul ESP8266EX

Modulul ESP8266EX este o componenta cheie in aplicatia de monitorizare a monoxidului de carbon, fara ea nu s-ar putea face trasmiterea de date intre partea hardware si cea software. Modulul este un modul Wi-fi care permite conectarea la internet, acesta poate fi folosit singur, sau poate fi folosit ca “slave” de catre un microcontroller.In cazul de față, modulul ESP8266EX este folosit ca “slave” de catre microcontroller.



Pentru transmiterea datelor si conectarea la server sau la client, modulul foloseste protocolul TCP/UDP.

Interfata:

|  |  |  |
| --- | --- | --- |
| **Numar Pin** | **Simbol** | **Descriere** |
| 1 | VCC | Sursa : 3.3V |
| 2 | GND | Ground : 0V |
| 3 | Rx | Receptor- folosit la receptia datelor de la un alt dispozitiv |
| 4 | Tx | Transmițator- folosit la transmiterea datelor catre alte dispositive |
| 5 | Ch\_En | “Chip enable”- activare modul, de obicei conectat la 3.3V |
| 6 | GPIO | Pin Generic – poate fi programat |
| 7 | GPIO 2 | Pin Generic 2 |
| 8 | RST | Reset – folosit la resetare, de obicei conectat la GND(0V) |

Modulul ESP8266EX este o alegere foarte buna deoarece este compact si este foarte accesibil din punt de vedere financiar. Acesta poate folsi chiar si end-point-uri pentru dezvoltarea aplicaților de tip IoT(Internet of Things).

Pentru a comunica cu modulul, microcontroleru are nevoie de o serie de comenzi de tip AT. Daca totul a fost configurat, modulul o sa comunice cu microcontrolerul folosind comunicația seriala sau UART la un baud rate specific.

Comenzi AT suportate de catre modulul ESP8266EX

|  |  |  |
| --- | --- | --- |
| Functie | Comanda AT | Raspuns |
| Functionare | AT | OK |
| Restart | AT + RST | OK  ...  Ready |
| Versiune de Firmware | AT + GMR | <Verisiunea AT> informatii despre versiunea AT  <Versiunea SDK> informatii despre versiunea de SDK  <timp de compilare> timpul in care bin-ul a fost compilat  OK |
| Lista cu punctele de acces | AT + CWLAP | +CWLAP:<ecn>,<ssid>,<rssi>,  <mac>,<ch><freq offset>  OK |
| Detealii despre punctele de acces la care se poate conecta modulul | AT + CWJAP ? | + CWJAP: <ssid>,<bssid>,<channel>,<rssi>  OK |
| Conectarea modulului la un punct de acces | AT + CWJAP =”SSID”, “Password” | WIFI Connected  WIFI GOT IP  OK |
| Deconectarea de modului de la un punct de acces | AT + CWQAP | OK  WIFI Disconnected |
| Afisarea adresei IP | AT + CIFSR  (Prima data trebuie setat  AT + CWMODE = 3) | +CIFSR:APIP, <adresa IP>  +CIFSR:APMAC, <adresa mac>  +CIFSR:STAIP, <adresa IP>  +CIFSR:STAMAC, <adresa mac>  OK |
| Afisarea modului Wi-fi | AT + CWMODE ? | +CWMODE:<mod> |
| Setearea modului Wi-fi | AT + CWMODE = <mode>  1 = STA (station)  2 = AP (Access Pont)  3 = Ambele (STA & AP) | OK |
| Detalii despre UDP | AT + CIPMUX ? | +CIPMUX:<mod> |
| Setare UDP | AT + CIPMUX = <mode>  0 = Single Connection  1 = Multiple Connection | OK |
| Detalii despre TCP/IP | AT + CIPSTATUS | STATUS:<status>  Statusuri posibile:  2 : Got IP  3:Connected  4:Disconnected |
| Verificarea modului de transmitere TCP | AT + CIPMODE ? | +CIPMODE:<mod> |
| Setarea modului de trasmitere TCP | AT + CIPMODE = <mode>  0 = Normal mode  1 = Transparent mode | OK |
| Setarea conexiunii UDP | (CIPMUX = 0)  AT + CIPSTART = <type> , <address> , <port>  (CIPMUX = 1)  AT + CIPSTART = <id>, <type>, <address>, <port> | CONNECT  OK |
| Transmitere de date | (CIPMUX = 0)  AT + CIPSEND = <data length>  (CIPMUX = 1)  AT + CIPSEND = <id>, <data length> | OK  Recv<data length> bytes  SEND OK |
| Inchidere conexiune | AT + CIPCLOSE | CLOSED  OK |

Transmission Control Protocol(TCP)

Este o metode de comunicare intre doua sau mai multe dispozitive care pot accesa o rețea pe internet. TCP este unul dintre metodele de comunicare standard pe internet, acesta este conceput sa trimite pachete intr-o retea asigurandu-se totodată ca informația a fost trimisă și primita cu succes prin rețea. Este cel mai folosit protocol de transmitere de date in cadrul comunicaților in rețea deoarece verifica transmiterea informației de la un capat la altul. TCP organizeaza si structureaza datele pentru ca ele sa poată fi transmise intre server si client, in plus, garantează si pentru integritatea datelor care sunt trimise prin rețea. Înainte ca datele sa fie trimise, TCP stabilește o conexiune intre surse si asigură stabilitatea si integritatea conexiunii pana la incepere comunicației, după care, desparte cantitatea mare de informație in pachete mai mici si se asigură ca nimic sa nu fie pierdut în timpul procesului. Ca rezultat al faptului ca integritatea datelor este garantată de TCP, toate protocoalele de nivel înalt care au nevoie de transmitere de date îl folosesc, cum ar fii rețelele peer-to-peer(egal-la-egal), File Transfer Protocol (FTP), Secure Shell(SSH), si Telnet.

User Datagram Protocol(UDP)

UDP este un protocol de trasmitere de date pe internet, folosit în special la trasmisiile la care nu conteaza integritatea datelor trimise, ci viteza cu care ele au fost trimise prin rețea. Viteza crescuta rezultă din faptul ca UDP nu stabileste o conexiune înainte ca datele sa fie trimise, el permite ca datele sa fie trimise la o viteza foarte mare dar, in urma neglijării pachetelor trimise, unele ar putea fi pierdute in timpul transmiterii, creând oportunitați pentru atacuri de tip DDoS(Distributed Denial of Service). UPD realizează transferul de date intr-o maniera mai simplă, trimite pachetele direct la un dispozitiv din rețea fara a mai stabili o conexiune in prealabil, indicand doar ordinea in care s-au trimis pachetel, fara a mai verifica dacă ele au ajuns cu succes sau nu.

Internet Protocol(IP)

Reprezintă o metoda de transmitere de date pe internet de la un dispozitiv la altul. Fiecare dispozitiv are o adresa unica, numita adresa IP, si care permite comunicarea si schimbul de date intre dispozitivele conectate la internet. IP este considerat un standard pentru transmiterea directă de date intre dispozitivele mobile. El este responsabil cu definirea modului, formatul si regulile pentru schimburile de date intre retele locale sau retele conectate la internet.

