Universitatea Politehnica Timișoara
Facultatea de Automatică și Calculatoare
Automatică și Informatică Aplicată

Autentificare bazată pe recunoaștere facială

Proiect de diplomă

Coordonatori:

Șl.dr.ing. Adriana ALBU
As.univ. Cristian ZIMBRU

Candidat:

Paul NOVAC

Timișoara,
2018

Conținut

Capitolul 1. Introducere 6
1.1. Context 6
1.2. Tema proiectului 7
Capitolul 2. Fundamentare teoretică 8
2.1. Reţele Neuronale 8
Capitolul 3. Sisteme și tehnologii folosite 19
3.1 Python 19
3.1.1 Dezvoltare 20
3.1.2. *Python Dlib* 21
3.1.3. *Speech to Text* 24
3.2. *SQLite Data Base* 24
Capitolul 4. Prezentarea Aplicaţiei 26
4.1. Scopul aplicației 26
4.2. Prezentarea modulelor 27
4.2.1. Captură de imagini (*ImageCapture*) 27
4.2.2. *DataBase* 27
4.2.3. *FaceRecognition* & *Neural Network Training* 27
4.2.5. *Speech2Text* 28
4.2.6. *Text2Speech* 28
4.3. Implementarea modulelor 28
4.3.1. Implementarea generală: 29
4.3.2. Implementare în detaliu: 30
4.4. Aplicaţii existente 38
4.5. Probleme întâmpinate 40
Capitolul 5. Utilizarea sistemului 41
5.1. Ghid de instalare 41
5.2. Limitările sistemului 41
Capitolul 6. Direcţii de continuare a dezvoltării 42
Capitolul 7. Concluzii 45
Capitolul 8. Bibliografie 46

Figura 1.1. Model poză pentru rețeaua neuronală. 6
Figura 2.1. Straturile de baza ale unei reţele neuronale 8
Figura 2.2. Mostra de imagine care conţine o faţă 11
Figura 2.3. Detectarea feţei din imagine 12
Figura 2.4. Analiza caracteristicilor faciale din imagine 12
Figura 2.5. Compararea feţei detectatea, cu cea din baza de date 12
Figura 2.6. Predicţia pentru imaginea curentă 13
Figura 2.7. Transformarea imaginii color în alb-negru 14
Figura 2.8. Analiza caracteristicilor faciale 14
Figura 2.9. Fluxul de la lumină la întuneric, pentru intreaga imagine 15
Figura 2.10. Identificarea celor 68 de repere faciale 16
Figura 2.11. Detectarea feţei, identificarea reperelor faciale, extragerea reperelor şi forfecarea imaginii 17
Figura 3.1. Schemă de funcționare a pachetului *dlib* 21
Figura 3.2. Reprezentarea caracteristicilor faciale 23
Figura 3.3. Indicii celor 68 de repere faciale 24
Figura 3.4. Structura bazei de date. 25
Figura 4.1. Schiţarea posibilităţii de utilizare a aplicaţiei. 26
Figura 4.2. Arhitectura sistemului 29
Figura 4.3. Interfaţa grafică a aplicaţiei 30
Figura 4.4. Design-ul implementării butonului de pornire a aplicaţiei. 31
Figura 4.5. Predicţia reţelei neuronale 32
Figura 4.6. Interfaţa grafică, pentru accesul permis. 32
Figura 4.7. Etichetare specifică persoanelor care nu fac parte din baza de date. 33
Figura 4.8. Interfaţa grafică de adăugare nume utilizator. 34
Figura 4.9. Design-ul implementării butonului de inserare a unui nou utilizator. 35
Figura 4.10. Structura de foldere a bazei de date. 35
Figura 4.11. Design-ul pentru ștergerea unui utilizator. 36
Figura 4.12. Interfaţa grafică pentru introducerea parolei de administrator 37
Figura 4.13. Design-ul implementării pentru reantrenarea reţelei neuronale 37
Figura 4.14. Detecţia utilizând aplicaţia *Kairos* 38
Figura 4.15. Logo-ul aplicaţiei AiCure. 39
Figura 4.16. Interfața aplicației Walmart 39

Capitolul 1. Introducere

1.1. Context

Recunoaşterea facială reprezintă în prezent o zonă foarte cercetată în ştiinţa calculatoarelor. Securitatea prin utilizarea soluţiilor de recunoaştere facială este valul viitorului. Utilizatorii pot accesa conturi, pot primi acces în diferite instituţii şi toate acestea cu un singur *selfie*. În prezent recunoașterea facială este o zonă foarte cercetată și de interes. Diferențele de abordare ale acestui domeniu sunt reprezentate de tehnicile utilizate sau rata de precizie [1].

Autentificarea prin recunoaştere facială funcţionează în felul următor, un simplu *selfie,* realizat cu o cameră de fotografiat (fie că este vorba de o cameră de telefon sau o cameră web), creează un şablon biometric, foarte precis şi securizat, al feţei unui utilizator folosind algoritmi sofisticaţi de învăţare automată [2].

Viteza se obţine evitând îndelungatul proces de creare unui cont, pentru acest lucru utilizatorul va opta pentru o serie de fotografii făcute pe loc, cu o cameră web pentru a face parte din baza de date, așa cum se poate observa în *figura 1.1* [3].

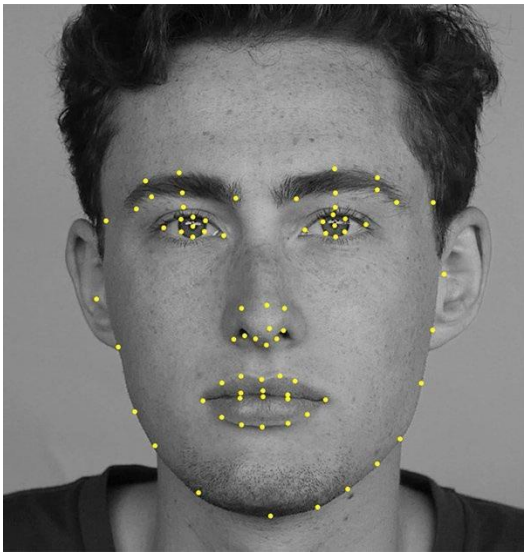

Figura 1.1. Model poză pentru rețeaua neuronală.

Baza de date este folosită de o reţea neuronală bine antrenată, pentru ca recunoaşterea facială să aibă loc în cel mai rapid şi eficient mod cu putinţă. Confirmarea persoanei în cauză cu una existentă în baza de date, îi va permite accesul persoanei respective într-o companie, confirmarea se face printr-un mesaj audio.

1.2. Tema proiectului

Limbajul de programare pentru aplicaţia curentă este Python, fiind o aplicaţie *desktop* pentru sistemul de operare Windows, aceasta foloseşte feţele utilizatorilor pentru popularea bazei de date SQLite.

Popularea bazei de date are loc la fiecare nouă introducere în sistem a unui utilizator, cu ajutorul unei camere web pozele sunt făcute pe loc, din diferite unghiuri, iar mai apoi vor fi folosite de reţeaua neuronală, în prima fază pentru antrenare, iar mai apoi pentru a acorda accesul în clădire.

Acordarea accesului se face prin confirmarea persoanei detectate cu ajutorul unei camere web, cu una din pozele deja existente în baza de date, în cazul în care persoana în cauză nu există în baza de date, o persoană autorizată va trebui să o adauge în sistem. Dacă persoana în cauză face parte din baza de date, i se va permite accesul în locaţia în care aplicaţia activează. Întreg sistemul funcţionează şi pe bază de comenzi vocale, sistem care la rândul lui va putea informa utilizatorul cu un mesaj audio corespunzător situaţiei curente (exemplu: *The access is grant.*). Datorită acestei interacţiuni biunivoce, sistemul este foarte uşor de gestionat şi de întreţinut.

Întreaga aplicaţie va rula pe un singur sistem de calcul, pe care se va regăsi baza de date cu persoanele care au drept de acces şi fişierul responsabil cu datele antrenate pentru reţeaua neuronală.

Capitolul 2. Fundamentare teoretică

2.1. Reţele Neuronale

Reţelele neuronale reprezintă acea ramură a ştiinţei inteligenţei artificiale şi constituie totodată, un instrument de cercetare pentru neuroinformatică. Reţelele neuronale artificiale se caracterizează ca fiind elemente simple de procesare, fiind puternic interconectate şi paralel operaţionale, având ca rol interacţiunea cu mediul înconjurător într-o manieră asemănătoare creierelor biologice, prezentând o capacitate avansată de învăţare [4].

Reţelele neuronale sunt compuse din neuroni artificiali, fiind parte a inteligenţei artificiale, având conceptual originea ca şi neuroni artificiali, în biologie. Reţelele neuronale pot fi definite ca fiind reţele de elemente simple, dar puternic interconectate prin intermediul unor legături, care poartă denumirea de interconexiuni, prin care se propagă informaţie numerică. Rețelele neuronale sunt compuse din cel puțin trei straturi, la fel cum se poate observa și în *figura 2.1*.

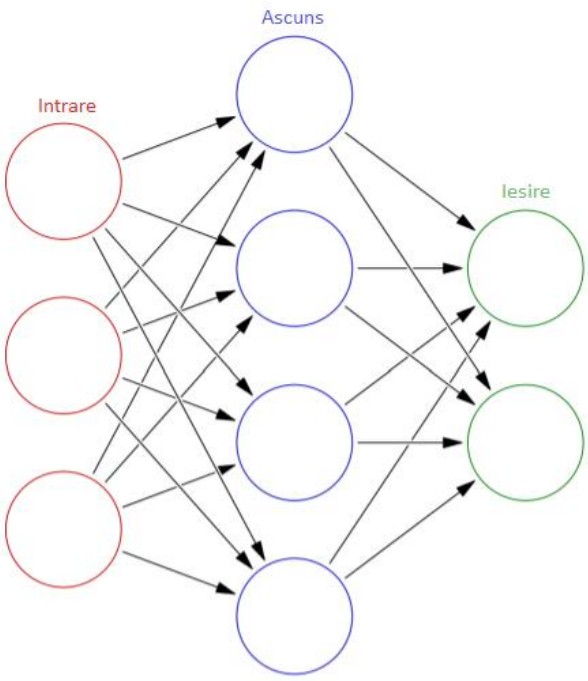

Figura 2.1. Straturile de baza ale unei reţele neuronale.
Sursă: Comparison of Face Recognition Neural Networks

Principala trăsătură a acestor reţele este capacitatea de învăţare pe baza unor exemple, utilizând experienţa precedentă în scopul îmbunătăţirii rezultatelor.

Deşi asemănarea cu creierul uman, atunci când vine vorba de funcţionare este destul de mare, reţelele neuronale se diferenţiază ca şi structură faţă de creierul uman. Reţeaua neuronală este mult simplificată în comparaţie cu corespondentul său uman, însă la fel ca şi creierul uman, se compune din unităţi puternice cu capacitate de calcul, cu toate acestea sunt mult inferioare neuronului.

