

UNIVERSITATEA POLITEHNICA BUCUREȘTI
FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE
DEPARTAMENTUL CALCULATOARE



PROIECT DE DIPLOMĂ

Smart City – Air Quality Monitoring

Claudiu-Daniel Marinescu

Coordonator științific:
Prof. dr. ing. Francisc Iacob

BUCUREȘTI

2019

UNIVERSITY POLITEHNICA OF BUCHAREST
FACULTY OF AUTOMATIC CONTROL AND COMPUTERS
COMPUTER SCIENCE DEPARTMENT



DIPLOMA PROJECT

Smart City – Air Quality Monitoring

Claudiu-Daniel Marinescu

Thesis advisor:
Prof. dr. ing. Francisc Iacob

BUCHAREST

2019

CUPRINS

Sinopsis	4
Abstract	4
1 Introducere	5
1.1 Context	5
1.2 Problema	5
1.3 Obiective	6
1.4 Soluția propusă.....	6
1.5 Structura lucrării.....	7
2 Analiza și specificarea cerințelor	8
3 Studiu de piață / Soluții existente	11
3.1 Rețeaua Națională	11
3.2 Airly	12
3.3 Dispozitive independente	13
4 Soluția propusă	15
4.1 Aplicația server	16
4.2 Aplicația web	19
4.3 Soluția hardware	25
5 Evaluarea rezultatelor	29
6 Concluzii.....	33
7 Bibliografie.....	35
8 Anexe	36

SINOPSIS

Calitatea aerului pe care îl respirăm în fiecare zi ne influențează în mod direct sănătatea, având efecte atât pe termen scurt, cât și pe termen lung. Fiindcă soluțiile existente sunt subdezvoltate și dificil de accesat, populația este dezinformată și măsurile de îmbunătățire a calității aerului lipsesc.

Din acest motiv am realizat o soluție completă pentru un oraș inteligent pentru monitorizarea calității aerului: de la o rețea de stații pentru monitorizare, răspândită pe suprafața orașului, la o aplicație capabilă să gestioneze fluxul de date în timp real și să le ofere utilizatorilor într-un mod atrăgător, familiar, ușor de accesat.

Rezultatele obținute confirmă utilitatea acestei soluții: aplicația este scalabilă, eficientă, rapidă, ușor de folosit, oferă informații relevante privind calitatea aerului și poate recomanda, pentru o rută introdusă de utilizator, alternative mai puțin poluate.

ABSTRACT

The air quality we breathe every day directly affects our health, with effects both on the short and the long term. As the existing solutions are underdeveloped and difficult to access, the population is misinformed and measures to improve air quality are missing.

Consequently, we built a complete solution for a smart city to monitor air quality: from a network of monitoring stations spread across the city to an application capable of managing real-time data flow and of exhibiting them in an attractive, familiar, accessible way.

The results confirm the utility of this solution: the application is scalable, efficient, fast, easy to use, provides relevant air quality information and may recommend less polluting alternatives for a user-initiated route.

1 INTRODUCERE

București – orașul în care tranzitează zilnic peste 3 milioane de oameni, cel mai dezvoltat oraș din România din punct de vedere industrial și comercial, dar și cel mai populat oraș. În fiecare zi, cele 3 milioane de oameni respiră aerul din București fie că merg la serviciu sau facultate, participă la un eveniment cultural sau pur și simplu se relaxează în unul dintre numeroasele parcuri ale acestui oraș. Din acest motiv (și din altele, pe care le voi prezenta în continuare), calitatea aerului pe care îl respirăm are o importanță deosebită, aceasta influențând în mod direct calitatea vieții pe care o trăim.

1.1 Context

Proiectul propus are în vedere monitorizarea calității aerului în București, putând fi extins în toate orașele României. Mai concret, pentru a evalua calitatea aerului vom măsura un număr de parametri, cum ar fi: monoxid de carbon, particule în suspensie (PM10 și PM2.5), compuși organici volatili (COV) și oxizi de azot, cu ajutorul cărora vom calcula nivelul AQI (Air Quality Index), metrica cea mai importantă în ceea ce privește calitatea aerului.

Din punct de vedere legislativ, monitorizarea calității aerului este reglementată prin publicarea în Monitorul Oficial a Legii nr.104/15.06.2011 privind calitatea aerului înconjurător, în care sunt specificați poluanții care trebuie monitorizați (printre care și cei enumerați mai sus) și limitele superioare în care aceștia trebuie să se încadreze de-a lungul unui an: valorile reglementate sunt reprezentate ca medie a mai multor măsurători de-a lungul unui an.

Totodată, Uniunea Europeană a publicat în Jurnalul Oficial al Uniunii Europene (JOUE) nr. L152 din 11 iunie 2008 prevederi pentru domeniul calității aerului, prevederi incluse în Legea nr.104/15.06.2011 amintită mai sus.

1.2 Problema

Cu toate că monitorizarea calității aerului este reglementată prin lege și poluarea influențează în mod direct viața de zi cu zi a fiecărui om din București, lipsa de interes a autorităților în acest sens a condus la o infrastructură subdezvoltată sau chiar absentă și, în consecință, la lipsa informării populației cu privire la calitatea aerului.

În București există opt stații de monitorizare deținute de Agenția pentru Protecția Mediului București, dintre care, însă, doar una funcționează. Pe lângă aceasta, Rețeaua Națională de Monitorizare a Calității Aerului deține aproximativ 150 de stații, dintre care doar 7 se află pe teritoriul orașului București, adică pe o suprafață de 230 de kilometri pătrați. În consecință, autoritățile nu cunosc calitatea aerului, populația de asemenea nu cunoaște calitatea aerului, însă cu toții respirăm același aer zi de zi și resimțim efectele poluării, fie acestea pe termen scurt sau lung.

De remarcat este că nu doar pietonii și bicicliștii sunt afectați de calitatea aerului, ci și cei aflați în interiorul unui autoturism, contrar opiniei populare. Ba mai mult, aerul din interiorul unui autoturism poate ajunge chiar de până la 15 ori mai poluat decât aerul din afara acestuia, în

principal din cauza faptului că poluanții, odată introduși în autoturism, au tendința de a rămâne acolo și chiar de a se agrava sub influența directă a soarelui și a căldurii. De asemenea, fiindcă niciun automobil nu este ermetic, chiar și atunci când recircularea aerului este pornită, o mare parte din poluanții emanați de autoturismele din fața noastră vor ajunge în interiorul autoturismului cu care ne deplasăm.

1.3 Obiective

Obiectivul principal al acestui proiect este oferirea unei soluții complete pentru informarea populației cu privire la calitatea aerului: pornind de la o rețea de stații pentru monitorizarea calității aerului, răspândită pe suprafața orașului București, la o aplicație capabilă să gestioneze fluxul de date în timp real și în mod eficient și scalabil. În plus, această soluție își propune a fi ușor de accesat și folosit în viața de zi cu zi de către locuitorii orașului, astfel devenind o soluție pentru un oraș inteligent.

1.4 Soluția propusă

Pentru atingerea obiectivelor acestui proiect, soluția propusă este alcătuită din 3 componente: un prototip hardware pentru măsurarea poluanților, o aplicație server pentru colectarea și gestionarea datelor și o aplicație web cu triplu rol: de a informa utilizatorii de impactul calității aerului în viața de zi cu zi, de a monitoriza nivelul diferiților poluanți din aer în ziua curentă, de a recomanda, la introducerea unei rute, o alternativă mai puțin poluată, serviciu care va fi integrat cu Google Maps.

Prototipul hardware este un dispozitiv propus pentru măsurarea poluanților. Acesta va fi construit dintr-o plăcuță Arduino, senzori pentru măsurarea nivelului poluanților, precum monoxid de carbon, particule în suspensie (PM10 și PM2.5), compuși organici volatili (COV) și oxizi de azot, și dintr-un canal de comunicație cu aplicația server. La momentul implementării acestui proiect într-un oraș, acest prototip va fi instalat în locații multiple, astfel încât orașul să fie monitorizat complet. Mai multe detalii vom prezenta în capitolul 4-Soluția propusă.

Aplicația server, detaliată în capitolul 4-Soluția propusă, este responsabilă pentru colectarea datelor. Folosind un Scheduled Task (un task programat să ruleze la un anumit interval de timp), aplicația server interoghează prototipul hardware, prototipul efectuează măsurători pentru fiecare senzor existent și le trimite ca răspuns serverului. Datele primite sunt stocate într-o bază de date MySQL timp de 24 de ore. De asemenea, aplicația server poate colecta date de la aplicațiile existente în acest domeniu și care permit citirea datelor pentru un anumit interval de timp și un număr de poluanți (spre exemplu, Airly, detaliat în capitolul 3-Metode existente, pune la dispoziție un API în mod gratuit în anumite limite de cereri pe zi; cu toate acestea, dispozitivele instalate de Airly sunt mult prea puține pentru ca aplicația prezentă să funcționeze corect având doar date provenite din această sursă).

Aplicația web este destinată publicului larg și îndeplinește mai multe roluri. Un prim rol este cel de informare: utilizatorii se pot documenta, într-o pagină dedicată, cu privire la diferitele tipuri de poluanți existenți, la impactul lor asupra sănătății noastre și la riscurile la care ne

expunem în fiecare zi, respirând aerul din București (CITAT). De asemenea, pagina principală a aplicației conține statistici despre nivelul calității aerului din ultimele ore. Se pot accesa date de până la 24 de ore în trecut și specifice unei anumite zone din București. Utilitatea acestei funcționalități este de a oferi utilizatorilor date în timp real cu privire la nivelul poluanților din București. O a treia și ultimă funcționalitate a clientului web este integrarea cu Google Maps într-o aplicație de rutare: pentru o anumită rută introdusă de utilizator, aplicația poate recomanda o alternativă mai puțin poluată, decizia luându-se pe baza măsurărilor efectuate în punctele prin care va trece ruta introdusă. Utilizatorul va avea posibilitatea să salveze această alternativă în contul personal de Google și să o acceseze în momentul dorit.

1.5 Structura lucrării

În continuare vom aprofunda motivațiile acestui proiect în capitolul 2: vom răspunde la întrebări precum de ce este nevoie de o asemenea aplicație, ce valoare aduce utilizatorului aplicația, care este impactul pe care calitatea aerului o are în viața de zi cu zi și de ce soluțiile existente nu sunt suficiente pentru împlinirea scopului acestui proiect.

În capitolul 3 vom analiza soluțiile existente pe piață în acest sens, care sunt avantajele și dezavantajele acestora, urmând ca în capitolul 4 să fie prezentată soluția propusă pentru a acoperi neajunsurile soluțiilor existente. Vom vedea care este arhitectura soluției propuse, cum intercomunică componentele constitutive, cum sunt construite și ce rol îndeplinesc aceste componente și, în final, ce aduce în plus această soluție față de cele prezentate în capitolul 3.

În capitolul 5 vom evalua aplicația din mai multe perspective și vom vedea dacă aceasta și-a atins scopul și poate deservi utilizatorii în sensul anunțat anterior. La final, în urma acestor evaluări vom trasa concluziile acestui proiect (în capitolul 6).

2 ANALIZA ȘI SPECIFICAREA CERINȚELOR

Dezvoltarea prezentului proiect este motivată din cel puțin trei direcții: din punct de vedere legislativ, din punct de vedere al impactului poluării asupra sănătății oamenilor și din punct de vedere informatic, al integrării tehnologiei în viața de zi cu zi cu scopul de a ne îmbunătăți și ușura viața, de a transforma un simplu oraș într-un oraș inteligent. În continuare vom dezvolta pe rând aceste motivații.