Internet Protocol Suite(TCP/IP)

TCP/IP este un set de protocoale de comunicații care permite o trasmitere a datelor foarte exactă și corectă din punct de vedere al integritații datelor.El desparte datele in pachete pentru a evita retrimiterea integrală a datelor în cazul în care trimiterea esuează. Fiecare pachet poate lua o rută diferita intre cele doua dispozitive, dacă ruta pe care a fost trimis pachetul anterior este indisponibilă.La final, după ce toate pachetele au fost trimise cu succes, ele se reasamblează automat formând mesajul integral. TCP/IP simplifica comunicarea, aceasta fiind împarțită in patru straturi fara a avea implicari hardware sau software. Pachetele trebuie sa treaca prin toate cele patru straturi ale protocolului înainte de a ajunge la destinația finală, dupa ce pachetele au ajuns, TCP/IP parcurge invers cele patru straturi, reasalmblând mesajul final. TPC/IP, fiind un protocol care se bazează pe conexiunea intre dispozitive, acesta folosește “three-way handshake” pentru a stabili conexiunea. Serverul primeste un pachet pentru sincronizare, prin care știe ca urmeaza sa se conecteze un client, in al doilea pas, serverul trimite un raspuns clientului, prin care il notifica ca a primit pachetul, iar in final, clientul trimite inca un raspuns serverului pentru și pentru a stabili conexiunea.

Cele 4 straturi care definesc protocolul TCP/IP

1.Interfața de rețea.

Interfața de rețea defineste modul in care datele ar trebui trimise, se ocupa de actul fizic de recepție si transmitere de date si este responsabilă cu transmiterea datelor intre aplicatii sau dsipozitive in rețea. Acestea includ modul in care datele ar trebui semnalate de componetele hardware si de celelalte dispozitive din rețea.

2. Internet

Stratul Internet este responsabil cu trimiterea pachetelor și cu controlul miscării lor de-a lungul rețelei, pentru a se asigura ca pachetele ajung la destinație.

3.Transport

Stratul de transport implică producerea unei conexiuni solide si stabile intre aplicatia sau dispozitivul care trimite date si destinația dorită. Acesta este nivelul unde data este desparțită in pachete si este numerotată pentru a se crea o secvența. Totodată stratul de transport, determină cantitatea de date care se trimite, unde trebuie trimisă, si rata la care se trimit datele.

4.Aplicație

Stratul aplicație se refera la faptul ca dispozitivele au nevoie de TCP/IP pentru a le ajuta la comunicare. Este stratul unde utilizatori interacționează, cum ar fi sisteme de e-mail sau platforme de comunicare prin mesaje.

Ghid UI



Interfața de logare: Utilizatorul trebuie sa se conecteze cu un cont, email-ul care este setat la contul de utilizator este email-ul la care se v-or trimite alertele cand se va detecta o crestere semnificativa de monixid de carbon.

A screen shot of a login form

Description automatically generated with low confidence

Interfata de înregistrare, poza “nr pozei”.In aceasta interfața, utilizatorul este nevoie sa își creeze cont în cazul in care nu are deja un cont creat.

A screenshot of a device

Description automatically generated with low confidence

Interfața principala: In aceasta interfață sunt afișate datele inregistrate de catre senzor, câmpul pentru setarea threshold-ului, oprirea notificarilor pe mail si setarea timpului de evaluare al datelor citite de la senzor.

A screenshot of a phone

Description automatically generated with low confidence

Interfata de setari: Sunt afișate toate setarie, se pot trimite email-uri pentru a verifica daca este funcțională trimiterea, se poate seta alta adresa la care sa se trimita email-uri, totodata se poate afișa un istoric pentru email-urile ce au fost trimise, si se poate contacta developer-ul.

A picture containing text, screenshot, software, multimedia software

Description automatically generated

Interfața pentru setarea adresei de email: Aici se pot configura adresa de email, subiectul mailului, cat si mesajul care sa fie trimis in caz ca exista cresteri de monoxid de carbon.

A picture containing text, screenshot

Description automatically generated

Interfata istoricului. In interfata aceasta regasim un mic istoric cu toate email-urile ce au fost trimise, data la care s-au trimis si threshold-ul care a fost atins.

A picture containing text, screenshot, multimedia software, software

Description automatically generated

Interfata de contact. Aici se pot trimite email-uri catre email-ul de contact al developerului, in cazul in care se semnalează comportamente neprevazute ale aplicației.

A picture containing text, screenshot, multimedia software

Description automatically generated

Interfata cu graficul valorilor masurate. Reprezentarea grafica in timp real ale valorilor masurate de catre senzorul de monoxid de carbon



Diagrama UI pentru Aplicația Mobilă

Legatura dintre Java si Android Studio

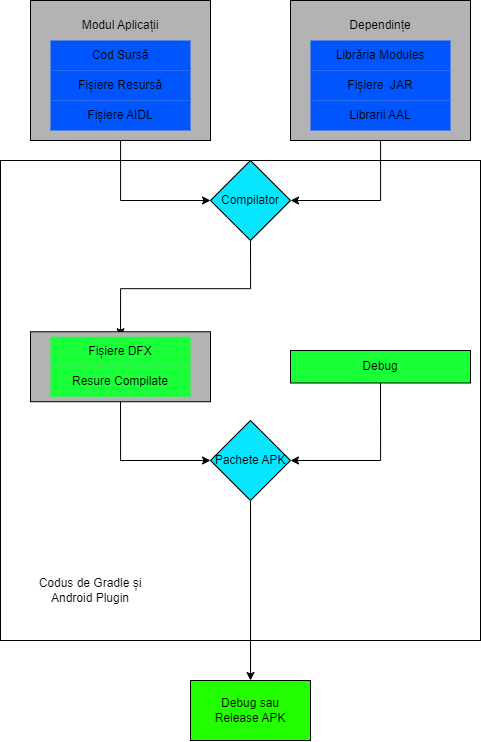
Dupa cum am precizat in capitolul “Programe folosite” (referinta catre capitol), Android Studio este un IDE care suporta un numar foarte mare din librariile care fac parte din Java 5.0, de aceea putem spune ca Android Studio este bazat pe Java. Majoritatea librariilor care nu sunt suportate de catre Android Studio din Java, fie sunt înlocuite cu alte librari fie nu sunt necesare. Una dintre diferențele majore care apare in Android studio este faptul ca metoda main nu exista, aceasta fiind inlocuita cu metodele onCreate, onPause sau onResume, metode care trebuie suprascrise de catre dezvoltatori.

Pentru a intelege mai bine legatura dintre Java si Android Studio avem urmatoarea diagrama “vezi nr diagrama”. Putem observa o vedere in ansamblu a sistemului de Build pe care il folosește Android Studio.

Javac and dx

Acesta a fost toolchain-ul lansat drept parte a Android SDK, codul java este luat și convertit in Java bytecode, prin compilatorul Java, javac, care incrulde sursele generate de java annotation processor, (ex. Lombok). Dupa care, in final codul este tradus in dex bytecode de catre dx.

Datorita acestor lucruri, putem spune ca Android Studio este bazat in mare masura pe Java si pentru a construii aplicații. Divergența apare in toolchain cand codul este tradus in cod specific Andriod Studio. Dar oricum ar fii, rezultatul final este mai mult sau mai puțin la fel cu cel din Java.



Ce este Java?