Reţelele neuronale artificiale sunt caracterizate pe baza a 3 elemente:

- Modelul utilizat pentru procesarea individuală
- Arhitectura
- Învățare automată

Există mai multe moduri de a clasifica modelele neuronului elementar, acestea implicând domeniul definiţiilor pentru semnalele folosite, natura datelor utilizate, tipul funcţiei apelate şi prezenţa memoriei.

La nivel macro se pot identifica doua clase de arhitecturi:

- Cele care propagă informația doar dinspre intrare spre ieșire
- Rețele cu reacție

Reţelele neuronale au deasemenea câteva dezavantaje, aceste dezavantaje sunt reprezentate de lipsa teoriei care este responsabilă pentru precizarea tipului reţelei şi a numărului de neuroni elementari, precum şi modalităţile de interconectare.

Există mai multe deosebiri între reţelele neuronale şi sistemele electronice de prelucrare a informaţiilor, însă principala diferenţă este reprezentată de capacitatea de învăţare, care se desfăşoară în urma interacţiunii cu mediul înconjurător şi îmbunătăţirea continuă a performanţelor [5].

Reprezentarea corectă a informaţiilor, cu scopul de a permite interpretarea, predicţia şi răspunsul sistemului la stimuli externi, pot permite reţelei neuronale să dezvolte un model propriu al procesului în cauză, astfel acest model, în cele din urma va putea răspunde unor stimuli care nu au fost utilizaţi în procesul anterior de învăţare a reţelei. Informaţiile folosite

în cadrul procesului de învăţare, pot fi: informaţii care se cunosc dinainte sau perechi de intrare-ieşire (cu rol în stabilirea relaţiei de tip cauză-efect), modul de reprezentare internă, urmând un set de reguli foarte bine documentate.

Conceptele rețelelor neuronale ne permit să extragem o gamă largă de caracteristici din imagini. Extragerea caracteristicilor pentru recunoașterea facială se foloseşte de această idee.

Învățarea profundă, este zona incipientă a informaticii care revoluționează inteligența artificială, permițându-ne să construim mașini și sisteme ale viitorului. Deși învățarea profundă ușurează viața, înțelegerea modului în care funcționează poate fi dificilă.

Învăţarea automată este o zonă a Inteligenței Artificiale (*AI*) care le permite calculatoarelor să *învețe*. În mod tradițional, întotdeauna am folosit computerele pentru a face lucruri, oferind un set strict de instrucțiuni (de exemplu, aplicații *desktop*). Învățarea automată utilizează o abordare foarte diferită, în loc să oferim computerului un set de instrucțiuni despre cum să facem ceva, îi oferim instrucțiuni despre cum să înveți să faci acel lucru.

Oferindu-i date și programând pentru a folosi diferite modele matematice și statistice care să convingă datele și să învețe să ia decizii bazate pe acestea. Aceasta abordare nu face altceva decât să învețe algoritmul să urmărească diferenţele de aspect dacă vorbim de imagini. Acest lucru se poate face folosind o serie de algoritmi diferiți [6].

În acest fel, computerul *învață* singur, în loc să fie programat în mod specific să facă o anumită sarcină. Procesul de predare calculatorului (adică oferirea datelor de învățare) poartă denumirea de antrenament.

Învățarea profundă, așa cum sugerează și numele, reprezintă un subdomeniu al învățării mașinilor. Învățarea în profunzime implică, în cea mai mare parte, utilizarea unor rețele neuronale profunde (algoritmi sau modele computaționale) pentru a aborda problemele de învățare automată.

Uneori folosirea algoritmilor de luare a deciziilor și/sau a altor algoritmi de învățare automată poate fi, de asemenea, numită învățare profundă dar, în cea mai mare parte, învățarea profundă implică utilizarea rețelelor neuronale [7].

Recunoașterea facială presupune bineînțeles rezolvarea câtorva probleme:

În prim pas, avem nevoie de cel puţin o imagine în care să găsim, cel puţin o faţă, în cazul în care imaginea conţine mai multe feţe, acestea trebuiesc detectate. În pasul doi, algoritmul trebuie să se concentreze pe fiecare față și să poată înțelege că şi atunci când o față este într-o direcție ciudată sau într-o iluminare proastă, ea este încă aceeași persoană.

În al treilea rând, trebuie să existe o disponibilitate de a alege trăsăturile faciale unice, care pot fi folosi pentru a face distincţia între oameni, cum ar fi: cât de mari sunt ochii, cât de alungită este faţa, etc. În cele din urmă, se compară caracteristicile unice ale acelei fețe cu toți oamenii pe care deja îi cunoaște, pentru a determina numele persoanei. Ca om, creierul este legat să facă toate acestea automat și instantaneu. De fapt, oamenii sunt prea buni la recunoașterea fețelor și ajung să vadă fețe în obiectele obișnuite. Computerele nu sunt capabile de acest gen de generalizare la nivel înalt (cel puțin nu încă), așa că trebuie să le învățăm cum să facă fiecare pas în acest proces separat [8].

Trebuie să construim o conductă în care să rezolvăm separat fiecare pas de recunoaștere a feței și să trecem rezultatul etapei curente la pasul următor. Cu alte cuvinte, vom alătura împreună mai mulți algoritmi de învățare automată:

Paşii pentru recunoaşterea facială sunt descrişi în cele ce urmează:

1. Avem nevoie de o imagine care să conţină cel puţin o faţă, la fel ca și in *figura 2.2.*

Figura 2.2. Mostra de imagine care conţine o faţă [8].

2. Cat timp avem cel puţin o faţă în imagine, această faţă va trebui detectată, detecția feței este încadrată cu galben în *figura 2.3.*

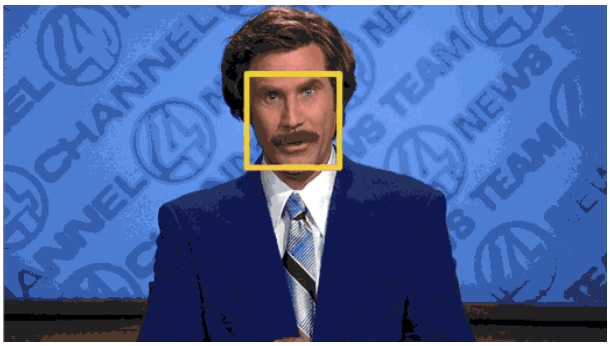

Figura 2.3. Detectarea feţei din imagine [8].

3. După ce detecţia a avut loc, se vor analiza caracteristicile faciale, ale subiectului în cauză, caracteristicile pot fi observate în *figura 2.4.*

Figura 2.4. Analiza caracteristicilor faciale din imagine [8].

4. Se va compara faţa găsită în imagine cu toate feţele care se găsesc în baza de date ca în *figura 2.5.*

Figura 2.5. Compararea feţei detectatea, cu cea din baza de date [8].

5.După care se va face o predicţie, predicție care va afișa numele persoanei detectate la fel ca în *figura 2.6*.

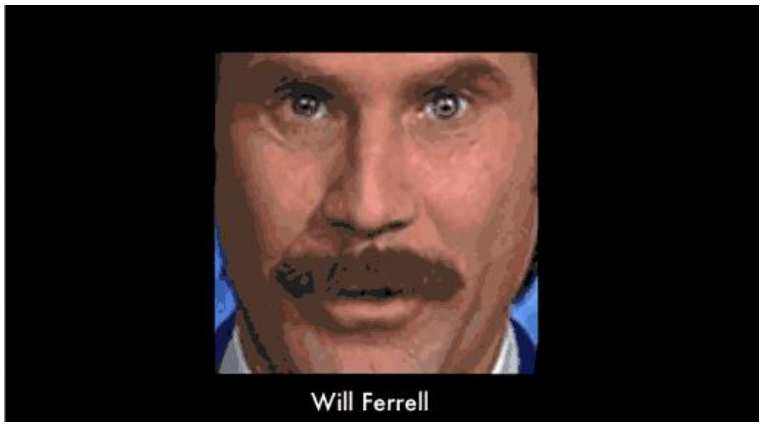

Figura 2.6. Predicția pentru imaginea curentă [8].

În cele ce urmează, abordarea problemei se va face pas cu pas, algoritmul de învăţare automată se va prezenta în linii mari.

Pasul 1 – Găsirea tuturor feţelor

Primul pas constă in detectarea feței. Evident, trebuie să găsim fețele într-o fotografie înainte de a începe analiza caracteristicilor faciale.

Detectarea feței a fost integrată la începutul anilor 2000, când Paul Viola și Michael Jones au inventat o modalitate de a detecta fețe, metodă care era destul de rapidă pentru a rula pe camere ieftine. Cu toate acestea, există soluții mult mai fiabile acum. Vom folosi o metodă inventată în 2005 numită Histogramă de gradienți orientați. [8]

Pentru a găsi fețe într-o imagine, vom începe prin a transforma imaginea noastră într-o imagine alb-negru precum în *figura 2.7*, deoarece nu avem nevoie de date despre culoare pentru a găsi fețe:

Figura 2.7. Transformarea imaginii color în alb-negru [8].

După care ne vom uita la fiecare pixel din imaginea noastră unul câte unul. Pentru fiecare pixel, dorim să privim pixelii care îl înconjoară, la fel ca în *figura 2.8.*

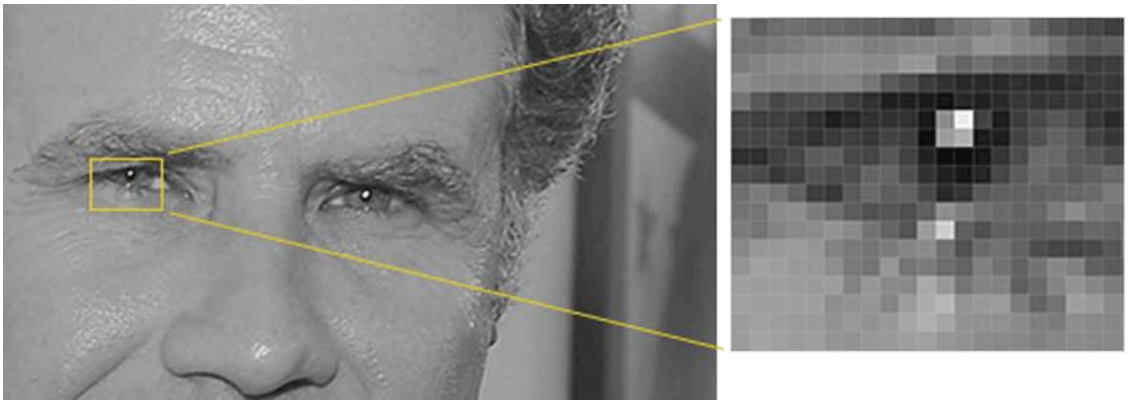

Figura 2.8. Analiza caracteristicilor faciale [8].

Scopul este acela de a ne da seama cât de întunecat este pixelul curent în comparaţie cu pixelii care îl înconjoară. Apoi vrem să desenăm o săgeată care să arate în ce direcție imaginea devine mai întunecată. Dacă acest proces este repetat pentru fiecare pixel din imagine, în cele din urma pixeli vor fi înlocuiți de o săgeată. Aceste săgeți sunt numite gradienți și arată fluxul de la lumină la întuneric pe întreaga imagine [8], precum in *figura 2.9.*

Figura 2.9. Fluxul de la lumină la întuneric, pentru intreaga imagine [8].