Motivația legislativă a fost prezentată succint în capitolul 1-Introducere. Legea nr.104/15.06.2011 privind calitatea aerului reglementează normele și cadrul în care trebuie monitorizată calitatea aerului: „În scopul monitorizării impactului poluării atmosferice asupra sănătății populației și mediului, măsurarea și evaluarea calității aerului înconjurător în puncte fixe de măsurare deținute și exploatate de către instituții publice sau autorități ale administrației publice locale, de către operatori economici, organizații neguvernamentale sau alte persoane juridice private se realizează cu respectarea prevederilor prezentei legi.”.¹

Pe lângă aceasta, monitorizarea calității aerului este reglementată și în cadrul Uniunii Europene. Fiindcă în România nu s-au implementat metode pentru gestionarea monitorizării calității aerului, Uniunea Europeană și-a manifestat îngrijorarea și la acest capitol, amintind că “autoritățile române NU au făcut nimic pentru a reduce poluarea din București și [U.E.] a dat statul român în judecată, anul trecut. România risca să plătească amenzi uriașe, între 100 și 400 de mii de euro pe zi, dacă nu rezolvă problema poluării.”.² România se află pe locul 8 în topul celor mai poluate țări din Uniunea Europeană, având indici de poluare cuprinși între 73 și 131. La polul opus, Helsinki, are valori între 13 și 21, potrivit aceluiași studiu.

Impactul poluării asupra sănătății a crescut semnificativ în ultimii ani și a ajuns să capete un caracter global, în principal datorită faptului că poluarea provenită din activitatea industrială (fabrici), centrale termoelectrice și traficul rutier a depășit poluarea rezultată în mod natural din erupții vulcanice sau dispersia polenului. Printre cei mai frecvenți poluanți se numără particulele (PM10, PM2.5), monoxidul de carbon, dioxidul de sulf, azotul și compușii organici volatili (COV). [5]

Dimensiunea particulelor este direct legată de potențialul de a cauza efecte. O problemă importantă o reprezintă particulele cu diametrul aerodinamic mai mic de 10 micrometri, care trec prin nas și gât și pătrund în alveolele pulmonare provocând inflamații și intoxicații. Sunt afectate în special persoanele cu boli cardiovasculare și respiratorii, copiii, vârstnicii și astmaticii. Expunerea pe termen lung la o concentrație scăzută de pulberi poate cauza cancer și moartea prematură. [1]

¹ Legea nr.104/15.06.2011, articolul 5, subpunctul 6

² “Bucureștiul, locul 7 în Europa la poluare”, <http://www.stiri.tvr.ro> (9 Februarie 2019)

Monoxidul de carbon provine atât din surse naturale, cât și antropice, însă principala sursă pentru producerea de monoxid de carbon o reprezintă arderea incompletă a combustibililor fosili. Acesta afectează capacitatea organismului de a transporta oxigenul în sânge, fiind periculos în special pentru persoanele cu probleme cardiace, expunerea la concentrații mari putând fi fatală. De asemenea, poate afecta sistemul nervos central, poate cauza oboseală acută și poate determina iritabilitate, migrene, amețelă.

Dioxidul de sulf este un gaz provenit în mod natural din erupții vulcanice, iar în mod antropic din emisiile provenite de la motoarele diesel, centrale termoelectrice, procese industriale. Acesta poate produce efecte diverse, de la dificultăți respiratorii severe la infecții ale tractului respirator. Cele mai afectate persoane sunt copiii, persoanele cu astm și persoanele cu boli cronice ale căilor respiratorii.

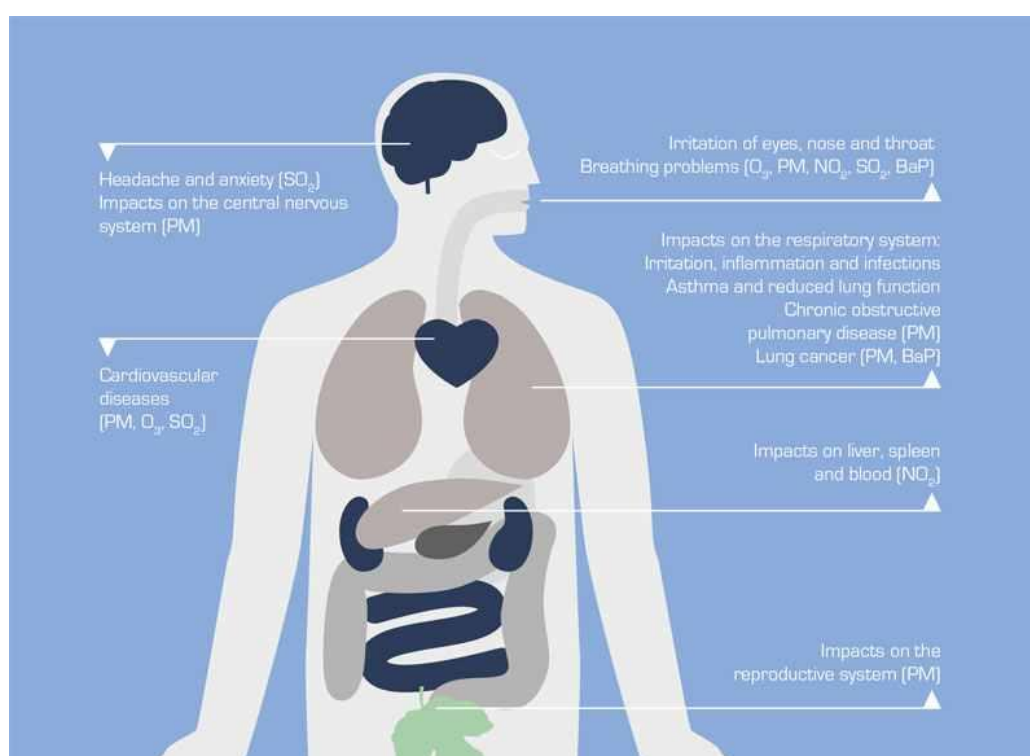


Figura 1. Efectele poluanților asupra corpului uman

După cum se poate observa și în imaginea de mai sus, poluarea ne afectează diferite părți ale corpului uman. Nu este de mirare că poluarea aerului ocupă locul 4 în lista factorilor cu risc extrem pentru decesele premature. Astfel, considerând această motivație, a impactului poluării asupra sănătății, putem spune că monitorizarea aerului ar trebui să fie o necesitate pentru orice oraș și ar trebui tratată riguros de către autorități și chiar de către locuitorii fiecărui oraș.

Nu în ultimul rând, dezvoltarea acestui proiect este motivată de informatizarea oraşului Bucureşti. Tehnologia ne-a uşurat şi îmbunătăţit viaţa şi nivelul de trai şi va continua să o facă: de la calculatoare, laptopuri, telefoane inteligente, maşini digitalizate, până la aproape orice electrocasnice inteligente. Toate acestea ne-au impactat viaţa personală într-un fel sau altul.

Proiectul prezent propune o soluţie pentru a îmbunătăţi viaţa colectivă, în afara spaţiului personal al casei. O soluţie pentru momentul când suntem pe stradă ori pe trotuar, la un festival sau pur şi simplu la o plimbare prin parc. Având la dispoziţie o modalitate accesibilă de a se informa corect, un locuitor al oraşului Bucureşti îşi poate planifica o rută sau chiar programul unei zile în funcţie de acele informaţii.

Informaţiile pe care ne propunem să le oferim prin acest proiect se referă la calitatea aerului pe care îl respirăm în fiecare zi. Astfel, utilizatorii vor avea la dispoziţie o soluţie uşoară şi accesibilă de a afla calitatea aerului în orice moment al zilei, în aproape orice zonă a Bucureştiului. Desigur, conştientizarea faptului că aerul pe care îl respirăm nu este cel mai curat ar trebui să aducă după sine câteva soluţii în acest sens, însă această lucrare are ca scop informarea populaţiei, adică conştientizarea.

3 STUDIU DE PIAȚĂ / SOLUȚII EXISTENTE

Cu toate că în momentul actual pe piață nu există o abordare care să acopere toate nevoile unui oraș inteligent în sensul monitorizării calității aerului, ne putem opri asupra unor soluții parțiale. Așadar, în acest capitol vom urmări câteva dintre soluțiile existente pentru a monitoriza poluarea în orașul București, caracteristici ale acestor metode, cât și limitările pe care le au. Soluțiile existente oferă informații utilizatorilor în mod static: utilizatorii pot observa calitatea aerului în anumite zone, în funcție de numărul de stații de monitorizare pe care le deține o anumită soluție, însă interacțiunea cu utilizatorul se oprește aici. În capitolul următor vom prezenta ce aduce soluția propusă de această lucrare în plus față de cele prezenta în acest capitol.

3.1 Rețeaua Națională

Rețeaua Națională de Monitorizare a Calității Aerului (RNMCA) este soluția autorităților pentru a monitoriza calitatea aerului din România. Această rețea însumează 142 de stații de monitorizare, dintre care 7 se află pe teritoriul Bucureștiului.

Stațiile rețelei naționale sunt dotate cu echipamente automate pentru măsurarea concentrațiilor principalilor poluanți atmosferici: dioxid de sulf (SO_2), oxizi de azot (NO_2/NOX), monoxid de carbon (CO), ozon (O_3), pulberi în suspensie (PM_{10} și $\text{PM}_{2.5}$), benzen (C_6H_6), metale grele (plumb, cadmiu, nichel, arsen, mercur), hidrocarburi aromatice policiclice.³

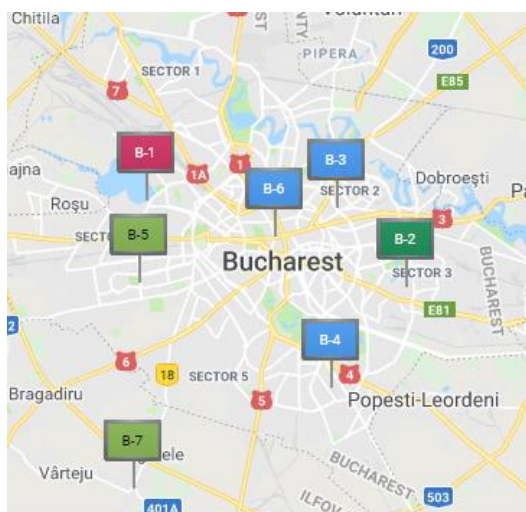


Figura 2. Rețeaua Națională de Monitorizare a Calității Aerului

Cele 7 stații de monitorizare prezente pe teritoriul Bucureștiului sunt, după cum se poate observa, departe de a fi suficiente. În Cracovia există cel puțin 70 de stații de monitorizare pentru o suprafață de aproximativ 300 de kilometri pătrați. Dat fiind faptul că suprafața

³ "Rețeaua Națională de Monitorizare automată a calității aerului", <http://www.anpm.ro> (13 Octombrie 2010)

orașului București este de aproape 230 de kilometri pătrați, este evident că rețeaua națională de monitorizare a calității aerului nu poate fi o sursă concludentă pentru informarea populației.

Mai mult decât atât, această soluție nu este ușor de accesat, iar datele prezentate sunt dificil de înțeles de către un utilizator dezinformați în domeniu.

3.2 Airly

Cea de-a doua soluție, deși în stadiu incipient în România, este cu mult mai promițătoare decât cea anterioară. Airly este o companie dedicată realizării unei rețele de monitorizare a calității aerului în Polonia, unde a reușit să instaleze mai mult de 2000 de senzori, Polonia fiind una dintre cele mai poluate țări din Uniunea Europeană. Dintre aceștia, aproape 70 de stații se află în orașul Cracovia, un oraș comparabil cu orașul București ca suprafață. Mediul urban în care trăim ne impactează viața de zi cu zi. Chiar dacă unele amenințări nu se văd, nu putem pretinde că acestea nu există.

Compania Airly susține că primul pas spre a îmunătăți situația este să crească nivelul de informare cu privire la problemă și să ofere acces la date privind calitatea aerului local. Acesta este și scopul acestei lucrări, după cum am menționat anterior, în primul capitol.

