Universitatea Tehnică de Construcții București

Facultatea de Hidrotehnică Domeniul Ingineria Sistemelor

Specializarea Automatică și Informatică Aplicată

**PROIECT DE DIPLOMĂ**

**Aplicatie de mobile pentru identificarea si rutarea statiilor electrice**

Coordonator: Absolvent:

Șl. dr. ing. Ionela Halcu Brinzaru Marius-Andrei

-2020-

CUPRINS

[CAPITOLUL 1. INTRODUCERE 3](#_Toc39756151)

[1.1. Alegerea temei 5](#_Toc39756152)

[1.2.Obiectele lucrarii 8](#_Toc39756153)

[CAPITOLUL 2. Sistemul de operare Android 10](#_Toc39756154)

[2.1Platforma Hardware 10](#_Toc39756155)

[2.2Arhitectura MCP 11](#_Toc39756156)

[2.3 Sistemul de operare 13](#_Toc39756157)

[2.4 Rezumat Android 13](#_Toc39756158)

[CAPITOLUL 3. Aspecte teoretice 14](#_Toc39756159)

[3.1. Perspectiva Server-Side C# .NET CORE 14](#_Toc39756160)

[3.1.1 Tablouri 14](#_Toc39756161)

[3.1.2 Clase 14](#_Toc39756162)

[3.1.3 Drepturi de acces 15](#_Toc39756163)

[3.1.4  Date membru 16](#_Toc39756164)

[3.4.5 Constructori 16](#_Toc39756165)

[3.4.6 Moştenire 16](#_Toc39756166)

[3.4.7. Metode 17](#_Toc39756167)

[3.4.8 Proprietăţi 17](#_Toc39756168)

[3.4.9 Destructori 18](#_Toc39756169)

[3.2 Perspectiva interfata utilizator IONIC 19](#_Toc39756170)

[3.3. Autentificare Auth0 19](#_Toc39756171)

[3.4. Baza de date MonogoDB 19](#_Toc39756172)

[3.5. Algoritmul de rutare Google Directions API 19](#_Toc39756173)

[3.6. Rezultate obtinute 19](#_Toc39756174)

[CAPITOLUL 4. CONCLUZII SI DEZVOLTARI ULTERIOARE 19](#_Toc39756175)

[BIBLIOGRAFIE 19](#_Toc39756176)

[ANEXE 19](#_Toc39756177)

# CAPITOLUL 1. INTRODUCERE

Știința prelucrării informațiilor (datelor) cu ajutorul computerelor se numește informatică. Istoria informaticii începe cu mult timp înainte ca disciplina modernă a informaticii să apară în secolul al XX-lea. Progresia, de la invenții mecanice și teorii matematice până la mașini și concepte moderne, a format un domeniu major academic și baza unei industrii uriașe la nivel mondial.

Cel mai vechi instrument cunoscut pentru utilizarea în calcul este abacul. Se crede că a fost inventat în Babilon circa 2400 î.Hr.. Prima oară era folosit prin trasarea unor linii în nisip cu pietricele. Acesta a fost probabil primul computer și cel mai avansat sistem de calcul cunoscut din aceea perioadă- precedând metoda elenă cu 2000 ani. Abacul este o tăbliță dreptunghiulară, folosită de oameni în antichitate pentru efectuarea calculelor. Abace cu un design modern sunt încă folosite astăzi ca instrumente de calcul.

Dispozitivele mecanice analogice de calcul au apărut din nou o mie de ani mai târziu, în lumea islamică medievală. Exemple de dispozitive din aceasta perioada includ: equatoriumul lui Arzachel (equatoriumul este un instrument astronomic de calcul a pozițiilor aștrilor) sau astrolabul lui Al-Biruni (astrolabul este un instrument astronomic istoric folosit de astronomi, navigatori, și astrologi).

În 1623, Wilhelm Schickard a proiectat o mașină de calcul, dar a abandonat proiectul, deoarece clădirea în care se afla prototipul a fost distrusă de un incendiu în 1624.

În jurul anului 1640, Blaise Pascal, un cunoscut matematician francez, a construit primul dispozitiv mecanic bazate pe un design descris de matematicianul grec Heron din Alexandria.

Apoi, în 1672, Gottfried Wilhelm Leibniz a inventat Socotitorul în trepte care a fost finalizat în 1694. Nici unul dintre aceste dispozitive timpurii de calcul nu au fost calculatoare în sensul modern al cuvântului, și a fost nevoie de un progres considerabil în matematică înainte ca primele calculatoare moderne să poată fi concepute.

În secolul al VII-lea, matematicianul indian Brahmagupta a dat prima explicație a sistemului de numerație hinduso-arab și utilizarea lui zero atât ca substituent cât și ca cifră zecimală. (Un substituent este un element care poate înlocui un alt element, cu care are proprietăți asemănătoare.)

În 1703, Gottfried Leibnitz a dezvoltat logica într-un sens matematic formal, în scrierile sale despre sistemul de numerație binar. În sistemul său, valorile unu și zero reprezintă valorile adevărat și fals (true și false) sau pornit/oprit (on/off). Dar a fost nevoie de mai bine de un secol pentru ca George Boole să publice algebra booleană în 1854 cu un sistem complet care permite proceselor de calcul să fie modelate matematic. (Algebra booleană este o algebră formată din: elementele {0,1}; două operații binare numite SAU și SI, notate simbolic cu + sau Ú și × sau U; și o operație unară numită NU (negație), notată simbolic 0 sau O.

În perioada 1833-1835, Charles Babbage a inventat „mașina analitică”, care se baza direct pe cartelele perforate pentru programare ale lui Joseph Marie Jacquard. Această mașină era alimentată de un motor cu aburi. Babbage nu a reușit să o construiască până la moartea sa, din cauza limitărilor tehnologice ale vremii. Cu toate acestea, o mașină construită în 1991 după schițele sale s-a dovedit a funcționa perfect.

Un computer -numit în limba română și calculator, calculator electronic sau ordinator (din franceză) este o mașină care prelucrează informații (date) pe baza unui program (o listă de instrucțiuni).

Calculatoarele sau computerele pot administra, memoriza, transmite și prelucra o mare cantitate de date într-un timp scurt. Pentru efectuarea unor astfel de operații este necesară o interacțiune complexă între partea fizică sau echipament (în engleză hardware) și partea logică sau programele soft care reprezintă domeniile fundamentale de cercetare în Informatică.

Termenul informatică desemnează știința procesării sistematice a informației, în special a procesării cu ajutorul calculatoarelor.Istoric, informatica s-a dezvoltat ca știință din matematică, în timp ce dezvoltarea primelor calculatoare își are originea în electrotehnică și telecomunicații. Marele avantaj al sistemelor computaționale constă în capacitatea lor de a prelucra în mod schematic cantități enorme de informații la o viteză foarte mare. S-a încercat și implementarea capacităților perceptive ale omului în sistemele informatice, însă până în prezent cu un succes foarte limitat. Un exemplu în această direcție îl constituie sistemele de recunoaștere a chipului uman, sau/și de luare a deciziilor atunci când nu se dispune de toate datele necesare. Astfel de procese sunt studiate de o ramură specializată a informaticii, inteligența artificială. Astfel, în anumite discipline restrânse pot fi obținute deja rezultate remarcabile. Totuși nu se poate încă vorbi despre o modelizare a inteligenței umane.

De-a lungul timpului, formele de comunicare și mijloacele folosite pentru a comunica s-au dezvoltat în funcție de progresul tehnologiei. Modul de comunicare al oamenilor a fost revoluționat de apariția calculatorului, internetului și a telefoniei inteligente. Acesta a reușit să spargă bariera de timp și spațiu, ajutându-i să interschimbe intr-un mod eficient informații. Aparția telefoanelor inteligente ( smartphones) a marcat momentul în care tehnologia a reuțit să conecteze oamenii oriunde și oricând. [1]

În prezent, informatica își găsește aplicații în toate domeniile vieții. Prezența ei este puternic amplificată de impactul pe care îl are internetul. Rețeaua la nivel mondial a revoluționat comunicarea dintre companii, logistica, mass-media, dar și viața privată a fiecărui individ. Mai puțin vizibil, dar totuși omniprezent, informatica și-a câștigat un loc stabil până și în aparatele casnice. Printre altele, telefoanele inteligente, care sunt adevărate calculatoare, au permis democratizarea utilizării informatice pe scară extrem de largă.