Pentru a găsi chipuri în această imagine *HOG*, tot ce trebuie făcut este să găsim acea parte a imaginii care arată cel mai asemănător cu un model *HOG* cunoscut care a fost extras dintr-o grămadă de alte fețe de antrenament. Folosind această tehnică, găsirea feţelor în fiecare imagine este mult mai simplă.

Pasul 2 - Poziţionarea şi proiectarea feţelor

După izolarea tuturor feţelor dintr-o imagine, există posibilitatea ca acele feţe să fie orientate în diferite direcţii și vor arăta complet diferit pentru un computer. Pentru a ține cont de acest lucru, se va încerca denaturarea fiecărei imagini, astfel încât ochii și buzele să fie întotdeauna în locul eșantionului din imagine. Acest lucru va face mult mai uşoară compararea fețelor în etapele următoare.

Pentru a face acest lucru, se va folosi un algoritm numit estimarea caracteristicilor faciale (*face landmark estimation*). Există multe modalități de a face acest lucru, dar vom folosi abordarea inventată în 2014 de Vahid Kazemi și Josephine Sullivan.

Ideea de bază este aceea de a folosi 68 de puncte specifice (numite repere) care există pe fiecare față - partea superioară a bărbiei, marginea exterioară a fiecărui ochi, marginea interioară a fiecărei sprâncene etc . Apoi se va pregăti algoritmul de învățare automată pentru a putea găsi aceste 68 de puncte specifice pe orice față [8], punctele fiind reprezentate in *figura 2.10*.

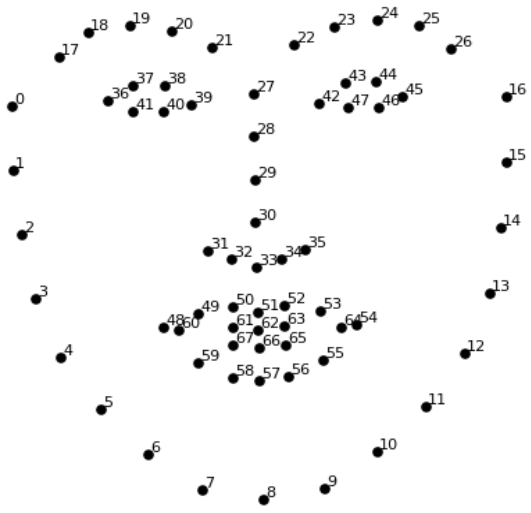

Figura 2.10. Identificarea celor 68 de repere faciale [8].

Acum știind unde se află ochii și gură, se va roti pur și simplu, se va scala și se va forfeca imaginea, astfel încât ochii și gura să fie centrate cât mai bine posibil, acest proces se poate observa in *figura 2.11*. Se va folosi doar transformări de bază ale imaginii, cum ar fi rotația și scalarea, care păstrează liniile paralele (numite transformări afine):

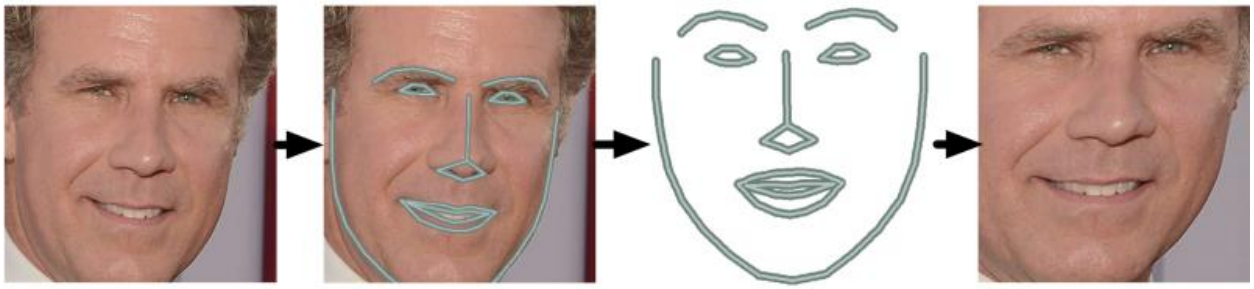

Figura 2.11. Detectarea feţei, identificarea reperelor faciale, extragerea reperelor şi forfecarea imaginii [8].

Indiferent de modul în care se roteşte faţa, algoritmul este capabili să centreze ochii și gura în aproximativ aceeași poziție ca şi în imagine. Acest lucru va face următorul pas mult mai precis.

Pasul 3 – Codificarea feţei

Cea mai simplă abordare a recunoașterii faciale este aceea de a compara direct fața necunoscută găsită în pasul 2 cu toate imaginile pe care le avem ale persoanelor care au fost deja etichetate. Când găsim o față marcată anterior, care arată foarte asemănătoare cu fața necunoscută, trebuie să fie aceeași persoană, însă acest proces este foarte greoi şi încet.

Avem nevoie de o modalitate de a extrage câteva măsurători de bază de la fiecare față. Apoi am putea măsura fața necunoscută în același fel și am găsi fața cunoscută cu cele mai apropiate măsurători. De exemplu, am putea măsura dimensiunea fiecărei urechi, distanța dintre ochi, lungimea nasului etc.

Măsurătorile precum culoarea ochilor, nu au sens cu adevărat pentru un calculator care privește la pixelii individuali dintr-o imagine. Cercetătorii au descoperit că cea mai corectă abordare este aceea de a permite computerului să-și dea seama ce măsurători să colectează. Învățarea profundă face o treabă mai bună decât cea a oamenilor în a determina care părți ale feței sunt importante pentru măsurare [8].

Pasul 4 – Găsirea numelui persoanei codificate

Ultimul pas este, de fapt, cel mai simplu pas în întregul proces. Tot ce a rămas de făcut este găsirea persoanei din baza de date cu cele mai apropiate măsurători imaginii testului nostru. Acest lucru se obţine folosind orice algoritm de bază de clasificare a învățării automate [8].

Capitolul 3. Sisteme și tehnologii folosite

3.1 Python

Python este un limbaj de programare, interpretor, orientat pe obiecte, cu semantică dinamică. Structurile de date construite la nivel înalt, combinate cu tastarea dinamică și legarea dinamică, îl fac foarte atractiv pentru dezvoltarea rapidă a aplicațiilor, precum și pentru utilizarea ca limbaj de *scripting* sau lipire pentru a conecta împreună componentele existente. Sintaxa simplă și ușor de învățat a *Python-ului* accentuează lizibilitatea și prin urmare, reduce costul întreținerii programelor. *Python* acceptă module și pachete, ceea ce încurajează modularitatea programelor și reutilizarea codului. Interpretul *Python* și biblioteca extensivă standard sunt disponibile atât în formă sursă cât şi în formă binară [9].

Python este un limbaj de programare dinamic, a fost dezvoltat în 1989 de către un programator de origine olandeză Guido van Rossum. O multitudine de companii folosesc *Python* ca şi limbaj de programare, câteva dintre acestea ar fii: *Google* sau *Yahoo*! Folosit pentru programarea aplicaţiilor web. *Python* poate fii folosit şi pentru crearea unor aplicaţii ştiinţifice sau de divertisment, aplicaţii care pot fii programate fie parţial fie în întregime în Python [10].

Datorită popularităţii şi puterii foarte ridicate, limbajul de programare este foarte apreciat în rândul programatorilor specializaţi, fiind un instrument foarte apreciat în cadrul mediilor universitare. Accentul este pus pe curăţenia şi simplitatea codului, sintaxa le permite dezvoltatorilor să îşi exprime ideile într-o manieră mult mai clară şi mai concisă spre deosebire de alte limbaje de programare.

Aplicaţia care face subiectul acestei lucrări a fost dezvoltată în *Python* 3.6, versiune lansată în anul 2016, acest lucru însemnând că toate îmbunătăţirile recente ale bibliotecilor, sunt disponibile în mod implicit numai în *Python* 3.X.

3.1.1 Dezvoltare

Limbajul de programare *Python* a fost conceput la sfârșitul anilor 1980, însă implementarea sa a fost inițiată în decembrie 1989 de Guido van Rossum în Țările de Jos, fiind capabil de tratarea excepțiilor. Van Rossum este autorul principal al limbajului de programare *Python,* iar principalul sau rol este de a continua să decidă direcția limbajului de programare. *Python* 2.0 a fost lansat pe 16 octombrie 2000, cu multe caracteristici noi, inclusiv un colector de gunoi care detectează cicluri (în plus față de numărarea referințelor) pentru gestionarea memoriei. Cu toate acestea, cea mai importantă schimbare a fost procesul de dezvoltare în sine, cu trecerea la un proces mai transparent și mai bine susținut de comunitate. *Python* 3.0 este o versiune majoră, incompatibilă cu versiunile anterioare, a fost lansată pe 3 decembrie 2008 după o perioadă lungă de testare. Multe dintre caracteristicile sale majore au fost, de asemenea, returnate la versiunile anterior compatibile cu *Python* 2.6 și 2.7 [11].

Pentru dezvoltarea aplicaţiei *Python* au fost necesare:

- *PyCharm IDE*
- P*utty-0.67*
- *CMake*
- *Active Python DataBase*
- *DB Browser for SQLite*

Câteva din pacheţele *Python* instalate:

- *dlib v19.13.0* – pachetul pentru rețele neuronale și învățare automată
- *face-recognition* – pachetul pentru recunoaștere facială
- *Py-Audio* – pachetul pentru semnale audio
- *pypiwin32* – pachetul pentru accesarea *API-urilor*
- *pyspeech* – pachetul pentru comenzi vocale
- *pyttsx* – pachetul pentru convertire unui șir caractere în semnal audio
- *SpeechRecognition* – pachetul pentru recunoaștere vocală

Pentru a utiliza fiecare funcţionalitate a aplicaţiei este necesară o conexiune la internet, deoarece apelarea *API-ului Google* cere acest lucru. În cazul în care nu există o conexiune la

internet, utilizatorul va fi informat. Modulul responsabil de transformarea comenzi vocale date de utilizator în mesaj text, folosește *API-ul* de la *Google*, acest lucru făcând conexiunea la internet o necesitate.

3.1.2. *Python Dlib*

3.1.2.1 Introducere

Dlib este o bibliotecă software generală cu multiple platforme, (emblema se află în *figura13.)* scrisă în limbajul de programare C++ și folosită pentru rezolvarea problemelor reale. În primul rând, reprezintă un set de componente software independente. Este un software *open-source* lansat sub licența *Boost Software License*.

Davis King a fost autorul principal al bibliotecii *dlib* încă de la începutul dezvoltării sale în 2002, aceasta a crescut pentru a include o mare varietate de instrumente. Începând cu anul 2016, aceasta bibliotecă, conține componente software pentru rezolvarea problemelor legate de rețea, fire, interfețe grafice, structuri de date, algebră liniară, învățare automată, prelucrare de imagini, extragere de date, parsarea textului, optimizare numerică și multe alte sarcini [12]. În *figura 3.1.* este reprezentată schema de funcționare a pachetului *dlib*.

