|  |  |  |
| --- | --- | --- |
|  | **UNIVERSITATEA „VASILE ALECSANDRI” DIN BACĂU**  **Facultatea de Științe**  **Str. Calea Mărăşeşti, nr. 157, Bacău, 600115**  **Tel. ++40-234-542411, tel./ fax ++40-234-571012** [www.ub.ro](http://www.ub.ro/); e-mail: [stiinte@ub.ro](mailto:stiinte@ub.ro) |  |

**PROGRAMUL DE STUDII INFORMATICĂ IFR**

**Recunoașterea și clasificarea sunetelor din mașină**

**Coordonator științific:**

Lect. Univ. dr. Vârlan Simona **Absolvent:**

Untu Andrei

**Bacău**

**2021**

[Introducere 3](#_Toc74843345)

[Capitolul I – Aplicații și tehnologii auto existente pentru siguranța copiilor și a animalelor 5](#_Toc74843346)

[I.1 Necesitatea apariției tehnologiilor privind siguranța copiilor și a animalelor 5](#_Toc74843347)

[I.2 Senzor pentru bancheta din spate 8](#_Toc74843348)

[I.3 Tehnologia radar 9](#_Toc74843349)

[I.4 SensorSafe 11](#_Toc74843350)

[I.5 Waze 12](#_Toc74843351)

[Capitolul II – Python, pachetele Python și tehnologiile folosite în realizarea aplicației 14](#_Toc74843352)

[II.1 Istoric și delimitări conceptuale 14](#_Toc74843353)

[II.2 Caracteristicile limbajului *Python* 15](#_Toc74843354)

[II.3 De ce Python? 16](#_Toc74843355)

[II.4 Python 3.x vs Python 2.7.x 16](#_Toc74843356)

[II.5 Cum execută Python codul 17](#_Toc74843357)

[II.6 Programarea modulară în Python 18](#_Toc74843358)

[II.7 Interpretorul *PyCharm* 19](#_Toc74843359)

[II.8 Învățarea automată 20](#_Toc74843360)

[II.9 Învățarea automată folosind Python 22](#_Toc74843361)

[II.10 Pachetul Python scikit-learn 23](#_Toc74843362)

[II.11 Pachetul Python librosa 24](#_Toc74843363)

[II.12 Pachetul Python tkinter 26](#_Toc74843364)

[II.13 Pushbullet 26](#_Toc74843365)

[Capitolul III – Aplicație de recunoaștere și clasificare a sunetelor din mașină 27](#_Toc74843366)

[III.1 Scopul și obiectivele care stau la baza dezvoltării aplicației 27](#_Toc74843367)

[III.2 Prezentarea aplicației 28](#_Toc74843368)

[III. 3 Pași preliminari în lansarea aplicației 29](#_Toc74843369)

[III.4 Funcționalitățile aplicației 32](#_Toc74843370)

[Concluzii și direcții de dezvoltare viitoare 41](#_Toc74843371)

[Bibliografie 42](#_Toc74843372)

**Introducere**

Tema proiectului meu de licența este „*Recunoașterea și clasificarea sunetelor înregistrate din mașină”*.

Industria automobilistică a cunoscut o dezvoltare accelerată în ultimul deceniu, iar marile companii se întrec în descoperirea noilor tehnologii care ar putea revoluționa această industrie. De la mașini clasice cu cutie manuala, fără servodirecție și cu un mecanism destul de simplu până la mașinile cu cutie automată, cu servodirecție care au ajuns să se conducă singure, ajungem să fim doar simpli pasageri în propria mașină.

Majoritatea sistemelor și componentele mecanice au fost îmbunătățite sau reinventate pentru a oferi utilizatorului mai mulă siguranță și confort. Sistemele de frânare sunt mai eficiente și mai sigure, stabilitatea automobilului pe șosea a fost îmbunătățită, consumul de carburant a fost redus cu ajutorul noilor motoare mult mai optimizate și chiar siguranța din interiorul mașinii a fost îmbunătățită prin sistemul de airbag-uri.

Noile tehnologii care se pot găsi in interiorul mașinii oferă chiar și conexiune la internet, cu ajutorul căruia poți controla unele funcții ale acesteia din telefonul mobil. Cu ajutorul acestei tehnologii, se pot diagnostica problemele mașinii de la depărtare, fără ca mașina să fie dusă fizic în service. Un alt sistem dezvoltat de curând, care se bazează pe conexiunea la internet a mașinii, este sistemul eCall, Emergency Call care poate fi apăsat manual sau în momentul unui accident se declanșează automat, iar acesta apelează serviciul de urgență din țară în care mașina este înregistrată, transmițând un semnal SOS împreună cu coordonatele mașinii. Un sistem asemănător este cel de bCall, breakdown call, acesta fiind de asemenea un buton fizic, care sună automat la un service auto prestabilit, în caz ca mașina are probleme tehnice.

Aplicația pe care am dezvoltat-o pentru lucrarea de licență se bazează pe recunoașterea sunetelor dintr-un fișier audio înregistrat in anumite condiții sau manual și telefonul mobil. Programul dezvoltat va putea fi integrat în sistemul automobilului foarte simplu, și va putea fi folosit ca o aplicație împreuna sistemul acesteia pentru a înregistra sunetele din interior și de a recunoaște sunetele pentru care aplicația a fost antrenata.

Principala motivație pentru care am ales să dezvolt și să implementez această idee este motivată de incidențele tot mai des întâlnite în care sunt implicați copii sau animale de companie, uitați sau lăsați intenționat în automobile. Problema apare în momentul în care temperaturile sunt foarte ridicate, deoarece interiorul automobilului poate atinge temperaturi foarte ridicate, chiar dacă temperatura de afara nu depășește 25 de grade Celsius, acestea pot fi fatale pentru ocupanții automobilului. De asemenea și temperaturile scăzute pot cauza neplăceri și pot duce la evenimente care se pot încheia tragic.

Aplicația dezvoltată pentru această lucrare, simulează temperatura din interiorul mașinii, iar în momentul în care temperatura este mult prea ridicată sau mult prea scăzută se declanșează un lanț de evenimente pentru a recunoaște daca în interiorul mașinii se află un bebeluș care plânge, câine sau pisica, și trimite notificări utilizatorului pentru ca acesta să fie conștient de situație.

Obiectivele principale care sunt atinse în această lucrare de licența sunt în primă fază analiza aplicațiilor, dispozitivelor sau tehnologiilor care încearcă să împiedice tragediile care se pot întâmpla atunci când șoferul uită sau lasă voluntar copilul sau animalul de companie în mașină, analiza resurselor necesare realizării aplicației de recunoaștere a sunetelor și prezentarea aplicației.

Capitolul I cuprinde prezentarea și analiza unor aplicații și tehnologii care încearcă să combată aceste evenimente care în unele cazuri se pot încheia tragic. Această prezentare este necesară pentru a avea o vedere generală despre aplicațiile și tehnologiile deja existente și cum încearcă acestea să combată aceste incidente.

Capitolul II cuprinde o scurtă prezentare a limbajului de programare Python și cum funcționează acesta iar apoi sunt prezentate tehnologiile și pachetele Python care sunt folosite pentru dezvoltarea aplicației.

Capitolul III cuprinde prezentarea detaliată a aplicației mai exact care sunt pașii pentru a antrena un model care este folosit pentru recunoaștere și care sunt evenimentele declanșatoare pentru recunoaștere.

Lucrarea se încheie cu o serie de concluzii și direcții de dezvoltare viitoare pentru aplicația dezvoltată și prezentată în capitolul III.

Capitolul I – Aplicații și tehnologii auto existente pentru siguranța copiilor și a animalelor

I.1 Necesitatea apariției tehnologiilor privind siguranța copiilor și a animalelor

În tandem cu evoluția și apariția noilor tehnologii apare și expunerea la noi riscuri , noi situații care pot pune în pericol viața omului . Necesitățile oamenilor sunt în continuă dezvoltare și pe parcursul accesării noilor soluții auto întâmpinăm noi cerințe ce trebuiesc satisfăcute în urma unor evenimente mai puțin favorabile.

O astfel de situație poate fi atunci când o persoană își uita accidental copilul sau animalul de companie în mașină. Astfel de incidente se pot întâmpla pe tot parcursul anului dar pot fi deosebit de mortale în perioada Aprilie-Septembrie. Într-o zi caldă de primăvara sau de vară, o mașină se poate încălzi în interior la o temperatură care poate deveni foarte periculoasă pentru cineva care se află în interior, în doar 10 minute. Moartea prin hipertermie este denumirea medicală oficială în care se încadrează decesele survenite în interiorul unei mașini foarte încinse.

Copii mici, persoanele în vârsta, persoanele cu dizabilități și animalele rămase singure în mașină prezintă riscuri mai mari de a suferi de insolație în aceste situații.

În America acest incident a fost numit „Forgotten Baby Syndrome” – sindromul copilului uitat și se referă la părinții care accidental își lasă bebelușul sau copii mici într-o mașină încuiată iar aceste acțiuni au rezultat tragic. Acest sindrom apare atunci când părinții sunt concentrați la alte sarcini în loc să fie atenți la ce au de făcut în acel moment. Această stare poate fi extrem de periculoasă pentru un copil care doarme sau pentru un animal lăsat în mașină.

Nu există un profil definit al persoanei care își uită copilul sau animalul în automobil, acest sindrom afectându-i pe toți oamenii indiferent de rasă, sex sau educație. De cele mai multe ori aceste incidente se întâmplă persoanelor care de obicei nu își iau copilul sau animalul de companie cu ei.

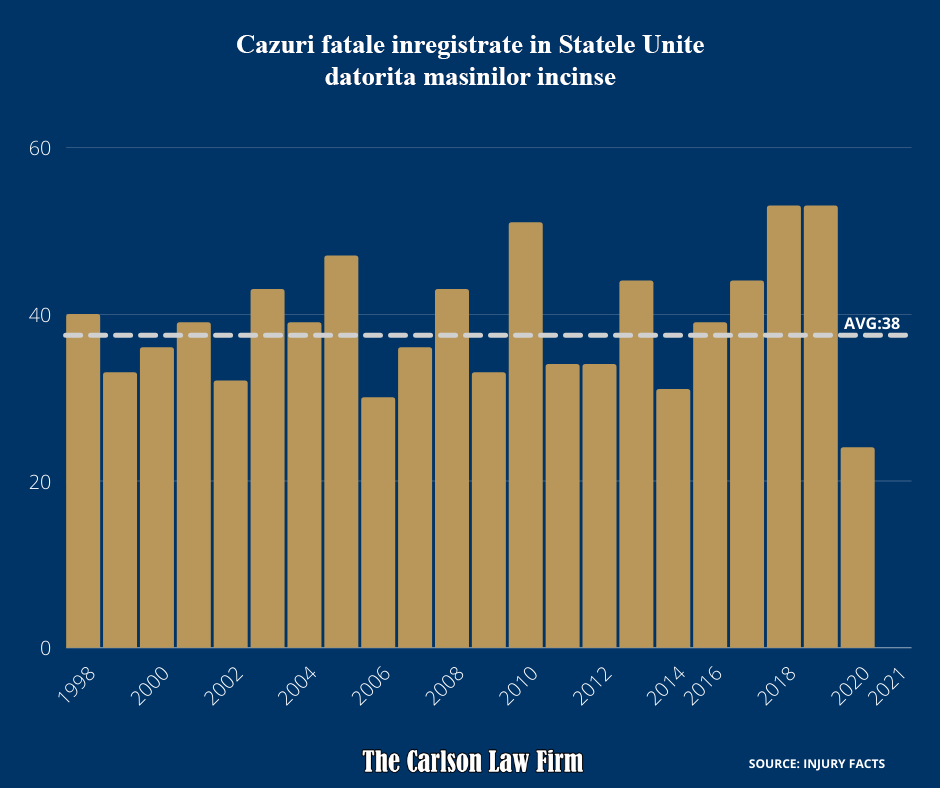
În timp ce 54% dintre incidente se întâmplă atunci când oamenii își uită animalele sau copii în mașină, sunt și alte moduri în care copii ajung în această situație gravă. In 28% din cazuri copii ajung în mașini fără voia părinților.[[1]](#footnote-1)



*Fig.1.1 Cauzele deceselor în mașină*

Datorită numărului mare de decese survenite, multe companii au început să dezvolte diverse tehnologii pentru a împiedica astfel de incidente. În anul 2020 numărul copiilor care au decedat în mașinile uitate în soare cu tot cu pasageri s-a înjumătățit față de anul 2019. 2020 a fost primul an, după aproape șase ani în care s-a înregistrat un declin de decese în Statele Unite survenite datorită mașinilor lăsate în soare cu un copil în interior.