Oamenii au necesitatea de a depăși orice prag în privința interrelaționării umane, iar acest lucru este vizibil în evouția vânzărilor e smartphone-uri. Evoluția vânzărilor se poate observa în figura următoare de mai jos.

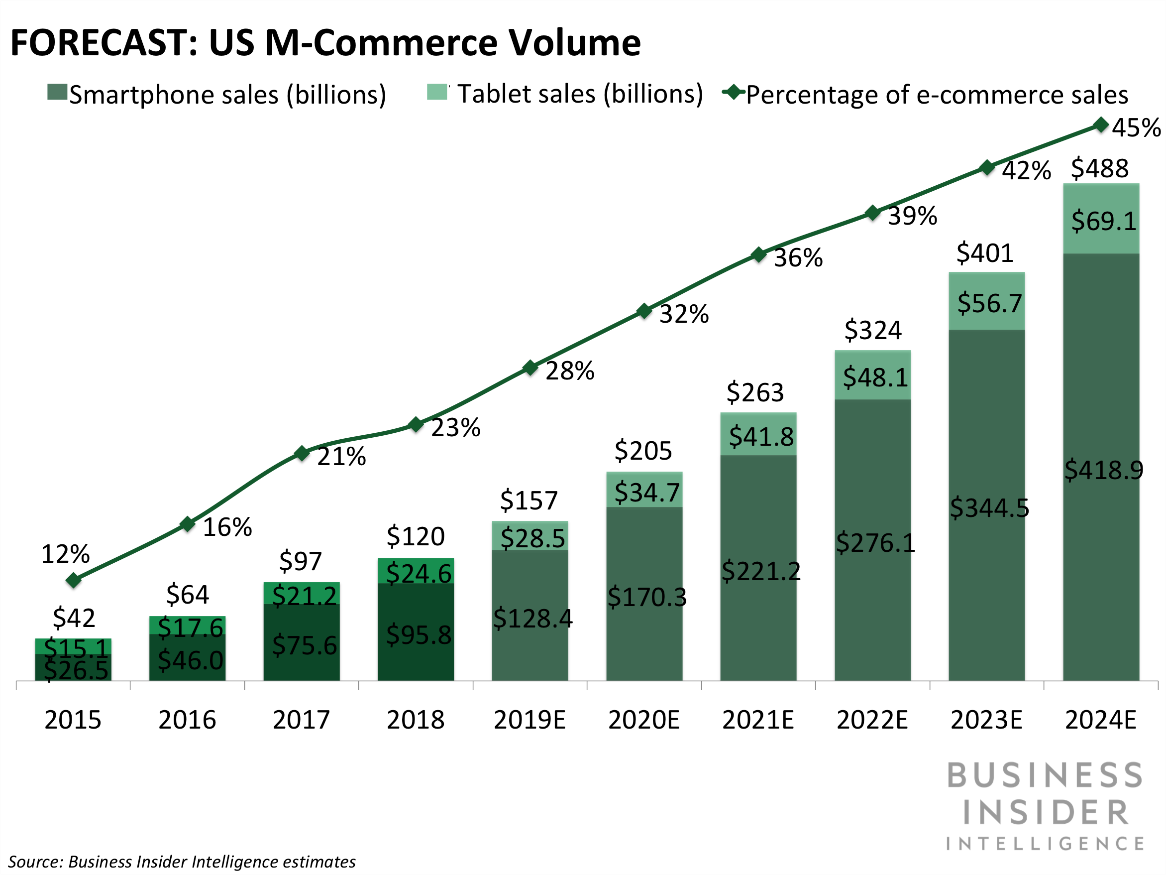


Figura 1.1. Evoluția vânzărilor de smartpone-uri

După cum se poate observa în figura de mai sus vânzărle de smartpone-uri sunt in continuă creștere. Anul 2015 a înregistrat o creștere de 12% față de anul precedent, 2016 a înregistrat o creștere de un procent față de anul 2015, însă anii 2017 și 2018 au înregistrat o creștere de peste 20%. Pentru perioada 2020-2024 se preconizează o creștere de peste 45% pentru vânzarea de smartpone-uri.

## 1.1. Alegerea temei

În prezent aplicațiile de telefon sunt în continuă ascensiune și vor domina piața pentru mult timp de acum încolo. Momentan nu exista nici o aplicație de telefon mobil, fie el Android sau iOS, care să iți permită vizualizarea unui traseu specific de parcurs pentru a ajunge exact la destinația în care se află stația de încarcare electrică.

Tema abordată în cadrul proiectului de diplomă se intitulează “Aplicație de mobil pentru identificarea și rutarea stațiilor electrice**”.**

Unul dintre motivele principale care a adus la alegerea ca tema de diplomă a fost crearea unui sistem informatic care să informeze utilizatorii aplicației de starea stațiilor electrice de încarcare a automobilelor aflate în imprejurimea utilizatorului.

Un alt motiv este determinat de studiul pe care l-am realizat asupra unor aplicații existente și care m-a determinat să creez o nouă aplicație care să înglobeze toate facilitățile de care are nevoie un utilizator.

Alt motiv ar fi acela ca industria auto se orienteaza din ce in ce mai mult spre modelele cu o propulsie hibridă sau doar electrică. Urmând o creștere accelerată a automobilelor și respective a stațiilor electrice ne confruntăm cu o problemă de indentificare a stațiilor disponibile din zona în care vrem să le accesam.

O scurta istorie a automobilelor hibride, adică a mașinilor care erau propulsate de motoare electrice ajutate de un motor cu ardere interna, ne duce până în zorii secolului al XX-lea, în anul 1901. În acel an, Ferdinand Porsche a proiectat automobilul Mixte, o versiune hibridă a mașinii electrice “System LohnerPorsche”, lansat cu un an înainte la Salonul Auto de la Paris. Mixte avea o autonomie de 50 de kilometri și atingea o viteză maximă de 50 km/h.

În figura următoare este prezentată prima mașină hibridă.



Figura 1.2. Prima mașină hibridă, Mixte a. 1901.

În momentul de față există aplicații web care indică poziția stațiilor electrice numită “PlugShare”. Rețeaua de încărcare pentru mașini electrice din România se îndreaptă cu pași repezi spre ieșirea din faza de pionierat. În toate orașele importante din țară există cel puțin câteva puncte de încărcare de putere medie, adică 22 Kw, și minimum o stație rapidă , 50 Kw. Prețul pentru 1 kwh este de 0,5 lei, în medie pentru un consum casnic. Timpul de încărcare pentru o mașină electrică a anului 2019, cu o capacitate de 40 kWh, poate dura între o oră și 12 ore, în funcție de capacitatea bateriei, puterea prizei/stației și puterea încărcătorului.

În 1917, compania Woods Motor Vehicle din Chicago a produs un automobil hibrid al cărui motor electric îl propulsa până la 25 km/h. Până în 1918 au ieșit de pe linia de fabricație 600 de astfel de produse, dar mașina nu a avut succes deoarece era prea lentă, scumpă și greu de reparat.

Istoria maşinilor electrice este una fascinantă, iar evoluţia acestui tip de automobile a fost una spectaculoasă, cu multe urcuşuri şi coborâşuri în preferinţele publicului. Dacă în anii 1900 aceste maşini dominau piaţa, în 1935 reprezentau deja o raritate, iar producţia lor la scară largă s-a făcut abia 90 de ani mai târziu.

Marii constructori de automobile, inclusiv General Motors, Volkswagen, Daimler AG, nu neglijează acest sector, în prezent fiind alocate fonduri consistente pentru a dezvolta vehicule electrice ce pot satisface chiar şi cele mai rafinate gusturi. Tesla s-a axat în special pe dezvoltarea bateriilor care alimentează autoturismele electrice. Primul automobil electric care a stârnit pasiuni a fost conceput la sfârşitul anilor 1800 şi începutul anilor 1900. În 1899 şi 1900, vehiculele electrice surclasau toate celelalte tipuri de maşini. În fapt, potrivit datelor statistice, 28% din cele 4.192 de maşini produse în Statele Unite erau electrice.