Rețeaua independentă Airly este prezentă în România din octombrie 2018, având în momentul actual 15 stații de monitorizare amplasate pe teritoriul României (14 dintre ele în București, una în Ploiești) și oferind date în timp real cu privire la poluanți precum PM10, PM2.5, PM1, temperatură, umiditate și presiune.

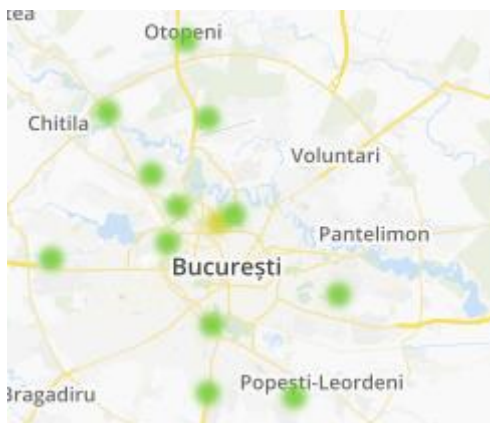


Figura 3. Rețeaua Airly

Harta Airly este disponibilă atât online, accesând site-ul <http://airly.eu/map/>, cât și în aplicația proprie, accesibilă atât pe dispozitive Android cât și iOS. Informațiile sunt afișate atractiv, aplicația este ușor de folosit și oferă date relevante cu privire la calitatea aerului. Cea mai nouă funcționalitate a acestei aplicații este predicția pentru următoarele ore a nivelului

calității aerului. Predicția nu este deloc precisă la momentul actual, însă ideea poate avea într-adevăr un impact semnificativ.

Pe lângă această aplicație, Airly oferă acces la resursele pe care le gestionează printr-un API deschis. Acest API este destinat dezvoltatorilor de aplicații care vor să folosească date cu privire la calitatea aerului în proiectele lor. Acesta este și cazul proiectului curent: folosind un task programat, apelăm API-ul Airly și salvăm în baza de date proprie datele primite ca răspuns. Desigur, numărul stațiilor Airly este mult prea mic pentru a ne putea baza doar pe această sursă ca furnizor de date, însă nu este nici de neglijat. Mai multe detalii despre toate acestea vom prezenta în capitolul următor.

În ceea ce privește limitările acestei soluții, aplicația Airly este o aplicație informativă, asemenea Rețelei Naționale, care nu permite interacțiunea cu utilizatorul în vreo formă: datele pot doar a fi observate. În ceea ce privește API-ul, acesta este limitat la 1000 de apeluri pe zi și la 50 de apeluri pe minut (varianta gratuită a serviciului). Așa cum am menționat anterior, numărul stațiilor localizate în București este foarte mic, însă putem observa potențialul acestei soluții în exemplul din Cracovia (70 de stații instalate).

3.3 Dispozitive independente

Pe lângă soluțiile centralizate prezentate până acum există și o multitudine de dispozitive independente, care pot fi achiziționate în mod individual și care pot reprezenta soluții acolo unde nu există nicio rețea pentru monitorizarea calității aerului.

Printre aceste dispozitive putem aminti: Flow, Digital Radon Monitor (CNR-05), Pollution Indicator (CDL-210), Particle Counter (ML-DZ6800).



Figura 4. Dispozitivul Particle Counter ML-DZ6800

În continuare ne vom axa pe dispozitivul Particle Counter ML-DZ6800, însă toate dispozitivele existente pe piață măsoară mai mult sau mai puțin aceiași poluanți și au aceleași funcționalități disponibile.

Dispozitivul Particle Counter poate măsura poluanți precum: PM2.5, PM10, CH₂CO, VOC și temperatura și umiditatea din aer. Este un dispozitiv mic, de dimensiuni 150 x 70 x 43 mm care poate fi transportat cu ușurință în rucsac sau chiar în buzunar. Acesta are un display pe care sunt afișate valorile poluanților din aer în timp real și funcționează cu ajutorul unei baterii.

Desigur, un astfel de dispozitiv poate fi foarte folositor în situații unde nu există și nu vor exista rețele pentru monitorizarea poluării (în interiorul locuinței, într-un oraș mai mic, în zone greu accesibile, cum ar fi zonele montane). Însă în capitala unei țări membre a Uniunii Europene, o astfel de soluție poate fi văzută ca una de urgență, în lipsa unei rețele pentru monitorizarea calității aerului integrată în structurile orașului. În plus, pentru a măsura nivelul poluării dintr-o anumită zonă trebuie să fii prezent în acea zonă, deja respirând aerul de acolo. Așadar, informațiile oferite de acest dispozitiv pot fi folosite doar pentru constatare.

După cum am putut observa în cele prezentate, deși există anumite soluții pe piață, acestea sunt fie subdezvoltate, fie greu accesibile, fie sunt construite cu totul pentru alt scop, însă folosite pentru monitorizare din lipsă de alternative. Din aceste motive, necesitatea unei soluții care să îndeplinească condițiile (sau obiectivele) anunțate anterior este justificată. În următorul capitol vom prezenta o soluție pentru rezolvarea acestei probleme.

4 SOLUȚIA PROPUȘĂ

Soluția pe care o propunem este formată din 3 componente principale: aplicația server, aplicația web și o rețea de senzori distribuită pe suprafața orașului, alcătuită din unități individuale asemenea prototipului propus în continuare. Fiecare dintre aceste componente vor fi analizate, pe rând, în capitolul curent.

Arhitectura soluției propuse este descrisă în diagrama următoare:

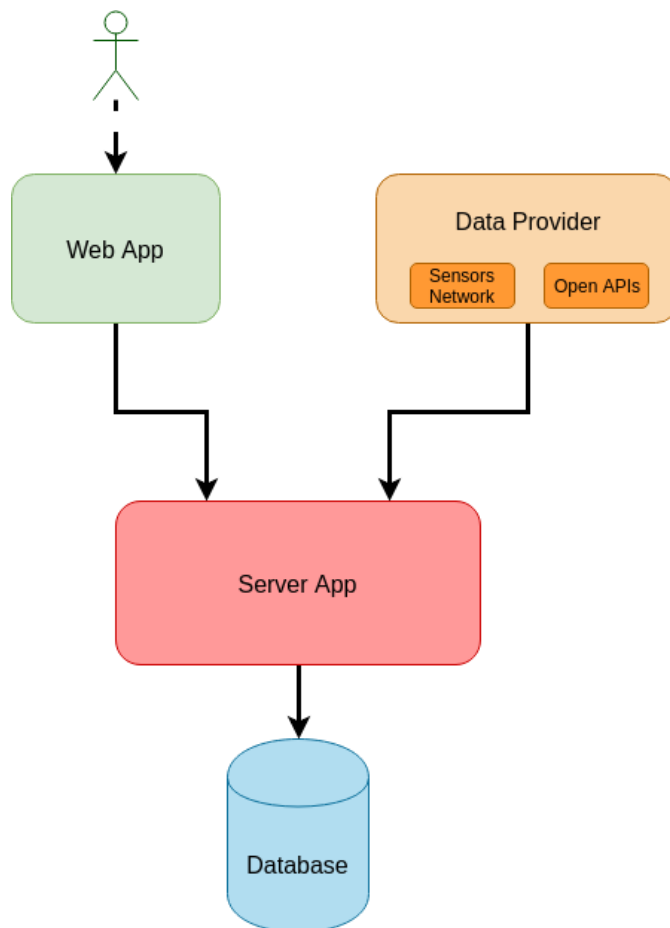


Figura 5. Arhitectura soluției

Aplicația server va avea ca furnizori de date API-ul celor de la Airly și rețeaua proprie de senzori, pe care o va putea interoga la orice moment de timp pentru a realiza măsurători și a întoarce date în timp real cu privire la calitatea aerului. Datele vor fi salvate într-o bază de date timp de 24 de ore, acestea fiind accesibile utilizatorilor prin clientul web.

Interfața cu utilizatorul va fi simplă, ușor de folosit și va beneficia de funcționalități precum prezentarea sub forma unor grafice a nivelului calității aerului din diferite zone și recomandarea unei alternative mai puțin poluate la introducerea unei rute de către utilizator.

4.1 Aplicația server

Aplicația server este construită cu ajutorul Spring Framework. Spring este o platformă Java care oferă suport pentru infrastructură pentru dezvoltarea aplicațiilor Java. Spring se ocupă de infrastructură pentru ca dezvoltatorii să se poată concentra pe aplicație. [2]

```
@SpringBootApplication
public class BackendApplication {

    public static void main(String[] args) {
        SpringApplication.run(BackendApplication.class, args);
    }
}
```

Figura 6. Funcția main a aplicației server

Baza de date folosită este MySQL. Ca sistem de versionare al bazei de date s-a folosit Liquibase. Un sistem de versionare pentru baze de date presupune memorarea fiecărei modificări asupra tabelelor, utilizatorilor, rolurilor, memorarea realizându-se prin înregistrarea într-o tabelă specifică a statusului rulării fiecărui script (dacă s-a rulat cu succes sau nu). Dacă la un moment dat se va alege migrarea bazei de date, structura acesteia poate fi recreată doar prin pornirea aplicației conectată la noua bază de date, iar Liquibase va observa că niciun script nu a fost rulat (nu mai există tabela în care Liquibase salva scripturile rulate) și le va rula din nou.

Fiecare modificare asupra bazei de date este salvată într-un fișier diferit, după cum se poate observa în imaginea de mai jos:

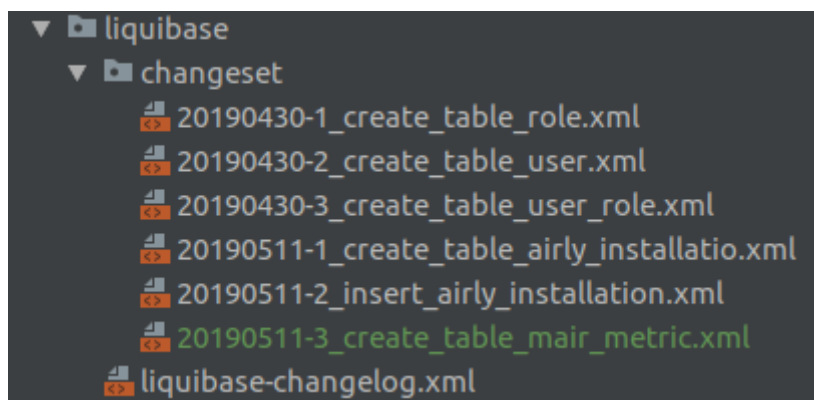


Figura 7. Fișierele Liquibase

Securitatea aplicației server este asigurată la nivel de endpoint API: orice apel la unul dintre endpoint-uri trebuie să conțină în header un nume de utilizator și o parolă pentru autentificare. La nivelul aplicației server există câteva roluri predefinite pentru aplicația web și pentru prototipul hardware. Fiecare apel făcut de una dintre cele două componente va

conține în header numele de utilizator și parola corespunzătoare. Pentru aplicația web, rolul asigurat este doar de citire de date, pe când prototipul hardware are și rol de scriere, putând crea intrări noi în baza de date. Rolurile sunt salvate în baza de date, alături de numele de utilizator și de parolă. Parola este criptată folosind *BCryptPasswordEncoder* disponibil în framework-ul Spring. [3]

În imaginea de mai jos se poate observa metoda care configurează securitatea aplicației. Este acceptat orice apel HTTP către aplicație, însă inițiatorul (specificat în header prin nume de utilizator și parolă) va trebui să dețină rolul specificat pe endpoint-ul apelat pentru a fi autorizat. În caz contrar, aplicația va răspunde cu 401 Unauthorized sau 403 Forbidden.

```
@Override
protected void configure(HttpSecurity http) throws Exception {

    http
        .authorizeRequests() ExpressionInterceptUrlRegistry
        .anyRequest() ExpressionUrlAuthorizationConfigurer<HttpSecurity>.AuthorizedUrl
        .permitAll() ExpressionUrlAuthorizationConfigurer<HttpSecurity>.ExpressionInterceptUrlRegistry

        .and() HttpSecurity
        .httpBasic() HttpBasicConfigurer<HttpSecurity>

        .and() HttpSecurity
        .csrf() CsrfConfigurer<HttpSecurity>
        .disable();
}
```

Figura 8. Configurarea securității

Pentru a furniza măsurători în timp real, aplicația server trebuie să actualizeze periodic datele stocate în baza de date. Această operație se realizează cu ajutorul unor taskuri programate (Scheduled Task). Un task programat este o metodă adnotată cu *@Scheduled* care va rula periodic, la un interval de timp prestabilit. Spre exemplu, în aplicația noastră, există două astfel de taskuri: unul pentru a interoga API-ul Airly și unul pentru rețeaua de senzori propusă de această lucrare. Taskul programat pentru API-ul Airly este prezentat în imaginea de mai jos. Perioada la care este executat este specificată în fișierul de configurare (momentan rulează la fiecare oră); de asemenea este specificată și o întârziere inițială (după ce aplicația este pornită se va aștepta un număr de secunde până ce va porni primul task) pentru a evita diferite erori ce pot apărea. Dacă se dorește modificarea intervalului la care să ruleze acest task, se poate modifica fișierul de configurare *application.yml*. Pentru ca schimbările să aibă loc, este necesar doar să restartăm aplicația, nefiind necesară compilarea, fiindcă fișierul modificat este un fișier extern de tip text.