Din anul 1998, de când au început să fie înregistrate aceste statistici au fost înregistrate în medie 38 cazuri fatale pe an.[[2]](#footnote-2)



*Fig.1.2 Numărul deceselor înregistrate în Statele Unite între anii 1998-2020*

Oamenii ar trebui să fie foarte atenți atunci când iau un pasager cu ei mai ales când acești pasageri sunt copii sau animale, deoarece se poate întâmpla ca șoferul să părăsească mașina și să lase fără știință pasagerii în ea. Dacă o mașină este lăsată în soare la 26 grade Celsius căldura din interiorul acesteia poate ajunge la 50 grade Celsius în mai puțin de 60 minute.

Pentru a fi mai prudenți oamenii pot urma niște pași simpli pentru a evita aceste incidente și pot aplica următoarele[[3]](#footnote-3):

* Să verifice de fiecare dată scaunul din față și scaunele din spate înainte de a părăsi autovehiculul. Pur și simplu conducătorul verifică încă o dată mașina în caz că a uitat ceva.
* Deschide ușa din spate a autovehiculului de fiecare dată după ce ai parcat mașina. Conducătorul auto verifică bancheta din spate și vede dacă nu are un pasager.
* Plasează scaunul copilului în mijlocul banchetei din spate. Așa conducătorul poate vedea în oglinda retrovizoare dacă are copilul în scaun.
* Conducătorul ar trebui să evite apelurile telefonice când este la volan. Acestea îl pot distrage iar dacă acesta este nevoit să parcheze și să părăsească autovehiculul uita să mai verifice dacă are pasageri.
* Cel mai bun mod de a evita aceste incidente este să fim responsabili pentru siguranța pasagerilor.

I.2 Senzor pentru bancheta din spate

Un grup de producători auto au fost de acord în mod voluntar să instaleze sisteme de detectare a pasagerilor pentru scaunele din spate pentru fiecare mașină nou produsă și vândută în SUA pănă în anul 2025. Sistemele vizează prevenirea uitării copiilor în automobile. Peste 90 de cazuri care au implicat copii uitați în mașină au fost fatale în SUA de la începutul anului 2018.

Acest nou acord include unii dintre cei mai mari producători de mașini incluzând cei de la General Motors, Fiat Chrysler Automobiles, Ford, Honda, Nissan sau Toyota. Acest acord nu specifică ce tip de abordare tehnică ar trebui să adopte fiecare producător sau ce fel de cerințe de performanță ar trebui să îndeplinească, doar că ar trebui să conțină o combinație de alerte sonore și vizuale atunci când vehiculul a fost oprit.

Unii susținători ai acestor idei avertizează ca acest acord nu este impus de către guvernul sau conducerea unui stat și în consecința tehnologiile dezvoltate de către companiile producătoare pot conține sisteme care nu monitorizează activ în vehicul prezența unui copil pe locurile din spate.

În unele țări, ca de exemplu SUA, cei de la „Consumer Reports” – protecția consumatorilor, care sugerează că producătorii de automobile ar trebui să integreze în mașină niște funcții care trebuie să reamintească șoferilor sau să îi alerteze cu privire la prezența copiilor pe bancheta din spate. [[4]](#footnote-4)

Unii producători de autovehicule au început deja să implementeze aceste sistem. General Motors a fost primul producător care a implementat un sistem care determină daca ușa din spate a mașinii a fost deschisă cu 10 minute înainte ca mașina să fie pornită sau oricând după ce mașina a fost pornită. Șoferul primește un mesaj pe panoul de bord după ce motorul este oprit, împreuna cu un semnal sonor. Această idee a fost împrumutată și implementată și de alți producători auto cum ar fi Hyundai, Kia sau Nissan cu unele diferențe de implementare. Producătorii General Motors și Honda au anunțat ca vor introduce acest sistem în majoritatea vehiculelor cu patru uși până la sfârșitul anului 2022.

În SUA, fiind una dintre țările în care rata acestor incidențe este destul de ridicată față de alte țări, sunt demarate procese pentru ca implementarea unor tehnologii, pentru a împiedica uitarea în mașina a unui copil, să devină reglementate prin lege și să oblige toți producătorii de mașini să instaleze un sistem pentru a împiedica aceste tragedii. Doar că, acest proces poate dura ani, iar pentru a accelera acest lucru, producătorii de automobile au creat acorduri voluntare pentru implementarea acestor tehnologii. De exemplu in 2016 „National Highway Traffic Safety Administration” și „Insurance Institute for Highway Safety” din SUA au încheiat un acord voluntar între 20 de producători de automobile pentru a echipa mașinile cu echipamente standard de frânare de urgență automată (AEB) pentru toate modelele pănă în 2022.

Dar și aceste acorduri voluntare pot avea probleme, deoarece nu există niște standarde și cerințe definite pe care un producător trebuie să le respecte pentru a implementa acest sistem și din acest motiv pot apărea probleme în implementarea și siguranța acestor sisteme.[[5]](#footnote-5)

I.3 Tehnologia radar

Universitatea din Waterloo, Canada lucrează la dezvoltarea unui senzor care ar putea salva vieți declanșând o alarmă atunci când copii sau animalele sunt lăsați singuri în mașină.

Noul dispozitiv dezvoltat de acești cercetători combină tehnologia radar cu inteligența artificială pentru a detecta copii sau animalele lăsate nesupravegheate cu o acuratețe de 100%.[[6]](#footnote-6)

Acest dispozitiv este destul de mic încât încape într-o palmă, având doar trei centimetri în diametru. Dispozitivul este conceput pentru a fi atașat la oglinda retrovizoare sau montat în tavanul mașinii.

Senzorul transmite semnale radar care sunt reflectate înapoi către el, dacă aceste unde întâlnesc un om sau un animal în raza lor de acțiune. Inteligența artificială incorporată, analizează apoi aceste semnale. Având aceste funcționalități, dispozitivul poate fi folosit pentru rezolvarea diferitelor probleme cu care lumea se confruntă, fiind foarte accesibil, ceea ce îl poate transforma într-o echipare standard pe toate mașinile.

Proiectarea si dezvoltarea acestui dispozitiv, în forma de disc, a fost finanțată parțial de un important producător de piese auto, care își propune să îl introducă pe piața cât mai curând.

Dispozitivul determină numărul de ocupanți și locația acestora în mașină. Aceste informații pot fi utilizate de aplicațiile de ridez sarong sau pe drumurile cu taxă.

Principalul scop este de a detecta când un copil sau un animal este uitat în mod accidental sau deliberat în mașină, un scenariu care poate rezulta în rănirea sau chiar decesul în condiții de vreme caldă sau rece. În asemenea cazuri autovehiculul împiedică blocarea ușilor și transmite o alertă sonoră pentru a alarma șoferul, pasagerii sau alți oameni din jur, anunțând ca există o problemă.

Spre deosebire de camere, acest device păstrează intimitatea și nu are un punct mort, deoarece sistemul radar poate penetra scaunele, de exemplu, pentru a determina dacă există un pasager pe scaunul din spate.

Dispozitivul are un consum redus de energie folosind bateria autovehiculului, putând distinge ființele vii de obiectele neînsuflețite prin detectarea mișcărilor de respirație subtile.

Cercetătorii explorează acum utilizarea acestei capacitați pentru a putea monitoriza semnele vitale ale conducătorului auto pentru a putea detecta semnele de oboseală, distrageri, tulburări, boli sau alte probleme. [[7]](#footnote-7)

Și marii producători de mașini au început să dezvolte astfel de dispozitive care se bazează pe radar, iar prima mașină care va ajunge pe piață cu un asemenea dispozitiv special conceput pentru a detecta copii uitați in mașina, este un SUV GV70 produs de cei de la Generis.[[8]](#footnote-8)

I.4 SensorSafe

SensorSafe este o tehnologie dezvoltată de cei de la Cybex. Combină tehnologia smart cu cea a designului smart pentru a oferi siguranță copilului aflat pe bancheta din spate.[[9]](#footnote-9)



*Fig.1.3 Dispozitivul SensorSafe*

Cybex produc scaune auto, cărucioare pentru copii și mobilier care nu sunt doar sigure dar sunt adaptate pentru stilul de viață urban. Aceasta companie se bazează pe principiul inovației care combină un design unic, siguranța, calitate și funcții inteligente. Având niște strategii bine definite acestea au condus la multe premii internaționale la concursuri de profil.

SensorSafe a fost dezvoltat pentru a ajuta și preveni situațiile critice care se pot întâmpla în mașină. Acest sistem alertează atunci când un copil încearcă să își desfacă centura, când temperatura ambientală din jurul copilului este prea ridicată sau prea scăzută, dacă copilul este lăsat nesupravegheat în mașină sau chiar dacă copilul a stat prea mult timp așezat în scaun. Acest sistem furnizează o monitorizare continua a copilului aflat în scaun și asigură siguranța și sănătatea acestuia.

Toate alertele acestui dispozitiv sunt primite prin Bluetooth pe smartphone printr-o aplicație ușor de utilizat.

Mai mult de 51% dintre părinți au raportat situația în care copilul lor a desfăcut centura de siguranță de la scaun de unul singur, ceea ce este clar ca aceasta este o problemă cu care părinții se confruntă. De aceea sistemul SensorSafe avertizează printr-o notificare către smartphone dacă copilul a desfăcut centura de siguranță.

De asemenea SensorSafe monitorizează și temperatura din interiorul mașinii iar atunci când temperatura scade sau crește prea mult trimite o alertă pe telefon care te notifică dacă temperatura din mașină a ajuns sau a depășit 35 grade Celsius sau a scăzut sub 7 grade Celsius.

O altă caracteristică importantă al acestui dispozitiv este că avertizează șoferul în cazul în care acesta părăsește mașina iar scaunul copilului este încă blocat cu acesta în el. Utilizatorul va primi o notificare pentru a-i anunța că copilul este încă în mașină, nesupravegheat.