Îngrijorarea legată de poluare a fost parţial responsabilă pentru interesul privind dezvoltarea tehnologiei pentru automobile electrice. Toyota Prius a fost un alt model care a crescut interesul pentru maşinile cu consum redus de carburant. Primul model a fost produs în Japonia, în 1997. A devenit disponibil în întreaga lume în anul 2000. Prius a fost unul din primele vehicule hibrid produse în masă. Succesul a fost unul remarcabil, în primul an de la lansare compania vânzând 50.000 de astfel de maşini.Tayota Prius înregistrează recorduri peste recorduri în vânzări în SUA, Japonia și cresc constant în Europa de Vest.

În următorii ani, modelele propuse de Nissan, General Motors şi Volkswagen au făcut ca numărul celor care aveau maşini electrice să fie de ordinul milioanelor, iar tendinţa este una de creştere. De altfel, în 2016 vor fi lansate pe piaţă nu mai puţin de şase modele noi de maşini electrice. Mai mult, un nou tip de motor electric, după cel oferit de Tesla, propus de Faraday Future, ar putea rula pe şoselele lumii până în 2020. Deşi compania nu a dat detalii despre gama exactă a vehiculelor sale, Nick Sampson - şeful deparamentului de dezvoltare - a precizat că bateriile vor avea o autonomie mult mai mare decât cele marca Tesla .

Așa-zisele „mașini verzi” sunt proiectate să producă niveluri reduse sau inexistente de gaze cu efect de sera, in special dioxid de carbon, metan și oxid nitros. Acest lucru se poate obține prin construirea unor mașini funcționale pe baza altor mijloace energetice decât combustibilii fosili, benzina sau motorina. Este vorba despre surse de energie alternativa si regenerabila, care, în privința mașinilor cel puțin, pot îmbracă forma de celule de combustie, bazate pe biocombustibil (etanol, butanol, metanol), hidrogen, alimentare fotovoltaica, baterii sau super capacitoare .

Primul punct de încărcare pentru mașini electrice din România a fost înaugurat în data de 22 iunie 2011, la sediul Transelectrica din Municiupiul București. Acest punct de încărcare era o priză de 230 V, 16 A, pozițioată pe stradă, loc de parcare dedicat și puteai să îți încarci un autovehicul electtic în acel loc. [2]

În anul 2015 industria mașinilor electrice a avut un nou început, începând cu instalarea primei stații de mare putere, 50 de Kw, la sediul Porsche România din Pipera. În anul 2016 a urmat o altă stație similară, tot de 50 Kw, la sediul BMW Group și tot atunci deveneau funcționale și primele stații de 50 Kw plasate în parcările marilor centre comerciale. S-au făcur remarcați Kaufland/e-charge Renovatio (la nivel național) și Lidl (pe București).

Anii 2017 și 2018 au marcat dezvoltarea rețelei de încărcare pe bazele deja puse anterior, plus puncte de încărcare noi. În această perioadă au început să apară noi nume precum Polyfazer, Rotric, Romstal, Electromagnetica, Penny, Mega Image, EVconnect, OMV + Electrica, Rompetrol, Mol, ETU, MCM Petrol, Enel, E.ON, engie, Uber (Green), BCR (eGO), practic încep să se facă remarcate companii noi pe subiectul electromobilității, iar marile benzinării diversifică tipurile de carburant oferite, adăugând și stațiile de curent electric pe lângă cele de benzină și disel.

În anul 2019 au fost așteptate primele stații de putere foarte mare, adică 150 Kw, iar concomitent vor apărea primele mașini electrice ce pot fi încărcate la acceastă putere.

În cele mai multe locuri este gratuit să încarci o mașină electrică, iar cel mai scump preț pe unitate (1 unitate = 1 kWh) este de 1,95 Lei. Termenul de unitate apare ca echivalent pentru kWh din motive de legislație. Furnizorii de electricitate nu pot vinde curent electric către un automobil (adică spre un bun care se deplasează), ci doar către un punct fizic fix (clădire, stație). Așa că furnizorii s-au adaptat și au botezat kWh în unitate, să fie cât mai simplu de înțeles de către utilizatori. [3]

## 1.2.Obiectele lucrarii

Lucrarea are ca obiectiv realizarea unei aplicații de mobile (Android) care sa puna la dispoziția utilizatorilor un serviciu de indetificare a stațiilor electrice pe o rază de aproximativ 400-500 m a utilizatorului, precum și o rută care să te conducă cu succes la stația cea mai apropiată și potrivită din punct de vedere al configurației vehiculului pe care utilizatorul dorește să îl încarce.

Android este un sistem de operare pentru dispozitive și telefoane mobile bazată pe nucleul Linux, dezvoltată inițial de compania Google. Android permite dezvoltatorilor să scrie un cod gestionat în limbajul Java, controlând dispozitivul prin intermediul bibliotecilor Java dezvoltate de Google. Lansarea platformei Android la 5 noiembrie 2007 a fost anunțată prin fondarea Open Handset Alliance, un consorțiu de 48 de companii de hardware, software și de telecomunicații, consacrat dezvoltării de standarde deschise pentru dispozitive mobile. Google a lansat cea mai mare parte a codului Android sub licența Apache, o licență de tip free-software și open source.

Fiind o platformă destinată dezvoltării de aplicaţii pentru telefoane mobile, în realizarea ei s-a avut în vedere conservarea energiei acumulatorului precum şi gestionarea în mod eficient a resurselor de memorie existente.

În figura următoare este prezentată evoluția pieței de telefonie mobilă între anii 2017-2023.

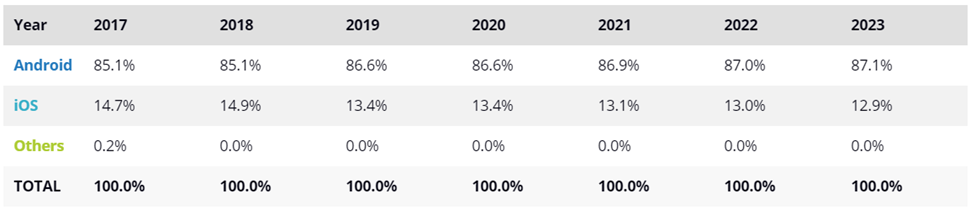


Figura 1.3. Evolutia pieței de telefoane mobile în următorii ani

Android deține în prezent o pondere a pieței de 86% care este în continuă ascensiune pe piața telefoanelor și tabletelor. O analiză facută de IDC Corporate prezice urmatoarea evoluție a pieței de telefoane smart și tablete.

Problema abordata este aceea de a indetifica statiile electrice intr-un mod cat mai accesibil si la indemana oricui.Aproape oricine detine un smartphone care il ajuta in activitatile de zi cu zi de aceea o aplicatie care sa te ajute sa indetifici statiile electrice mai ales daca detii un automobil cu propulsie electrica.

Avantajul acestei metode de indentificare este faptul ca este usor accesibila nu necesita cunostinte avansate de utilizare a smartphone-urilor.In scopul evitarii oricarei aglomerari asupra uneia sau mai multe statii.

Dezavantajul este ca daca nu este folosit de o mare majoritate de utilizatori pot aparea interferente in cea ce priveste starea actuala a statiei.

Pentru realizarea lucrării de diplomă se va realiza o aplicație mobile pentru identificarea stațiilor de încărcare, pentru vehiculele electrice.

Un vehicul hibrid este un vehicul care are mai multe sisteme de propulsie, spre deosebire de vehiculele convenționale (cu motor cu ardere internă), cele electrice sau cele cu pile de combustie, care au câte un singur sistem de propulsie. Adesea, în special pentru autoturisme, termenul se folosește în sensul de vehicul electric hibrid adică pentru vehiculele echipate cu motoare cu ardere internă și cu motoare electrice. Însă sunt posibile și alte combinații: hidraulic hibrid (en) (cu motor cu ardere internă și pompă/motor hidraulic), sau propulsat muscular și electric, de exemplu bicicletele electrice. În această lucrare se studiază istoria dezvoltării mijloacelor de transport hibrid.