Java este un limbaj de programare orientat pe obiecte, totodata este un software care ruleaza pe miliarde de dispozitive din intreaga lume, incluzand calculatoare, telefoane mobile, console de gaming si dispozitive medicale. Sintaxele din java sunt bazate pe sintaxele limbajelor de programare C si C++. Unul dintre cele mai mari avantaje pe care aplicațile scrise in Java este protabilitatea, deoarece daca se scrie cod pentru un software-ul unui calculator, codul este foarte ușos de adaptat pentru trecerea la un telefon mobil. Pentru a scrie programe in Java, avem nevoie de Java Developement Kit. La fel ca si in Android Studio, programele scrise in java sunt compilate si transformate in Java bytecode. Totodata Java este un limbaj orientat pe obiecte, in alte cuvinte, java folosește obiecte ca prima sursa de implementare. Limbajele orientate pe obiect au ca obiectiv implementarea de entitați din viața de zi cu zi, precum moștenirea si polimorfismul.

Conceptele programarii orientate pe obiect:

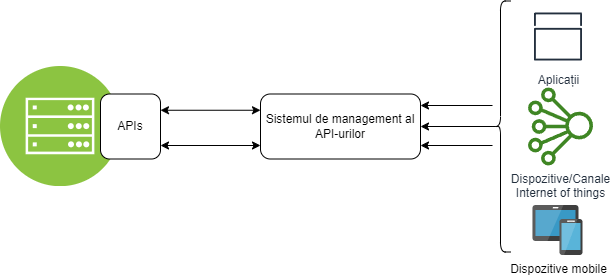
1.Clasele: sunt un set de obiecte care au atribute caracteristici si proprietați comune.

2.Obiectele: sunt o entitate de baza a limbajelor OOP, si reprezinta entitați din viata reala.

3.Metodele: sunt o colecție de pasi, care execută și returnează un rezultat apelantului.

REST API’s

Pentru a afla ce este un REST API, prima data trebuie sa înțelegem conceptul de API. Un API sau application programming interface este un set de definiții si protocoale pentru construirea si integrarea in aplicații software. Api-urile mai pot fi vazute ca și o ințelegere intre un furnizor de informație si un consumator de informație. Daca se dorește comunicarea cu un sistem pentru a primi informații sau pentru a apela o funcție, un API este cel care ajută și face posibilă comunicarea.



REST, sau representational state transfer, este un tip de arhitectura software care a fost creată pentru a face sigura operabilitatea intre mai multe sisteme. Servicile care sunt construite conform arhitecturii REST pot comunica intre ele mult mai usor.

Cum am precizat, REST este un tip de arhitectura pentru programele software, iar REST API face referire la un API care face un apel de tip HTTP pentru a accesa si a folosi datele. Datele pot fi folosite pentru mai multe metode precum GET, PUT, POST si DELETE, fiecare are un rol diferit, GET citește datele, PUT face update, POST creaza date, iar DELETE, dupa cum spune si numele, le șterge. REST API-urile sunt folosite foarte mult in partea de cloud, deoarece ajuta utilizatorii sa se conecteze si sa acceseze servicii de tip cloud, oferindu-le flexibilitate foarte mare.

Cum functioneaza REST API-urile:

REST API-urile folosesc metodele HTTP pentru a obtine resurele necesare, aceste metode sunt definite de catre protocolul RFC 2616, metode precum:

* GET
* PUT
* POST
* DELETE

Fiecara metodă are o sarcină speciala,totodata aceste metode sunt folosite si la identificare REST API-urilor, cand endpoint-urile sunt la fel.

* Metoda GET: folosita pentru a recupera o resursă
* Metoda PUT: folosite pentru a modifica o resursă
* Metoda POST: folosita pentru a crea o resură
* Metoda DELETE: folosita pentru a șterge o resursă

Cand se apeleaza un REST API, apelul poate sa conțina un corp care poate fi in mai multe formate de text, cele mai cunoscute si folosite ar fi:

* Application/json
* Application/xml
* Application/x-wbe-xml

REST API-uri dezvoltate de ThingSpeak

API care permite scriere de date pe un canal :

<https://api.thingspeak.com/update>

Acesta v-a trebui sa fie insoțit neaparat de api\_key, care este necesară pentru a indica canalul pe care se va scrie informația, Alte campuri care sunt opționale ar mai fii:

* field<X>, unde x este id-ul campului in care vrem sa se scrie informatia
* lat, latitudinea in grade, cu o valoare specifica intre -90 si 90
* long, logitudinea in grade cu o valoare specifica intre -180 si 180
* status, se face actualizare la mesaje.

API care permite sa se citească date de pe un canal :

[https://api.thingspeak.com/channles/<channel\_id>/feeds.<format](https://api.thingspeak.com/channles/%3cchannel_id%3e/feeds.%3cformat)>

La fel ca si API-ul va trebui sa fie însoțit neaparat de api\_key, care din nou este necesară pentru a indica chanalul de pe care se va citi informația.Campuri care nu sunt necesare:

* result : numarul de entitați care sunt extrase
* min : valoarea minima inclusa in răspuns
* max : valoarea maxima inclusa in răspuns
* location: longitudinea si latitudinea daca campul „location” este setat pe adevarat
* status : include statusul in raspuns si seteaza campul „status” pe adevarat

Implementare:

Pentru a citi datele de la senzor am scris un program in Arduino IDE care preia datele de la senzorul de monixid de carbon MQ-7, acesta le citește pe portul analogic al plăcii de dezvoltare, dupa care le trimite catre ThingSpeak.

Librăria SoftwareSerial.h

Librăria pe care am importat-o, SoftwareSerial.h, face comunicarea dintre modului ESP8266 si placa de dezvoltare Arduino posibilă. Metodele din libraria importata sunt :

* + SoftwareSerial(rx, tx): metoda care ne permite sa creăm un obiect de tipul SoftwareSerial
  + begin(viteza(baud\_rate)): setarea vitezei de transfer
  + pritnln(“mesaj”): scriere mesajului, comanda aceasta permite configurarea modulului ESP

A screenshot of a computer program

Description automatically generated with medium confidence

Am început prin importarea librăriei SoftwareSerial.h, aceasta ne permite sa configuram modulul ESP8266. Dupa care am creat un obiect de tip SoftwareSerial, la care am setat RX si TX pe porturile 10, respectiv 11. Acest obiect o sa ne ajute mai târziu cu configurarea modulului, si cu trimiterea de date catre ThingSpeak.

Setup-ul

In metoda de setup, am setat modul pinului la care este conectat senzorul de monoxid de carbon ca fiind un pin de input, deoarece pe pinul acesta se va face citirea datelor senzorului.Dupa care am setat ca transmiterea seriala sa se inceapa cu un baud rate, adica cați biti sa se transmita pe secundă, de 9600, iar pentru modulul ESP8266 am setat baud rate-ul ca fiind 115200.

Dupa ce am facut setarile de baza, am început configurarea modulului ESP8266. Am început prin a seta modul Wifi-ului, cu comanda AT+CWMODE = 1, aceasta comandă setează modul Wifi-ului pe Station, modul acesta ne permite sa conenctăm modulul ESP8266 la un punct de acces Wifi. Comanda AT + CWJAP conectează modulul la rețeaua care este menționata, primul parametru fiind SSID-ul rețelei la care vrem sa ne conectăm iar cel de al doilea parametru este parola cu care ne conectăm la rețea. Dupa fiecare comanda am setat un delay, pentru a aștepta răspunsurile in linia de comandă.