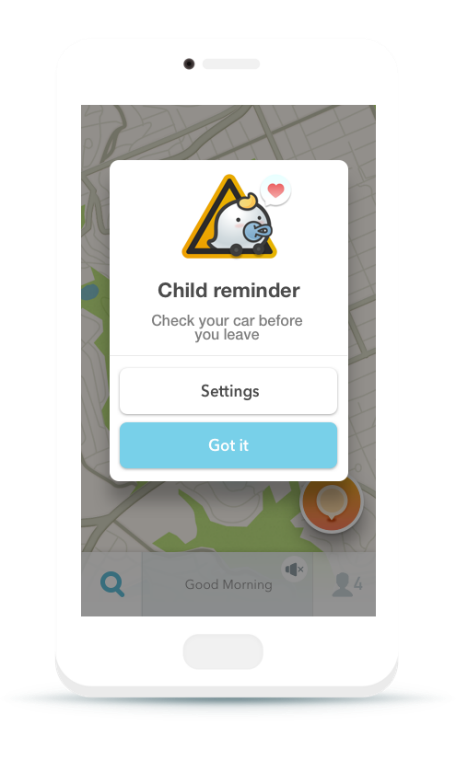
Pentru a evita momentul în care copilul stă în mașină mai mult timp de cât este considerat sigur pentru acesta, dispozitivul SensorSafe trimite o alertă utilizatorului în care îi sugerează să ia o pauza deoarece copilul a stat pe scaun prea mult timp. Dispozitivul recomandă ca utilizatorul să ia copilul și să îl scoată din scaun pentru al plimba.[[10]](#footnote-10)

I.5 Waze

Celebra aplicație pentru navigare prin GPS, Waze încearcă să prevină aceste tragedii prin dezvoltarea unei noi caracteristici care se numește „child reminder”. Această setare va trimite o notificare atunci când utilizatorul ajunge la destinație pentru ai reaminti să verifice bancheta din spate pentru a nu uita copilul în mașină.

Waze a devenit popular deoarece arată în timp real condițiile de trafic și reglează automat ruta pentru a ajunge în timp cât mai scurt la destinație, fără a fi nevoie ca șoferul să interacționeze cu telefonul pentru reconfigurarea traseului. În plus, dacă te întâlnești cu prietenii, odată ce ai plecat din casa, aplicația îi ține la curent pe prietenii care o folosesc, și nu mai este nevoie ca ei să fie notificați.

Noua funcție de notificare pentru copii nu este disponibilă la nivel global, ci doar în anumite țări. Utilizatorul va putea să personalizeze notificările, ceea ce înseamnă ca utilizatorul va putea seta aplicația pentru ai reaminti să verifice scaunul din spate pentru bebeluși sau pentru orice alt lucru trebuie să și-l amintească șoferul când ajunge la destinație.[[11]](#footnote-11)



*Fig.1.4 Noua caracteristică a aplicației Waze*

Există multe alte metode eficiente pe care părinții le folosesc pentru ași aminti să verifice bancheta din spate înainte de a ieși din mașina, cum ar fi să lase geanta bebelușului pe scaunul pasagerului din față. Folosind aplicația Waze cu această funcționalitate, este încă o metoda de a împiedica incidente nedorite.

Capitolul II – Python, pachetele Python și tehnologiile folosite în realizarea aplicației

II.1 Istoric și delimitări conceptuale

*Python* este un limbaj de programare dinamic multi-paradigmă, un limbaj orientat obiect, cu semantică dinamică, utilizat pentru programarea cu scop generat. Python a fost creat de olandezul Guido van Rossum, și a fost lansat pentru prima data în data de 20 Februarie 1991. Numele acestui limbaj nu a fost ales făcând referire la șarpele piton ci a fost ales după un serial de comedie difuzat de BBC numit *Monty Python’s Flying Circus.*

*Python* este un limbaj multifuncțional folosit de exemplu de către companii ca *Google* sau *Yahoo!* pentru programarea aplicațiilor web, însă există și o serie de aplicații științifice sau de divertisment programate parțial sau în întregime în *Python*. Popularitatea în creștere, dar și puterea limbajului de programare Python au dus la adoptarea să ca limbaj principal de dezvoltare de către programatori specializați și chiar și la predarea limbajului în unele medii universitare. Din aceleași motive, multe sisteme bazate pe Unix, inclusiv Linux, BSD și Mac OS X includ din start interpretatorul CPython.” [[12]](#footnote-12)

Fiind un limbaj de programare ușor de învățat, cu o sintaxă mult mai simpla față de celelalte limbaje de programare, rapid și puternic, a făcut ca acesta să fie ales de multe companii de succes pentru dezvoltarea back-end-ului site-urilor pe care le dețin, folosind Python și framework-urile pe care acesta le deține. De exemplu *Instagram*, cea mai mare aplicație online de partajare a fotografiilor, *Google* motorul de căutare pe care milioane de oameni din întreaga lume îl folosește, *Spotify*, o aplicație care permite utilizatorului să redea muzica, *Netflix* cea mai importantă rețea de televiziune online sau *Uber*, serviciul de ridesharing folosit în zeci de țări.

II.2 Caracteristicile limbajului *Python*

Printre principalele caracteristici ale limbajului *Python* se numără următoarele:

* *Ușor de programat*: Python este un limbaj de programare de nivel înalt. Python este foarte ușor de învățat comparativ cu alte limbaje de programare ca C, C#, Javascript, etc. Este foarte ușor de programat în Python și oricine poate învăța lucrurile de baza în câteva ore sau zile.
* *Gratis și open source*: Limbajul de programare este disponibil gratuit pe pagina oficială și poate fi descărcat de oricine. Fiind open source, înseamnă ca codul sursa este de asemenea disponibil pentru public.
* *Limbaj orientat obiect*: Una dintre caracteristicile principale ale acestui limbaj este programarea orientată pe obiect. Python suportă limbajul orientat pe obiect și conceptele de clase, încapsularea obiectelor și altele.
* *Limbaj de nivel înalt*: Când se scriu programe în Python, nu trebuie să se țină cont de arhitectura sistemului și nici nu trebuie să se gestioneze memoria sistemului*.*
* *Caracteristici extensibile*: Se poate scrie cod Python în limbajul C sau C++ și de asemenea se poate compila acel cod în limbajul C sau C++.
* *Python este un limbaj portabil*: Daca codul Python este scris în sistemul de operare Windows putem executa același cod pe alte platforme ca Linux, Unix sau Mac și nu este necesar să aducem schimbări codului.
* *Suport de programare pentru GUI*: Interfețele grafice pentru utilizator pot fi realizate folosind pachetele oferite de Python cum ar fi PyQt5, PyQt4, wxPython sau TK/
* *Biblioteci standard*: Python deține o mulțime de librării standard care oferă un set de module și funcții pentru ca programatorul să nu fie nevoit să scrie cod pentru fiecare lucru pe care dorește sa îl implementeze.
* *Limbaj dinamic*: Tipul unei variabile este decis la executare nefiind necesar să fie specificat tipul variabilei.[[13]](#footnote-13)

II.3 De ce Python?

Python este printre cele mai vechi limbaje de programare, fiind de asemenea și foarte populari printre dezvoltatorii de software. Python este open source, deci codul sursă este pus la dispoziție gratuit și poate fi modificat după nevoile fiecăruia.

Fiind open source, acest limbaj are parte de o comunitate foarte mare de oameni care îl folosesc. Avantajul acestui aspect este că atunci când scrii un program în Python și întâmpini o problemă poți cauta rapid prin documentația acestuia, sau o soluție mult mai rapidă ar fi căutarea acelei probleme într-un forum online, în care se discută despre Python.

Limbajul de programare Python este un limbaj dinamic, ceea ce înseamnă ca orice cod nou poate fi executat în timpul rulării, fără nici o limitare. Nu este necesar procesul de dezvoltare pentru a adauga câteva linii de cod.

Un alt mare avantaj în alegerea limbajului de programare Python în defavoarea altor limbaje este acela că Python este aprobat de către oamenii de știința. Daca Python este utilizat în mod eficient de către oamenii de știința, înseamnă ca Python poate deveni rapid unul dintre cele mai utilizate și eficiente limbaje de programare din întreaga lume.

Un alt motiv pentru a alege Python este că acesta folosește mult mai puține resurse față de alte limbaje de programare. Chiar și o echipă mică poate lucra eficient la un proiect dacă utilizează Python ca limbaj principal pentru acel proiect. Practic, orice program scris în Python are nevoie de mult mai puțin cod scris decât același program scris în alt limbaj de programare, de exemplu C++.

În concluzie, având atâtea argumente putem înțelege de ce tot mai multe companii mici și mari încep să utilizeze Python în defavoarea altor limbaje de programare.[[14]](#footnote-14)

II.4 Python 3.x vs Python 2.7.x

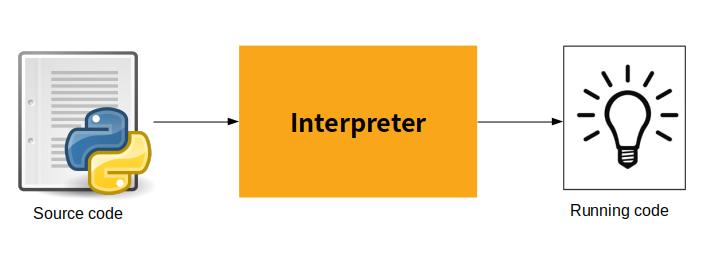
Python 3 a fost introdus in 2008 cu scopul de a face Python un limbaj de programare mai ușor de utilizat și de a schimba modul în care șirurile de caractere sunt gestionate pentru a corespunde cerințelor actuale.

Alegerea unei versiuni de Python se reduce la alegerea librăriilor de care dezvoltatorul are nevoie. Python 2 este mult mai vechi decât noua versiune, Python 3, acesta fiind un avantaj deoarece nu toate librariile disponibile in Python 2 au fost portate pe Python 3. Versiunea de Python 3 este considerată superioară celeilalte versiuni deoarece multe probleme descoperite la versiunea de Python 3 au fost rezolvate în noua versiune. Python 3 este de asemenea mult mai ușor de învățat și de înțeles. [[15]](#footnote-15)

II.5 Cum execută Python codul

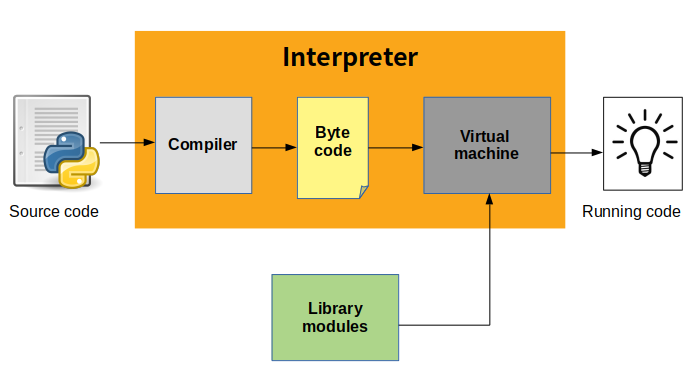
Interpretorul nu este altceva decat un software care execută scriptul Python și poate fi implementat în orice limbaj de programare. Unul dintre principalele interpretoare ale limbajului Python este CPython, care este scris în limbajul de programare Python. Un alt interpretor este Jython acesta fiiind scris în limnajul de programare Java.

Din punctul de vedere al unui programator un interpretor este un simplu software care execută codul sursa linie cu linie.



*Fig.2.1 Interpretor din punct de vedere al programatorului*

Din punctul de vedere al limbajului de programare Python, interpretorul este format din doua părți, compailerul și o mașină virtuală.



*Fig.2.2 Interpretor din punct de vedere al limbajului*

II.6 Programarea modulară în Python

Python este un limbaj de programare care dispune de două mecanisme care facilitează programarea modulara, acestea fiind modulele și pachetele Python.

Programarea modulară se referă la procesul de divizare a unui program, care conține foarte multe funcționalități cu mii sau chiar milioane de linii de cod, în subtaskuri sau module separate, mai mici și mai ușor de gestionat. Modulele individuale pot fi utilizate împreuna pentru a crea o aplicație mai mare.