În momentul în care un task programat este lansat în execuție, acesta rulează pe un fir de execuție nou. Astfel, chiar dacă un task durează o perioadă mai lungă de timp, acesta nu va afecta rulările ulterioare ale aceluiași task.

```

@Override
@Scheduled(fixedRateString = "${airly.job-frequency}", initialDelayString = "${airly.initial-delay}")
public void executeJob() {
    List<AirlyInstallation> installations = airlyInstallationRepository.findAll();

    installations.forEach(this::executeJobForInstallation);
}

```

Figura 9. Airly Scheduled Task

Un alt task programat necesar aplicației este responsabil de ștergerea datelor mai vechi de 24 de ore. Fiindcă soluția prezentă își propune să ofere date în timp real utilizatorilor și fiindcă utilitatea acestor măsurători scade odată cu trecerea timpului, necesitatea stocării tuturor datelor pe o perioadă mai mare de 24 de ore devine nejustificată. Desigur, un al doilea argument în detrimentul salvării pe o perioadă mai îndelungată a datelor este performanța bazei de date: o bază de date încărcată va funcționa și reacționa mult mai greu (cu întârziere) în comparație cu una a cărei încărcări este menținută în anumite limite. Datele mai vechi de 24 de ore pot fi agregate și pot contribui la diferite statistici, însă nu mai sunt relevante la nivel de oră, ci poate la nivel de zi, săptămână sau chiar lună.

Toate endpoint-urile din aplicația server pot fi apelate și din Swagger. Swagger este un proiect open-source folosit pentru a documenta API-uri de tip REST. Pe lângă documentație, Swagger UI permite dezvoltatorilor sau utilizatorilor să interacționeze cu resursele API-ului.

Astfel, toate resursele puse la dispoziția utilizatorilor de către aplicația server pot fi testate sau apelate din Swagger UI, o interfață prin intermediul căreia se pot genera cereri către orice endpoint disponibil, având posibilitatea de a specifica valori pentru parametri (dacă este cazul). Desigur, și din această interfață va fi nevoie de autorizare pentru a accesa o anumită resursă securizată. Câteva exemple din interfața Swagger a aplicației sunt disponibile în anexa acestei lucrări.

În cele ce urmează vom prezenta un exemplu real din aplicația server. Endpoint-ul *api/metric/get-latest-measurements/* va oferi ca răspuns o listă de metrice de tipul specificat, măsurate la data cea mai apropiată de cea a apelului. Tipul metricei dorite se poate specifica prin setarea parametrului *measurementType* (spre exemplu, un URL valid ar arăta astfel: *api/metric/get-latest-measurements?measurementType=pm10*; acesta ar întoarce o listă de metrice de tipul PM10). Acest endpoint este securizat: pentru a putea fi apelat, în header-ul cererii vor fi necesare credențialele echivalente unui rol existent în aplicație, iar rolul deținut să corespundă celui configurat la nivelul endpoint-ului. În exemplul prezentat este necesar ca apelantul să aibă rol de citire pentru a avea acces la acest endpoint. Se pot specifica diferite descrieri pentru operațiile disponibile în Swagger UI, cum ar fi tipul valorii returnate, tipul parametrilor sau autorizația necesară realizării unui apel.

```

@GetMapping("/get-latest-measurements")
@ApiOperation(value = "Find last measurements",
    produces = "List of measurements",
    response = List.class,
    authorizations = {@Authorization(value = "basicAuth")})
@PreAuthorize(SecurityConstants.HAS_READ_ROLE)
public ResponseEntity getCaqiForHeatmaps(@RequestParam MeasurementType measurementType) {
    List<MairMetric> metrics = metricService.findAllBy(measurementType, LocalDateTime.now());

    return ResponseEntity.ok().body(metrics);
}

```

Figura 10. Exemplu endpoint

4.2 Aplicația web

Interacțiunea cu utilizatorii este un obiectiv important al acestui proiect. Așa cum am amintit în primul capitol, soluția propusă are ca mijloc de interacțiune cu utilizatorii o a doua componentă constitutivă, și anume o aplicație web ușor de folosit, cu o interfață user-friendly și cu utilitate ridicată. Utilitatea acestei aplicații constă tocmai în faptul că locuitorii orașului București o pot accesa la orice moment de timp, aceasta oferind informații actualizate cu privire la nivelul calității aerului și posibilitatea incorporării unor funcționalități în viața de zi cu zi, cum ar fi recomandarea unei alternative pentru o rută introdusă de utilizator pe baza calității aerului de pe parcursul acesteia.

Tehnologiile folosite pentru dezvoltarea aplicației web sunt: Angular 6, Typescript, HTML, CSS. Angular este un framework open-source menit să îmbunătățească codul HTML în browser, creând o fundație care simplifică dezvoltarea de aplicații puternice. Proiectul este dezvoltat de Google [6]. Aplicațiile construite cu ajutorul Angular au în spate, de obicei, un model arhitectural de tip MVC (Model-View-Controller).⁴

Aplicația este formată din trei pagini: Dashboard, Map, Wiki. În continuare le vom prezenta pe rând, menționând funcționalitățile fiecăreia și cum se realizează legătura cu aplicația server sau alte aplicații externe.

Metrica cea mai importantă pe baza căreia sunt expuse datele este AQI (Air Quality Index). Indexul calității aerului este calculat în funcție de nivelul poluanților disponibili (în cazul nostru, în funcție de PM2.5, PM10 și CO; alți poluanți care pot fi considerați în viitor ar fi: SO2, NO2 și COV - Compuși Organici Volatili). Fiecare poluant este încadrat în anumite limite pentru a putea corespunde nivelelor AQI (0 - 25, 25 - 50, 50 - 75, 75 - 100, gt100). Mai multe despre aceste limite se pot observa în tabelul următor.

⁴ Freeman, Adam. (2018). Pro AngularJS.

Index class	Grid	Traffic						City Background							
		core pollutants			pollutants			core pollutants				pollutants			
		NO2		PM10	PM2.5		CO	NO2	PM10	O3		PM2.5	CO	SO ₂	
		1-h.	24-h.		1-h.	24-h.		1-h.	24-h.			1-h.	24-h.		
Very low	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	25	50	25	15	15	10	5000	50	25	15	60	15	10	5000	50
Low	25	50	25	15	15	10	5000	50	26	15	60	15	10	5000	50
	50	100	50	30	30	20	7500	100	50	30	120	30	20	7500	100
Medium	50	100	50	30	30	20	7500	100	50	30	120	30	20	7500	100
	75	200	90	50	55	30	10000	200	90	50	180	55	30	10000	350
High	75	200	90	50	55	30	10000	200	90	50	180	55	30	10000	350
	100	400	180	100	110	60	20000	400	180	100	240	110	60	20000	500
Very High*	> 100	> 400	>180	>100	> 110	>60	>20000	> 400	>180	>100	>240	> 110	>60	>20000	>500

Figura 11. Tabel limite AQI

Pagina *Dashboard* este formată din mai multe diagrame. Prima și poate cea mai relevantă este diagrama AQI (Air Quality Index). Aceasta prezintă, într-o diagramă liniară, calitatea aerului de-a lungul a 24 de ore. După cum se poate observa și în legenda diagramei, un index cuprins între 0 și 25 este marcat în diagramă de culoarea verde și de caracteristica *Great* (aerul este foarte bun), între 25 și 50 culoarea galben și caracteristica *Good* (aerul este bun), între 50 și 75 culoarea portocaliu și caracteristica *Bad* (aerul nu este bun), între 75 și 100 culoarea roșu și caracteristica *Harmful* (aerul este foarte dăunător).



Figura 12. Exemplu diagramă

Pe lângă această diagramă, în pagina *Dashboard* mai sunt disponibile diagrame pentru poluanți precum PM2.5, PM10, CO și pentru temperatură. Toate aceste diagrame prezintă date obținute prin apeluri către aplicația server și măsurate din 2 în 2 ore. Pe axa X se poate observa ora la care o măsurătoare a avut loc, iar pe axa Y nivelul respectivei metrici.

Toate aceste metrice disponibile în pagina *Dashboard* sunt grupate pe zone. Spre exemplu, zonele disponibile momentan pentru astfel de grafice sunt Piața Unirii, Universitate și Piața Victoriei. În momentul în care se alege o altă zonă, se fac cereri către aplicația server pentru a oferi date specifice zonei selectate și diagramele vor fi actualizate în momentul în care serverul va oferi datele interogate. Soluția țintă ar fi ca utilizatorul să poată alege orice punct de pe harta Bucureștiului și astfel să fie afișate diagrame pentru un punct ales în mod dinamic, așa cum vom menționa și în ultimul capitol.

Datele pe baza cărora se trasează aceste grafice sunt obținute prin apeluri la aplicația server. Pentru fiecare tip de metrică dorit se realizează un apel asincron către un endpoint care poate fi parametrizat cu tipul metricii, latitudinea și longitudinea, data și ora. Un apel asincron inițiază o cerere HTTP către server, însă nu așteaptă ca acesta să răspundă. În schimb, se oferă o metodă care să fie apelată în momentul în care răspunsul este disponibil. În cazul de față, atunci când serverul răspunde cu lista de măsurători interogată, datele graficelor sunt actualizate cu cele primite ca răspuns. Dacă din anumite motive serverul nu răspunde în timp util sau răspunde cu un cod de eroare (spre exemplu 403 Forbidden) pentru o anumită cerere, celelalte funcționalități ale paginii vor fi în continuare disponibile utilizatorului, nealterând funcționarea aplicației.

Cea de-a doua pagină a aplicației, pagina *Map*, este orientată mai mult spre utilizatori. Mai exact, cea de-a doua pagină își propune să rezolve o problemă care se poate rezuma prin următoarea întrebare: Prin ce zonă ar fi mai bine să merg? În cele ce urmează vom prezenta soluția propusă.

Toate soluțiile existente pe piață informează utilizatorii cu privire la nivelul calității aerului din anumite zone: Airly - prin plasarea unei culori în locația senzorilor în funcție de nivelul de poluare pe o hartă a orașului București, alte dispozitive prin măsurarea de către utilizator din locația acestuia și în momentul respectiv. Acest obiectiv îl atinge și soluția propusă în această lucrare prin pagina *Dashboard*. De asemenea, în pagina *Map* utilizatorii pot consulta o hartă a capitalei cu un strat reprezentând nivelul calității aerului, după cum se poate observa și în exemplul de mai jos.