A screenshot of a computer program

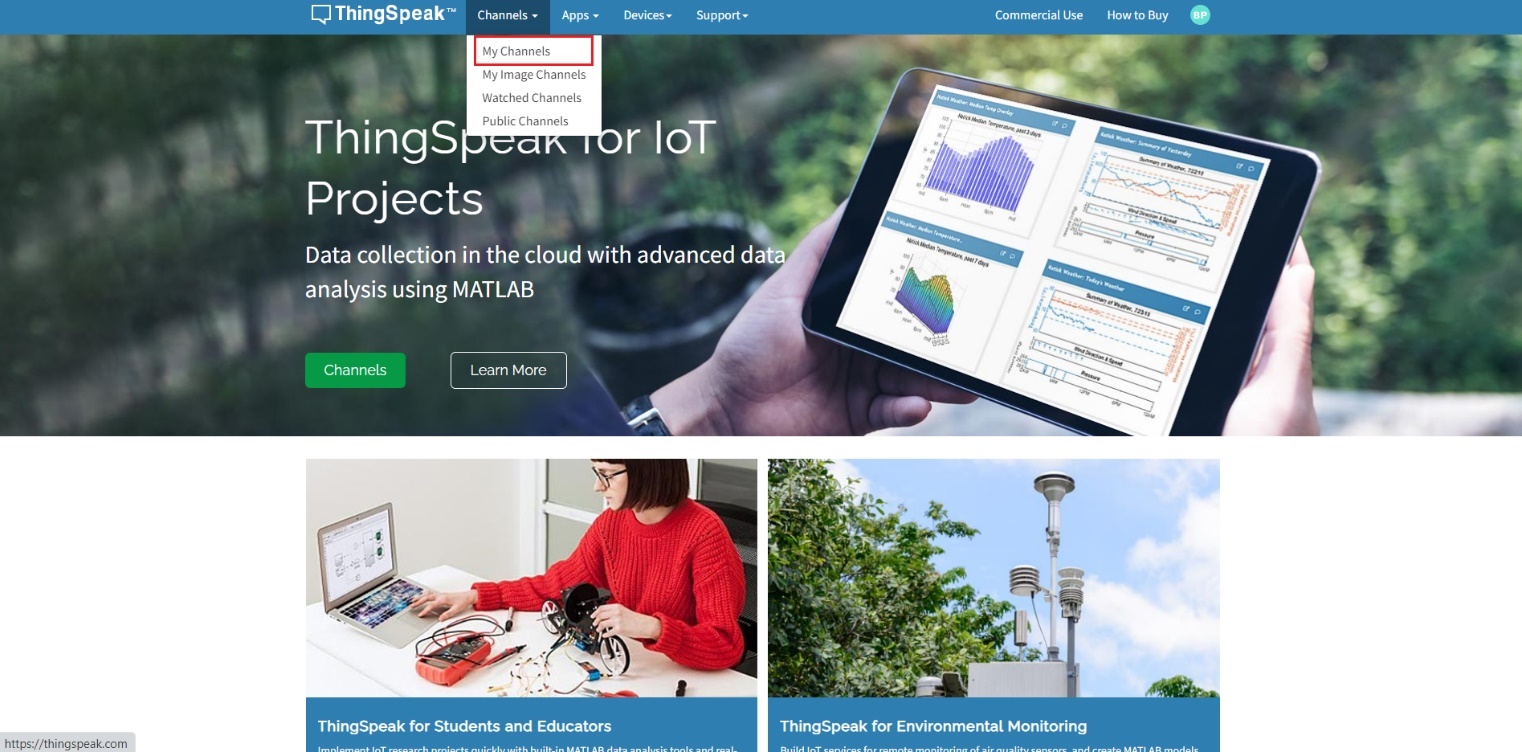
Description automatically generated with low confidence

Metode loop

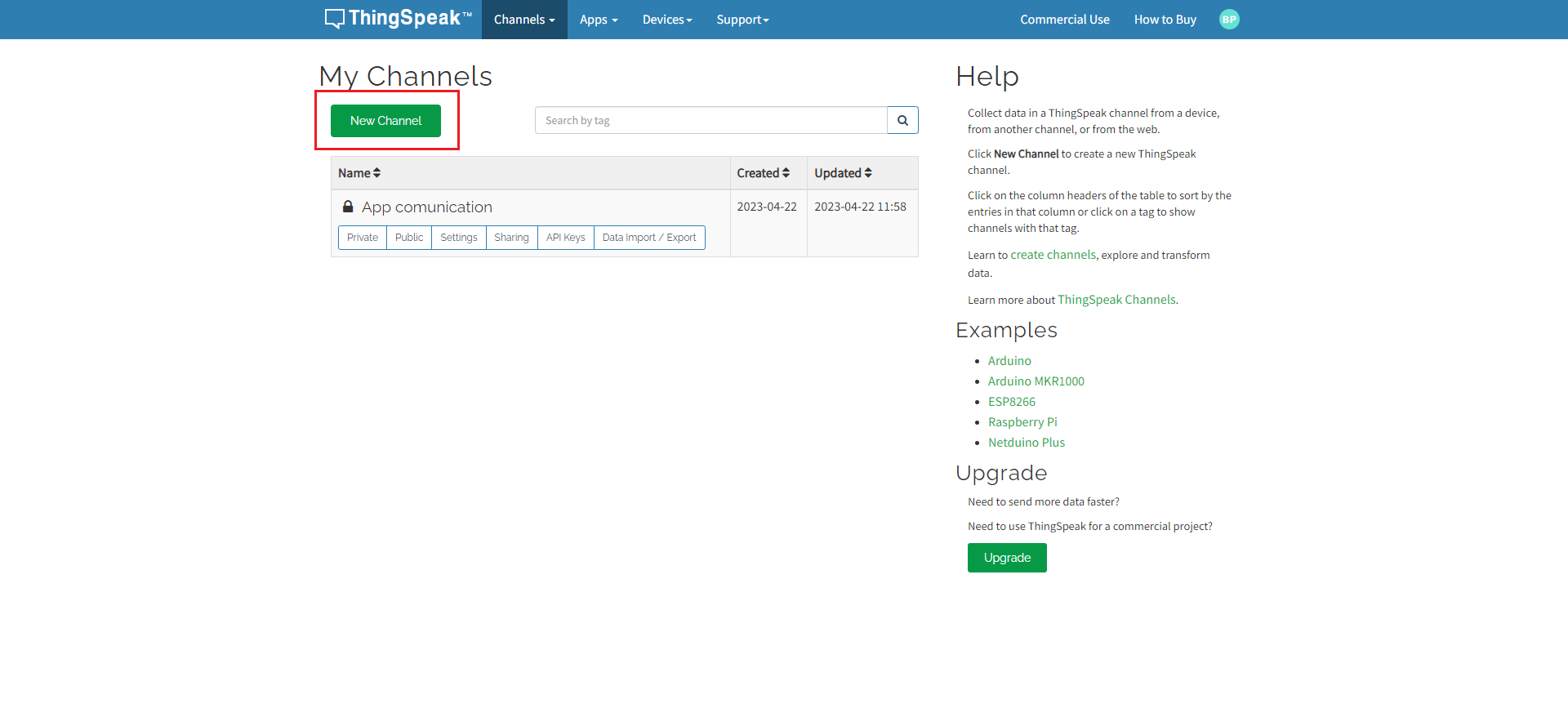
In metoda loop am preluat datele de la senzorul MQ-7.Cu metoda analogRead, care ia ca parametru numarul pinului la care este conectat senzorul, aceasta metoda citeste valorile specifice pinului analogic. Dupa care se seteaza AT+CIPMUX pe 0, astfel se notifica modulul ESP2866 ca va urma sa realizam o conexiune de tip “Single Connection”. Cu urmatoarea commandă, AT+CIPSTART, se realizeaza o conexiune de tip TCP, la adresa “api.thingspeak.com”, pe portul 80. AT+CIPSEND ne indica lungimea sirului de date pe care vrem sa-l trimitem, in cazul nostru, lungimea sirului de date trimis este 51. Mai apoi, se face o cerere cu metoda HTTP, GET, catre API-ul de update dezvoltat de catre ThingSpeak, in care este precizata cheia canalului une urmează sa fie scrise datele senzorului cat si valoare citita de le senzorul de monoxid de carbon. Ultima comandă, este cea care inchide conexiunea intre modulu Wifi ESP2866, si canalul de ThingSpeak. Delay-ul setat dupa fiecare comandă este o măsura de precauție, si este necesară deoarece unele comenzi au durează o perioada de timp până cand se îndeplinesc, iar acel delay, reprezintă siguranța ca toate comenzile au fost primite si realizate cu succes.