Există mai multe avantaje în modularizarea codului într-o aplicație mare:

* *Simplitatea;*
* *Mentenabilitatea;*
* *Reutilizabilitatea;*
* *Definirea domeniului.*

Funcțiile, modulele și pachetele sunt construite în Python, și promovează modularizarea codului.[[16]](#footnote-16)

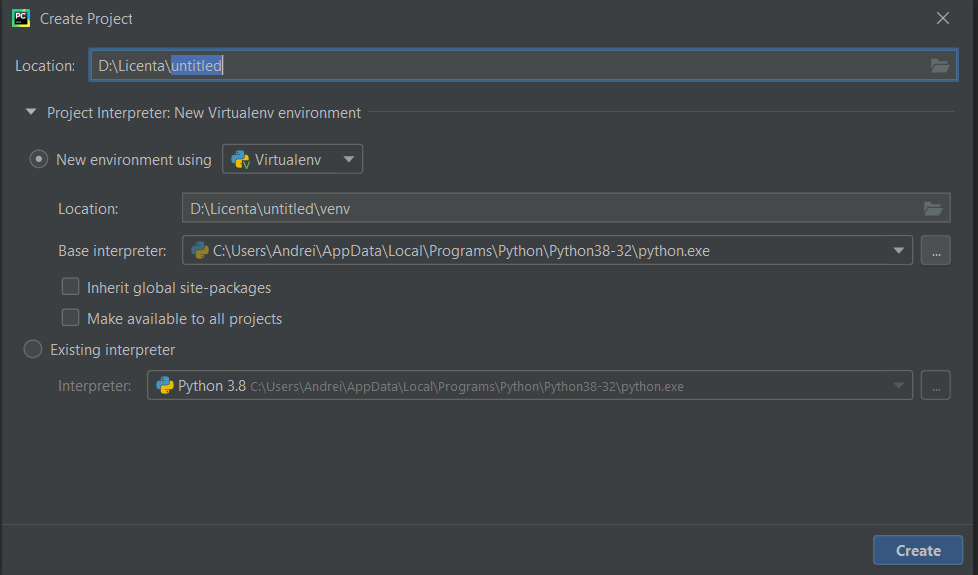
II.7 Interpretorul *PyCharm*

*PyCharm* este o platformă hibridă dezvoltată de Jeat Brains ca IDE pentru Python. Este frecvent utilizată pentru dezvoltarea aplicațiilor Python iar marile companii ca Twitter, Facebook, Amazon și Pinterest utilizează acest interpretor ca IDE Python.

Un *IDE* constă într-un editor și compilator folosit pentru a scrie și compila programe. Conține o combinație de funcționalități necesare dezvoltării software-ului.

Prezenta unui IDE face procesul de dezvoltare și programare mult mai ușor. Interpretează ceea ce programatorul scrie și sugerează un cuvânt cheie relevant pentru a fi inserat. Poate distinge între o clasă și o metodă, deoarece IDE-ul alocă culori diferite, și oferă de asemenea culori diferite pentru cuvinte cheie corecte și greșite.

Pycharm este un interpretor IDE care suportă ambele versiuni de Python, 2.x sau 3.x, și este compatibil cu sistemele de operare Windows, Linux sau Mac OS.



*Fig.2.3 Crearea unui nou proiect folosind PyCharm*

Putem selecta două opțiuni cu care interpretorul de proiect să lucreze. Opțiunea care creează un nou mediu cu numele de *Virtualenv* conține toate pachetele sau modulele care sunt deja instalate în mod implicit în interpretor. Cealaltă opțiune este direct Python, instalat pe mașina, care moștenește automat toate pachetele și modulele instalate.[[17]](#footnote-17)

II.8 Învățarea automată

Învățarea automată este un subdomeniu al informaticii și o ramură a inteligenței artificiale, al cărui obiectiv este de a dezvolta tehnici care dau mașinilor posibilitatea de a învață.[[18]](#footnote-18)

Domeniul învățării automate se suprapune cu cel al statisticii computaționale, deoarece cele două discipline se bazează pe analiza datelor.

Învățarea automată are o gamă largă de aplicații, inclusiv motoare de căutare, diagnosticare medicală, detectarea de frauda în utilizarea cardului de credit, clasificarea secvențelor de ADN, recunoaștere vocală și al limbajului scris, jocuri și chiar robotică. Învățarea automată are ca rezultat un model pentru a rezolva o anumită sarcină.

|  |  |  |
| --- | --- | --- |
| Modele geometrice | Modele probabilistice | Modele logice |
| - Sunt construite în spațiul de instanțe și pot avea unul, două sau mai multe dimensiuni;  - Dacă există o margine de decizie liniara între clase, se spune că datele sunt liniar separabile;  . - O limită de decizie liniară este definită ca w \* x = t, unde w este un vector perpendicular pe limita de decizie, x este un punct arbitrar de pe granița de decizie, iar t este pragul de decizie. | - Acestea încearcă să determine distribuția între probabilitatea de descripție pentru funcția care leagă valorile de caracteristici cu valorile determinate;  - Unul dintre conceptele cheie pentru a dezvolta modelele probabilistice este statistica bayesiana. | - Transformă și exprimă probabilitățile în normele organizate sub formă de arbori de decizie. |

*Tabel 2.1 Modele de rezolvare folosind învățare automată[[19]](#footnote-19)*

Modelele pot fi, de asemenea, clasificate ca *modele de grupară și modele de gradient.* Primele încearcă să împartă spațiul de cazuri în grupuri. Unele, cum numele lor indică, reprezintă un gradient în care se poate diferenția între fiecare instanță. Clasificatoarele geometrice, cum ar fi mașinile vectoriale de suport, sunt modele de gradient.

Învățarea automata conține diferiți algoritmi care sunt grupați într-o taxonomie, în funcție de popularitatea acestora. Aceste tipuri de algoritmi pot fi:

* *Învățarea supervizată total*: Algoritmul produce o funcție care stabilește o corespondență între intrările și rezultatele dorite de la sistem. Un exemplu de acest tip de algoritm este o problemă de clasificare, în cazul în care sistemul de învățare este de a eticheta (clasifica) o serie de vectori folosind oricare din mai multe categorii (clase). Baza de cunoștințe a sistemului este formată prin exemple de etichetat. Acest tip de învățare se poate dovedi a fi foarte utilă în domenii precum cercetările biologice, biologie computațională sau bioinformatică.
* *Învățarea fără supervizare*: Tot procesul de modelare se realizează pe un set de exemple format numai de intrări în sistem. Nu există informații despre categoriile de aceste exemple. Prin urmare, în acest caz, sistemul trebuie să fie capabil să recunoască modele, pentru a fi capabil de a eticheta noi intrări.
* *Învățarea supervizată parțial*: Acest tip de algoritmi combină cei doi tipi de algoritmi de mai sus cu scopul de a clasifica în mod corespunzător. Acesta ia în considerare date etichetate și neetichetate.
* *Învățarea prin armare*: Algoritmul învață prin observarea lumii din jurul lui. Informațiile de intrare este răspunsul care îl primește de la lumea exterioară ca un răspuns la acțiunile lor. Obiectivul principal este de a învăța funcția de valoare care va ajuta agentul inteligent pentru a maximiza semnalul de recompensă și astfel să poată optimiza politicile lor în așa fel încât să înțeleagă comportamentul mediului și de a lua decizii bune pentru îndeplinirea obiectivelor sale.
* *Transducție:* Similar cu învățarea supervizată, dar nu constituie în mod explicit o funcție. Încearcă a prezice categoriile de viitor ale exemplelor, bazându-se pe exemplele de intrare, respectivele categorii și exemplele noi al sistemului. [[20]](#footnote-20)
* *Învățare multi-task*: Metode de învățare care utilizează cunoștințe anterior învățate de către sistemul de față-în-față cu probleme similare cu cele deja văzute.[[21]](#footnote-21)

II.9 Învățarea automată folosind Python

Python este ales de către dezvoltatori pentru o serie întreagă de aplicații, dar motivele pentru care acesta este ales pentru proiectele care implică inteligența artificială sau învățarea automată sunt simplitatea, multe librarii și framework-uri sau suport din partea comunității mari.

|  |  |  |
| --- | --- | --- |
| Librării și Framework-uri | Simplitatea | Suport |
| - Multitudinea de librării și framework-uri care facilitează codul și salvează timpul de dezvoltare. | - Python este renumit pentru codul sau concis, lizibil și este aproape de neegalat când vine vorba de ușurința în utilizare și simplitate;  - Învățarea automată se bazează pe algoritmi extrem de complecși și fluxuri de lucru în mai multe etape, astfel încât cu cât un dezvoltator trebuie să se concentreze mai puțin pe complexitatea codului, cu atât mai mult se poate concentra pe găsirea soluțiilor la probleme;  - Sintaxa simplă față de celelalte limbaje de programare permite dezvoltatorului să testeze rapid algoritmi fără a fi nevoie să ii implementeze;  - Codul ușor de citit nu va pune probleme în cazul în care la un proiect lucrezeă mai mulți programatori sau echipe. | - Fiind un limbaj de programare open source, are o comunitate mare și o documentație de înalta calitate. |

*Tabel 2.2 Argumente pentru a utiliza Python împreună cu învățarea automată*

II.10 Pachetul Python scikit-learn

*Scikit-learn* a fost inițial dezvoltat de *David Cournapeau* într-un internship de vară pe care acesta la avut la Google în 2007. Mai târziu *Matthieu Brucher* s-a alăturat echipei și a continuat dezvoltarea acestui pachet. Acum acest proiect are mai mult de 30 de contribuitori și multe companii îi sponsorizează, ca Google, INRIA sau Tinyclues, pentru ca aceștia să dezvolte în continuare acest pachet Python.[[22]](#footnote-22)

Scikit-learn oferă o gamă largă de algoritmi de învațare supervizată și nesupervizata printr-o interfața consistentă Python. Aceasta are o licență permisiva BSD simplificată și este distribuită în multe distribuții Linux, încurajând utilizarea academică și profesională.

Biblioteca este construită pe SciPy (Scientific Python) care trebuie instalate înainte de a putea folosi scikit-learn. Viziunea acestei biblioteci este un nivel de robustețe și suport necesar pentru utilizarea în sisteme de producție. Acest lucru se concentrează pe probleme precum ușurința utilizării și învățării codului, calitatea codului, documentația și performanța.

Deși interfața este Python, bibliotecile C sunt pârghii pentru performanță, cum ar fi pachetele numpy pentru matrici și operații cu matrici, și utilizarea atentă a CPython.

Biblioteca este axataă pe modelarea datelor. Nu este axată pe încărcarea, manipularea și rezumarea datelor.

Pachetul scikit-learn furnizează unele grupuri de modele:

* *Clustering*: pentru gruparea datelor neetichetate.
* *Validarea încrucișată (Cross Validation)*: Pentru estimarea performanței modelelor supervizate pe date nevăzute.
* *Seturi de date (Dataset)*: Pentru seturi de date de testare și pentru generarea de seturi de date cu proprietăți specifice pentru investigarea comportamentului modelului.
* *Reducerea dimensiunii (Dimensionality Reduction)*: Pentru reducerea numărului de atribute din date pentru sumarizare, vizualizare și selectarea caracteristicilor, cum ar fi analiza componentelor principale.
* *Metode de ansamblu (Ensemble methods)*: Pentru combinarea predicțiilor mai multor modele de supervizare.
* *Extragerea caracteristicilor (Feature extraction)*: Pentru definirea atributelor în datele de imagine și text.
* *Selectarea caracteristicilor* *(Feature selection):* Pentru identificarea atributelor semnificative din care se creează modele supervizate.[[23]](#footnote-23)
* *Reglarea parametrilor (Parameter Tuning)*: Pentru a beneficia la maximul de modelele supervizate.
* *Învățarea multiplă (Manifold Learning)*: Pentru sumarizarea și reprezentarea datelor complexe multi-dimensionale.
* *Modele supervizate (Supervised Models)*: O gamă vastă nelimitată de mulțimi pentru a generaliza modele liniare, pentru analize discriminatorii, *„naive bayes”*, metode lente, rețele neuronale, mașini vectoriale și arbori de decizie.