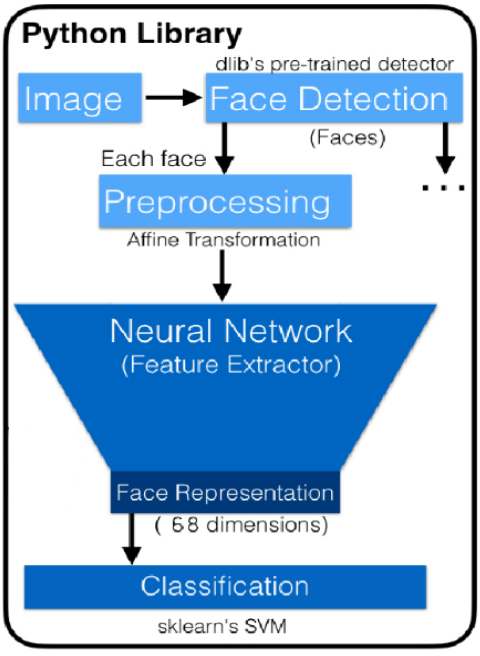

Figura 3.1. Schemă de funcționare a pachetului *dlib*.
Sursă: https://blog.algorithmia.com/understanding-facial-recognition-openface/

În ultimii ani, o mare parte a dezvoltării a fost axată pe crearea unui set larg de instrumente statistice de învățare a mașinilor. Cu toate acestea, *dlib* rămâne o bibliotecă cu scop general și salută contribuțiile unor componente software de înaltă calitate, utile în orice domeniu.

Este folosit atât în industrie, cât și în mediul academic într-o gamă largă de domenii, inclusiv robotică, dispozitive încorporate, telefoane mobile și medii mari de calcul de înaltă performanță. Licența *dlib* va permite utilizarea sa în orice aplicație, gratuit.

Spre deosebire de o multitudine de proiecte *open-source*, aceasta oferă o documentație completă și precisă pentru fiecare clasă și funcție. Există, de asemenea, moduri de depanare care verifică precondițiile documentate pentru funcții. Când acest lucru este activat, va capta marea majoritate a erorilor cauzate de funcțiile de apelare incorectă sau de utilizare a obiectelor într-un mod incorect.

Corelarea cu filozofia de dezvoltare a bibliotecii *dlib* este o dedicație pentru portabilitate și ușurință în utilizare. Prin urmare, tot codul bibliotecii este conceput astfel încât să fie cât mai portabil posibil și, în mod similar, să nu necesite ca utilizatorul să configureze sau să instaleze ceva. Pentru a realiza acest lucru, codul specific platformei este împachetat in *API-uri*. În prezent, biblioteca este accesibilă următoarelor platforme: OS X, MS Windows, Linux și Solaris [13].

3.1.2.2 Mod de funcționare

Detectarea reperelor faciale este un subgrup al problemei de predicție a formei. Având o imagine de intrare (și în mod normal o regiune de interes care specifică subiectul), un predictor de formă va încerca să localizeze punctele cheie de interes de-a lungul formei. Reperele faciale sunt folosite cu succes pentru alinierea feței, pentru postura capului, pentru schimbarea feței şi pentru detectarea intermitentă. În contextul reperelor faciale, scopul este acela de a detecta structuri faciale, folosind metode de predicție a formei. Repere reale sunt folosite pentru a localiza și reprezenta regiunile vizibile ale feței, cum ar fi:

- ochii
- sprâncenele
- nasul
- gura
- conturul feţei

Detectarea reperelor faciale este deci un proces în două etape:

Pasul 1: Localizarea feței în imagine.

Pasul 2: Detectarea structurilor faciale cheie.

În urma *pasului 1*, se obține o casetă de limitare a feței (adică, coordonatele (x, y) ale feței din imagine):

Având în vedere regiunea feței, putem aplica apoi *Pasul 2* şi anume, detectarea structurilor faciale cheie în regiunea feței. Există o varietate de detectori faciali, însă toate metodele încearcă, în esență, să localizeze și să eticheteze următoarele regiuni faciale: gura, sprânceana dreaptă, sprânceana stângă, ochiul drept, ochiul stâng, nasul şi falca.

Detectorul de repere faciale inclus în biblioteca dlib, are rolul de a eticheta imaginile în care se găsesc structuri faciale. Aceste imagini sunt marcate manual, specificând coordonatele (x, y) ale regiunilor care înconjoară fiecare structură facială [14].

Având în vedere aceste date de antrenament, un ansamblu de arbori de regresie este instruit să estimeze pozițiile reperului facial direct din intensitățile pixelilor, precum in imaginea din *figura 3.2*.

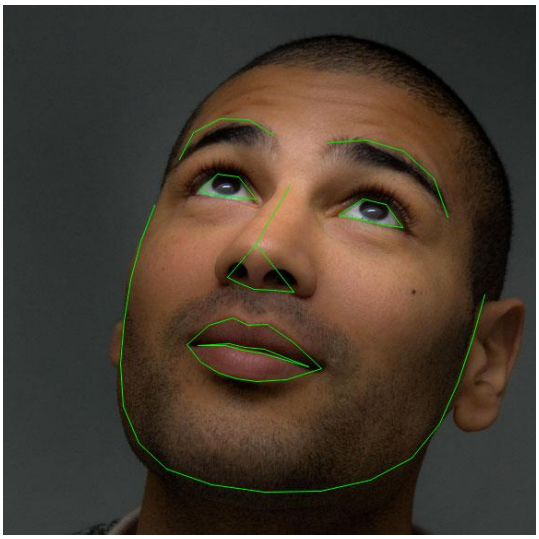

Figura 3.2. Reprezentarea caracteristicilor faciale [14].

Rezultatul final este un detector de repere faciale care poate fi folosit pentru a detecta reperele faciale în timp real, cu predicții de înaltă calitate. Detectorul de repere faciale pre-instruit, din

interiorul bibliotecii *dlib* este utilizat pentru a estima locația a 68 de coordonate (x, y), care sunt mapate pe structura feţei. Indicii celor 68 de coordonate pot fi vizualizaţi în *figura 3.3*.

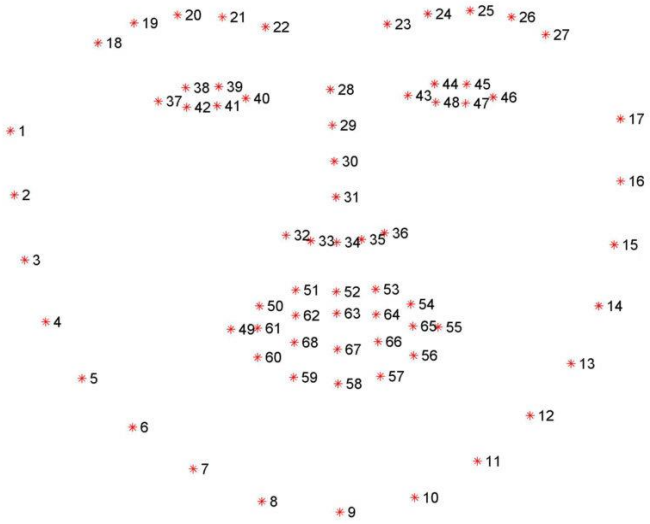

Figura 3.3. Indicii celor 68 de repere faciale [14].

Indiferent de setul de date care se utilizează, același cadru se poate folosi pentru a forma un predictor de formă pe datele de antrenament de intrare, acest lucru este util dacă se dorește pregătirea detectorilor de reper facial sau a predictorilor personalizați pentru forma proprie [14].

3.1.3. *Speech to Text*

Recunoașterea vorbirii este un sub-câmp interdisciplinar al lingvisticii computaționale care dezvoltă metodologii și tehnologii care permit recunoașterea și traducerea limbii vorbite în text de către calculatoare [15].

Google are un *API* de recunoaștere al vorbirii, acest *API* convertește intrarea audio în șir de caractere, intrare care poate să provină de la un microfon [16].

3.2. *SQLite Data Base*

Folosirea unei baze de date este importanţă, deoarece gestionează eficient datele şi permite administratorilor să efectueze cu uşurinţă mai multe sarcini, câteva din avantajele folosirii unei baze de date sunt:

- Reduce redundanţa datelor
- Îmbunătăţirea accesului la date pentru administratori

- Securitatea îmbunătăţită a datelor
- Dezvoltarea facilă a programului

SQLite este un sistem de gestionare a bazelor de date relaţionale, scris în limbajul de programare C. Spre deosebire de multe alte sisteme de gestionare a bazelor de date, *SQLite* nu este un motor de căutare într-o baza de date de tip client-server, ci mai degrabă face parte din programul final.

Foloseşte o sintaxa dinamică de tip *SQL*, care nu garantează integritatea domeniului, fiind o alegere populară pentru bazele de date încorporate şi pentru stocarea locală în aplicaţii *software* cum ar fi *browserele web*.

În ziua de astăzi este probabil cel mai răspândit motor de baze de date, folosit de mai multe *browsere*, sisteme de operare şi sisteme încorporate. *SQLite* are legături cu multe limbaje de programare.

Baza de date pe care aplicaţia o foloseşte este alcătuită din următoarele câmpuri și se regăsește în *figura 3.4*:

- *ID* – Indexare automată, contabilizează numărul de persoane din baza de date
- *PICTURE* – Tipul de data este *BLOB* (*Binary Large Object*), stochează fotografiile persoanelor din baza de date
- *TYPE* – Tipul de data este *text*, conţine informaţii cu privire la extensia cu care fotografia a fost salvată
- *FILE_NAME* – Tipul de data este *text*, conţine numele imaginii salvate în baza de date şi totodată numele persoanei din fotografia respectivă.

ID	PICTURE	TYPE	FILE_NAME
Filter	Filter	Filter	Filter
1	*BLOB*	.jpg	Paul Novac
2	*BLOB*	.jpg	Cosmin
3	*BLOB*	.jpg	Timotei Redis
4	*BLOB*	.jpg	Luca
5	*BLOB*	.jpg	Radu

Figura 3.4. Structura bazei de date.

Capitolul 4. Prezentarea Aplicaţiei

4.1. Scopul aplicaţiei

Aplicaţia dezvoltată reprezintă un sistem de autentificare şi validare bazat pe recunoaştere facială, procesare de imagini, fiind o aplicaţie capabilă să interacţioneze cu utilizatorul prin intermediul sistemului audio.

Aceasta aplicaţie are rolul de a asigura accesul utilizatorului (angajatului) în clădirea în care acesta lucrează, prin intermediul unei camere web instalate deasupra turnichetului de la intrarea în clădire. Schița de prezentare se află în *figura 4.1*.