Creare Canal ThingSpeak

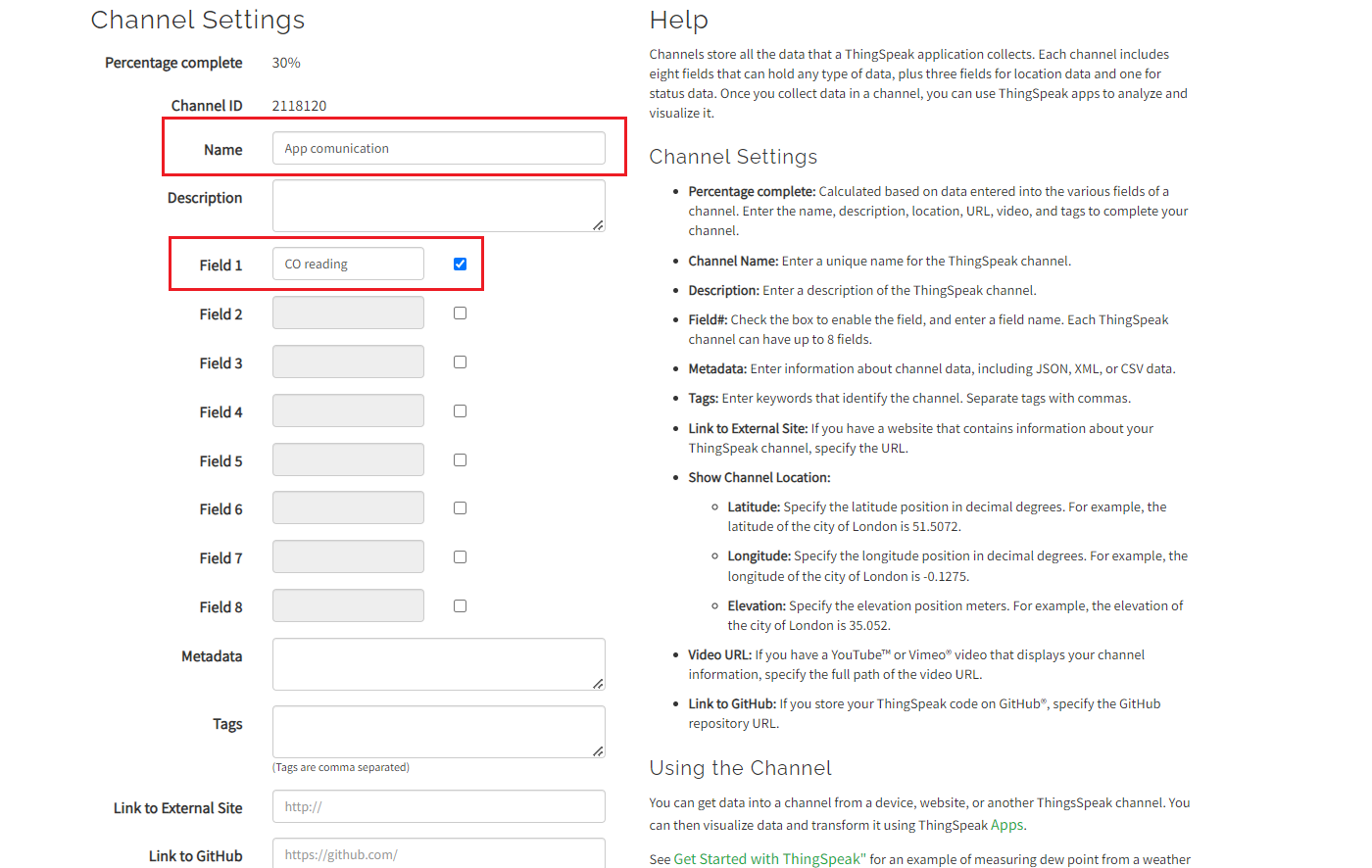
Pentru realizarea aplicației, este necesar un cont de utilizator pe platforma ThingSpeak, mai multe detalii despre platformă pot fii regasite la „REFERINTA CATRE DESCRIERE THINGSPEAK”, dupa ce utilizatorul este conectat, trebuie sa se creeze un canal pentru a primi si prelua date.



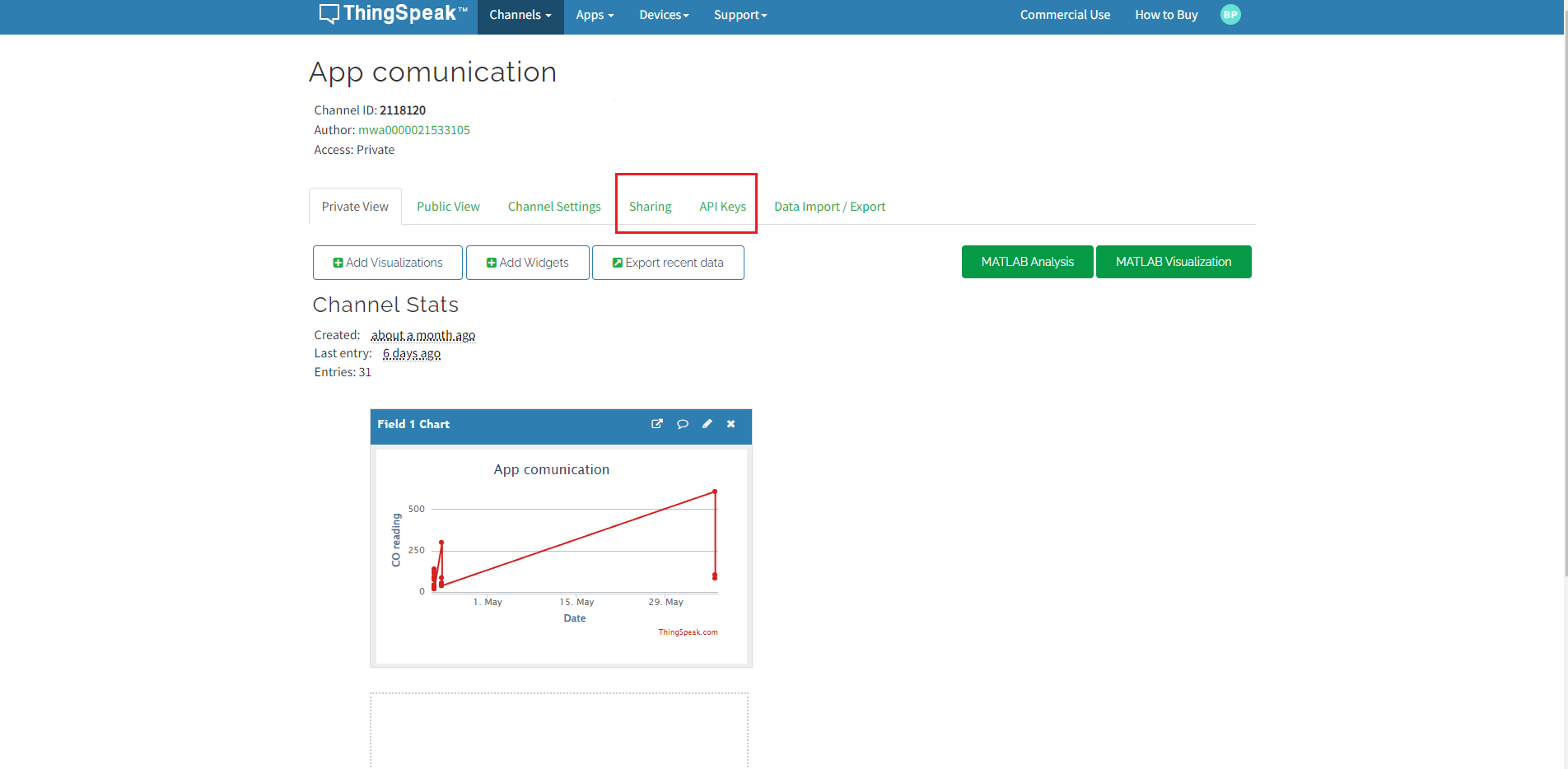
Trebuie sa merge la „My channels”, si sa apasam pe butonul „New Channel”, acesta o sa deschida o pagina in care va trebui sa realizăm setarile canalului.