II.11 Pachetul Python librosa

*Librosa* este un pachet Python folosit pentru procesarea semnalului audio și a muzicii. Acest pachet încearcă să ușureze tranziția cercetărilor MIR (Music Information Retrievel) in Python (si practicile moderne de dezvoltare software) și, de asemenea, pentru a face tehnicile MIR de bază disponibile pentru o comunitate mai larga de oameni de știința și programatori Python.

Concepte cheie în proiectarea pachetului Python *librosa*:

* Dezvoltarea pachetului a avut în vedere ca cercetătorii care lucrează cu MATLAB să poată învață cu ușurință să înțelegă acest pachet.
* Standardizarea interfețelor, a numelor de variabile și a setărilor de parametri în diferite funcții de analiză. Acest lucru a fost complicat de faptul că implementările de referință din care derivă implementările din *librosa* provin de la diverși autori și sunt concepute ca scripturi unice și nu ca niște funcții care aparțin unei biblioteci, cu interfețe bine definite.
* Funcțiile din pachetul *librosa* sunt concepute pentru a fi modulare, permițând programatorilor să acceseze funcțiile dorite din pachet atunci când aceștia au nevoie. Acest lucru permite cercetătorilor să valorifice funcțiile pachetului existente în timp ce experimentează îmbunătățiri ale altor componente specifice.[[24]](#footnote-24)
* Dezvoltatorii pachetului încearcă să obțină un cod cat mai lizibil, o documentație amănunțita și o testare exhaustiva. Toate aceste informații sunt open source și sunt disponibile pe GitHub. Toate funcțiile sunt implementate in Python, bine documentate folosind Sphinx și includ un exemplu de cod care demonstrează utilitatea. Implementarea îndeplinește in mare parte recomandările PEP-8.

În general, funcțiile pachetului *librosa*  tind să expună toți parametrii relevanți celui care încearcă să îi apeleze. Deși acest lucru oferă o flexibilitate utilizatorilor experți, poate fi mai complicat pentru un utilizator începător care are nevoie de o interfața consistentă pentru a procesa fișiere audio. Pentru a satisface ambele nevoi, au fost definite un set de convenții generale și valori standardizate ale parametrilor impliciți împărțiți între mai multe funcții. Un semnal audio este reprezentat ca o matrice unidimensionala *numpy*, notata ca *y* in *librosa*. De obicei, semnalul *y* este însoțit de o rată de eșantionare (notata *sr*) care denota frecventa în *Hz* la care sunt eșantionaje valorile lui *y*. Durata unui semnal poate fi calculată prin împărțirea numărului de eșantioane la rata de eșantioane.

În mod implicit, la încărcarea unui fișier audio stereo, funcția *librosa.load()* amestecă la mono, calculând media canalelor din stânga și dreapta, iar apoi reasamblează semnalul monofonic la rata implicita *sr=22050 Hz.* Majoritatea metodelor de analiză audio funcționează nu la frecvența de eșantionare nativă a semnalului, ci pe cadre mici ale semnalului.

Majoritatea funcțiilor de analiză implementate de *librosa* produc ieșiri bidimensionale stocate ca *numpu.ndarray*, de exemplu, *S[f, t]* ar putea conține energia dintr-o anumită banda de frecvență *f* la indexul de cadru *t*. Matricele de caracteristici sunt organizate în coloană majoră (stil Fortran) în memorie, astfel încât modelele de acces comune beneficiază de localitatea cache.[[25]](#footnote-25)

II.12 Pachetul Python tkinter

Tkinter este o legătura între Python și setul de instrumente Tk GUI. Este interfața standard Python cu setul de instrumente Tk GUI, și este standardul de facto GUI. Tkinter este inclus în instalările standard de Python pentru Linux, Windows și Max OS.

La fel ca majoritatea celorlalte legături Tk moderne, tkinter este implementat ca un wrapper Python în jurul unui interpret Tcl complet incorporat în interpretul Python. Apelurile tkinter sunt traduse în comenzi Tcl care sunt încărcate către acest interpret incorporat, făcând astfel posibilă combinarea Python și Tcl într-o singură aplicație[[26]](#footnote-26)

II.13 Pushbullet

Pushbullet este un serviciu web care conține o aplicație de mobil, Android sau iOS, prin care, cu ajutorul unei librării Python pe nume *pushbullet.py* se pot trimite notificări din scriptul Python către un dispozitiv mobil.

Cu ajutorul pachetului *pushbullet*, programatorul poate trimite din programul sau Python notificări, fișiere și chiar imagini.

Transferul acestora se face folosind protocolul API, cu ajutorul căruia se realizează transferul către aplicația de mobil Pushbullet. Notificările ajung să fie afișate pe ecran iar fișierele sau imaginile sunt descărcate în telefon cu aprobarea utilizatorului de mobil.

De asemenea pentru a folosi protocolul API utilizatorul are nevoie de o cheie API care aparține aplicației Pushbullet, cu ajutorul căreia Python poate comunica cu telefonul. Cheia API este calea de acces securizată pentru transferul dintre programul Python și telefon. [[27]](#footnote-27)

Capitolul III – Aplicație de recunoaștere și clasificare a sunetelor din mașină

III.1 Scopul și obiectivele care stau la baza dezvoltării aplicației

În acest capitol voi prezenta implementarea aplicației pentru aceasta temă, arhitectura aplicației, modul de implementare și scenariile de funcționare. De asemenea voi prezenta cum am gândit aplicația în întregime și cum poate fi aceasta folosită.

Aplicația se bazează pe recunoașterea sunetelor dintr-un fisier audio și analiza acestora pentru a identifica prezența unui copil care plânge sau unui animal de companie, un câine care latră sau pisica care miaună.

Fiind o aplicație destinată pentru a fi utilizată pentru masină voi asocia interfața și inregistrările cu mediul care se regăsește in mașina. Aplicația se dorește a fi o simulare pentru interiorul automobilului, atât interfața acesteia cât și fișierele audio care există sau care se înregistrează.

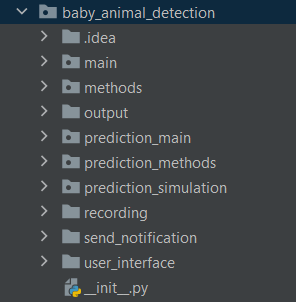
Am ales să dezvolt această aplicație care se bazează pe detectarea sunetelor deoarece această abordare nu a fost dezvoltată de nici un producător de automobile. Sistemele dezvoltate sau aflate în dezvoltare iîn acest moment se bazează pe senzori sau sisteme radar de detectare a semnelor de viață din mașină.

De obicei un copil uitat în mașină, atunci când se simte în pericol sau singur, începe să plângă. De asemenea un câine dacă este lăsat în mașină începe să latre. Bazându-mă pe aceste fapte am ales această abordare pentru a dezvolta un program care să alarmeze cu o notificare pe telefon șoferul autovehiculului ca și-a uitat copilul sau animalul de companie în mașină.

Scopul dezvoltării acestei aplicații este de a împiedica tragediile rezultate din urma uitării unui copil sau animal în mașină. Obiectivele urmărite sunt de a introduce o nouă soluție la pachetul de aplicații si thenologii din aria siguranței si protecției consumatorului , soluție ce poate fi ușor implementată și utilizată si poate combate situațiile critice.

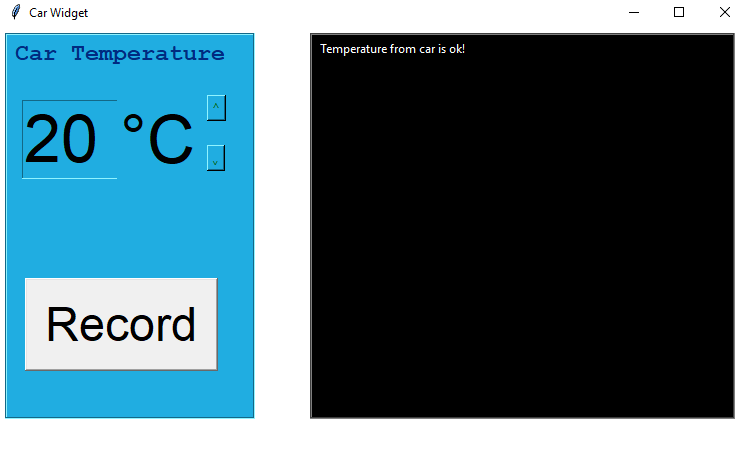
III.2 Prezentarea aplicației

Aplicația este împărțita pe mai multe directoare, această structură fiind mai ușor de înțeles sau de parcurs. Fiecare folder conține definirea unei clase sau a unei metode.



*Fig.3.1 Structura de directoare ale programului Python*

Aplicația a fost dezvoltată pentru a simula temperatura din interiorul unei mașini sau utilizatorul poate interacționa cu interfața, în caz că acesta si-a lăsat copilul sau animalul de companie în interiorul mașinii, pentru a verifica dacă copilul plânge sau câinele dacă latră.



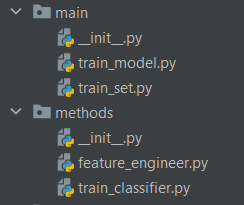
*Fig.3.2 Interfața aplicației*

Interfața ține locul unor senzori sau a unor microfoane, care ar trebui să fie instalate în mașină și care ar trebui să preia toate sunetele din interior. Cel mai important senzor care ar trebui să existe în interiorul automobilului este un senzor de temperatură, care ar trebui să declanșeze un lanț de evenimente de la înregistrarea sunetelor din mașină și până la recunoaștere și trimiterea unei notificări proprietarului în cazul în care acesta a uitat că a venit însoțit dar a plecat fără să verifice mașina.

III. 3 Pași preliminari în lansarea aplicației

Înainte de a putea folosi funcționalitatea din interfața aplicației și de recunoaștere a sunetelor din mașina, sunt necesari câțiva pași în prealabil. Acești pași sunt obligatorii și sunt necesari pentru a instrui un model de învățare automată care, în primul rând va citi datele și apoi va antrena un model.

Acesta model predictiv generat va fi folosit în pasul de recunoaștere. Pentru acest pas de antrenare este necesară o mașină puternică pentru ca această etapă să nu necesite o perioadă lungă de timp.



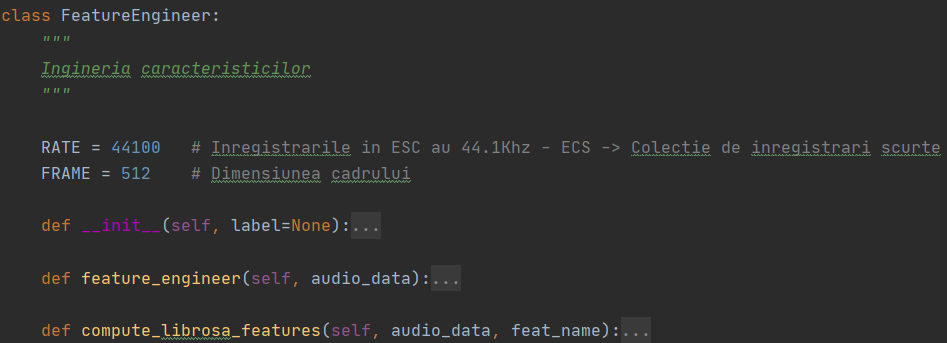
*Fig.3.3 Funcțiile și metodele folosite pentru antrenare*

Pentru a începe antrenarea avem nevoie de un set mare de date, mai exact, pentru programul dezvoltat avem nevoie de o mulțime de fișiere audio. Cu cât mai multe fișiere audio, cu atât modelul pe care îl antrenam să facă predicția va fi mai exact și ne va întoarce rezultatul așteptat. Fișierele audio trebuie să conțină sunete pe care le dorim ca programul nostru să le recunoască cum ar fi, recunoașterea unui copil care plânge, un câine care latră sau o pisica care miaună.