În ceea ce privește numărul de mașini electrice pe șoselele din întreaga lume, aceasta a crescut până la aproximativ 2 milioane în 2016, potrivit Agenției Internaționale pentru Energie. Guvernul britanic s-a angajat ca aproape toate autoturismele și autoutilitarele să fie cu emisii zero până în 2050.Ca urmare, se estimează că cererea pentru automobile electrice va accelera dramatic în următoarele decenii. OPEC și-a revizuit prognoza la 266 milioane de mașini electrice până în 2040, aproape de șase ori mai mult decât estimarea anterioară de 46 milioane.

Avantajele majore ale automobilului electric sunt consumul redus, poluare aproape de zero în afara producției, eficiență/randament foarte mare, accelerare instantă, zgomot foarte redus, întreținere ușoară prin eliminarea unei mari părți a motorului.

Dat fiind faptul că autoturismele electrice sunt în continuă ascensiune asta arată graficul facut de cercetatorii de la IENE(Institutul de energie pentru sud-estul Europei).

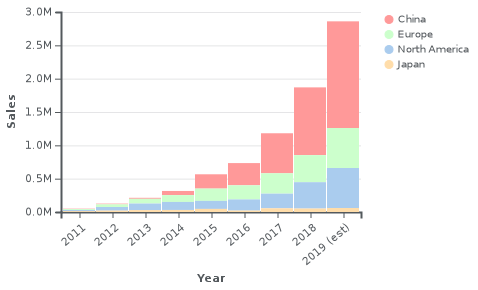


Figura 1.3. Evoluția vânzării de mașini electrice în următorii ani

Ascensiunea statiilor electrice este direct proporțională cu piața autovehiculelor cu propulsie electrică și hibird. Au apărut mai multe standarde de putere electrică și mufe de încarcare specifice fiecărei mărci de autovehicule.

# CAPITOLUL 2. Sistemul de operare Android

Următoarele două subcapitole oferă o privire de ansamblu asupra platformei Android, atât din punct de vedere hardware, cât şi software şi au fost realizate utilizăndu-se ca referinţe: lucrarea Android Forensics - Investigation, Analysis, and Mobile Security for Google Android scrisă de Andrew Hoog şi publicată la editura Elsevier în 2011 [6], precum şi pagina web destinată programatorilor Android [7]. Ultimul subcapitol descrie componenţa unei aplicaţii Android şi a fost elaborată consultând materialul Professional Android 2 Application Development scris de Reto Meier şi publicat la editura Wiley Publishing în 2010 [34].

## 2.1Platforma Hardware

Android a fost proiectat pentru a fi compatibil cu o gamă largă de componente hardware. Aceasta constituie o caracteristică importantă, întrucât permite producătorilor să conceapă şi să integreze elementele necesare dispozitivului Android, făcându-l atractiv atât în rândul programatorilor cât şi al consumatorilor. Fiind o platformă destinată dezvoltării de aplicaţii pentru telefoane mobile, în realizarea ei s-a avut în vedere conservarea energiei acumulatorului precum şi gestionarea în mod eficient a resurselor de memorie existente. Aşa cum apare în [6], majoritatea dispozitivelor Android au utilizat încă de la început procesoare ARM, cunoscute a fi destul de puternice, raportat la consumul mic de energie. Cu toate acestea, unele corporaţiile care au portat Android pe alte platforme au ales să folosească propriile lor CPUuri (ex: Intel a utilizat propriul lor procesor Atom). Pentru a nu suprasolicita procesorul, designerii de celulare au realizat o componentă separată care să gestioneze complexitatea comunicării mobile.

Modem-ul baseband şi radioul sunt sisteme hardware şi software care furnizează dispozitivelor Android o conexiune la telefonia mobilă, lucru ce permite atât transfer de voce, cât şi de date. După cum se poate vedea in Figura 2.1, la telefoanele inteligente întâlnim două tipuri de memorie: memorie cu acces aleator (RAM) şi memorie nevolatilă flash NAND. Conform [6], memoria RAM este folosită de sistem pentru încărcarea, executarea şi manipularea diferitelor părţi ale sistemului 12 de operare, iar cea flash pentru stocarea sistemului de operare precum şi a datelor dorite de utilizator. Datorită limitării impuse de spaţiul restrâns, cele două tipuri de memorie sunt combinate într-o singură componentă, cunoscută sub numele de pachet multicip (MCP).

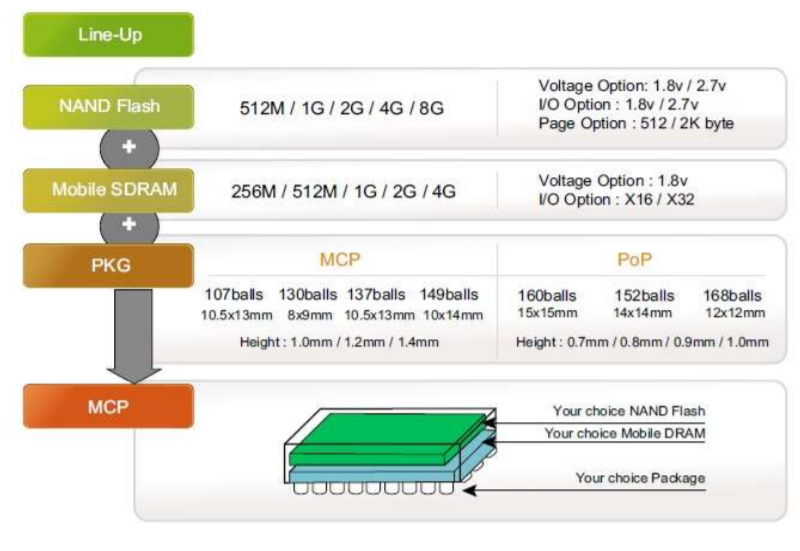


Figura 2.1. Arhitectura MCP

## 2.2Arhitectura MCP

Arhitectura MCP Una dintre cele mai interesante componente hardware este sistemul de poziţionare globală (GPS) care este oferit gratis în pachetul de bază. Acesta nu identifică doar locaţia telefonului, folosindu-se de sateliţii GPS, dar oferă şi aplicaţiilor posibilitatea de a accesa şi de a se folosi de aceste informaţii. Pe lângă reţeaua de telefonie mobilă existentă, Android permite conectarea la internet prin activarea opţiunii Wireless, cât şi conectarea la alte telefoane sau dispozitive externe, prin activarea opţiunii Bluetooth. 13 Majoritatea telefoanelor prezintă un card de memorie detaşabil care este cunoscut prin numele de card SD (Secure Digital). La fel ca memoria flash, cardurile SD sunt tipuri de memorie nevolatilă ce utilizează tehnologia flash NAND.

Fiind concepute ca elemente portabile, acestea aderă la anumite specificaţii fizice, făcând posibil interoperabilitatea cu celelalte dispozitive. Acest aspect constituie una dintre notele distinctive vis-a-vis de telefoanele Apple, care nu prezintă card SD, ci doar memorie flash cu o capacitate de la 4 GB la 32 GB, după cum este relatat în [6]. Această alegere făcută de producătorii Android nu este doar o soluţie mai ieftină ci şi practică, acest card putând fi utilizat şi de către alte telefoane.

Ecranul este interfaţa sistemului de operare pentru interacţiunea cu utilizatorul, fiind important nu doar pentru impactul pe care îl produce din punct de vedere vizual, cât şi pentru răspunsul pe care îl oferă la atingere. Acesta este într-o continuă dezvoltare urmărindu-se creşterea rezoluţiei, a luminozităţii, precum şi capacităţii de răspuns la stimulii externi. Camera telefonului şi-a depăşit atribuţiile cunoscute de a face poze sau de a filma, utilizând celelalte componente hardware pentru a-şi lărgi aria de utilitate. Majoritatea dispozitivelor combină funcţionalitatea acesteia pentru a înregistra pe lângă data şi timpul la care s-a realizat o poză sau un video şi coordonatele GPS la momentul utilizării acesteia.