Figura 13. Hartă cu indicatori pentru poluare

Fiecare zonă colorată reprezintă locația unui senzor din rețeaua de senzori propusă în această lucrare. Datele sunt oferite de către aplicația server și reprezintă măsurătorile de tip AQI (Air Quality Index) cele mai recente. În funcție de valoarea AQI, un punct de interes va fi colorat corespunzător (spre exemplu, pentru valori între 0 și 25, punctul va fi colorat cu verde, pentru valori între 25 și 50 cu galben și așa mai departe).

Pentru a răspunde la întrebarea adresată anterior, soluția oferită de această aplicație este ca, pentru o rută introdusă de utilizator, să recomande o alternativă mai puțin poluată. Pentru a atinge acest obiectiv am fi putut folosi o hartă normală, proprie, pe care să evidențiem ruta introdusă și câteva alternative propuse de aplicație. Însă, fiindcă în acest fel nu am fi avut acces la alte informații cu privire la trafic, străzi blocate sau în construcție și altfel de evenimente neprevăzute pe parcursul traseului, am ales să folosim serviciile oferite de Google în acest sens.

Decizia de a folosi serviciile Google se bazează în primul rând pe facilitățile pe care acesta le oferă, și anume accesul la informații cu privire la trafic. Un al doilea motiv, poate la fel de important ca primul, pentru folosirea acestor servicii este chiar popularitatea Google în rândul utilizatorilor, astfel fiind un serviciu familiar și ușor de folosit și de asimilat în cadrul unei alte aplicații (cum este cea prezentă). Desigur, folosirea acestor servicii aduc cu sine și anumite limitări, pe care le vom discuta mai târziu în acest capitol.

Utilizarea hărților pentru vizualizarea diferitelor elemente este un lucru natural în ziua de azi. Folosim hărțile pentru a vedea locația unor lucruri, pentru a căuta o adresă, pentru a obține rute de navigare și pentru multe alte lucruri. Cele mai multe informații au o locație, iar dacă ceva are o locație, poate fi vizualizat pe o hartă. Cea mai populară soluție în acest sens este Google Maps și, pentru a putea fi incorporată și în alte aplicații, cei de la Google au creat un API deschis pentru a putea folosi funcționalitățile Google Maps. [4]

Primul serviciu Google folosit în aplicație este *Place Autocomplete*, un serviciu web care returnează predicția unei locații pe baza unei cereri în care este specificat un câmp textual și, opțional, limite geografice în care să se încadreze răspunsurile.

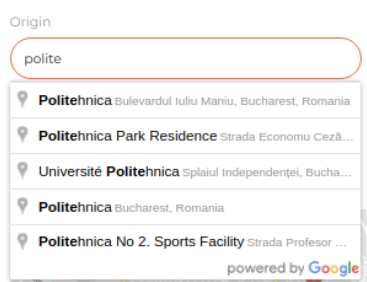


Figura 14. Place Autocomplete

În cazul de față, am conectat acest serviciu la căsuțele de căutare a destinației și a originii (de unde începe ruta și unde se termină). De asemenea, am limitat răspunsurile la locații din România. Astfel, la fiecare caracter introdus în oricare dintre cele două căsuțe se realizează o cerere către *Place Autocomplete* conținând textul introdus până în momentul respectiv și se primește ca răspuns predicții de locații pe baza cererii.

Utilizatorul are posibilitatea de a opta pentru folosirea locației curente bifând căsuța specifică. În acest caz, aplicația va necesita drepturi pentru folosirea locației actuale din browser, pe care le va cere în mod dinamic. Se poate reveni în orice moment la selectarea manuală a unei locații de origine, debifând căsuța *Use current location*.

După selectarea unei locații de origine sau de destinație, pe hartă va apărea un marker indicând locația dorită. Acesta poate fi mutat manual de către utilizator prin drag-and-drop oriunde pe hartă. După ce au fost validate cele două locații, se va debloca butonul *Compute route*. Apăsând pe acest buton, aplicația va folosi cel de-al doilea serviciu Google, și anume Google Directions.

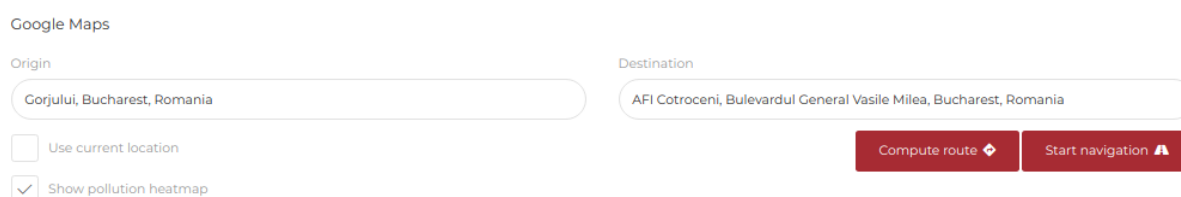


Figura 15. Google Maps User Input

Google Directions API este un serviciu care calculează ruta dintre două locații. În cazul de față, aplicația va realiza un apel la acest serviciu pentru a obține ruta dintre origine și destinație. Prin setarea parametrului *alternatives* pe *true*, serviciul este capabil să returneze mai multe alternative pentru ruta dorită în mod dinamic.

Una dintre limitările Google cu privire la serviciile sale este că nu se poate naviga către destinația introdusă din aplicații externe (cum este aplicația de față). Singura soluție pentru această limitare este să redirectăm utilizatorul către aplicația oficială Google, oferind ca parametri ruta introdusă și câteva puncte de referință de-a lungul rutei pentru a putea reproduce aceeași rută. Dacă se transmite ca parametru doar destinația dorită, ruta se va recalcula în aplicația Google Maps și nu avem nicio siguranță că va fi identică cu cea calculată în aplicația noastră, astfel fiind necesare alte referințe de-a lungul rutei.

În cazul accesării aplicației de pe un dispozitiv mobil ce are Google Maps instalat, atunci când utilizatorul va accesa butonul *Start navigation* i se va oferi posibilitatea să deschidă această rută direct în aplicația nativă Google Maps. O altă limitare care apare în acest proces este numărul de referințe ce pot fi pasate: pentru un cont gratuit această limitare este de 25 de referințe, ceea ce, pentru o rută mai complexă, ar putea fi insuficient.

După ce se realizează apelul către Google Directions API pentru generarea rutelor pe baza traficului din momentul respectiv, aplicația primește ca răspuns aceste rute. Pentru a determina care este cea mai bună alternativă, se calculează pentru fiecare rută un AQI (Air Quality Index) mediu astfel: pentru fiecare punct care formează ruta curentă se caută cel mai apropiat senzor și se adună valoarea AQI măsurată de acesta. La final se împarte valoarea rezultată la numărul total de senzori consultați pentru a rezulta o valoare medie. În cazul în care nu este disponibil niciun senzor în limita a 1 kilometru, se trece la următorul punct din rută fără a aduna nicio valoare pentru punctul curent. Dintre toate rutele oferite de Google Directions API se va alege ca rută recomandată cea cu valoarea AQI mai mică. Aceasta va fi indicată pe hartă prin culoarea albastru și va apărea mesajul “Vă recomandăm să folosiți ruta evidențiată”.

Alături de ruta recomandată, pe hartă vor fi disponibile și celelalte rute colorate cu gri. Utilizatorul poate selecta o altă rută, după preferință, selectând ruta dorită. Aceasta se va colora cu albastru și va deveni ruta principală.

Mai multe informații despre ruta principală la un moment dat pot fi consultate în colțul stânga-sus al hărții. Acolo pot fi găsite informații precum: distanța și durata rutei, valoarea medie AQI și calitatea aerului. Selectând o altă rută, aceste informații se vor actualiza cu datele rutei dorite.

În continuare vom prezenta un exemplu pentru ruta Gorjului – AFI Cotroceni.

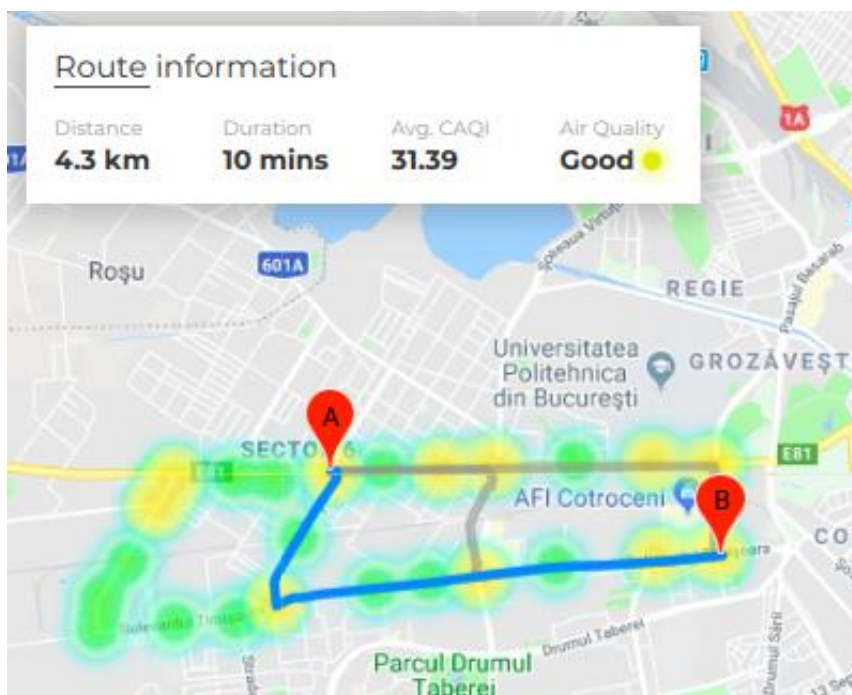


Figura 16. Exemplu rută

Pentru a ajunge la destinație, Google Directions ne oferă trei variante: prin Bulevardul Iuliu Maniu, prin Strada Moinești și pe lângă Plaza România. Cu toate că varianta prin Bulevardul Iuliu Maniu este cea mai scurtă ca distanță, aceasta este de multe ori foarte aglomerată. Dat fiind numărul mare de mașini care tranzitează acest bulevard și intersecțiile aferente, valoarea calității aerului începe să se degradeze. Pe de altă parte, Bulevardul Timișoara fiind tranzitat de mai puține autoturisme are o valoare mai scăzută a indicelui calității aerului.

Așadar, aplicația ne recomandă să folosim ruta Gorjului – Moinești – Bulevardul Timișoara – AFI Cotroceni pe care vom avea o valoare AQI destul de scăzută, mai exact 31.39, cu toate că această rută este cea mai lungă (4.3km).

Vom observa, prin selectarea pe rând a celorlalte alternative, că valoarea AQI a rutei recomandate este, într-adevăr, cea mai mică, cu toate că celelalte rute sunt mai rapide sau au o distanță mai mică. Timpul pe care l-am câștiga selectând o alternativă mai poluată ar însemna deteriorarea sănătății pe termen lung. Astfel, un minut în plus în schimbul unui aer mai curat devine un schimb atrăgător.

4.3 Soluția hardware

Din punct de vedere hardware, rețeaua de stații pentru monitorizarea calității aerului pe care o propunem este formată din două dispozitive: unul intermediar, pentru a intercepta cererile de date de la aplicația server și care le va înainta către cel de-al doilea dispozitiv (dispozitiv final, stația de monitorizare), care este conectat la senzorii doriți și care va efectua măsurătorile, întorcând datele dispozitivului anterior.