Această antrenare care generează modelul are de asemenea nevoie de fișiere audio care sunt diferite de cele pe care vrem să le recunoască programul nostru. Acestea pot fi diverse de la un fișier audio în care nu se aude nimic pana la sunete de ploaie, tunete sau sunete de mașini. Pentru a genera acest model predictiv folosim ingineria caracteristicilor.

Procesul de inginerie a caracteristicilor este procesul de selectare și transformare a caracteristicilor, proprietăților sau atributelor din date brute atunci când se creează un model predictiv folosind învățarea automată sau modelare statistică cum ar fi învățarea profunda, arbori de decizie sau regresia. Procesul implică o combinație de analiză a datelor, aplicarea unor reguli generale, hotărâre și uneori este denumit pre-procesare. Datele utilizate pentru a crea un model predictiv constau într-o variabilă rezultată, care conține date care trebuie să prezică rezultatul, o serie de variabile predictive care conțin date care trebuie să fie preconizate și o serie de variabile predictive care conțin date despre care se crede că trebuie să prezică rezultatul pentru variabila care trebuie prezică rezultatul.[[28]](#footnote-28)[[29]](#footnote-29)

Pentru a genera setul de date, în cazul nostru un set de date numpy, care vor fi folosite de model pentru a putea face recunoașterea, este nevoie să rulam fișierul *train\_set.py* unde avem o funcție principală care folosește metodele din directorul *methods*, fișierele Python *feature\_engineer.py* și *\_\_init\_\_*.

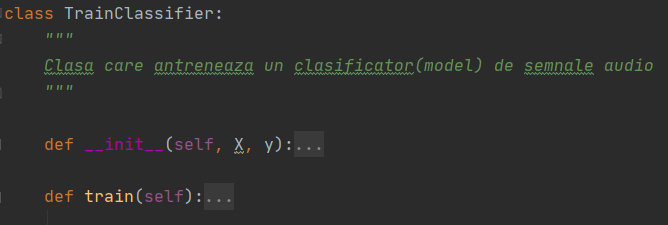


*Fig.3.4 Clasa FeatureEngineer din fișierul feature\_enginner.py*

Fișierul Python *feature\_engineer.py.* conține clasa *FeatureEngineer și* două funcții, *feature\_engineer și compute\_librosa\_features*. Pentru a genera setul de date se apelează funcția *feature\_engineer* care va returna o matrice care conține dintr-un fișier audio.

După executarea scriptului Python *train\_set.py* acesta va genera un set de date care va fi folosit în continuare pentru angrenarea modelului.

Următorul pas este antrenarea modelului, iar pentru acest pas trebuie să apelam scriptul *train\_model.py* care folosește metode din fișierul *train\_classifier.py* unde avem clasa *TrainClassifier*.



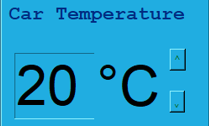
*Fig 3.5 Clasa TrainClassifier din fișierul train\_classifier.py*

Funcția *train* pe care scriptul *train\_model.py* îl folosește, generează modelul.

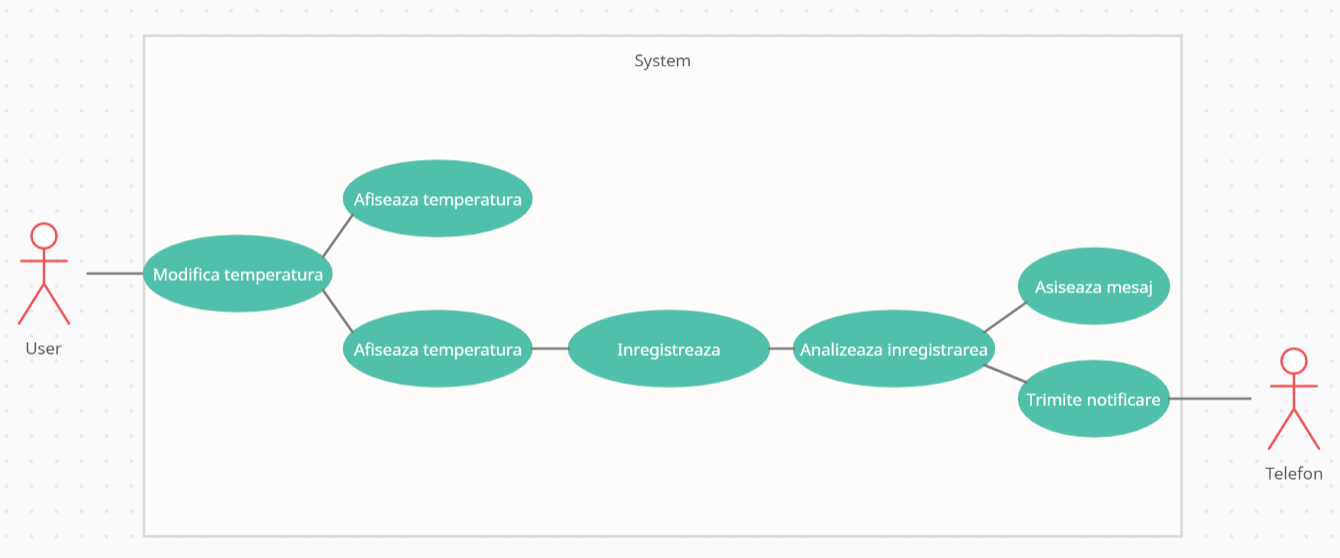
Un model de învățare automată este acel fișier care este instruit pentru a recunoaște un anumit tip de tipar. Instruirea unui model se face cu ajutorul unui set de date, în cazul nostru unor fișiere audio, și cu ajutorul pachetului Python *sklearn* care este o librărie software de învățare automată. Odată antrenat modelul, acesta este pregătit pentru a fi folosit pentru a face previziuni asupra unor date noi, pe care modelul nu a fost antrenat.

III.4 Funcționalitățile aplicației

Prima funcționalitate și cea mai importanta este simularea temperaturii din interiorul automobilului.



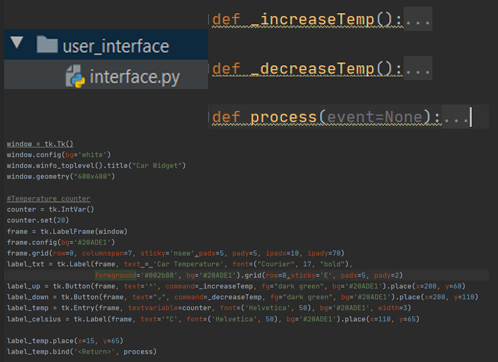
*Fig.3.6 Simularea temperaturii*



*Fig.3.7 Diagrama cazurilor de utilizare – Simularea temperaturii*

Utilizatorul poate modifica cu ajutorul săgeților temperatura. Daca utilizatorul dorește să crească temperatura acesta va trebui să apese cu ajutorul mouse-ului săgeata îndreptata în sus iar dacă acesta dorește să scadă temperatura va apasă pe săgeata îndreptata în jos. Utilizatorul mai poate introduce manual temperatura dorita în căsuța în care sunt afișate valorile numerice.

Înregistrarea și recunoașterea din interiorul automobilului se face doar în anumite intervale de temperatura definite în program.



*Fig.3.8 Fișierul python interface.py*

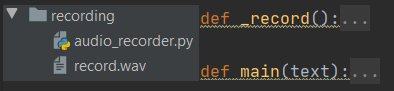
Fișierul *interface.py* este scriptul Python principal, care rulează acest program. Pe lângă codul care a ajutat la construcția interfeței avem funcțiile care ne ajuta să modificam temperatura. *\_increaseTemp și \_decreaseTemp* sunt funcțiile care ordonă creșterea și descreșterea temperaturii iar funcția *process* este construită pentru a putea procesa introducerea manuala a valorii. Aceste funcții generează acțiuni care duc la recunoașterea sunetelor din interiorul mașinii.

În interiorul fiecărei funcții sunt definite și limitele pentru care se vor lua acțiuni de către program.

|  |  |  |
| --- | --- | --- |
| **Interval** | **Mesaj** | **Notificare** |
| 100 .... 0  30 .... 100 | ALERT!!!  <<*valoare*>>°C in the car. Really, now I’m freezing!/ It's really hot now!  Baby cry!/Dog barking!/Cat meow!  Go to car now! | Yes |
| 0 .... 5  25 .... 27 | Warning!  <<valoare>>°C in the car. Now it's really cold!/ It's getting pretty hot here!  Baby cry!/Dog barking!/Cat meow!  You should hurry up, your BABY is in the car!/You should hurry up, your CAT is in the car!/You should hurry up, your DOG is in the car! | Yes |
| 5 .... 10  25 .... 27 | Notification!  <<valoare>>°C in the car. It's a little too cold!/ It's getting warm!  Baby cry!/Dog barking!/Cat meow!  Don't forget that your BABY is in the car!/Don't forget that your DOG is in the car!/Don't forget that your CAT is in the car! | Yes |
| 10 .... 24 |  | NO |

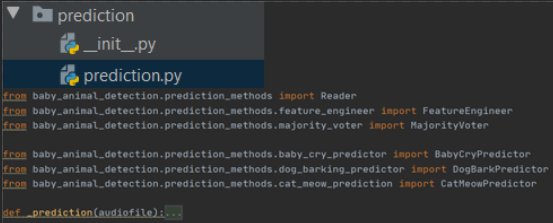
*Tabel 3.1 Mesaje de notificare*

Atunci când se ating intervalele critice unde se consideră că șoferul automobilului ar trebui notificat, pentru al alerta sau pentru a-i aduce aminte că a lăsat pe cineva în mașină în primul rând se va apela din fișierul *audio\_recorder.py* funcția *\_record* care va înregistra pentru câteva secunde sunetele din interiorul mașinii. Acest sunet este trecut printr-un proces de recunoaștere pentru a se verifica dacă există semne de viață în mașină, dacă posesorul automobilului și-a uitat sau lăsat bebelușul în mașină acesta e posibil să plângă, în caz ca și-a uitat sau lăsat cățelul sau pisica aceștia pot scoate sunete, cățelul poate latră sau pisica mieuna. În cazul în care unul dintre aceste sunete este recunoscut de către program, și dacă temperatura se încadrează într-un interval prezentat în tabel, posesorul automobilului va primi o notificare pe telefonul personal cu mesajul și alerta aferentă intervalului de temperatura. În cazul în care nici un sunet nu este înregistrat, nu va fi trimisă nici o notificare.



*Fig.3.9 Fișierul python audio\_recorder.py*

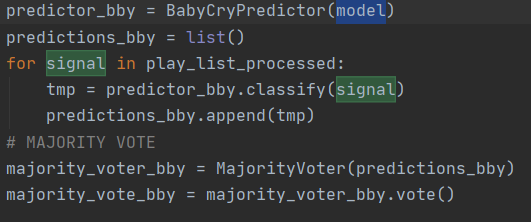
Funcția *main* din fișierul *audio\_recorder.py* gestionează în continuare recunoașterea pe fișierul audio pe care îl generează cu funcția \_record. Pentru a putea primi un rezultat în urma înregistrării fișierului audio, recunoașterea se face apelând scriptul Python *prediction.py*.