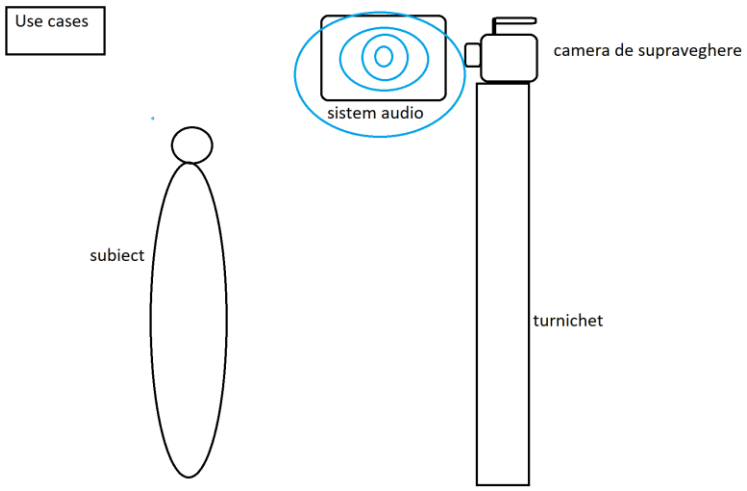

Figura 4.1. Schiţarea posibilităţii de utilizare a aplicaţiei.

Sistemul de faţă scoate din ecuaţie nevoia unui card de acces, pe care angajatul ar trebui să îl aibe zilnic în posesie, pentru a putea intra în clădire, astfel în cazul în care acest card este uitat acasă sau este pierdut nu va influenţa cu nimic prezenţa la servici, deoarece sistemul se bazează pe detecţia facială a angajatului.

Accesul în clădire este restricţionat doar la persoanele care sunt autorizate, persoanele noi angajate vor fi adăugate în baza de date, opţiunea de adăugare în baza de date va fi atribuită unui administrator, deoarece acest lucru presupune reantrenarea reţelei neuronale.

4.2. Prezentarea modulelor

4.2.1. Captură de imagini (*ImageCapture*)

Modulul este responsabil de captura imaginilor, acesta va porni camera la care aplicaţia este legată şi va cere utilizatorului în primul rând să îşi introducă numele complet, după care trei poze i se vor face din unghiuri diferite, pentru ca ulterior detecţia pe care reţeaua neuronală o va face să fie mai precisă.

4.2.2. *DataBase*

Acest modul implementează baza de date de care aplicaţia se foloseşte, fiind vorba de o baza de date locală, în aceasta bază de date se vor introduce poze cu angajaţii cu drept de acces în clădire. Introducerea în baza de date se va face fie în urma utilizării *API-ului* de la *Facebook* fie prin apelarea modulului *ImageCapture*.

4.2.3. *FaceRecognition* & *Neural Network Training*

După cum îi spune şi numele, modulul va fi responsabil cu recunoaşterea facială, pentru ca procesul să fie rapid şi eficient s-a apelat la o reţea neuronală antrenată în acest scop. Recunoaşterea facială se face prin compararea fiecărei persoane detectate în cadrul fluxului video cu cele deja existente în baza de date, în cazul în care persoana nu există în baza de date se va înştiinţa administratorul despre acest lucru, tot administratorul are obligaţia să reantreneze reţeaua neuronală după ce o persoana nouă a fost adăugată în baza de date, acest lucru îi va permite accesul în clădire [17].

4.2.5. *Speech2Text*

Prin implementarea acestui modul utilizatorul va putea interacţiona cu aplicaţia prin intermediul sistemului audio. Utilizând un microfon, utilizatorul va putea răspunde sistemului. Răspunsul dat sistemului va fii convertit într-un şir de caractere, folosind *Google API* [18].

4.2.6. *Text2Speech*

Acest modul va converti un şir de caractere într-o comandă audio, comanda pe care sistemul o va reda utilizatorului folosind un difuzor (boxele laptopului). Acest modul este mult mai simplist decât cel de *Speech2Text*, deoarece redarea conținutului audio se face folosind biblioteci *Python*.

4.3. Implementarea modulelor

Aplicaţia a fost dezvoltată pe baza modulelor mai sus descrise. Modulele comunică între ele, după cum urmează, atunci când pornim aplicaţia, adică funcţionalitatea de baza, aceasta fiind detecţia şi recunoaşterea facială, o instanţă a camerei web este pornită, în spate este rulat algoritmul reţelei neuronale, algoritm responsabil cu detecţia facială în primă instanţă. Atunci când detecţia este confirmată în captura de imagine, captura va fi redimensionată, rămânând doar faţa persoanei în cauză, după care i se vor extrage caracteristicile faciale, caracteristici care mai apoi vor fi comparate cu cele deja existente în fişierul antrenat. Fişierul antrenat nu este altceva, decât un fişier în care se găsesc caracteristicile faciale ale tuturor persoanelor din baza de date locală. În cazul în care persoana face parte din baza de date, un chenar roşu va apărea peste faţa acestuia, iar în partea de jos a chenarului se află o etichetă cu numele persoanei în cauză.

4.3.1. Implementarea generală:

Sistemul este împărţit în trei module mari ca și in *figura 4.2*:

- Modulul Audio
- Modulul Video
- Baza de Date

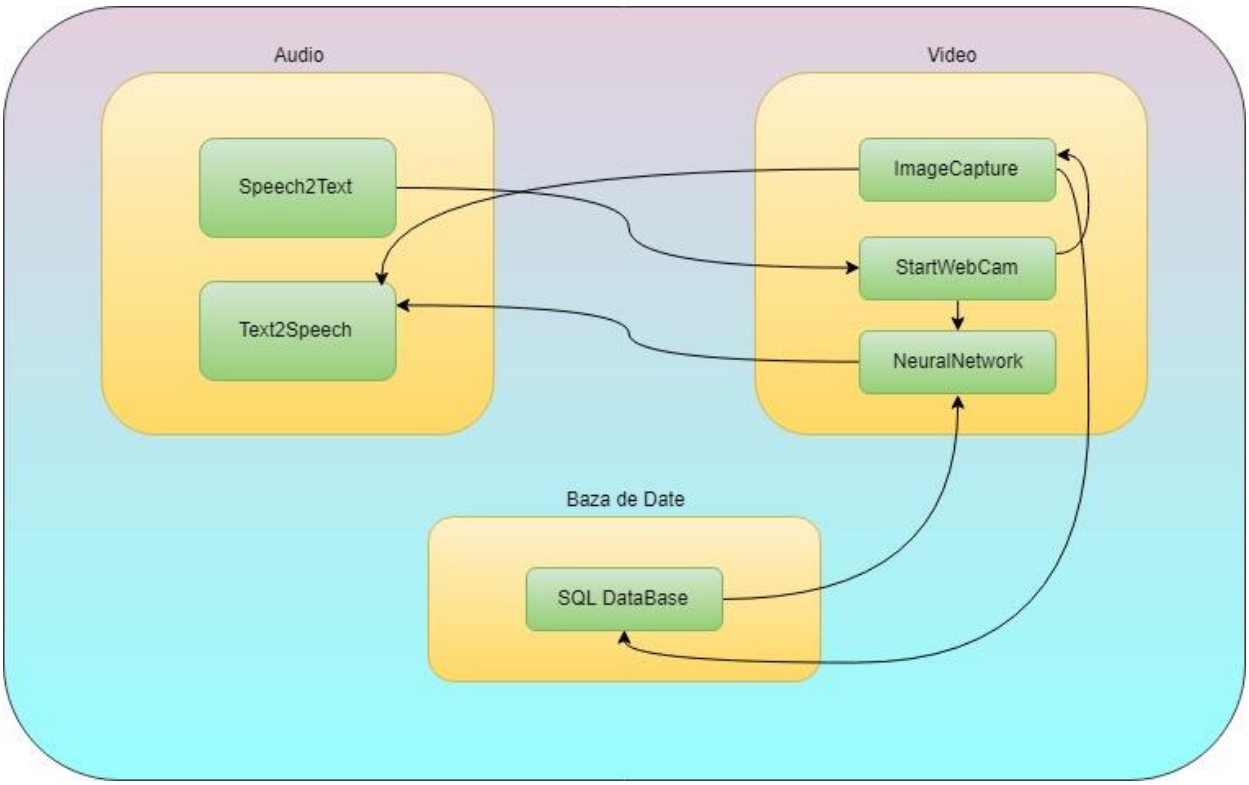

Figura 4.2. Arhitectura sistemului.

Modulul audio, este reprezentat de submodulele: *Speech2Text* şi *Text2Speech*, responsabile de convertirea semnalului audio într-un şir de caractere, respectiv convertirea unui şir de caractere într-un semnal audio. Submodulul *Speech2Text* are rolul de a oferi utilizatorului, ocazia de a interacţiona cu aplicaţia folosind comenzi vocale, comenzi pe care *API-ul Google* să le transforme într-un şir de caractere, ca mai apoi acest şir de caractere să constituie o comandă pentru aplicaţia de faţă.

Modulul video, este alcătuit din următoarele submodule: *ImageCapture*, *WebCamDeploy* şi *NeuralNetwork*. Când comanda vocală data de utilizator a fost procesată, submodulul *WebCamDeploy* va intra în acţiune, pornind o instanţă a camerei web integrate a laptopului, după care submodulul *ImageCapture*, va face trei capturi de ecran, din trei unghiuri diferite, utilizatorul fiind informat cu ajutorul submodulului *Text2Speech*, referitor la schimbarea poziţiei feţei. Submodulul *NeuralNetwork* va reantrena reţeaua ori de câte ori un nou utilizator este adăugat în baza de date, acest submodul mai este responsabil şi de confirmarea sau infirmarea apartenenţei unui utilizator în baza de date.

Baza de date, este una locală, stocată pe laptop şi conţine informaţii cu privire la utilizatori. Acest modul interacţionează cu modulul *Video*, prin faptul ca atunci când o noua persoana este introdusă sau ştearsă, baza de date este responsabilă cu această sarcină.

4.3.2. Implementare în detaliu:

În acest subcapitol se va face o descriere amănunţită referitoare la design-ul aplicaţiei şi anume ce se întâmplă în spate, atunci când se apasă un buton din design-ul grafic al interfeţei aplicaţiei. Interfața grafică poate fi observată în *figura 4.3*.

Figura 4.3. Interfaţa grafică a aplicaţiei.