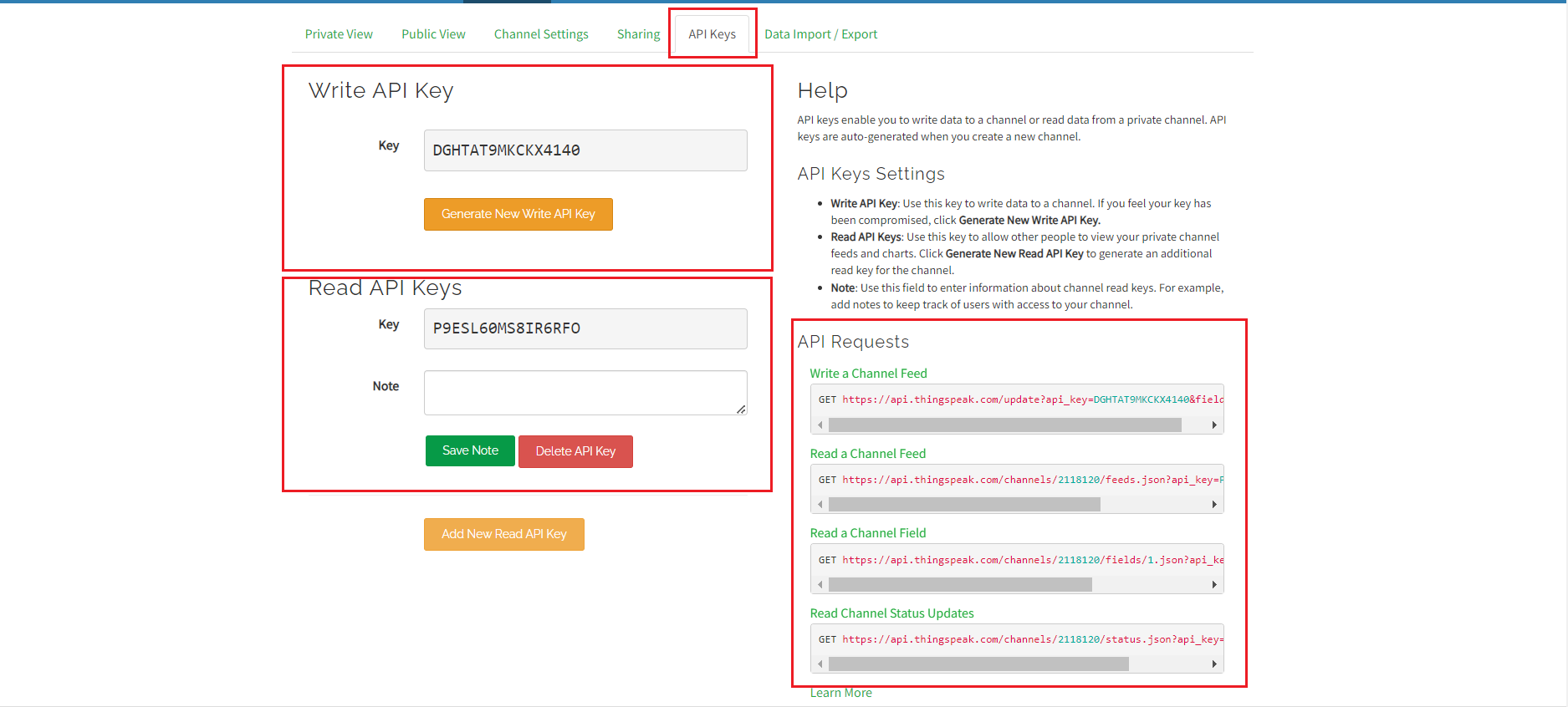
Dupa care, în urmatoarea pagina va trebui sa se seteze canalul, sa fie setat numele, numele campurilor pe care vom avea nevoie sa scrie sau sa citim date, și multe alte setari care nu sunt neaparat necesare.



Dupa ce tot a fost setat si configurat, interfata ar trebui sa arate asa. Urmatorul lucru care necesita abordat este salvarea cheii reprezentative canalului, ea poate fi gasită la „API Keys‘



In pagina „API Keys” se gasesc diferite informații despre cheile care ne ajuta la scrierea si citirea datelor de pe canal, cheile sunt diferite, una este pentru citirea datelor, iar cealaltă este pentru scrierea lor. Ba mai mult, sunt cateva exemple cu API-uri, pentru a fi mai ușor de inteles.



Gradle si dependințe folsite

Datorita faptului ca Android Studio ruleaza pe un sistem de build de tip Gradle, am folosit diferite dependințe.

A screen shot of a computer program

Description automatically generated with low confidence

Fiecare dependința din Gradle are un scop, in cazul acesta scopul dependințelor este de a defini grafice și alte entitați asemanatoare și de a permite trimiterea de mail-uri din aplicația noastra catre un cont prestabilit.

Dependințe care definesc grafice

Definiție : implemenatation ‘com.github.PhilJay:MPAndrodiChart:v3.1.0‘

Cu aceasta dependință se adaugă in cod o mulțime de clase care ne permit sa definim noi obiecte de diferite tipuri, precum LineChart, BarChart sau CandleStickChart.

Definiție : implemenation ‘com.github.Gruzer:simple-gauge-android:0.3.1’

Dependința Simple Gauge ne permite crearea de obiecte de tip „Gauge”, care ne indica intervalele in care se alfa valoarea de monoxid de carbon masurată.