După cum am amintit în capitolul *3-Soluții existente*, una dintre principalele probleme ale rețelelor existente este dimensiunea redusă. Pentru ca aplicația propusă să funcționeze corespunzător va fi necesară o rețea de stații de monitorizare dezvoltată care să acopere tot orașul București (sau cel puțin zona centrală și zonele periferice mai aglomerate). În acest sens, este necesar ca în fiecare kilometru pătrat să fie prezentă o stație pentru monitorizare.

După cum se poate vedea și în figura de mai jos (fiecare pătrat reprezintă aproximativ un kilometru pătrat), pentru a acoperi zonele mai importante ar fi necesare 100 de stații de monitorizare. Pentru a îmbunătăți rezultatele, distanța pentru care este responsabil un senzor ar trebui micșorată și, în consecință, ar fi necesare 150-200 de stații de monitorizare.

Desigur, numărul de stații este variabil, fiind doar o aproximare. În parcuri, spre exemplu, distanța dintre stații poate fi mai mare, pe când în zonele mai aglomerate se poate dubla numărul stațiilor pentru măsurători mai exacte.

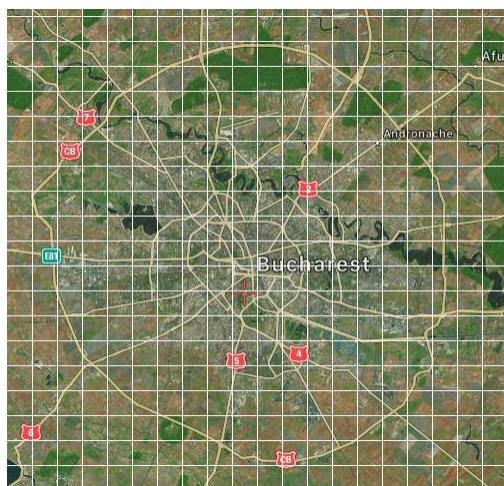


Figura 17. Localizarea instalațiilor

După cum am menționat anterior, rețeaua de stații propusă este formată din două dispozitive. Primul dispozitiv va juca rolul unui server: va primi cereri de la aplicația server, le va înainta către cel de-al doilea dispozitiv pentru a realiza măsurătorile și va răspunde cu date în timp real. La prezentarea aplicației server am menționat că aceasta va interoga rețeaua de senzori pentru a primi date actuale. Dacă aplicația server ar fi interogată în mod direct dispozitivele care gestionează senzorii (să presupunem aproximativ 200 de dispozitive), atunci toată operația de preluare de date de la rețea ar dura semnificativ mai mult.

În plus, pentru fiecare dispozitiv ar fi necesar câte o operație de inserare în baza de date, ceea ce ar putea conduce la alte probleme (de sincronizare, datele sunt actualizate pentru unele zone și pentru altele nu). În schimb, soluția pe care o propunem este ca să existe un alt dispozitiv între aplicația server și dispozitivul final care să comunice cu serverul și să gestioneze un număr de stații de măsurare. Astfel, numărul de operații de inserare în baza de date ar fi semnificativ mai scăzut, iar timpul de citire ar fi sincronizat între dispozitive.

Mai mult, se pot grupa stațiile de monitorizare pe zone (spre exemplu, în Sectorul 6 toate stațiile de monitorizare să fie disponibile printr-un singur dispozitiv intermediar). În acest fel, diferența dintre datele actualizate și cele încă neactualizate (diferența poate apărea, așa cum am menționat anterior, din diferite motive obiective ce țin de senzori și funcționarea acestora) nu ar fi resimțită foarte puternic.

Complexitatea unei astfel de rețele poate să crească sau să scadă în funcție de dimensiunea orașului în care este implementată. Într-un oraș mai mare, se mai poate adăuga un nivel de dispozitive intermediare, astfel responsabilitatea fiind împărțită și favorizând scalabilitatea și eficiența. Într-un oraș mai mic se poate renunța complet la dispozitivele intermediare. Pentru orașul București considerăm că arhitectura potrivită este cu un nivel de 10-15 dispozitive intermediare responsabile pentru 15-20 de stații de monitorizare (dispozitive finale).

Aplicația server va comunica doar cu dispozitivele intermediare (evidențiate cu galben în diagrama de mai jos), acestea la rândul lor comunicând cu dispozitivele finale pentru a realiza măsurători (evidențiate cu verde în diagrama de mai jos).

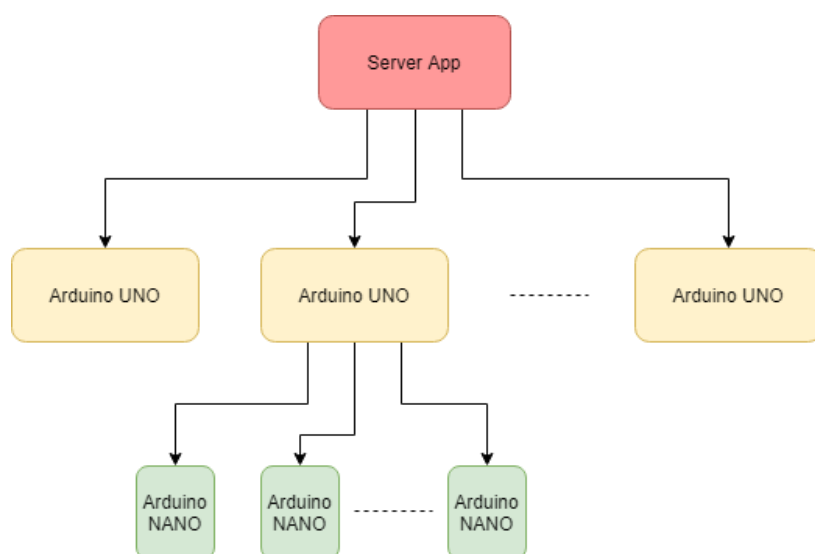


Figura 18. Arhitectura rețelei de stații

Dispozitivele intermediare vor fi construite cu ajutorul plăcuței de dezvoltare Arduino UNO (sau altă alternativă asemănătoare) și vor avea în componență două module de comunicare: unul pentru comunicarea cu aplicația server și unul pentru comunicarea cu dispozitivele finale. Aceste dispozitive intermediare se pot comporta și ca dispozitivele finale, legând senzorii disponibili, astfel economisind un număr de stații de monitorizare.

Dispozitivele finale au responsabilitatea de a oferi măsurători la cerere. Acestea vor fi compuse dintr-un modul de comunicare (pentru a comunica cu dispozitivul intermediar aferent) și senzorii doriți: senzor pentru monoxid de carbon, pentru particule PM2.5 și PM10 și pentru temperatură și umiditate.

Pentru a nu risipi energie atunci când dispozitivele nu sunt apelate, senzorii pot fi menținuți într-o stare de stand-by. În funcție de caracteristicile senzorilor, va fi nevoie de un apel înainte de a se realiza măsurătorile efective pentru a porni și pentru a se calibra senzorii (unii senzori pot necesita 1-3 minute pentru a se calibra și pentru a oferi date relevante din momentul în care sunt pornite).

Din punct de vedere al localizării, dispozitivele vor putea fi dispuse de-a lungul bulevardelor și străzilor, dat fiind faptul că de aici provine principala sursă de poluare. În cazul parcurilor, frecvența stațiilor de monitorizare poate fi scăzută, considerând fluctuațiile mici ale poluanților din aceste zone.

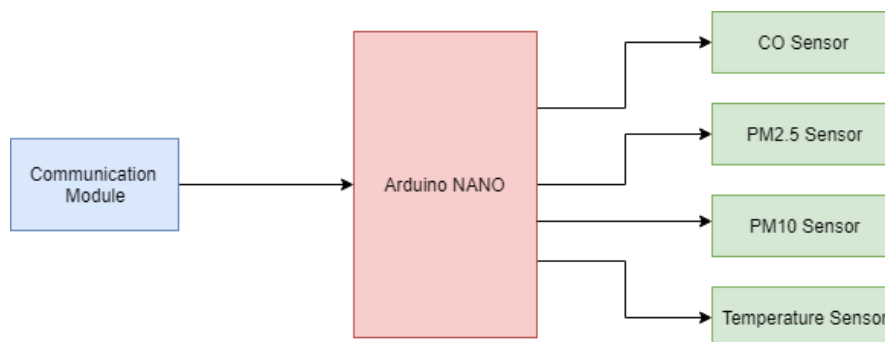


Figura 19. Dispozitiv final

Având toate componentele soluției propuse descrise, putem rezuma un flux al aplicațiilor astfel: aplicația server rulează task-uri programate pentru a colecta date în timp real din cele două surse: din propria rețea de stații pentru monitorizarea calității aerului, unde va înainta o cerere către dispozitivele intermediare. Acestea, la rândul lor, vor înainta cereri către dispozitivele finale conectate la acesta pentru a realiza măsurători. După ce va primit răspuns de la toate stațiile, va întoarce datele primite aplicației server. Aceasta din urmă le va salva în baza de date proprie. Cea de-a doua sursă pentru date este API-ul Airly, de unde vor fi primite date în timp real din rețeaua de stații Airly și care vor fi, de asemenea, salvate în baza de date proprie.

În momentul în care un utilizator se conectează la clientul de web, acesta din urmă va înainta cereri pentru date actuale către aplicația server și le va afișa utilizatorului (în diagrame sau pe hartă). În final, aplicația web va realiza, la cererea utilizatorului, apeluri către Google Maps API, Directions API sau Autocomplete API pentru a genera alternative mai puțin poluate pentru o rută anume.

5 EVALUAREA REZULTATELOR

În acest capitol vom evalua pe rând funcționalitățile aplicației, menționând acolo unde este cazul ce merge și ce nu merge conform așteptărilor, urmând ca în capitolul următor să prezentăm pașii următori, ce s-ar putea dezvolta în continuare pentru a îmbunătăți aplicația.

Din punct de vedere al interfeței cu utilizatorul, putem spune că aplicația s-a ridicat la așteptările inițiale: interfața grafică este simplă, curată, ușor de utilizat și de înțeles elementele componente ale vizualului. Aplicația poate fi asimilată rapid în viața de zi cu zi a utilizatorilor, în special datorită integrării serviciilor Google Maps, servicii care sunt folosite zilnic de un număr semnificativ de oameni.

Din punct de vedere al informațiilor disponibile utilizatorilor, rezultatele sunt satisfăcătoare, însă se pot îmbunătăți anumite aspecte. Așa cum am prezentat în capitolul anterior, în pagina *Dashboard* sunt afișate informații cu privire la calitatea aerului din anumite zone din București, sub forma unor grafice: unul pentru AQI (Air Quality Index), câte unul pentru fiecare poluant măsurat pentru acea zonă. Limitarea acestei funcționalități este faptul că nu se poate selecta decât o zonă prestabilită dintr-o listă de zone. În contrast, în aplicația Airly (prezentată anterior) se poate selecta orice stație de monitorizare pentru a vizualiza informațiile oferite de acea stație. Similar, în aplicația prezentă se poate dezvolta o funcționalitate prin care utilizatorul să poată selecta dinamic zona de interes.

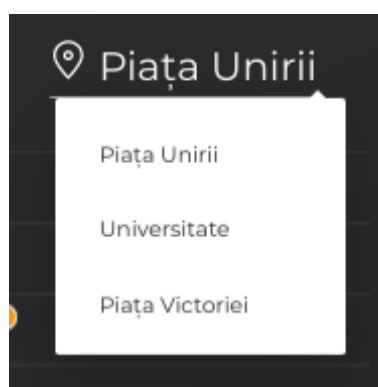


Figura 20. Selectarea unei zone

Informații cu privire la calitatea aerului din anumite zone se pot vizualiza și în cea de-a doua pagină a aplicației, pagina *Maps*. În această pagină este disponibilă o hartă (Google Map) peste care este afișat un strat colorat reprezentând nivelul de poluare din zonele unde sunt instalate stațiile pentru monitorizare. Această hartă este un instrument bun pentru asimilarea informațiilor mai bine (în comparație cu Rețeaua Națională, care doar prezintă locația senzorilor, sau dispozitivele care pot fi cumpărate în particular, care oferă informații doar despre zona curentă), însă nu oferă funcționalități precum posibilitatea de a vedea nivelul AQI (Air Quality Index) pentru o anumită stație de monitorizare (în comparație cu Airly, unde acest lucru este posibil).