Datele multimedia pot fi cu uşurinţă încărcate pe web şi partajate cu alte persoane. Interesantă este utilizarea acestei comoponente în domeniul realităţii augmentate, camera putând fi folosită pentru a identifica un produs pe baza codului de bare sau a unei clădiri pe baza coordonatelor geografice sau a unor etichete. Tastatura a fost şi ea înlocuită cu una ce încorporează tehnologia touch screen. Aceasta se poate adapta după poziţia telefonului şi poate suporta mai multe limbi, existând şi posibilitatea utilizării unor sisteme ce au rolul de a prezice cuvintele ce urmează a fi scrise. Durata de viaţă a bateriei constituie încă o problemă datorită aplicaţiilor existente ce sunt mari consumatoare de energie. Se doreşte depăşirea acestui neajuns, existând cercetări în acest sens, după cum apare în [6], ce urmăresc realizarea unor baterii din ce în ce mai puternice sau încărcarea bateriei fără utilizarea vreunui cablu, un exemplu fiind încărcarea în timpul mişcării. Majoritatea dispozitivelor Android oferă suport pentru USB (Universal Serial Bus) cu ajutorul căruia telefoanele pot fi conectate la un calculator. Prin cablu se poate realiza atât încărcarea bateriei, cât şi accesarea memoriei SD sau a consolei Linux de pe dispozitiv.

Android poate detecta şi schimba poziţia ecranului, în funcţie de felul în care telefonul este ţinut sau rotit, acest lucru fiind posibil datorită existenţei unui accelerometru care detectează aceste schimbări. Noile versiuni suportă şi un giroscop, care este mult mai senzitiv decât un accelerometru, aspect ce a dus la dezvoltarea jocurilor pe această platformă. Ultimele componente hardware sunt constituite de difuzorul şi microfonul telefonului, care la fel ca şi celelalte părţi sunt perfecţionate de la o versiune la alta. Unele dispozitive prezintă două sau trei microfoane, care combinate cu software-ul Android au abilitatea de a detecta şi elimina zgomotul de fundal pentru a oferi o calitate mai bună a sunetului.

## 2.3 Sistemul de operare

Sistemul de operare este bazat pe nucleul Linux 2.6 care oferă software-ul necesar pentru a coordona atât partea hardware, cât şi aplicaţiile Android. Aşa cum este ilustrat în Figura 1.2, funcţiile low-level includ administrarea energiei, a conexiunii wireless, a ecranului, a driverelor audio şi a memoriei flash. Peste kernel este construit un set de librarii ce oferă funcţionalităţile principale de care au nevoie atât dezvoltatorii, cât şi utilizatorii. Acestea includ librării precum cele ce gestionează lucrul cu bazele de date (SQLite), cel ce furnizează utilizarea de socket-uri securizate bazate pe protocolul SSL, dar şi librăria standard C.

Conform [7], librăriile principale sunt unite cu o maşină virtuală customizată Java, ce furnizează mediul de rulare pentru aplicaţiile Android. Dezvoltatorii scriu şi compilează programele cu kit-ul de dezvoltare Java oferit de companie Sun (Sun’s Java Development Kit) şi byte code-ul rezultat este transformat în fişiere de tip .dex ce sunt rulate apoi pe maşina virtuală Dalvik.

Fiecare program rulează în procesul său propria lui instanţă a maşinii virtuale Dalvik, care este concepută astfel încât mai multe multe instanţe ale acesteia să poată rula pe acelaşi dispozitiv într-un mod eficient.

Utilizarea acestor resurse este realizată prin intermediul unor API-uri şi a unui application framework, cu care programatorii interacţionează şi care oferă acces acestora la metodele necesare în crearea aplicaţiilor dorite.

## 2.4 Rezumat Android

Acest capitol oferă o privire de ansamblu asupra platformei Android atât din punct de vedere hardware, cât şi software. Deşi unele componente variază de la un producător la altul, structura este comună pentru fiecare dintre aceste dispozitive.

Telefonul poate fi privit asemenea unui mini computer, prezentând un procesor de tip ARM şi doua tipuri de memorie, combinate într-o singură componentă, numită pachet multicip (MCP). Conexiunea la reţeaua mobilă este realizată de către modem-ul baseband şi radioul existent, conectarea la internet prin activarea opţiunii Wireless, iar la alte telefoane sau dispozitive externe prin activarea opţiunii Bluetooth sau USB. Pe lângă acestea întâlnim o cameră performantă, un ecran touch screen şi un sistem de localizare GPS.

Sistemul de operare este bazat pe nucleul Linux 2.6 şi oferă software-ul necesar pentru a coordona atât partea hardware, cât şi aplicaţiile Android. Librăriile sunt accesate de programatori prin intermediul API-ului existent, iar rularea programelor se realizează pe maşinile virtuale Dalvik.

Din punct de vedere structural, o aplicaţie Android este compusă din două elemente principale: clasele ce descriu funţionalitatea programului şi resursele constituite din aspecte, imagini, şiruri de caractere sau culori ce descriu interfaţa programului. Definirea structurii şi metadatei aplicaţiei, componentele şi cerinţele acesteia sunt precizate în fişierul manifest al aplicaţiei, fişier fără de care o aplicaţie nu poate rula.

# CAPITOLUL 3. Aspecte teoretice

În acest subcapitol, principalele obiective vor avea drept scop înțelegerea evenimentelor petrecute în aplicația curentă, familializarea cu unele concepte folosite şi oferirea unor detalii legate de implementarea acestora.

## 3.1. Perspectiva Server-Side C# .NET CORE

În această lucrare, propun o metodă de creare a unei aplicații în Visual Studio 2019 Professional, într-un limbaj de programare de nivel înalt, C#, folosind o arhitectură REST(Representational State Transfer), însoțită de algoritmi eficienți de ultimă generație pentru procesarea unui traseu eficient.

Codul C# este compilat ca şi cod de bază, ceea ce înseamnă că beneficiază de serviciile CLR. Aceste servicii includ interoperabilitea limbajelor, garbage collection, securitate ridicată şi suportul versiunilor imbunătăţite.

În .NET (şi, implicit, în C#), tipurile de date se împart în două categorii principale: tipuri valoare şi tipuri referinţă. Diferenţa dintre ele este că variabilele de tip referinţă conţin referinţe (pointeri) spre datele propriu-zise, care se află în heap, pe când variabilele de tip valoare conţin valorile efective. Această deosebire se observă, de exemplu, la atribuiri sau la apeluri de funcţii.

La o atribuire care implică tipuri referinţă, referinţa spre un obiect din memorie este duplicată, dar obiectul în sine este unul singur (are loc fenomenul de aliasing - mai multe nume pentru acelaşi obiect). La o atribuire care implică tipuri valoare, conţinutul variabilei este duplicat în variabila destinaţie. Tipurile valoare sunt structurile (struct) şi enumerările (enum). Tipuri referinţă sunt clasele (class), interfeţele (interface), tablourile şi delegările (delegate).

Poate părea surprinzător, dar unele din tipurile \"primitive\" cu care suntem obişnuiţi din C (int, de exemplu), sunt tot structuri. Altele, cum ar fi string, sunt clase. Singurul lucru care le face mai speciale în C# este faptul că, la compilare, textul din codul sursă este identificat şi convertit automat în instanţe ale acestor tipuri. În plus, pentru aceste tipuri, există cuvinte cheie C# prin care sunt descrise, dar şi clase .NET care reprezintă implementarea propriu-zisa.

## 3.1.1 Tablouri

Un tablou este o structură de date care conţine un număr de variabile numite elemente, accesate prin intermediul unuia sau mai multor indici întregi. Tablourile sunt tipuri referinţă, iar pentru a crea un tablou trebuie alocat explicit spaţiu cu operatorul new. În acest caz, un tablou este efectiv creat, iar valorile elementelor sunt iniţializate cu valoarea implicită a tipului. Pentru a crea un tablou de obiecte (tipuri referinţă), trebuie creat întâi tabloul, elementele acestuia vor fi iniţializate implicit cu null, după care tabloul trebuie parcurs şi elementele sale iniţializate explicit de către programator cu valori diferite de null.