Aplicaţia contine patru butoane:

- Start Aplicaţie
- Inseraţi Persoana
- Antrenare Reţea
- Închide Aplicaţia

- Butonul Start Aplicaţie prezentat în *figura 4.4.*

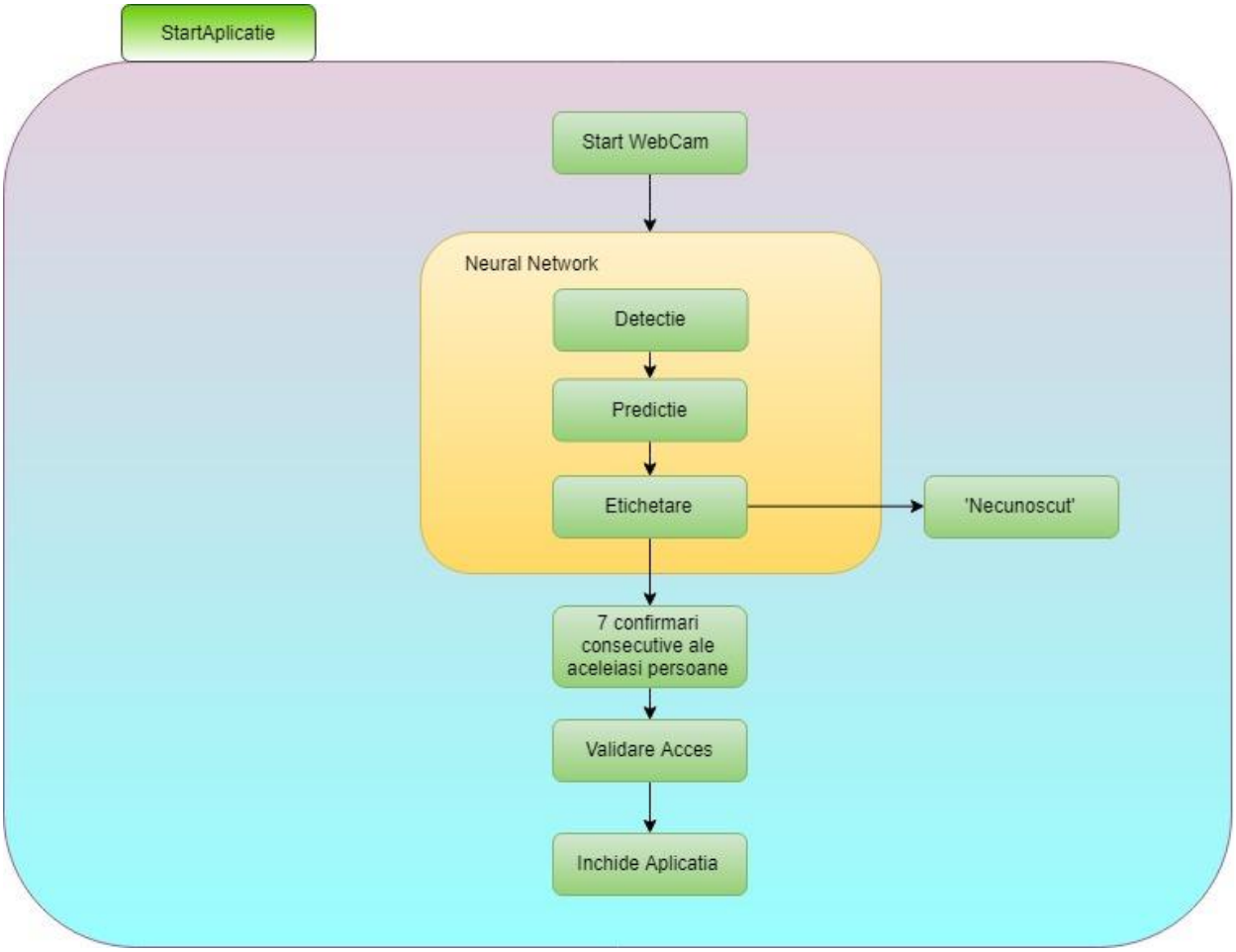

Figura 4.4. Design-ul implementării butonului de pornire a aplicaţiei.

La apăsarea acestui buton, se vor întâmpla următoarele: o noua instanţă a camerei web a laptopului va porni, fiecare cadru din noua instanţă va fi trimis către reţeaua neuronală şi va fi procesat, cu alte cuvinte în cazul în care apare o față în cadrul curent, se vor extrage

caracteristicile faciale şi se vor compara cu cele deja existente în fişierul antrenat (trained_knn_model. Clf). În cazul în care persoana există în baza de date, se va face o predicţie conturându-se faţa persoanei din cadrul curent precum în in *figura 4.5.* iar în partea de jos a conturului se va găsi o etichetă cu numele persoanei respective.

Figura 4.5. Predicţia reţelei neuronale.

În cazul în care vor fi şapte detecţii consecutive confirmate ale aceleiaşi persoane, accesul îi va fi permis, afișîndu-se imaginea din *figura 4.6.*

Figura 4.6. Interfaţa grafică, pentru accesul permis.

Dacă persoana nu se află în baza de date, i se va contura faţa şi va fi etichetat ca necunoscut, la fel ca în *figura 4.7*.

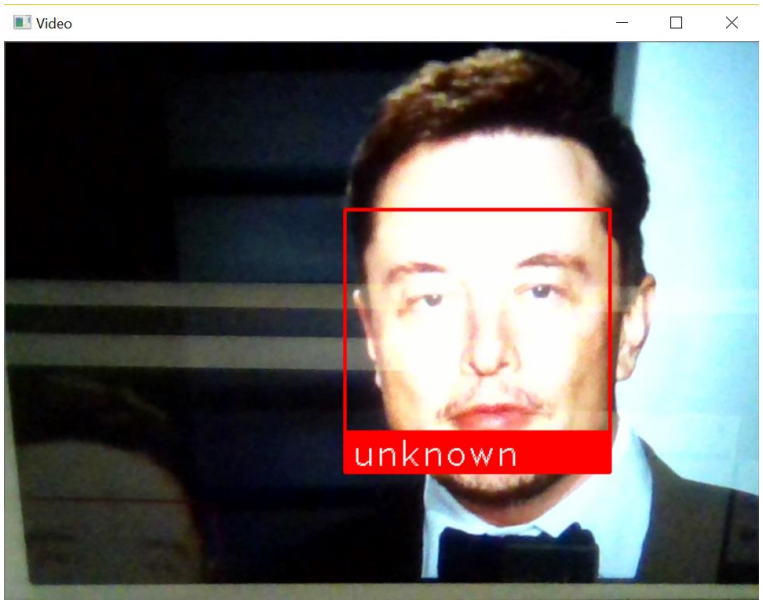

Figura 4.7. Etichetare specifică persoanelor care nu fac parte din baza de date.

După ce accesul persoanei în cauză a fost permis, se va face o reîntoarcere către meniul principal.

- Butonul Inseraţi Persoana

După apăsarea acestui buton utilizatorul va fi atenţionat printr-un mesaj vocal cu ajutorul submodulului *Text2Speech* cu privire la faptul ca va trebui să se poziţioneze în faţa camerei web a laptopului şi să folosească comanda vocală *take a photo*, comandă care va fi procesată de *API-ul Google* şi va avea ca efect pornirea unei noi instanţe a camerei web. După pornirea instanţei, utilizatorul are sarcina de a privi în cameră web şi a aştepta preţ de câteva secunde declanşarea automată a acesteia, după ce poză a fost făcută, utilizatorul va trebui să îşi introducă numele în interfaţa grafică care îi va apărea pe ecran (este de preferat să se introducă mai întâi prenumele urmat de nume, în următoarea formă: *Prenume_NUME*), după care acesta va trebui să apese pe butonul *Adaugă* din noua interfaţa, urmând să fie informat cu privire la faptul ca va trebui să îşi mute privirea puţin înspre stânga, urmând să i se facă o

noua poză, iar mai apoi înspre dreapta pentru aceeaşi acţiune. Fereastra pentru introducerea numelui se află în *figura 4.8*.

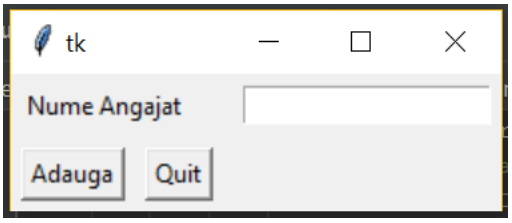

Figura 4.8. Interfaţa grafică de adăugare nume utilizator.

Aceste poze vor fi adăugate în baza de date, în total se vor găsi minim trei poze pentru fiecare utilizator. Instanţă camerei se va închide automat şi se va reveni la meniul principal. Schema implementării butonului se află în *figura 4.9*.

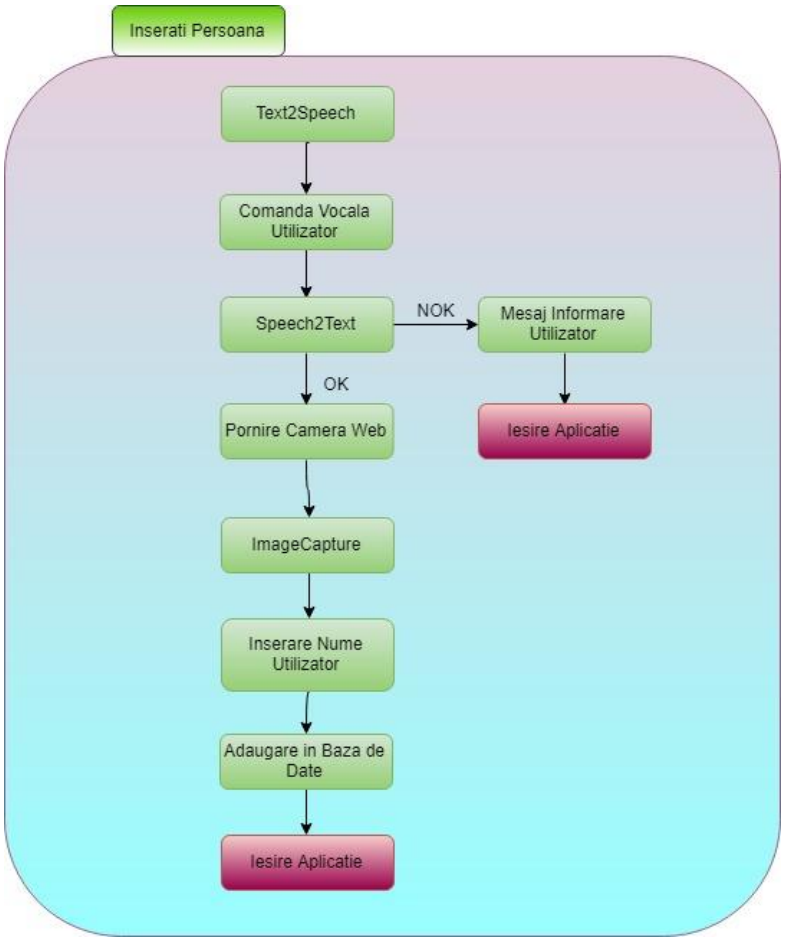

Figura 4.9. Design-ul implementării butonului de inserare a unui nou utilizator.

- Ştergerea unei persoane din baza de date

Pentru a şterge o persoana din baza de date, este necesară o navigare în structura de directoare stocată local (structura se află în *figura 4.10*), se va căuta folderul cu numele persoanei, care se doreşte a fi şters, se va şterge întreg directorul, iar apoi se va reantrena reţeaua neuronală din interfaţa grafică a aplicaţiei. Schema implementării butonului se află în *figura 4.11*.

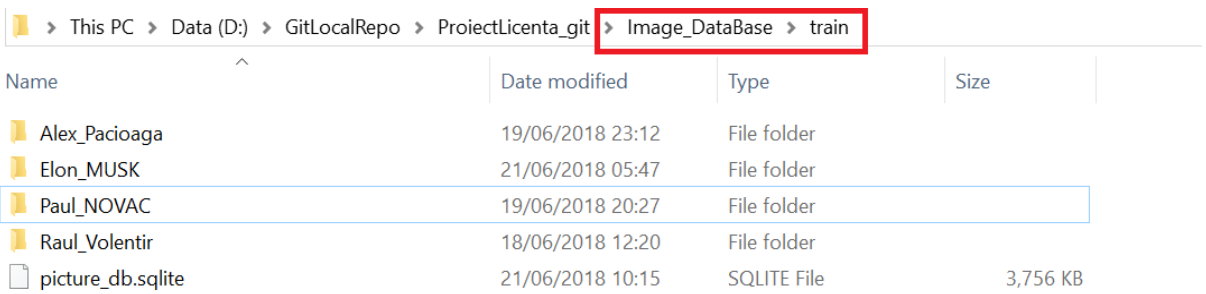

Figura 4.10. Structura de foldere a bazei de date.