Figura 21. Vizualizarea hărții cu un strat colorat pentru poluare

Funcționalitatea cea mai importantă a acestei aplicații se află în pagina *Map*. Rezultatele acestora sunt foarte satisfăcătoare și ne motivează să continuăm să dezvoltăm acest produs. Ceea ce aduce în plus față de soluțiile existente, prezentate în capitolele anterioare, este posibilitatea utilizatorului de a introduce o rută de interes și de a primi, în funcție de nivelul poluării, alternative mai puțin poluate.

În continuare vom prezenta un exemplu pentru a putea evidenția utilitatea acestei funcționalități. Dorim să parcurgem ruta Piața Gorjului – AFI Cotroceni la ora prânzului. La introducerea locației în cele două căsuțe am primit recomandări de locații pe baza textului introdus și am putut selecta cele două locații din alternativele oferite de serviciul Google Autocomplete. Apăsând pe butonul *Compute route*, aplicația a realizat un apel la serviciul Google Directions și a primit trei alternative pentru ruta introdusă: una via strada Moinești, una via bulevardul Iuliu Maniu și una via Plaza România. Pentru fiecare rută s-a calculat valoarea AQI (Air Quality Index) medie astfel: pentru fiecare punct ce formează ruta, s-a căutat cea mai apropiată stație de monitorizare și s-a adunat la valoarea totală a rutei, urmând ca la final să se împartă această valoare la numărul de stații interogate.

Obținând această valoare pentru fiecare rută, aplicația ne-a recomandat cea mai bună rută dintre cele trei, cu valoarea medie AQI cea mai mică: ruta via strada Moinești. Fiindcă tranzitul de mașini pe bulevardul Iuliu Maniu este semnificativ mai mare decât pe celelalte două rute, ne-am fi așteptat la această recomandare, validând astfel corectitudinea datelor.

Selectând pe rând celelalte rute am putut observa, în panoul din stânga-sus, că într-adevăr valoarea AQI cea mai bună este pentru ruta recomandată. Cu toate că ruta recomandată este cea mai lungă din punct de vedere al distanței, diferența nu este notabilă în comparație cu un aer mai curat.

Am putut observa că putem selecta folosirea locației curente și funcționalitatea rămâne neschimbată. De asemenea, am putut ascunde stratul colorat reprezentând nivelul poluării măsurat de stațiile existente pentru a putea observa străzile mai bine.

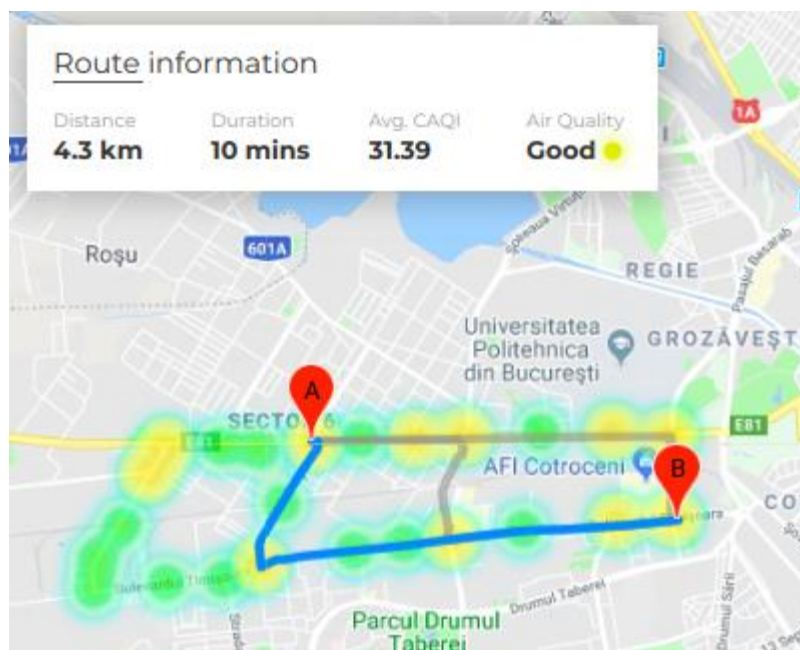


Figura 22. Exemplu rută

În final, selectând ruta recomandată inițial de aplicație, am observat apariția butonului *Start navigation*. Apăsând pe acesta, am fost redirecțiați către aplicația web Google Maps, având ruta generată în aplicația noastră setată și gata de a începe. Am observat mai multe puncte evidențiate pe parcursul rutei, care sunt punctele de referință pentru a menține aceeași rută între origine și destinație.

Realizând același exemplu de pe un dispozitiv mobil, comportamentul a fost identic, interfața grafică s-a adaptat corespunzător și, la apăsarea butonului *Start navigation* am primit posibilitatea de a deschide ruta în aplicația nativă Google Maps instalată pe dispozitiv.

În ceea ce privește baza de date, am obținut o arhitectură performantă și un flux de date eficient. Datele sunt introduse în baza de date la fiecare oră din cele două surse menționate anterior: API-ul Airly și rețeaua proprie de stații. Datele sunt păstrate 24 de ore în baza de date, urmând ca după cele 24 de ore să fie șterse (datele mai vechi de 24 de ore nu mai sunt relevante pentru scopul aplicației). Astfel, baza de date este eficientă și rapidă, menținând întotdeauna un număr minim de intrări în tabele.

Aplicația server s-a dovedit a fi o aplicație puternică, scalabilă, considerând cele două funcționalități majore ale acesteia. Prima, cea de colectare de date, se realizează periodic (la interval de o oră), poate dura de la câteva minute la câteva zeci de minute, în funcție de timpul de răspuns al furnizorilor de date. Considerând arhitectura rețelei proprii de stații de monitorizare, în care aplicația server comunică doar cu o parte dintre dispozitive (mai exact, cu dispozitivele intermediare), timpul de comunicare a scăzut semnificativ. De asemenea, timpul în care aplicația realizează operații de inserare în baza de date a scăzut semnificativ,

crescând astfel eficiența bazei de date și a aplicației, dat fiind faptul că se vor insera un număr mare de date pentru fiecare dispozitiv intermediar, în funcție de numărul de dispozitive finale pentru care acesta este responsabil.

Cea de-a doua funcționalitate, cea de a oferi aplicației web, la cerere, anumite date, este rapidă și scalabilă. Endpoint-urile aplicației sunt rapide, oferă date corecte și pot accepta mai multe cereri simultan. Fiecare apel către un endpoint se realizează pe un fir de execuție separat, astfel fiind limitați în ceea ce privește scalabilitatea doar de mașina pe care rulează aplicația.

Eficiența aplicației server poate fi îmbunătățită prin folosirea unor cache-uri. Fiindcă datele servite de către această aplicație sunt similare pe un interval de o oră, datele pot fi reținute într-un cache. Astfel se evită apelurile la baza de date și răspunsurile vor fi mult mai rapide. În prezent, aplicația realizează apeluri la baza de date, chiar dacă se aduc aceleași date de fiecare dată. Implementarea unui cache la nivel de aplicație server ar îmbunătăți nu numai eficiența aplicației, ci și scalabilitatea și timpul necesar pentru a răspunde unui apel la unul dintre endpoint-urile aplicației.

6 CONCLUZII

În acest capitol vom prezenta obiectivele pe care le-am avut, ce am obținut și cum putem evalua produsul. După cum am amintit în primele capitole, obiectivul acestui proiect a fost crearea unei soluții pentru monitorizarea calității aerului într-un oraș inteligent, soluție care să fie ușor de folosit de către utilizatori și care să ofere date în timp real.

Am reușit să dezvoltăm o aplicație server rapidă, scalabilă, capabilă să colecteze date de la un număr de furnizori, printre care API-ul celor de la Airly și rețeaua proprie de stații de monitorizare a calității aerului. Aplicația este capabilă să colecteze aceste date în mod automat și periodic, la un interval predefinit de timp. Am creat diferite endpoint-uri pentru accesarea datelor colectate după anumite criterii. De asemenea, am securizat aplicația server prin definirea mai multor roluri (de citire, de scriere) necesare pentru autorizarea oricărui apel la endpoint-urile create.

Am obținut un flux de date eficient la nivelul bazei de date prin limitarea timpului la maxim 24 de ore vechime. Dat fiind faptul că obiectivul aplicației a fost să ofere date în timp real cu privire la calitatea aerului, datele mai vechi de 24 de ore nu mai sunt relevante. Astfel, prin ștergerea acestora, baza de date rămâne la încărcare minimă, rezultând în timpi mai rapizi de răspuns.

În ceea ce privește interacțiunea cu utilizatorii, am creat o aplicație web simplistă, ușor de folosit și familiară. Am integrat serviciile Google Maps, Google Directions și Google Autocomplete pentru funcționalitățile pe care acestea le oferă și pentru familiaritatea pe care le aduc aplicației, considerând că aceste servicii sunt larg folosite de către locuitorii orașului București și nu numai. Pe lângă acestea, am obținut o aplicație care oferă date relevante și în timp real privind calitatea aerului în diverse zone unde sunt instalate stațiile pentru monitorizarea diferiților poluanți, precum PM2.5, PM10, CO, SO2, temperatură și umiditate. Mai mult, am creat o pagină dedicată informării populației, în care sunt prezentați poluanții măsurați, sursele de proveniență ale acestora și efectele pe care le au asupra sănătății, limbajul utilizat fiind accesibil utilizatorilor.

În ceea ce privește soluția hardware, am creat o arhitectură scalabilă și eficientă. Rețeaua de stații de monitorizare este amplasată pe suprafața orașului București, fiecare stație fiind montată la maxim un kilometru depărtare de precedentă stație. Astfel, datele colectate de la această rețea acoperă în proporție semnificativă suprafața orașului. Pentru a eficientiza procesul de colectare de date, am propus ca aplicația server să nu comunice direct cu fiecare stație de monitorizare. În schimb, aplicația server înaintează cererea de colectare de date unor dispozitive intermediare, responsabile de comunicarea cu un număr de stații finale, care la rândul lor vor înainta cererea de măsurare către stațiile de monitorizare și vor răspunde cu datele primite de la acestea. Astfel, responsabilitățile sunt împărțite între componentele soluției, crescând scalabilitatea.

În cele ce urmează vom prezenta câteva idei pentru dezvoltare ulterioară. Dat fiind faptul că aplicația are un potențial major pentru un oraș inteligent, există multe modalități de integrare ulterioară și dezvoltare de noi funcționalități.

Un prim pas necesar a fost deja amintit și în capitolele anterioare: dezvoltarea unei funcționalități prin care utilizatorul să poată alege în mod dinamic orice punct de pe harta orașului București despre care să fie afișate informații privind nivelul calității aerului. În momentul actual se poate alege o zonă dintr-o listă prestabilită.