Pentru partea de trimitere de e-mail-uri sunt adaugate urmatoarele librării:

* implementation files(libs/activation.jar)
* implementation files(libs/additional.jar)
* implementation files(libs/mail.jar)

Toate aceste librării sunt folosite pentru configurarea și trimitearea de e-mail-uri către utilizatorii aplicației. Acestea trebuie descarcate si adaugate in fisierul „/libs” din proiect dupa care trebuie implementate in fisierul „build.gradle”.

A screenshot of a computer

Description automatically generated

Implemenatrea Claselor si Activitaților necesare:

Clasa DataTableDetails:

A screenshot of a computer program

Description automatically generated with medium confidence

Clasa care implementeaza interfata Parcelable si care are ca atribute urmatoarele:

* private String mContact
* private String mThreshold
* private String mDate

Aceste atribute se vor fi afișate mai tarziu in istoricum de e-mail-uri trimise. Clasa implementează interfata Parcelable deoarece este necesar ca instanțele clasei sa poată fi trimise prin Intent, dintr-o activitate in alta. Tot in această clasă regasim si un constructor, care ne permite sa creăm obiecte. Din moment ce clasa implementează interfața Parcelable, este nevoie si de implementarea metodelor care se regăsesc în interfață. Se poate observa metoda „writeToParcel” care are adnotarea „@Override”. Metoda care are ca parametrii un obiect de tip „Parcel” si flags de tip intreg, si care transfera datele in obiectul de tip „Parcel”.

A screen shot of a computer program

Description automatically generated with low confidence

Alte metode care sunt implementate in clasa DataTableDetails ar fii getter-ele si setter-ele care sunt folosite pentru setarea si extragerea informațiilor din obiectele create. Clasa „Creator” este o sub-interfață, care contine metodele „createFromParcel” si „newArray”. Aceste două metode trebuie implementate in clasa „DataTableDetails”.

A screenshot of a computer program

Description automatically generated with medium confidence

Clasa „EmailDto” are ca atribute urmatoarele câmpuri:

* private String email
* private String subject
* private String message

Clasa vine in ajutorul gestionarii utilizatorilor, si salvării datelor lor. Fiecare instața a clasei contine o adresa de e-mail, subiectul e-mail-ului si mesajul care va fi trimis in e-mail. In clasa sunt implemementați constructorii, unul cu toate atributele clasei, iar celalat fara nici un atribut. Tot in clasa, pentru accesarea și scrierea de date in obiectele create sunt implementate si metodele de „get” si „set”.

Clasa DBUtils :

Clasa ce ne ajuta la definirea si implementarea metodelor pentru baza de date in care sunt salvate datele utilizatorilor.

A screen shot of a computer code

Description automatically generated with low confidence

In clasă este declarata o constantă care retine numele bazei de date. Clasa „DBUtils” extinde clasa „SQLiteOpenHelper”, deci clasa „SQLiteOpenHelper” este clasa parinte pentru clasa „DBUtils”.De aceea în constructorul clasei „DBUtils” este cuvantul cheie „super”, acest cuvant cheie, apelează constructorul clasei parinte, „SQLiteOpenHelper”.

A screenshot of a computer program

Description automatically generated with low confidence

Metodele suprascrise din clasa DBUtils:

Metoda „onCreate(SQLiteDatabase myDb)” face ceea ce îi sugerează numele, adică crează o baza de date locală printr-un query de tip SQL. Baza de date este creată cu câmpurile „email” de tip TEXT si care este si cheia primară a bazei de date si în care este salvată adresa de e-mail, și „password” tot de tip TEXT care reține parola utilizatorului, și metoda „onUpgrade(SQLiteDatabase myDb)” care sterge baza de date.

A screenshot of a computer program

Description automatically generated with low confidence

Metoda „insertDataIntoDb(String email, String password)”:

Metodă creată pentru a insera utilizatori in baza de date. In metoda se deschide conexiune cu baza de date prin metoda „getWritableDatabase()”, iar prin cuvantul cheie „this” se face referire la obiectul curent al clasei. Dupa care se crează un obiect „ContentValues” in care se vor stoca datele utilizatorului. La final se vor insera datele utilizatorului in baza de date si se va verifica daca inserarea a fost realizată cu succes.

Metoda „checkIfUserExists(String email)”:

Elementele din baza de date trebuie sa fie unice, pentru ca un utilizator sa poată dispune doar de un cont pe o singura adresa de mail. Metoda „checkIfUserExists” executa un query în baza de date cu care selectează toate obiectele care au email-ul egal cu cel transmis ca parametru si salvează rezultatul intr-un obiect de tip Cursor, obiect care este returnat de metoda „rawQuery”. Dupa ce query-ul a fost executat, se verifică daca numarul de elemente din cursor este mai mare ca 0, returnând adevarat daca este mai mare, și semnalând faptul ca utilizatorul a fost deja creat.

Metoda „loginCheck(String email, String password)”

Metodă care are ca parametrii email-ul utilizatorului si parola acestuia, verifică daca utilizatorul are cont deja creat în baza de date sau daca parola acestuia se potriveste cu emai-ul. Și in această metodă se executa un query de tip SQL în baza de date care selecteaza toate obiectele care au email-ul si parola la fel ca cele transmise ca și parametrii, după care se verifica cate obiecte ce îndeplinesc aceste condiții sunt in cursor. Daca utilizatorul are un cont creat, rezultatul metodei „cursor.getCount()” va fi 1 și metoda va returna adevarat.

A screen shot of a computer

Description automatically generated with medium confidence

In clasa „MailUtils” am setat două constante, una care reprezintă adresa de email de pe care se v-or trimite email-uri catre utilizator, iar cealată este parola pentru aplicații care este creată în contul de Gmail, ambele fiind necesare pentru a trimite email-uri din aplicație.

A screen shot of a computer program

Description automatically generated with low confidence

Clasa JavaMailApi:

Țelul clasei JavaMailApi este acele de a trimite email-uri. Atributele acestei clase sunt :

* private Context mContext
* private Session mSession
* private String mEmail
* private String mSubject
* private String mMessage

In ea este declarat si constructorul care are are ca parametrii toate atributele clasei. Aceasta clasa extinde clasa „AsyncTask” deoarece este nevoie de suprascrierea metodelor care fac parte din aceasta clasa.

A screen shot of a computer

Description automatically generated with low confidence

Metoda „onPreExecute()”

Face parte din clasa „AsyncTask” si este suprascrisa în clasa „JavaMailApi”. Metoda deschide un dialog înainte de a se trimite email, si notifică utilizatorul ca va urma sa se trimită un email.

Metoda „onPostExecute(Void aVoid)”

Notifică utilizatorul cand un email sa trimis cu succes catre adresa setată.

A screen shot of a computer screen

Description automatically generated with low confidence

Metoda „doItInBackground(Void ... params)”

In aceasta metoda se trimite mesajul implicit catre utilizator. In prima parte a metodei se seteaza proprietațile și se configurează pentru a putea trimite mail-uri pe Gmail. Prima data se setează hostul, acesta fiind „smtp.gmail.com”, dupa care se indica port-ul si se trece campul pentru autentificare pe „true. Dupa care se crează o sesiune nouă cu credențialele setate in MailUtils. După toate acestea se creaza un obiect de tip „MimeMessage” cu sesiunea curenta și se configurează cu adresa de email de pe are se trimit email-uri, adresa de mail care primește email-uri, cu subiectul si mesajul email-ului. Cu metoda statica „send(MimeMessage mm)” se trimite email-ul. În caz ca ceva nu merge bine, sau se aruncă vreo excepție aceasta este prinsa si este tratată.