*Fig.3.10 Fișierul python prediction.py*

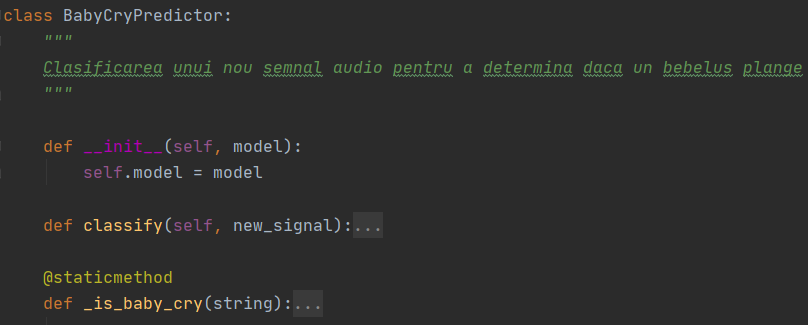
Funcția din fișierul *prediction.py, \_prediction* folosește pentru a putea face recunoașterea vocală, metodele definite pentru o anumită categorie, de exemplu, recunoașterea plânsului unui copil se folosește fișierul Python *baby\_cry\_redictor.py* care are definită o funcție *BabyCryPredictor* cu metodele sale septice care ajută la recunoaștere. Pentru recunoașterea lătratului unui câine folosim clasa *DogBarkPredictor* din *dog\_bark\_predictor.py* iar pentru recunoașterea mieunatului unei pisici avem clasa *CatMeowPredictor* din *cat\_meow\_predictor.py*.

Pentru a verifica dacă în urma înregistrării fișierului audio din mașină, există un copil care plânge, vom folosi metoda *\_prediction*.



*Fig.3.11 Codul pentru a verifica dacă un copil plânge*

In figura 3.11 putem vedea cum se creează un obiect folosind clasa *BabyCryPredictor* folosind un model deja antrenat pentru a face previziunea.

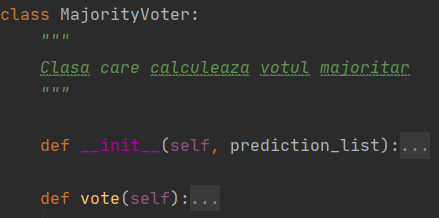


*Fig.3.12 Clasa BabyCryPredictor*

Fișierul audio care intra în procesul de recunoaștere prima oară este împărțit în cadre de către funcția *feature\_engineering* din clasa *FeatureEngineer*. Fiecare cadru al fișierului audio trece prin procesul de predicție, mai exact este apelată clasa *classify* pentru fiecare cadru și cu ajutorul modelului antrenat în prealabil se face predicția. Funcția *classidy* va returna 1 în caz că a recunoscut în acel cadru din fișierului audio dacă un bebeluș plânge iar daca nu, va returna 0.

Același proces se execută pentru fiecare cadru din fișierul audio iar la final vom avea o listă cu predicții care conțin 1 și 0.

Ultima etapă înainte ca programul să returneze dacă există în fișierul audio un copil care plânge sau nu se face folosind clasa *MajorityVoter*.

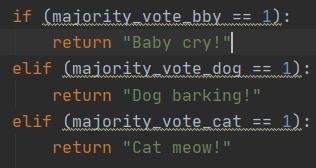


*Fig.3.13 Clasa MajorityVoter*

Folosind lista de previziuni care conține valori de 1 și 0 construim un obiect folosind clasa *MajorityVoter* căreia îi atribuim lista cu valori. Din această clasă, pentru a calcula votul majoritar din listă, apelăm funcția *vote* care va returna valoarea majoritară.

Fișierul audio va trece prin același proces de predicție pentru a fi verifica dacă nu există alt sunet înafara de cel al unui copil care plânge. Acesta este definit tot în fișierul *prediction.py.* Se verifică dacă în fișierul audio nu există un câine care latră sau o pisică care miaună. Pentru aceste cazuri vom folosi fișierele *dog\_bark\_predictor.py* care conțin clasa *DogBarkPredictor și CatMeowPredictor* din *cat\_meow\_predictor.py*. Aceste două clase sunt apelate asemănător ca și în cazul clasei *BabyCryPredictor*. De asemenea acestea sunt construite la fel ca și *BabyCryPredictor*. Ceea ce diferențiază predicția între acestea este modelul cu care este instanțială această clasă. Modelul va face diferența atunci când va prezice cu exactitate ce se aude mai exact în acel fișier audio.

După ce fișierul audio a trecut prin fiecare clasă prin care s-a făcut prezicerea există un set de valori de 1 sau 0 pentru fiecare clasă în parte.



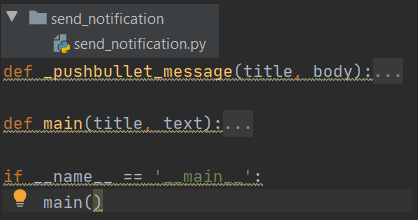
*Fig.3.14 Verificarea predicției*

Departajarea se face cu o verificare a valorilor, mai exact pentru care prezicere avem valoarea 1. Pentru acea valoare care este 1 se va întoarce mesajul aferent valorii. În cazul în care nu există valoarea 1 pentru nici o prezicere, programul nu va returna nici un mesaj.

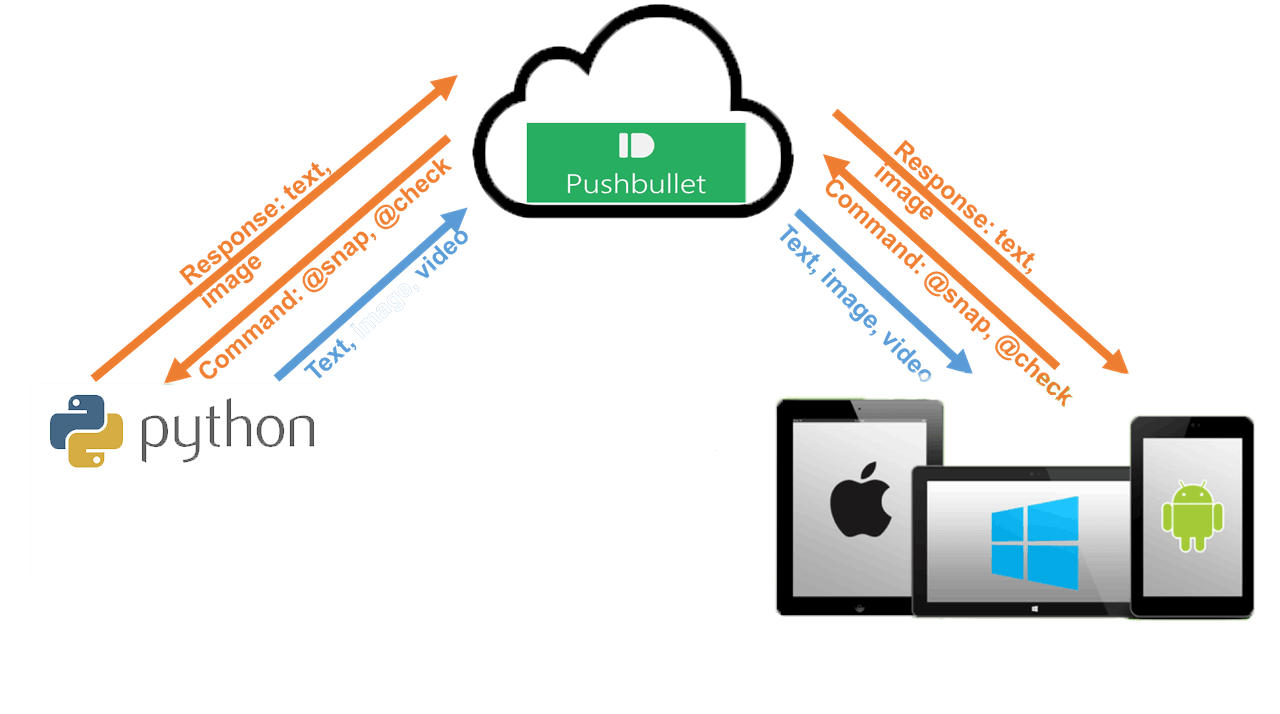
Prezicerea fișierului audio va fi folosita în *audio\_recorder.py* unde se verifică dacă există o recunoaștere sau nu.

Luând în considerare temperatura simulată și dacă există o recunoaștere pentru unul dintre cazurile definite va fi trimis un mesaj de notificare pe telefonul șoferului bazat pe cele 2 criterii.

Pentru a trimite acest mesaj de notificare folosim aplicația Pushbullet și fișierul Python *send\_notification.py*.



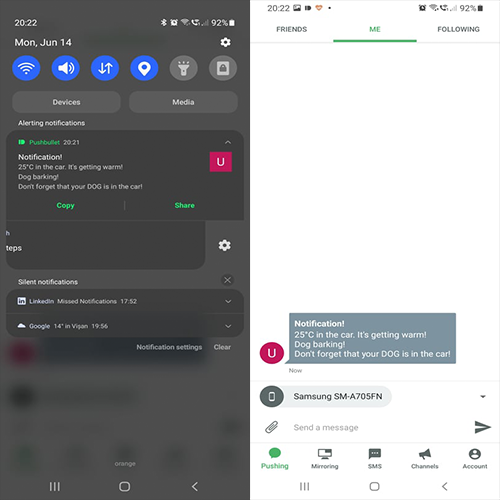
*Fig.3.15 Fișierul Python send\_notification.py*



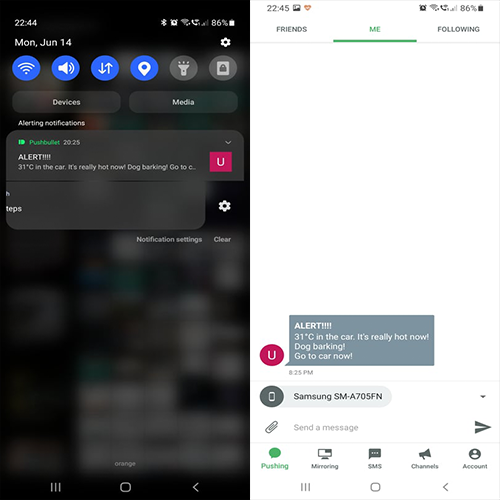
*Fig.3.16 Trimiterea notificărilor de catre programul Python*

Funcția *main* din fișierul *audio\_recorder.py* apelează funcția ­*\_pushbullet\_message* care trimite notificarea pe telefon.

Acesta este ultimul pas în procesul de recunoaștere începând de la modificarea temperaturii din interiorul mașinii și până la recunoașterea făcută pe fișierul audio cu notificările trimise la final în caz că a fost îndeplinit o condiție de recunoaștere.

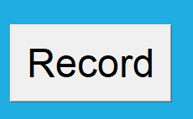


*Fig.3.17 Notificarea și mesajul primit pe telefonul mobil*

****

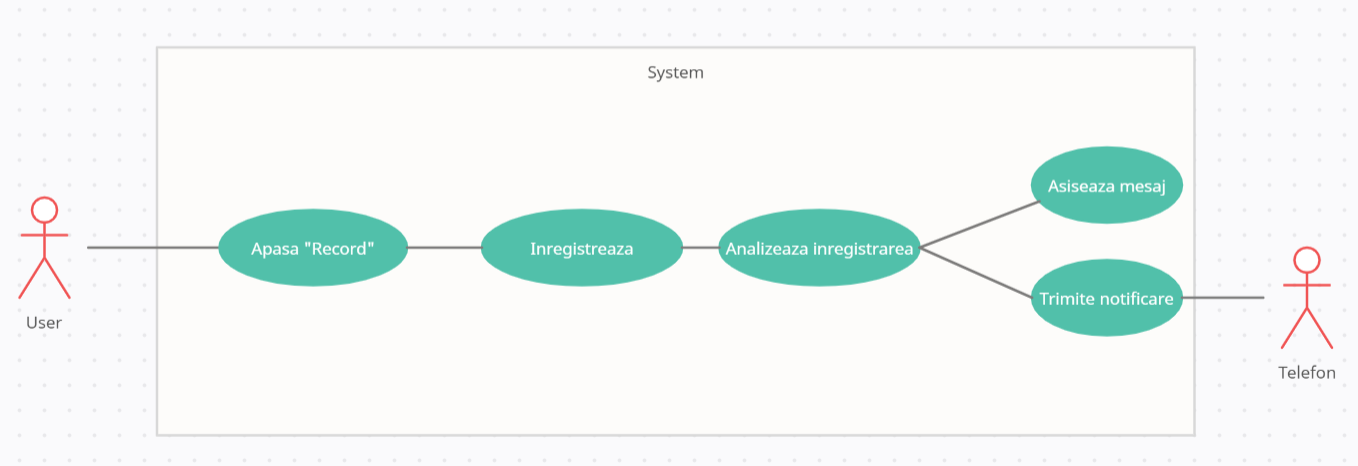
*Fig.3.18 Alerta și mesajul primit pe telefonul mobil*

O altă funcționalitate care este foarte folositoare în momentul în care utilizatorul își lasă intenționat copilul, câinele sau pisica în mașină este cea de înregistrare automată.