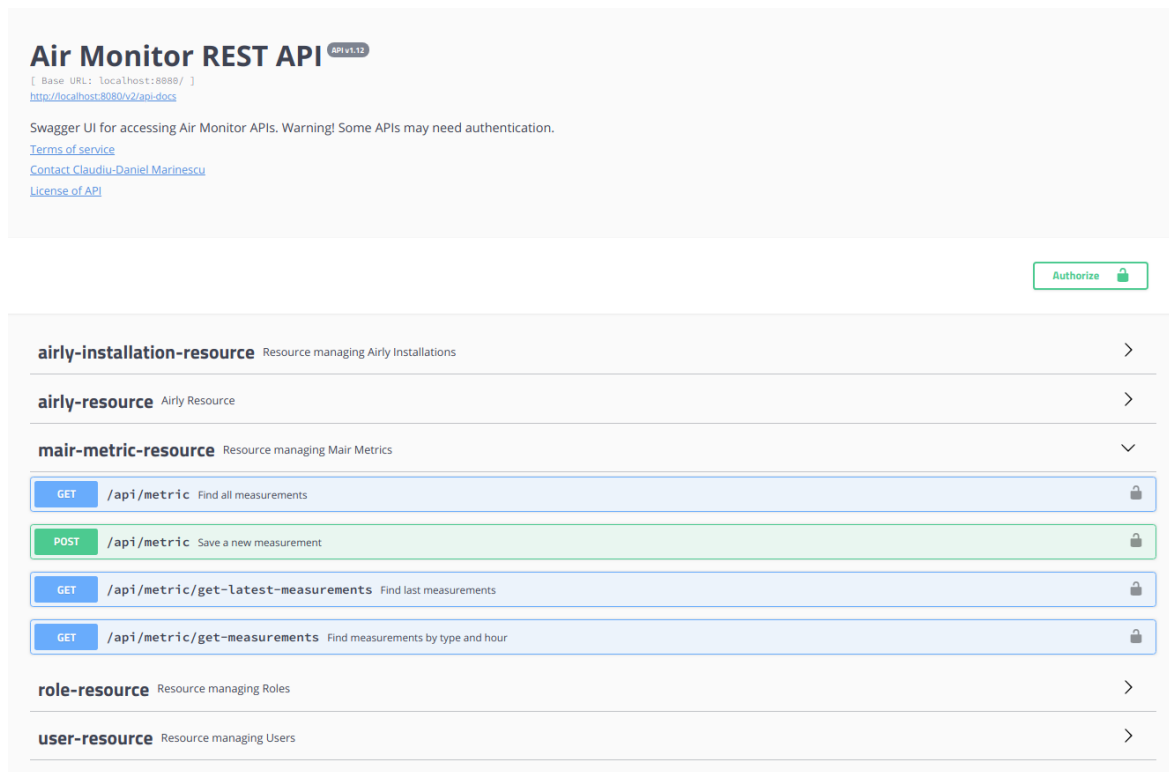
Un al doilea pas ulterior va fi integrarea învățării automate (Machine Learning) în aplicație. Prin antrenarea unui model cu date în timp real colectate de la stațiile de monitorizare, va putea fi dezvoltată o funcționalitate pentru predicția pe următoarele ore a nivelului calității aerului sau a poluanților din diferite zone. Această funcționalitate va crește semnificativ utilitatea aplicației, în special în comparație cu celelalte soluții existente pe piață.

De asemenea, aceeași funcționalitate va putea fi folosită și în procesul de selecție a celei mai bune alternative pentru o rută introdusă de utilizator. Folosind modelul antrenat pentru predicția nivelului calității aerului pe ruta introdusă, aplicația va putea recomanda plecarea la o oră specifică, amânată față de ora inițială de la câteva minute la câteva ore, în funcție de preferința utilizatorului, pentru a beneficia de aer mai curat.

7 BIBLIOGRAFIE

- [1] “Particule în suspensie PM10 și PM2.5”, calitateaer.ro. [Online]. Disponibil: <http://www.calitateaer.ro/public/assessment-page/pollutants-page/pulbere-suspensie-page>. [Accesat 09.05.2019]
- [2] Johnson, Rod, Juergen Hoeller, Keith Donald, Colin Sampaleanu, Rob Harrop, Thomas Risberg, Alef Arendsen et al., “The spring framework–reference documentation”, 2004.
- [3] VARANASI, Balaji; BELIDA, Sudha, “Spring REST”, Apress, 2015.
- [4] SVENNERBERG, Gabriel, “Beginning Google Maps API 3”, Apress, 2010.
- [5] Vallero, Daniel A., “Fundamentals of air pollution”, Academic press, 2014.
- [6] Freeman, Adam, “Pro Angular 6”, Apress, 2018.

8 ANEXE



Anexa 1. Swagger UI

```
spring:
  liquibase:
    change-log: classpath:liquibase/liquibase-changelog.xml
    enabled: true

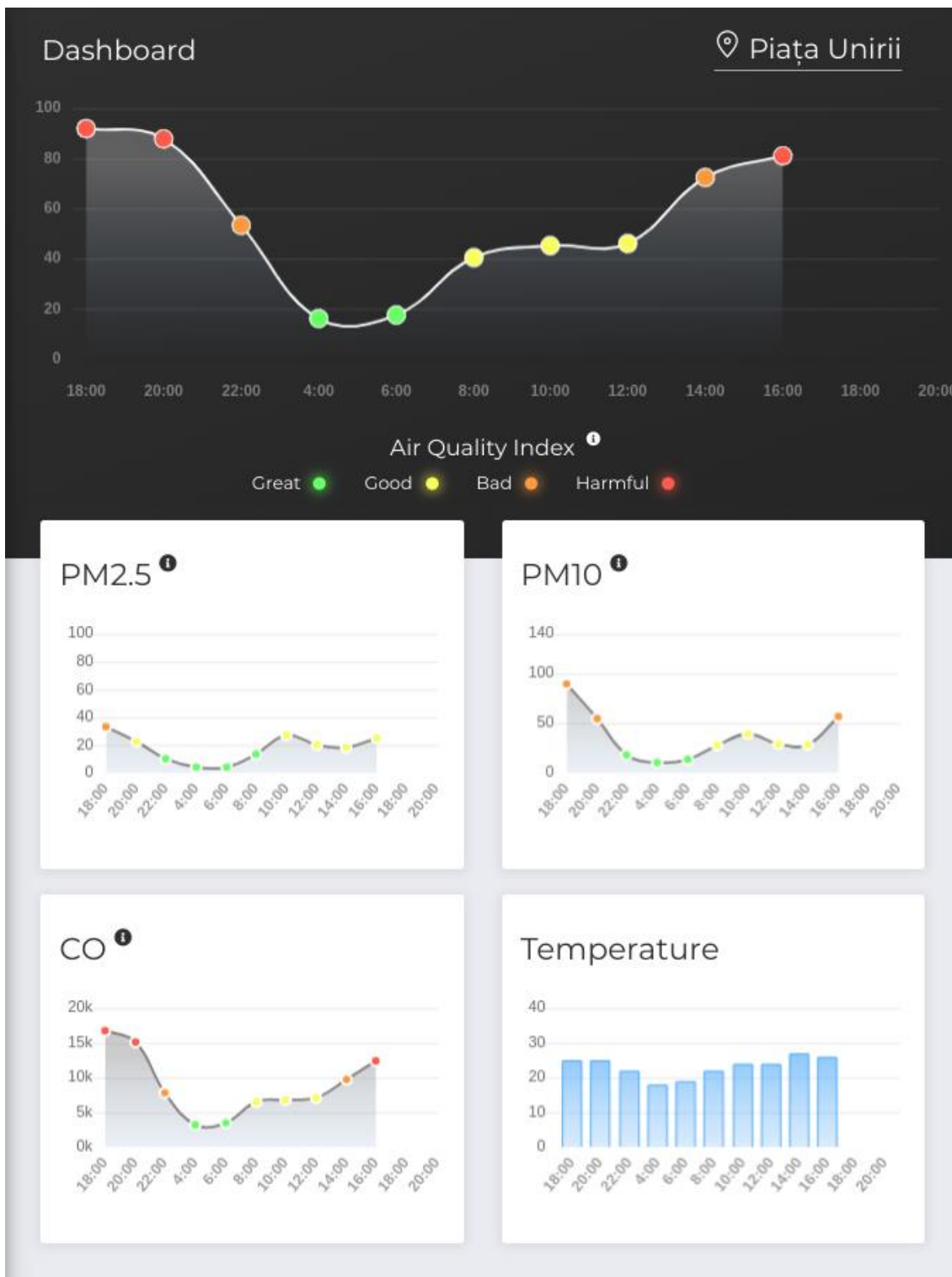
  application:
    name: Air Monitor

  datasource:
    url: jdbc:mysql://localhost:3306/airmonitor_dev?useSSL=false
    username: root
    password: |
    driver-class-name: com.mysql.jdbc.Driver

  jpa:
    hibernate:
      ddl-auto: none
      use-new-id-generator-mappings: false
      show-sql: true

  airly:
    initial-delay: 100_000
    job-frequency: 10_000_000
```

Anexa 2. Fișier de configurare aplicație server



Anexa 3. Pagina Dashboard

Map

Google Maps

Origin

Destination

☐ Use current location

☒ Show pollution heatmap

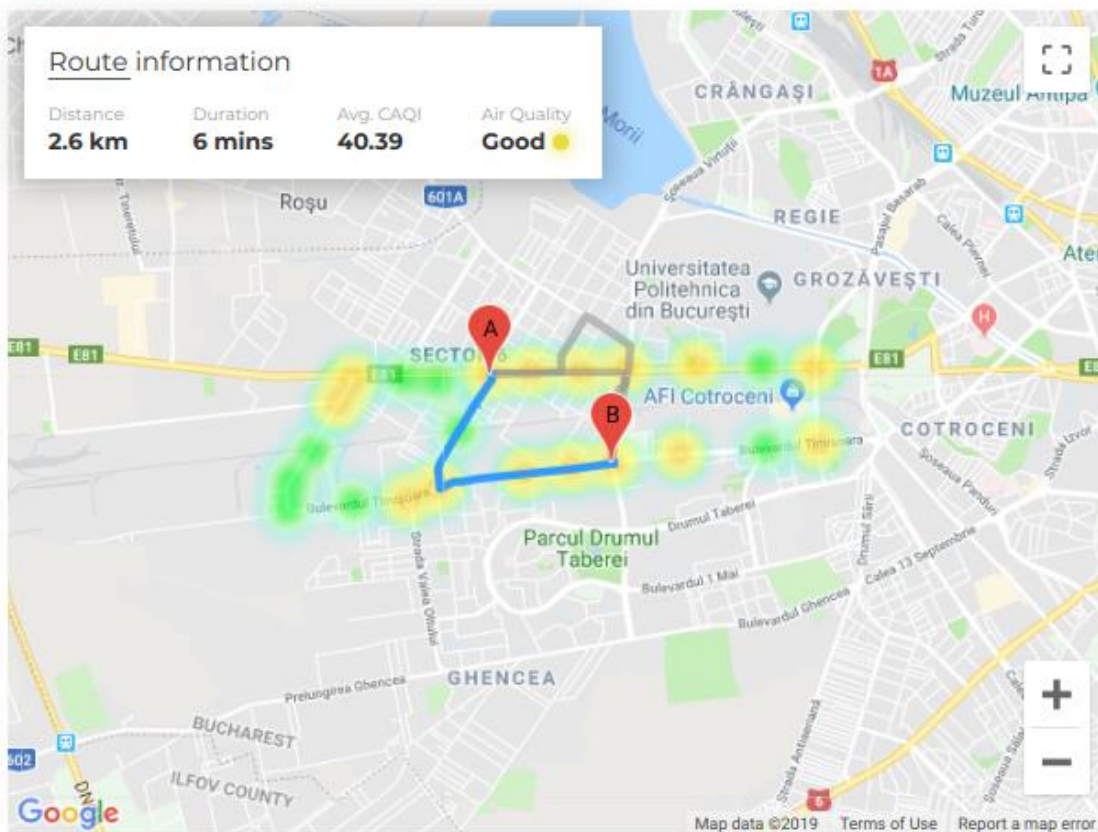
☒ Clear all

Compute route

Start navigation

Route information

Distance	Duration	Avg. CAQI	Air Quality
2.6 km	6 mins	40.39	Good



Anexa 4. Pagina Map