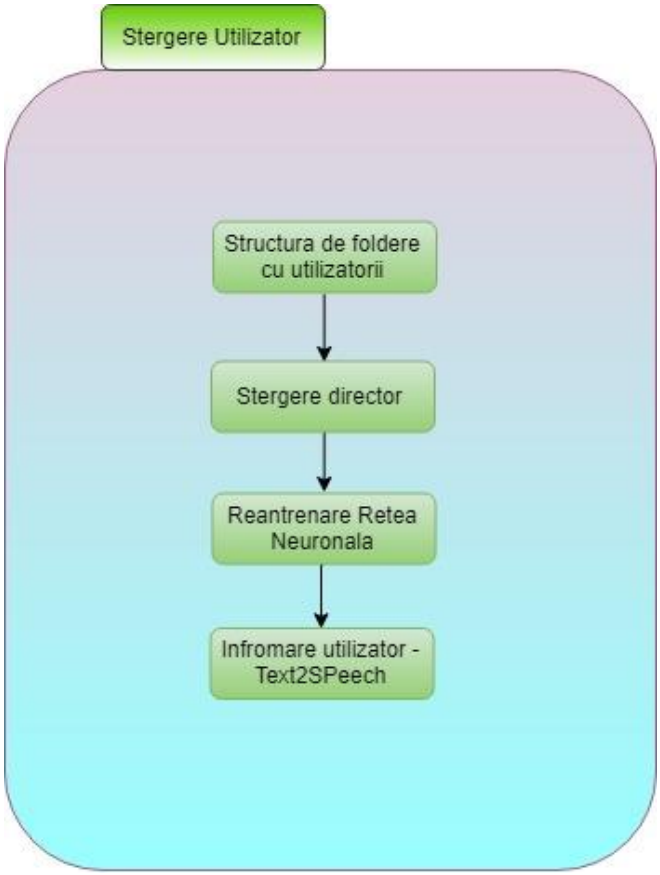

Figura 4.11. Design-ul pentru ştergerea unui utilizator.

- Butonul Antrenare Reţea

Pentru antrenarea reţelei, trebuie introdusă o parolă pentru evitarea adăugării în baza de date a utilizatorilor noi veniţi, de către persoane neautorizate, interfața grafică pentru inroducerea parolei se află în *figura 4.12*.

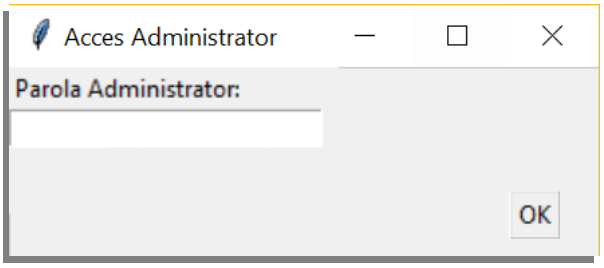

Figura 4.12. Interfaţa grafică pentru introducerea parolei de administrator.

Folosindu-se pozele persoanelor din baza de date, reţeaua va fi reantrenată, iar datele de antrenare vor fi salvate în următorul fişier: *trained_knn_model.clf*, după care administratorul va fi informat vocal cu privire la faptul ca persoana face parte din baza de date, revenindu-se la meniul principal. Schema implementării butonului se află în *figura 4.13.*

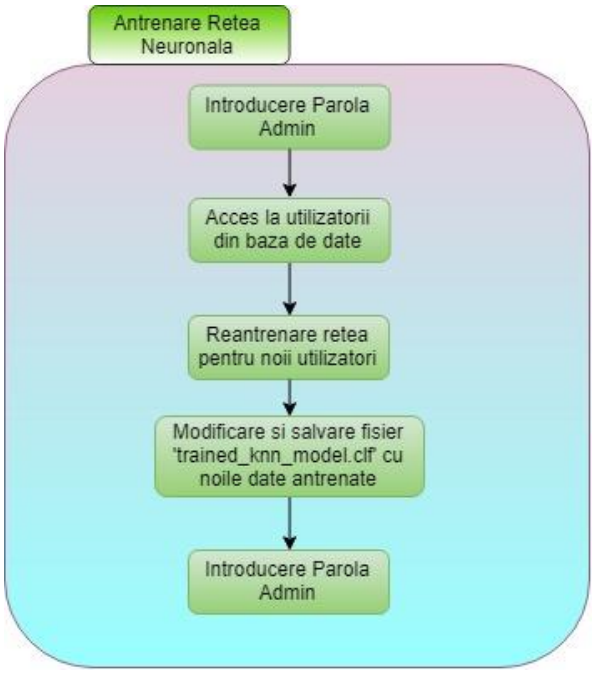

Figura 4.13. Design-ul implementării pentru reantrenarea reţelei neuronale.

4.4. Aplicaţii existente

Tehnologia de recunoaștere facială a fost asociată în mod tradițional cu sectorul de securitate, dar astăzi există o expansiune activă în alte industrii, inclusiv în marketing și sănătate.

Majoritatea cazurilor de utilizare a recunoașterii faciale se încadreze în trei categorii majore:

Securitate: companiile pregătesc algoritmi de învățare profundă pentru a recunoaște detectarea fraudei, pentru a reduce nevoia de parole tradiționale și pentru a îmbunătăți capacitatea de a face distincția între o față umană și o fotografie.

1. Aplicaţia *Kairos* – pentru detectarea fraudei

În mod evident, una dintre companiile mai mari din spațiul de recunoaștere facială a inteligenţei artificiale, *Kairos* utilizează învățarea automată și viziunea pe computer pentru a-și executa suita de instrumente care includ caracteristici de recunoaștere facială standard și alte opțiuni, cum ar fi detectarea sexului, vârstei și etniei, interfața grafică poate fi observată în *figura 4.14*.

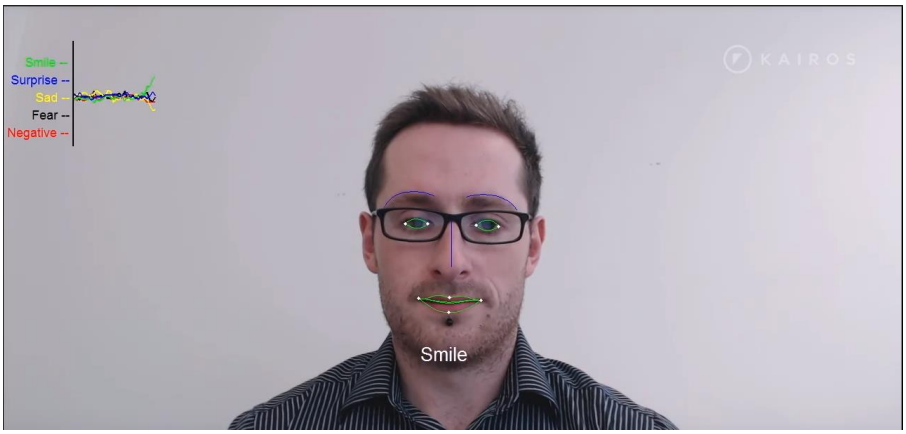

Figura 4.14. Detecţia utilizând aplicaţia *Kairos*.
Sursă: https://www.youtube.com/watch?v=vAAEpeLKyuQ

Asistența medicală: învățarea automată este combinată cu viziunea computerului pentru a urmări cu mai multă acuratețe consumul de medicamente pentru pacient și pentru a sprijini procedurile de gestionare a durerii.

2. *AiCure* - Aderare la medicație

Fondată în 2010, *AiCure (figura 4.15)* este o companie care utilizează tehnologia de recunoaștere a feței și viziunea pe computer pentru a îmbunătăți practicile de aderare la medicație.

Figura 4.15. Logo-ul aplicației AiCure.

Marketing: bogat în considerente etice, marketingul este un domeniu cu o mare creștere în inovația recunoașterii faciale.

3. *Walmart* – prevenirea furtului

În sectorul comerțului cu amănuntul, principalii comercianți par să exploreze tehnologia de recunoaștere a feței în scopuri de securitate. Cu toate acestea, s-au înregistrat unele eforturi cu privire la preocupările privind confidențialitatea consumatorilor.

În 2015, Walmart a început să testeze recunoașterea facială în unele magazine, într-un efort de identificare a ucigașilor, dar ulterior a încetat utilizarea acesteia. Interfața aplicației folosite de cei de la *Walmart* poate fi observata în *figura 4.16.*

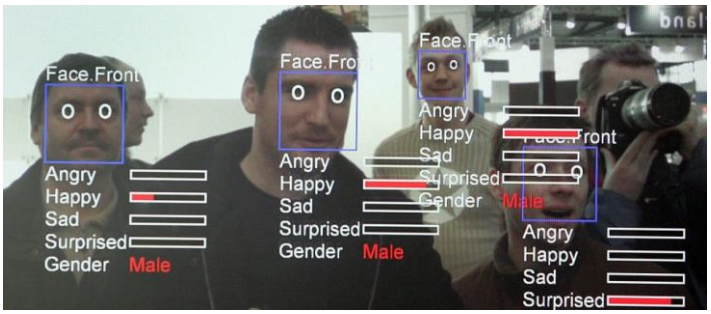

Figura 5.16. Interfața aplicației *Walmart*.

4.5. Probleme întâmpinate

Problemele întampinate de-a lungul dezvoltării au fost atât pe partea audio cât și pe partea video. Pentru a putea procesa fiecare cadru din fluxul video, în timp real și fără întârzieri, a fost necesară o redimensionare a cadrului original la o treime.

Pentru a putea rula în același timp submodulul responsabil de ascultarea comenzii vocale a utilizatorului și fluxul video în timp real, a trebuit creat pentru fiecare submodul un proces, astfel sistemul rezultat este unul cu procesare multiplă.

În cazul în care sistemul rulează într-un mediu zgomotos, este recomandat să fie folosit un microfon, pentru comenzile vocale ale utilizatorilor, deoarece sistemul este făcut să aștepte utilizatorul până când acesta termină comanda vocală, în cazul în care mediul este unul zgomotos, sistemul nu se va opri din ascultare.

O altă problemă întâmpinată a fost generarea jetoanelor de acces folosind API-ul de la Facebook, această problemă nu a fost rezolvată în versiunea curentă și este menționată în *Capitolul 6. Direcții de continuare a dezvoltării.*

Capitolul 5. Utilizarea sistemului

5.1. Ghid de instalare

Pentru utilizarea aplicației vom avea nevoie de următoarele:

Mediul de dezvoltare din care va rula aplicația (ca și recomandare *PyCharm).* Instalarea comenzii *pip,* pentru descărcarea si gestionarea pachetelor *Python*. Pachetele necesare pentru rularea aceastei aplicații sunt precizate în submodulul *3.1.1. Dezvoltare,* pentru instalarea pachetului *dlib,* este necesară instalarea instrumentului *CMake* și setarea variabilelor de sistem. Pentru vizualizarea datelor din baza de date se va instala *DB Browser for SQLite*.