*Fig.3.19 Butonul care înregistrează din interfața programului*

Această funcționalitate poate ajuta utilizatorul, de exemplu, dacă acesta își lasă copilul dormind în mașină pentru a face altă activitate pentru o perioadă scurtă de timp pentru a verifica dacă copilul nu s-a trezit și a început să plangă.



*Fig.3.20 Diagrama cazurilor de utilizare – Înregistrare manuala*

Singura diferență în procesul de recunoaștere, față de primul caz de utilizare unde acest proces este pornit de către temperatura din interiorul mașinii, este că utilizatorul intervine și din dorința lui pornește înregistrarea. Restul procesului de recunoaștere de la împărțirea fișierului în cadre până la previziune și trimiterea notificărilor pe telefon este același cu cel care este pornit de simularea temperaturii.

Concluzii și direcții de dezvoltare viitoare

Indiferent de tehnologie sau de aplicație această problemă reală cu care se confruntă omenirea este cât se poate de reală. Majoritatea companiilor producătoare de mașini s-au unit pentru a veni în întâmpinarea acestei probleme și de a găsi o rezolvare comună sau individuală a acesteia pentru a împiedica aceste incidente, care, din păcate se întâmpla din ce in ce mai des.

Lumea se află într-o continua schimbare și dezvoltare, un om obișnuit are atât de multe lucruri de făcut într-o zi și atât de puțin timp, încât uită de lucrurile importante din viață. Acesta este și motivul pentru care aceste tragedii se tot întâmpla. Dar și tehnologia este în continuă dezvoltare și trebuie să vină cu soluții în rezolvarea acestei probleme.

Ideea dezvoltată și prezentată în lucrarea de licență dorește să aducă un aport important în rezolvarea acestei probleme. Fiind o idee puțin întâlnită consider că poate aduce un ajutor considerabil. Această implementare pe care am prezentat-o în lucrarea de licența poate fi îmbunătățită.

În primul rând acest program poate fi integrat într-o mașina iar cu ajutorul microfoanelor din mașină, care pot fi amplasate în diferite parți ale interiorului mașinii. Testele efectuate în acest mediu sunt foarte importante deoarece recunoașterea făcută cu un fișier audio înregistrat efectiv din interiorul mașinii poate să nu mai fie așa precisă față de cazul prezentat în licență, unde toată acțiunea se întâmpla într-un mediu simulat.

Acest program de recunoaștere poate fi combinat cu alți senzori din mașina, de exemplu senzori radar, pentru a avea mai multe verificări. În cazul în care nu se poate face recunoașterea pe baza unei înregistrări din mașină, senzorii radar pot fi soluția de rezervă, sau invers.

O îmbunătățire importantă pe care programul dezvoltat o poate primi este antrenarea acestuia pentru a putea face diferența dintre sunetele redate de la un radio sau o boxa, sau chiar sunetele din exteriorul mașinii și cele din interior. Această îmbunătățire poate elimina cazul în care aplicația trimite notificări eronate deoarece a înregistrat un sunet din exteriorul mașinii.

Ideea prezentată în această lucrare poate fi o soluție sau o idee pe baza căreia se pot crea alte idei sau tehnologii pentru a preveni tragediile care se întâmplă.

Bibliografie

1. Alice Zheng, Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists, O’Reilly, 2012
2. Al Sweigart, Automate the boring stuff with Python, 2nd Edition, No Sarch Press, 2015
3. Mark Lutz, Learning Python, O’Reilly, 2008
4. Mircea-Bogdan Rădac, Tehnici de învăţare automată cu aplicaţii, Editura Politehnica, 2019
5. Sebastian Raschka, Vahid Mirjalili, Python Machine Learning, Packt>, 2015
6. <https://adc.bmj.com/content/104/Suppl_3/A30.2> - data accesării: 13.04.2021
7. <https://www.carlsonattorneys.com/news-and-update/forgotten-baby-syndrome> - data accesării: 13.04.2021
8. <https://www.consumerreports.org/car-safety/auto-industry-agrees-to-put-rear-seat-reminder-systems-in-most-new-cars-by-2025/> - data accesării: 13.04.2021
9. <https://www.sciencedaily.com/releases/2019/11/191111084909.htm> - data accesării 14.04.2021
10. <https://www.consumerreports.org/car-safety/new-radar-technology-approved-by-fcc-could-reduce-hot-car-deaths/> - data accesării: 14.04.2021
11. <https://www.naturalbabyshower.co.uk/blogs/b4baby/cybex-sensorsafe-safe-smart-connected> - data accesării: 16.04.2021
12. <https://www.naturalbabyshower.co.uk/blogs/b4baby/cybex-sensorsafe-safe-smart-connected> - data accesării: 16.04.2021
13. <https://abcnews.go.com/Technology/waze-releases-safety-feature-reminding-drivers-forget-child/story?id=41141255> – data accesării: 16.05.2021
14. <https://ro.wikipedia.org/wiki/Python> - data accesării: 05.05.2021
15. <https://www.geeksforgeeks.org/python-features/> - data accesării: 05.05.2021
16. <https://www.binpress.com/why-choose-python/> - data accesării: 20.05.2021
17. <https://www.appdynamics.com/blog/engineering/the-key-differences-between-python-2-and-python-3/> - data accesării: 20.05.2021
18. <https://realpython.com/python-modules-packages/#python-modules-overview> – data accesării: 21.05.2021
19. <https://realpython.com/python-modules-packages/#python-modules-overview> – data accesării: 21.05.2021
20. <https://www.jetbrains.com/pycharm/> - data accesării: 22.05.2021
21. <https://ro.wikipedia.org/wiki/%C3%8Env%C4%83%C8%9Bare_automat%C4%83> – data accesării: 22.05.2021
22. <https://ro.wikipedia.org/wiki/%C3%8Env%C4%83%C8%9Bare_automat%C4%83#Tipuri_de_algoritmi> – data accesării: 22.05.2021
23. <https://scikit-learn.org/stable/user_guide.html> - data accesării: 01.06.2021
24. <https://scikit-learn.org/stable/user_guide.html> – data accesării: 01.06.2021
25. <https://librosa.org/doc/latest/index.html> - data accesării: 03.06.2021
26. <https://librosa.org/doc/latest/index.html> - data accesării: 07.06.2021
27. <https://docs.python.org/3/library/tkinter.html> - data accesării: 09.06.2021
28. <https://pypi.org/project/pushbullet.py/0.9.1/> - data accesării:09.06.2021
29. <https://www.displayr.com/what-is-feature-engineering/> - data accesării: 10.06.2021

1. <https://adc.bmj.com/content/104/Suppl_3/A30.2> [↑](#footnote-ref-1)
2. <https://www.carlsonattorneys.com/news-and-update/forgotten-baby-syndrome> [↑](#footnote-ref-2)
3. <https://www.carlsonattorneys.com/news-and-update/forgotten-baby-syndrome> [↑](#footnote-ref-3)
4. <https://www.consumerreports.org/car-safety/auto-industry-agrees-to-put-rear-seat-reminder-systems-in-most-new-cars-by-2025/> [↑](#footnote-ref-4)
5. <https://www.consumerreports.org/car-safety/auto-industry-agrees-to-put-rear-seat-reminder-systems-in-most-new-cars-by-2025/> [↑](#footnote-ref-5)
6. <https://www.sciencedaily.com/releases/2019/11/191111084909.htm> [↑](#footnote-ref-6)
7. <https://www.consumerreports.org/car-safety/new-radar-technology-approved-by-fcc-could-reduce-hot-car-deaths/> [↑](#footnote-ref-7)
8. <https://www.consumerreports.org/car-safety/new-radar-technology-approved-by-fcc-could-reduce-hot-car-deaths/> [↑](#footnote-ref-8)
9. <https://www.naturalbabyshower.co.uk/blogs/b4baby/cybex-sensorsafe-safe-smart-connected> [↑](#footnote-ref-9)
10. <https://www.naturalbabyshower.co.uk/blogs/b4baby/cybex-sensorsafe-safe-smart-connected> [↑](#footnote-ref-10)
11. <https://abcnews.go.com/Technology/waze-releases-safety-feature-reminding-drivers-forget-child/story?id=41141255> [↑](#footnote-ref-11)
12. <https://ro.wikipedia.org/wiki/Python> [↑](#footnote-ref-12)
13. <https://www.geeksforgeeks.org/python-features/> [↑](#footnote-ref-13)
14. <https://www.binpress.com/why-choose-python/> [↑](#footnote-ref-14)
15. <https://www.appdynamics.com/blog/engineering/the-key-differences-between-python-2-and-python-3/> [↑](#footnote-ref-15)
16. [https://realpython.com/python-modules-packages/#python-modules-overview](https://realpython.com/python-modules-packages/%23python-modules-overview) [↑](#footnote-ref-16)
17. <https://www.jetbrains.com/pycharm/> [↑](#footnote-ref-17)
18. <https://ro.wikipedia.org/wiki/%C3%8Env%C4%83%C8%9Bare_automat%C4%83> [↑](#footnote-ref-18)
19. Mircea-Bogdan Rădac, *Tehnici de învăţare automată cu aplicaţii,* Editura Politehnica, 2019 [↑](#footnote-ref-19)
20. [https://ro.wikipedia.org/wiki/%C3%8Env%C4%83%C8%9Bare\_automat%C4%83#Tipuri\_de\_algoritmi](https://ro.wikipedia.org/wiki/%C3%8Env%C4%83%C8%9Bare_automat%C4%83%23Tipuri_de_algoritmi) [↑](#footnote-ref-20)
21. [https://ro.wikipedia.org/wiki/%C3%8Env%C4%83%C8%9Bare\_automat%C4%83#Tipuri\_de\_algoritmi](https://ro.wikipedia.org/wiki/%C3%8Env%C4%83%C8%9Bare_automat%C4%83%23Tipuri_de_algoritmi) [↑](#footnote-ref-21)
22. <https://scikit-learn.org/stable/user_guide.html> [↑](#footnote-ref-22)
23. <https://scikit-learn.org/stable/user_guide.html> [↑](#footnote-ref-23)
24. <https://librosa.org/doc/latest/index.html> [↑](#footnote-ref-24)
25. <https://librosa.org/doc/latest/index.html> [↑](#footnote-ref-25)
26. <https://docs.python.org/3/library/tkinter.html> [↑](#footnote-ref-26)
27. <https://pypi.org/project/pushbullet.py/0.9.1/> [↑](#footnote-ref-27)
28. <https://www.displayr.com/what-is-feature-engineering/> [↑](#footnote-ref-28)
29. Alice Zheng, *Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists*, O’Reilly, 2012 [↑](#footnote-ref-29)