Implementarea Claselor pentru Activitați

Clasa „LogIn”

A screen shot of a computer program

Description automatically generated with low confidence

In activitatea de log in s-a creat un buton care preia datele din casetele de text in care utilizatorul a adaugat adresa de e-mail si parola cu care s-a inregistrat, dupa care se salveaza in doua variabile si se verifica daca ambele casete de text au fost completate, in caz contrar utilizatorul va primi un mesaj in care este notificat ca trebuie să completeze ambele campuri, dupa care, se verifica cu metoda „loginCheck” din clasa „DBUtils” daca utilizatorul exista in baza de date, in cazul in care acesta nu exista, se afiseaza mesajul corespunzator, cum ca datele introduse nu se află in baza de date sau ca nu se potrivesc. Daca datele sunt corecte, se crează un obiect „intent” care se oupa cu deschiderea unei noi activitați. In intent se adaugă un câmp extra cu adresa de e-mail introdusa in caseta de text, pentru a putea fi folosită la trimiterea de mesaje. Tot in activitatea de „LogIn” se afla un buton care deschide activitatea de „Register”

A screen shot of a computer program

Description automatically generated with low confidence

Clasa „Register”

A screenshot of a computer screen

Description automatically generated with low confidence

Dupa completarea campurilor, cu adresa de email, parola si confirmarea parolei, utilizatorul va trebui sa apese pe butonul de inregistrare, care va apela metoda „onClick” si va salva datele utilizatorului. Verifica daca toate campurile au fost completate și notifica utilizatorul daca nu au fost completate. După care verifică ambele parole sa coincidă, în caz contrat afiseaza un mesaj corespunzător pe ecran. Tot aici se verifica daca utilizatorul există deja, cu metodele implementate în clata „DBUtils”, daca utilizatorul nu exista se inserează in baza de date utilizatorul cu email-ul. In cele din urma deschide activitatea de logare, pentru ca utilizatorul sa se poată inregistra cu noul cont creat.

Clasa „SenzorValueActivity”

Clasa „SenzorValueActivity” este clasa în care sunt definite metodele petru citirea datelor de pe ThingSpeak si prelucrarea lor.

A screenshot of a computer program

Description automatically generated with medium confidence

Aici sunt definite metodele „onClick” ale butoanelor, butonul de reimporspatare apeleaza metoda „getRequest(url)”, iar in celelalte metode se deschid activitatea care conține setarile si cea care conține graficul cu datele in timp real. La final se setează atributele obiectului „emailDto”, email-ul care provine din activitatea „LogIn”, prealuat din intent, și celelate atribute, care sunt declarate drept constante în activitatea curentă.

A picture containing text, screenshot, software, multimedia software

Description automatically generated

Metoda „afterTextChange” execută codul dupa ce se modifica textul in campurile din activitatea principală unde. Se actulizează „threshold”-ul, si se modifică si în mesajul email-ului, în caz ca se întamplă ceva neprevazut, excepția este tratată si este afisat un mesaj care sa notifice utilizatorul. A screen shot of a computer

Description automatically generated with low confidence

Setare care determină daca se trimit sau nu email-uri, acesta verifică daca butonul de tip „switch” este bifat sau nu, si seteză câmpul „isSendingEmails” in funcție de valoarea butonului.

A picture containing text, screenshot, multimedia software, software

Description automatically generated

Codul din imaginea ”vezi imaginea” apelează „run()” o dată la o perioadă de timp, în cazul de față, se apelează la valoarea setată de catre utilizator din interfață sau la o valoare standard care este setată.

A screen shot of a computer screen

Description automatically generated with low confidence

Metoda „checIfThresoldIsReached(int threshold, boolean isSendingEmails)” verifica daca valoarea citită de senzor este mai mare decât pragul setat si daca se butonul care oprește trimiterea de email-uri nu este adevărat, dacă conditia este adevarată, se trimite email si se adaugă intr-un obiect de tip „DataTableDetails” email-ul la care s-a trimis mesaj, data la care s-a trimis și valoarea pragului care a fost depasită.

A picture containing text, screenshot, software, multimedia software

Description automatically generated

„checkIfSendEmails(int threshold, boolean isSendingEmails)” verifica daca valoarea senzorului depașește pragul și totdată verifică starea in care se află butonul pentru setarea trimiterii de email-uri.

Metoda „sendEmail”, face exact ceea ce ii sugerează numele, trimite email-uri catre utilizatorul care este trimis ca parametru, cu mesajul si subiectul respectiv. În această metodă este definit un obiec de tip „JavaMailApi” in care se setează contactul, subiectul si mesajul, dupa care se apelează metoda „execute()” care declansează trimiterea email-ului.

A screen shot of a computer screen

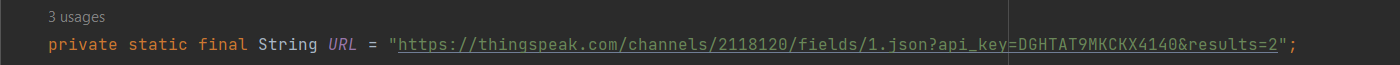
Description automatically generated with low confidence

Metodele din imaginea ”referința la imagine” sunt metode apelate cand se apasa butoanele, acestea crează un intent la care se adaugă câmpurile necesare in celelate activitați.

A screen shot of a computer program

Description automatically generated with low confidence

În metoda „getRequest”, se crează o coadă în care se adaugă un request cu contextul activitații curente. Se crează un obiect StringRequest si se configurează apelul catre ThingSpeak, metoda folosita este „GET”, metodă care ne permite sa retragem date din canalul creat, si se specifică si url-ul la care sa se faca apel.



URL-ul este prezentat în imaginea ”referința la imagine”, se poate observa ca are mai multi parametrii, printre care si cheia canalului de pe care se vor retrage date.

Metoda „onResponse” preia ca parametru raspunsul apelului către url, si il prelucrează. Se crează un obiect de tip „JSONObject” si se adauga răspunsul apelului, după care se preai toate datale din obiectul „arrayOfData” sub forma de „JSONArray”. Se ia ultimul obiect din listă și se transforma din „String” in obiect de tip „JSONObject” pentru a putea fi preluată data din câmpul „field1”, această valoare fiind de fapt valoarea senzorului. Pentru a intelege mai bine cum funcționează metoda s-a simulat un apel din PostMan și se poate vedea un exemplu al unui răspunsul primit de la ThingSpeak in imaginea „vezi imaginea”

A screenshot of a computer code

Description automatically generated with low confidence

Se poate observa campul „feeds” care este un vector de obiecte care conține câmpurile „created\_at”, „entry\_id” și „field1” care de fapt este valoarea de monoxid masurată de senzor.

A picture containing text, screenshot, software, multimedia software

Description automatically generated

Pentru a crea un grafic in timp real, acesta se folosește de aceeași metoda care se reapealează la un interval de timp setat. Se creaza un „lineDataSet” in care se adaugă valoarea citită de la senzor dupa care se seteaza in obiectul „mLineChart”.

Activitatea Istoricului.

A screen shot of a computer code

Description automatically generated with low confidence

Pentru afisarea tuturor datelor in istoric, datele sunt preluate de la celelalte activitați si sunt parcurse, iar pentru fiecare obiect se setează o nouă linie in tabelul „tableRowForData”, si se adauga contactul, valoarea pragului si data la care s-a trimis email-ul.