## 3.1.2 Clase

O clasă este o structură de date care poate conţine date (variabile şi constante) şi cod (metode, proprietăţi, evenimente, operatori, constructori şi destructori). Clasele pot fi derivate din alte clase, prin acest mecanism clasele derivate putând extinde, specializa sau modifica comportamentul clasei de bază.

Clasele reprezintă piatra de temelie a programării pe obiecte. Acest concept porneşte de la aserţiunea că datele sunt strâns legate de codul care le prelucrează şi că gruparea lor într-o structură comună (spre deosebire de limbajele procedurale, în care datele sunt separate de cod) duce la o proiectare, implementare, testare şi întreţinere mai simplă.

Clasele sunt tipuri referinţă, ceea ce înseamnă că datele membru ale instanţelor se află în heap şi programul are acces la ele prin referinţe. Referinţele sunt păstrate corespunzător de runtime şi în cazul în care nu mai există referinţe spre o instanţă, memoria folosită de aceasta este recuperată automat.

Sunt eliminate astfel o serie întreagă de defecte software legate de dealocarea memoriei - în C++ pot apărea situaţii în care memoria nu este dealocată şi este pierdută pâna la oprirea programului sau situaţii în care memoria este dealocată de o componentă dar altă componentă încearcă ulterior să scrie la acea adresă pentru că are un pointer pe care îl crede valid.

Membrii unei clase pot fi statici; în acest caz, există o singură instanţă a lor care este accesibilă din toate instanţele clasei respective; în situaţia în care membri statici sunt şi publici, ei vor fi accesibili din alte clase fără a avea nevoie de o instanţă a clasei respective. Mai mult, ei nu pot fi accesaţi prin intermediul unor instanţe, ci numai folosind numele clasei.

De asemenea, pot fi definite metode statice. În mod similar, acestea aparţin clasei şi nu instanţelor acesteia şi pot fi apelate folosind numele clasei. În general, acţionează asupra membrilor statici sau efectuează operaţii pentru care nu este necesară o instanţă a clasei.

## 3.1.3 Drepturi de acces

Încapsularea reprezintă un aspect esenţial al programării pe obiecte şi înseamnă, pe scurt, posibilitatea de a separa implementarea unui obiect de funcţionalitatea pe care acesta o pune la dispoziţie utilizatorilor săi. Prin implementarea unui obiect ne referim, în general, la variabilele membru ale obiectului respectiv şi la metodele interne, care contribuie la realizarea funcţionalităţii obiectului, dar care nu sunt necesare utilizatorilor obiectului respectiv.

Dacă specificatorul de acces lipseşte, membrii unei clase sunt implicit privaţi.

Drepturile de acces reprezintă modalitatea prin care putem ascunde de utilizatorii clasei detaliile de implementare ale acesteia. Cei cinci specificatori de acces în C# şi semnificaţia lor sunt:

**private** - Membrii privaţi sunt accesibili numai din clasa în care sunt declaraţi.

**internal** - Membrii interni sunt accesibili numai din clase care fac parte din acelaşi assembly cu clasa în care sunt declaraţi.

**protected** - Membrii protejaţi sunt accesibili din clasa în care sunt declaraţi şi din clasele derivate din aceasta.

**protected internal** - Membrii interni şi protejaţi sunt accesibili din clasa în care sunt declaraţi, din clasele care fac parte din acelaşi assembly şi din clasele derivate, indiferent în ce assembly sunt acestea.

**public** - Membrii publici sunt accesibili din orice clasă.

## 3.1.4  Date membru

O clasă este alcatuită din membrii moşteniţi din clasa de bază şi din membrii declaraţi în corpul clasei. Datele pot fi variabile, constante, sau variabile read-only. Diferenta dintre constante si variabile read-only este ca valoarea constantelor poate fi calculata la compilare si plasata în fisierul executabil ca atare, pe când valoarea variabilelor read-only este calculata la rulare. Implicit, pot fi declarate doar constante apartinând tipurilor recunoscute de compilator ca "primitive". Variabilele read-only pot fi de orice tip si pot avea valori care nu sunt cunoscute la compilare.

## 3.1.5 Constructori

Constructorul este o metodă care implementează acţiunile care au loc la crearea şi iniţializarea unei instanţe a clasei. Are acelaşi nume ca şi clasa din care face parte şi nu returnează nici o valoare, nici măcar void. Poate apela un alt constructor al aceleiaşi clase sau un constructor al clasei de baza. Poate avea orice nivel de acces.

Dacă într-o clasă nu este implementat nici un constructor, compilatorul inserează automat un constructor public, fără argumente, care iniţializează automat toate variabilele cu valorile lor implicite.

## 3.1.6 Moştenire

Prin moştenire se înţelege crearea unei clase derivate care conţine implicit toţi membrii (mai puţin constructorii, constructorul static şi destructorul) unei alte clase, numite de bază. Moştenirea se mai numeşte şi derivare sau, mai rar, specializare.

În C#, moştenirea se realizeaza punând ":" după numele clasei, urmat de numele clasei de bază. O clasă poate avea o singură clasă de bază. Dacă o clasă nu este derivată explicit din nici o clasă, compilatorul o face implicit să fie derivată din object. Object este rădăcina ierarhiei de clase din .NET.

Moştenirea este tranzitivă, în sensul că dacă A este derivată din B şi B este derivată din C, implicit A va conţine şi membrii lui C (şi, evident, pe cei ai lui B). Prin moştenire, o clasă derivată extinde clasa de bază. Clasa derivată poate adăuga noi membri, dar nu îi poate elimina pe cei existenţi.

Deşi clasa derivată conţine implicit toţi membrii clasei de bază, acest fapt nu înseamnă ca îi şi poate accesa. Membrii privaţi ai clasei de baza există şi în clasa derivată, dar nu pot fi accesaţi. În acest fel, clasa de bază îşi poate schimba la nevoie implementarea internă fără a distruge funcţionalitatea claselor derivate existente.

O referinţă la clasa derivată poate fi tratată ca o referintă la clasa de bază. Aceasta conversie se numeşte upcast, din cauză că în reprezentările ierarhiilor de clase, clasele de bază se pun deasupra, cele derivate dedesubtul lor, ca într-un arbore generalizat. Prin upcast se urcă în arbore.

Conversia inversă, de la clasa de bază la cea derivată, se numeşte downcast şi trebuie făcută explicit, deoarece nu ştim dacă referinţa indică spre un obiect din clasa de bază, spre un obiect din clasa derivată la care încercăm să facem conversia sau spre un obiect al altei clase derivate din clasa noastră de bază.

Accesibilitatea trebuie sa fie consistentă şi în cazul în care încercăm să derivăm o clasă din alta. Clasa de bază trebuie sa fie cel puţin la fel de accesibilă ca şi clasa derivată din ea.

O clasă derivată poate ascunde membri ai clasei de bază, declarând membri cu aceeaşi semnătură. Prin aceasta, membrii clasei de bază nu sunt eliminaţi, ci devin inaccesibili prin referinţe la clasa derivată. Ascunderea membrilor se face folosind cuvântul cheie new. Acest cuvânt cheie are rolul de a-l obliga pe programator să-şi declare explicit intenţiile şi face, în acest fel, codul mai lizibil.

## 3.1.7. Metode

În C#, este posibil, ca şi în C++ sau Java să avem mai multe metode cu acelaşi nume în cadrul aceleiaşi clase. În C#, la supraîncarcare se ţine cont de numele metodei, de tipul argumentelor, de categoria lor (de intrare, intrare-iesire, iesire), de numărul şi de ordinea lor. Semnătura unei metode nu include tipul returnat, numele argumentelor sau argumentele variabile de la sfârşitul listei de argumente.