5.2. Limitările sistemului

Sistemul este limitat să acorde acces unei singure persoane, însă el este capabil să recunoască toate persoanele din cadru. Acest lucru se întâmplă, deoarece este necesară o confirmare a prezenței aceleiași persoane pentru șapte cadre la rând, acest lucru ajută la îmbunătățirea performanței algoritmului.

Atunci când avem de-a face cu o persoană care poartă ochelari, iar locul în care este amplasat sistemul este slab iluminat, este posibil ca aplicația să nu poată detecta persoana în cauză, acest lucru se poate datora reflexiei ochelarilor.

În cazul în care sistemul este amplasat într-un mediu cu lumină foarte scăzută, detecția va avea mult de suferit, de cele mai multe ori procesarea cadrului nu poate avea loc, astfel permiterea accesului este compromisă.

Dacă utilizatorul folosește altă comandă în afară de cea pe care sistemul o acceptă, aplicația nu va face nimic.

O altă limitare a sistemului este necesitatea conexiunii la internet pentru decodificarea comenzii vocale a utilizatorului.

Capitolul 6. Direcţii de continuare a dezvoltării

Pentru a eficientiza procesul de verificare şi al îmbunătăţi, după ce utilizatorului i se fac cele trei poze, mai întâi se va verifica faptul ca există o față în fiecare poză, după care se va confirma faptul ca aceeaşi persoana se află în cele trei poze, în caz contrar, procesul trebuie reluat.

Implementarea unei baze de date, care să ţină evidenţa orei de venire, respectiv plecare pentru fiecare angajat, extinde aplicaţia, ajutând departamentele de resurse umane. Posibilitatea implementării unui raport la nivel de organizaţie, ajută managementul la o mai bună urmărire a situaţiei fiecărui angajat şi totodată este în avantajul angajatului care nu mai are nevoie de card de acces, pontarea manuală astfel devine istorie.

Înştiinţarea departamentului de IT printr-un mail, atunci când sistemul detectează o persoana fără drept de acces, este o funcţionalitate extrem de folositoare.

Pentru o mai bună predicţie a reţelei neuronale, este necesară o baza de date cât mai mare, cu alte cuvinte este necesar ca pentru fiecare utilizator să avem cât mai multe poze, din unghiuri diferite. Acest lucru se poate realiza foarte uşor, folosind pozele de profil de pe *Facebook* ale acestuia. Pentru acest lucru vom avea nevoie ca utilizatorul să se autentifice cu contul de Facebook, urmând a fi generat un jeton de acces, pentru a descărca pozele sale de profil. [19]

În programarea pe calculator, o interfaţa de programare a aplicatiilor (*API*) este un set de definiţii, protocoale şi instrumente de subrutină pentru construirea de software de aplicaţii. În termeni generali, este un set de metode clar definite de comunicare între diferitele componente software.

Un *API* bun face mai uşoară dezvoltarea unei aplicaţii *desktop,* prin furnizarea tuturor blocurilor de construcţie, care sunt mai apoi asamblate de către programator.

Specificaţiile *API* pot lua mai multe forme, dar adesea sunt incluse specificaţii pentru rutine, structuri de date, clase de obiecte, variabile sau apeluri la distanţă. Documentaţia pentru *API* este furnizată de obicei pentru a facilita utilizarea şi implementarea.

La fel ca o interfaţa grafică, facilitează utilizarea programelor de către dezvoltatori. Prin abstractizarea implementării care stă la baza şi prin expunerea obiectelor sau acţiunilor de care dezvoltatorul are nevoie, un *API* simplifică programarea [19].

Un jeton de acces conţine un identificator de securitate (*SID*) pentru utilizator, toate *SID-urile* pentru grupurile cărora le aparţine şi privilegiile utilizatorului. Un token de acces reprezintă un obiect încapsulând identitatea de securitate a unui proces sau a unui fir. Un jeton este folosit pentru a lua deciziile de securitate şi pentru a stoca informaţii despre o entitate a sistemului. În timp ce un token este utilizat în general pentru a reprezenta numai informații de securitate, el este capabil să dețină date suplimentare de formă liberă care pot fi atașate încă din timpul creări token-ului.

Ori de câte ori un fir sau un proces interacţionează cu un obiect securizat sau încearcă să efectueze o sarcină de sistem care necesită privilegii, sistemul de operare verifică eficacitatea accesului efectiv pentru a determina nivelul sau de autorizare.

Atunci când cineva doreşte să se conecteze cu o aplicaţie folosind *Facebook Login*, iar solicitarea îi este aprobată pentru permisiunile bifate, aplicaţia obţine un indicativ de acces temporar.

Un jeton de acces este un şir de caractere opac care identifică un utilizator, o aplicaţie sau o pagină şi poate fi folosit de aplicaţie pentru a efectua apeluri *API* grafice. Jetoanele de acces sunt obţinute printr-un număr de metode. Indicativul conţine informaţii despre momentul expirării token-ului şi a aplicaţiei care a generat jetonul. Din cauza controalelor de confidenţialitate, majoritatea apelurilor *API* pe *Facebook* trebuie să includă un jeton de acces.

Există diferite tipuri de jetoane de acces pentru a sprijini diferite cazuri de utilizare:

- Jeton pentru accesul către uşer, utilizat ori de câte ori aplicaţia solicită ca *API-ul* să citească, modifice sau să scrie date ale unei anumite persoane pe *Facebook* în numele lor.
- Jeton pentru accesul aplicaţiilor, necesar pentru a modifica şi citi setările aplicaţiei.

- Jeton pentru accesul unei pagini, aceste tipuri de jetoane sunt similare cu cele de acces din partea utilizatorului, cu excepţia faptului ca oferă permisiunea *API-urilor* care citesc, scriu sau modifică datele aparţinând unei pagini *Facebook* [19].

Pentru a genera un jeton cu acces pentru pagină, un administrator al paginii trebuie să acorde o permisiune extinsă numită *manage_pages*. Odată ce aceasta permisiune a fost acordată, jetonul de acces la pagină poate fii preluat.

Deşi fiecare platformă generează jetoane de acces prin *API-uri* diferite, toate platformele respectă strategia de baza pentru a obţine un jeton de utilizator:

Atunci când un nou utilizator (angajat) este adăugat în baza de date, acesta trebuie să se autentifice cu contul de *Facebook*, pentru ca generarea de token să aibă loc, generare care îi va permite sistemului să acceseze pozele sale de profil. Aceste poze vor fi adăugate în baza locală de date, însă pentru a avea acces în clădire, sistemul trebuie să valideze persoana în cauză cu una existentă în baza de date, acest lucru va fi posibil numai după ce administratorul sistemului, va reantrena reţeaua neuronală.

Toate jetoanele de acces trebuie să fie reînnoite la fiecare 90 de zile, cu acordul persoanei care utilizează aplicaţia. Acest lucru nu blochează sistemul, deoarece odată ce jetonul a fost primit, pozele de profil sunt salvate local în baza de date, astfel utilizatorul trebuie să facă o singură autentificare pentru a face parte din sistem.

Capitolul 7. Concluzii

În aceasta lucrare se prezintă o aplicaţie *Desktop* pentru autentificare, folosind recunoaştere facială, dezvoltată în limbajul de programare *Python*, care utilizează *Google API* pentru procesarea comenzii vocale a utilizatorului.

Aplicaţia este foarte uşor de utilizat datorită unui design grafic intuitiv şi datorită posibilităţii de a interacţiona cu aceasta aplicaţie, folosind comenzi vocale. Autentificarea este rapidă şi eficientă, înregistrarea în baza de date şi autentificarea se realizează în trei paşi simpli, introducerea în baza de date a utilizatorului, antrenarea reţelei neuronale şi în final autentificarea. Aplicaţia aduce în prim plan tehnologii noi, care sunt din ce în ce mai folosite şi vor avea un impact uriaş asupra omenirii.

Direcţiile de dezvoltare se petrec în sfera eficientizării şi securităţii, dar şi în zona de management prin implementarea şi generarea unor rapoarte.

Capitolul 8. Bibliografie

[1] E. Micheli-Tzanakou, K. N. Plataniotis, and N. V. Boulgouris, Biometrics theory, methods, and applications. Piscataway, N.J.; Hoboken, N.J.: IEEE Press ; Wiley, 2010.

[2] Department of Computer Science & Engineering, Velammal Engineering College, Tamil Nadu, India, P. P., B. Ganesh A, and TIFAC-CORE, Velammal Engineering College, Tamil Nadu, India, 'DETECTION OF HUMAN FACIAL BEHAVIORAL EXPRESSION USING IMAGE PROCESSING', ICTACT J. Image Video Process., vol. 01, no. 03, pp. 162–165, Feb. 2011.

[3] F. Y. Shih, Image processing and pattern recognition fundamentals and techniques. Piscataway, NJ; Hoboken, N.J.: IEEE Press ; Wiley, 2010.

[4] T. Rashid and E. B. July, 'A Gentle Introduction to Neural Networks', p. 85.

[5] I. N. da Silva, D. Hernane Spatti, R. Andrade Flauzino, L. H. B. Liboni, and S. F. dos Reis Alves, 'Artificial Neural Network Architectures and Training Processes', in *Artificial Neural Networks*, Cham: Springer International Publishing, 2017, pp. 21–28.

[6] Z. Uibo, 'Comparison of Face Recognition Neural Networks', p. 23.

[7] F. Chollet, *Deep learning with Python*. Shelter Island, New York: Manning Publications Co, 2018.

[8] Modern Face Recognition with Deep Learning, Sursă: https://medium.com/@ageitgey/machine-learning-is-fun-part-4-modern-face-recognition-with-deep-learning-c3cffc121d78

[9] J. V. Guttag, *Introduction to Computation and Programming Using Python: With Application to Understanding Data*. MIT Press, 2016.

[10] R. L. Halterman, 'LEARNING TO PROGRAM WITH PYTHON', p. 283, 2011.

[11] C. R. Severance, 'Python for Everybody', p. 243.

[12] D. L. Poole and A. K. Mackworth, 'Python code for Artificial Intelligence: Foundations of Computational Agents', p. 221.

[13] M. Ambrožič *et al.*, *The Digital Library of Slovenia development strategy - dLib.si 2007-2010*. Ljubljana: National and University Library, 2007.

[14] https://www.pyimagesearch.com/2017/04/03/facial-landmarks-dlib-opencv-python/

[15] X. Huang and L. Deng, 'An Overview of Modern Speech Recognition', p. 29.

[16] D. Rocchesso, *Introduction to sound processing*. Firenze: Mondo estremo, 2003.

[17] https://github.com/ageitgey/face_recognition

[18] https://cloud.google.com/speech-to-text/docs/

[19] M. Dodoo, 'Facebook SDK for Python Documentation', p. 18.