Supraîncărcarea metodelor şi operatorilor se mai numeşte şi polimorfism static, deoarece putem obţine comportamente diferite prin acelaşi apel de funcţie, dar comportamentul este totuşi bine determinat la compilare.

În C#, este posibil ca o metoda să primească un număr variabil de argumente. Argumentele variabile sunt transmise într-un tablou, iar numele acestui tablou trebuie precedat în declaraţia funcţiei de cuvântul cheie params.

Polimorfismul este a treia piatră de temelie a programării pe obiecte. Dacă încapsularea afectează în mod direct uşurinţa de implementare şi utilizare a claselor prin separarea implementării de funcţionalitate, iar moştenirea reprezintă o modalitate simplă de reutilizare a codului, polimorfismul reprezintă componenta esenţiala când vine vorba despre extensibilitate.

O metodă virtuală este o metodă care poate fi suprascrisă într-o clasă derivată.

## 3.1.8 Proprietăţi

Proprietăţile sunt membri care oferă acces la atributele unui obiect. Exemple de proprietăţi includ lungimea unui şir de caractere, lungimea unui tablou, mărimea unui font, titlul unei ferestre şamd. Proprietăţile pot părea sinonime cu variabilele membru ale unei clase - au nume, au tip şi sunt folosite în mod similar de utilizatorii clasei, cu aceeaşi sintaxă. Cu toate acestea, proprietăţile diferă de variabilele membru prin faptul că nu reprezintă spaţii de memorie, ci funcţii de acces.

Aceste funcţii de acces sunt executate când utilizatorii clasei încearcă să obţină sau să modifice aceste atribute. Ele sunt similare cu modelul get/set din alte limbaje de programare (aşa şi sunt implementate intern), dar oferă o modalitate mai coerenta de lucru împreună cu aceleaşi avantaje - posibilitatea de a valida argumentele furnizate la modificarea atributelor sau de a anunţa alte obiecte despre modificările apărute, de exemplu.

## 3.1.9 Destructori

În .NET, memoria ocupată de obiecte este automat recuperata de un garbage collector în momentul când nu mai este folosită. Uneori, însă, un obiect este asociat cu resurse care nu depind de .NET şi care trebuie dealocate explicit, într-un fel sau altul. Un exemplu de astfel de resurse îl reprezintă conexiunile TCP/IP, care nu pot fi lăsate la voia întâmplarii.

De obicei, este bine să se elibereze astfel de resurse în momentul când nu mai sunt necesare şi vom vedea puţin mai încolo cum se face acest lucru. Există însă şi o plasă de siguranţă oferită de compilator, reprezentată de destructori.

Destructorii sunt metode care au acelaşi nume cu clasă din care fac parte, precedat de semnul ~. Nu au drepturi de acces, nu au argumente şi nu permit nici un fel de specificatori (static, virtual samd). Nu pot fi invocaţi explicit, ci numai de librariile .NET specializate pe recuperarea memoriei. Ordinea şi momentul în care sunt apelaţi sunt nedefinite, ca şi firul de execuţie în care sunt executaţi. Este bine ca în aceste metode să dealocaţi numai obiectele care nu pot fi dealocate automat de .NET şi să nu faceţi nici un fel de alte operaţii.

## 

## 3.2 Perspectiva interfata utilizator IONIC + Angular

IONIC este un SDK(software development kit) mai exact un set de instrumente disponibile intr-un pachet destul de generos plin de instrumente si componente care este destinat crearii aplicatiilor de mobile hybrid, desktop si aplicatii web progresive.

Acest SDK se bazeaza pe un ansablu modern de tehnologii precum CSS, HTML5 si SASS. Aplicatiile pot fi construite ca o vizualizare web oferind portabilitate pentru orice fel de platforma si sitem de operare.Ofera servicii pentru dezovltatori pentru a crea applicatii in cloud , build-uri automate si un IDE pentru a face ca dezovltarea sa fie cat mai accesibila si suro de folosit numit Ionic Studio.Un instrument important pe care acest SDK il aduce este linia de comanda specifica pentru generearea proiectului, adaugarea de componente, servicii si pachete , instalarea de plugin-uri etc.

IONIC foloseste Cordova pentru a avea acces la toate functiile telefonului precum Camera, GPS, Lanterna, Gesturi .etc.Folosind componente vizuale web, metodologiile se bazeaza pe paradigmele din aria dezvoltarii web.

Penru Android suporta o versiune de software minima de 4.4 sau mai mare, pentur iOS versiunea 10 sau mai mult.

## 3.3. Autentificare Auth0

Autentifiacarea aplicatiei se bazeaza pe un provider autorizat de autentificare si anume Auth0.Este unul dintre liderii de pe piata serviciilor de autentificare renumit pentru securitatea sporita pentru care o ofera,accesibilitate in functie de orice tehnologie si mai ales faptul ca este bazat pe un standar renumit JWT.

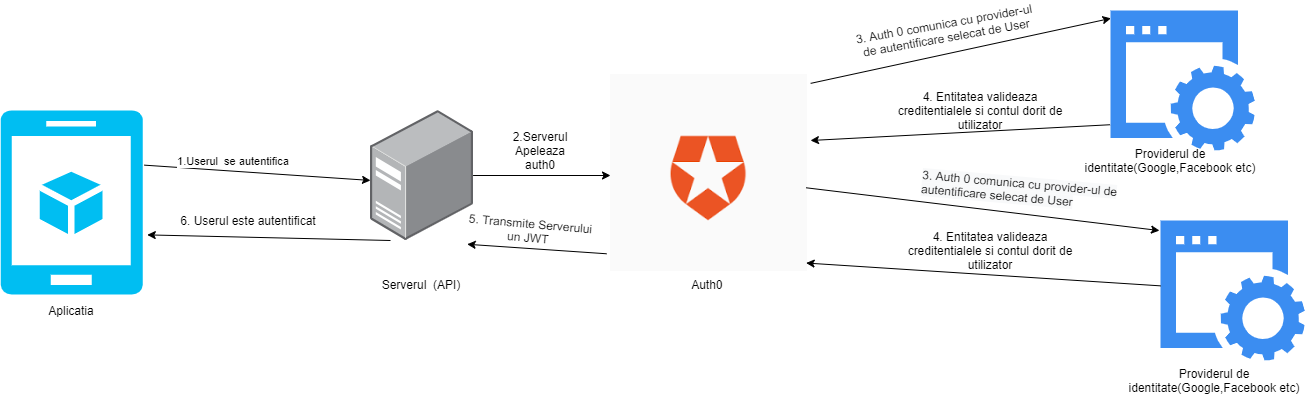
**JSON Web Token** (JWT) este un standard de internet pentru crearea de date cu semnătură opțională și / sau criptare opțională a cărei sarcină utilă deține JSON care afirmă un număr de revendicări.Jetoanele sunt semnate fie folosind un secret privat, fie o cheie publică / privată. De exemplu, un server ar putea genera un jeton care are revendicarea „conectat ca admin” și care îl poate oferi unui client. Clientul ar putea apoi să folosească acel jeton pentru a demonstra că este conectat ca administrator. Jetoanele pot fi semnate de cheia privată a unei părți (de obicei serverul), astfel încât partea care poate verifica ulterior simbolul este legitim. Dacă cealaltă parte, prin anumite mijloace adecvate și de încredere, deține cheia publică corespunzătoare, ei sunt capabili să verifice legitimitatea jetonului. Token-urile sunt concepute pentru a fi compacte, URL-safe și pot fi utilizate mai ales într-un context de conectare unică al browserului web. Revendicările JWT pot fi utilizate în mod obișnuit pentru a trece identitatea utilizatorilor autentificați între un furnizor de identitate și un furnizor de servicii sau orice alt tip de revendicări, după cum este necesar în procesele de afaceri.

Auth0 este o soluție flexibilă, pentru a adăuga servicii de autentificare și autorizare aplicațiilor dvs. Aplicatia poate evita timpul și riscul care vin cu construirea propriei soluții de autentificare și autorizare a utilizatorilor.

Auth0 poate conecta orice aplicație (scrisă în orice limbă sau pe orice stivă) la Auth0 și să definiți furnizorii de identitate pe care doriți să-l utilizați (cum doriți ca utilizatorii dvs. să se autentifice ex: Google,Facebook etc).

Pe baza tehnologiei aplicației apelează-ne la API- lui Auth0 și se conecteaza la aplicatie. Acum de fiecare dată când un utilizator încearcă să se autentifice, Auth0 își va verifica identitatea și va trimite informațiile necesare înapoi la aplicație.

Figura 3.1 reprezinta arhitectura pe care am implementat-o in acest proiect precum si modul de functionare.

Fig.3.1 Diagrama de functionare a autentificarii

## 3.4. Baza de date MonogoDB

Diversele baze de date NoSQL existente azi pe piaţă prezintă diferite abordari. Ceea ce au în comun este faptul că nu sunt relaţionale. Principalul avantaj este acela că permit lucrul eficient cu date nestructurate precum e-mail, multimedia, procesoare de text. În prezent există multe companii care au dezvoltat propriile baze de date NoSQL. Cele mai populare sunt cele dezvoltate de către companiile mari Web 2.0, precum Amazon and Google, din nevoia de a procesa cantităţi mari de date.

Avantajele folosirii sistemelor NoSQL constau în portabilitate, iar serverele acestora utilizează modelul DHT (Distributed Hash Table), astfel încât manipularea datelor obiect se realizează prin furnizarea de către utilizator a cheii primare a obiectului. DHT (Distributed Hash Table) fiind accesibil prin intermediul API Hashtable. Funcţia hash este un algoritm matematic ce stă la baza arhitecturii NoSQL. Acest algoritm are la intrare o lungime variabilă şi produce o lungime fixă la ieşire. Fiecare cheie este partiţionată, iar valoarea hash rezultantă este folosită pentru a se trimite la un server de baze de date NoSQL. Înregistrarea este memorată pentru căutări ulterioare. NoSQL permite accesul la date doar într-un model îngust de acces predefinit. Astfel modelul de acces la date este unul fiabil. Printre sistemele NoSQL din ziua de azi pot fi menţionate cele open-source, precum MongoDB.

Din punctul de vedere al arhitecturii bazelor de date NoSQL există probleme legate de interfeţe şi interoperabilitate. Modelul DHT (Distributed Hash Table) conţine interfeţele sale încă standardizate, însă nu conţine o semantică intrinsecă pentru indicii. Interoperabilitatea este importantă în momentul în care datele trebuie accesate de mai multe servicii, moment în care se poate pierde din performanţă. Spre deosebire de bazele de date relaţionale care s-au consacrat ca fiind stabile, sistemele NoSQL apar pe piaţă ca o alternativă ce poate fi pusă în aplicare doar cu precauţie, deoarece încă nu au ajuns la un nivel de maturitate cel puţin egal cu RDBMS-urile. De asemenea, în cazul apariţiei unor probleme, suportul sistemelor NoSQL este încă limitat, acestea fiind în mare proiecte open-source, iar companiile care oferă suport pentru bazele de date NoSQL nu oferă credibilitatea companiilor globale.Sistemele NoSQL se potrivesc foarte bine şi în tehnologia Cloud, care se bazează pe virtualizare. Există şi un punct slab al virtualizării ce ţine de performanţa I/O, limitările CPU-ului şi ale memoriei fiind de altfel în strânsă legătură. Bazele de date NoSQL folosesc proporţia cea mai mare pe memoria de disc, aceasta fiind locaţia principală de scriere, însă datorită scalării orizontale sunt capabile să memoreze datele eficient. Bazele de date NoSQL prezintă un dezavantaj din punctul de vedere al administrării, pentru că necesită un anumit efort pentru a fi menţinute şi cunoştiinţe solide pentru instalare. Persoanele calificate pentru bazele de date NoSQL sunt mai puţine decât cele cu experienţă pentru RDBMS. De aceea preluarea lor de către firme trebuie să fie făcută în mod corespunzător, întrucât beneficiile reale aduse de acestea pot fi însoţite uneori şi de anumite probleme.

MongoDB este o bază de date NoSQL open-source orientată pe documente. MongoDB face parte din familia de sistemelor de baze de date NoSQL. Diferența principală constă în faptul că stocarea datelor nu se face folosind tabele precum într-o bază de date relațională, MongoDB stochează datele sub formă de documente JSON cu scheme dinamice. Dezvoltarea MongoDB a fot începută de 10gen în 2007, când compania dezvolta un produs platform as a service similar cu Windows Azure sau Google App Engine. În 2009 MongoDB a fost transformat într-un produs open-source.

MongoDB este o bază de date open-source NoSQL scrisă în C++. Aceasta poate conține mai multe baze de date, colecții și indecși. În unele cazuri (baze de date și colecții ) aceste obiecte pot fi create implicit.Odată create,ele se găsesc în catalogul sistemului db.systems.collection, db.system.indexes. Colecțiile conțin documente (BSON). Aceste documente conțin la rândul lor mai multe câmpuri.

În MongoDB nu există câmpuri predefinite spre deosebire de bazele de date relaționale, unde există coloanele care sunt definite în momentul în care tabelele sunt create. Nu există schemă pentru câmpurile dintr-un document, acestea precum și tipurile lor pot varia. Astfel nu există operația de „alter table” pentru adăugare de coloane. În practică este obișnuit ca o colecție să aibă o structură omogenă, deși nu este o cerință, colecțiile putând avea structuri diferite. Această flexibilitate presupune ușurință în migrarea și modificarea imaginii de ansamblu asupra datelor.

MongoDB lucrează cu date nestructurate şi organizează aceste date în format document. Implementarea acestei baze de date este mai uşoară decat un RDBMS deoarece ea urmăreşte modelul cheie valoare şi nu are nevoie de o schemă predefinită a datelor. Conceptele acesteia pornesc de la concepte tradiţionale, de aceea întelegerea filosofiei acestei baze de date este ceva uşor de realizat.

**Caracteristicile MongoDB**

**Stocarea datelor sub formă de documente.** Baza de date MongoDB stochează obiecte (documente). Aceste documente reduc nevoia de join.

**Prezintă support pentru indexare.** Indexarea pe fiecare din atribute se face în modul tradițional (RDBMS) asupra cheilor de regăsire ale documentelor. Disponibilitatea datelor este asigurată printr-un proces automat de failover.

**Auto-Sharding.** Shardingul sau partiționarea datelor pe orizontală se face automat. Citirile și scrierile sunt distribuite pe partiții. Lipsa joinurilor face ca interogările distribuite să fie rapide.

**Limbajul de interogare**Limbajul de interogare este îmbunatatit și păstreaza principii SQL si C++.

**Modificări rapide.**MongoDB suportă operații de actualizare atomice căt și pe cele tradiționale. Operatiile de mai jos demonstrează flexibilitatea limbajului NoSQL.

**Map Reduce.** MongoDB folosește map/reduce pentru agregare și procesare batch.

**GridFS.**Poate stoca fișiere de dimensiuni mari fără a complica stiva de date.

**Suport.**Dispune de o comunitate care te poate ajuta treci peste anumte impedimente

**Paralelă MySQL versus MongoDB**

|  |  |
| --- | --- |
| **Termeni MySQL** | **Concepte MongoDB** |
| Bază de date | Bază de date |
| Tabelă | Colecție |
| Index | Index |
| Rând | Document BSON |
| Coloană | Câmp BSON |
| Join | Încapsulare și legatură |
| Cheie primară | Câmpul cheie unică \_id |
| Group by | Agregare |

## 3.5. Algoritmul de rutare Google Directions API

## 3.6. Rezultate obtinute

# CAPITOLUL 4. CONCLUZII SI DEZVOLTARI ULTERIOARE

# BIBLIOGRAFIE

<https://www.idc.com/promo/smartphone-market-share/os>

<https://www.iene.eu/heres-how-electric-cars-will-cause-the-next-oil-crisis-p3240.html>

[6] Hoog, A., 2011, Android Forensics Investigation, Analysis and Mobile Security for Google Android, Elsevier, Inc., Capitolele 2-3

[7] Android Developers: <https://developer.android.com/about>

# ANEXE