UNIVERSITATEA “ALEXANDRU IOAN CUZA” IAȘI

**FACULTATEA DE INFORMATICĂ**

****

LUCRARE DE LICENȚĂ

**Mimic**

propusă de

**Nechita Bogdan-Radu**

**Sesiunea:** Iulie, 2019

Coordonatori științifici

**Conf. Dr. Anca Vitcu și Conf. Dr. Adrian Iftene**

UNIVERSITATEA “ALEXANDRU IOAN CUZA” IAȘI

**FACULTATEA DE INFORMATICĂ**

**Mimic**

**Nechita Bogdan-Radu**

**Sesiunea:** Iulie, 2019

Coordonatori științifici

**Conf. Dr. Anca Vitcu și Conf. Dr. Adrian Iftene**

DECLARAȚIE PRIVIND ORIGINALITATEA ȘI RESPECTAREA DREPTURILOR DE AUTOR

Prin prezenta declar că Lucrarea de licență cu titlul „*titlu*” este scrisă de mine și nu a mai fost prezentată niciodată la o altă facultate sau instituție de învățământ superior din țară sau din străinătate. De asemenea, declar că toate sursele utilizate, inclusiv cele preluate de pe Internet, sunt indicate în lucrare, cu respectarea regulilor de evitare a plagiatului:

- toate fragmentele de text reproduse exact, chiar și în traducere proprie din altă limbă, sunt scrise între ghilimele și dețin referința precisă a sursei;

- reformularea în cuvinte proprii a textelor scrise de către alți autori deține referința precisă;

- codul sursă, imaginile etc. preluate din proiecte *open*-*source* sau alte surse sunt utilizate cu respectarea drepturilor de autor și dețin referințe precise;

- rezumarea ideilor altor autori precizează referința precisă la textul original.

Iași, *01.07.2019*

Absolvent *Autor*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(semnătura în original)

DECLARAȚIE DE CONSIMȚĂMÂNT

Prin prezenta declar că sunt de acord ca Lucrarea de licență cu titlul „*titlu*”, codul sursă al programelor și celelalte conținuturi (grafice, multimedia, date de test etc.) care însoțesc această lucrare să fie utilizate în cadrul Facultății de Informatică.

De asemenea, sunt de acord ca Facultatea de Informatică de la Universitatea „Alexandru Ioan Cuza” din Iași, să utilizeze, modifice, reproducă și să distribuie în scopuri necomerciale programele-calculator, format executabil și sursă, realizate de mine în cadrul prezentei lucrări de licență.

Iași, *01.07.2019*

Absolvent *Autor*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(semnătura în original)

**Cuprins**

[Motivație](#_Toc12842955)

[Introducere](#_Toc12842956)

[**Capitolul I: Încercări similare** 1](#_Toc12842957)

[**Serendipity** 1](#_Toc12842958)

[**ViBand** 2](#_Toc12842959)

[Contribuții personale 3](#_Toc12842960)

[**Capitolul II: Tehnologii utilizate** 4](#_Toc12842961)

[**II.1. Java - Spring Boot** 4](#_Toc12842962)

[**Motivație** 4](#_Toc12842963)

[**II.2. Samsung** 5](#_Toc12842964)

[**II.2.1 Tizen OS** 5](#_Toc12842965)

[**Capitolul III: Arhitectura sistemului** 8](#_Toc12842966)

[**IV.1. Pașii pentru implementare** 14](#_Toc12842967)

[**Crearea unei aplicații pentru un smartwatch cu Tizen OS** 14](#_Toc12842968)

[**Crearea unei conexiuni între server și ceas** 15](#_Toc12842969)

[**Observații** 16](#_Toc12842970)

[**Concluzii** 17](#_Toc12842971)

[**Crearea unei aplicații pentru vizualizarea, prelucrarea și analiza datelor** 17](#_Toc12842972)

[**Salvarea datelor** 18](#_Toc12842973)

[**Vizualizarea și prelucrarea datelor salvate** 19](#_Toc12842974)

[**Criterii de recunoaștere și algoritmul** 20](#_Toc12842975)

[**Algoritmul de antrenare** 23](#_Toc12842976)

[**Recunoașterea în timp real (aproape)** 24](#_Toc12842977)

[**Concluzii** 25](#_Toc12842978)

[**Păreri externe** 25](#_Toc12842979)

[**Vorbitor în public profesionist** 25](#_Toc12842980)

[**Vorbitor în public la conferințe tehnice** 25](#_Toc12842981)

[**Companie de training-uri tehnice** 26](#_Toc12842982)

[**InnovationLabs 2019** 26](#_Toc12842983)

[**Proiect european – BonAntreprenor** 26](#_Toc12842984)

[**Ce urmează** 26](#_Toc12842985)

[Calibrarea programului 26](#_Toc12842986)

[Testarea cu toți cei 3 senzori activi 26](#_Toc12842987)

[Determinarea celui mai relevant senzor în recunoașterea de gesturi în timp ce o persoană execută activități normale de zi cu zi 28](#_Toc12842988)

[Evitarea detectărilor false 28](#_Toc12842989)

[Crearea unui mod de procesare a datelor automat și folosirea mai multor canditați pentru adunarea de date și testarea sistemului 28](#_Toc12842990)

[Imbunatatirea vizuală a aplicației 28](#_Toc12842991)

[Transferarea algoritmului pe ceas folosind ori C, ori GoLang 29](#_Toc12842992)

[Gasirea unei formule pentru determinarea orientarii ceasului 29](#_Toc12842993)

[Crearea unei aplicatii care sa “reactioneze” la gesturile inregistrate 29](#_Toc12842994)

[Bibliografie 31](#_Toc12842995)

[ANEXE 34](#_Toc12842996)

# 

# Motivație

Tema generală a lucrării este augmentarea gesturilor efectuate de o persoană folosind un dispozitiv raspândit și uzual, precum un smartwatch.

Ideea a plecat de la faptul că dacă o persoană dorește să țină o prezentare și are atașate acestei prezentării și o prezentare Power Point, aceasta este constransă să țină un pointer sau telefon în mână sau să ramână în proximitatea laptopului pentru a putea schimba diapozitivele. Asfel, m-am gandit că o aplicație pentru un smartwatch care să înregistreze un anumit gest mai fin decât o mișcare amplă a mâinii și care comunică cu o prezentare Power Point (spre exemplu) ar aduce un plus de valoare unui prezentator, mai ales unuia care este foarte expresiv din punct de vedere al limbajului trupului.

Având în vedere ca vanzarile de smartwatch au crescut pe parcursul anilor (conform surselor 67-71) cu un număr considerabil de la an la an, mai exact cu 66 milioane de unități în 2018 fața de 2017, când s-au vândut 71 de milioane de unități, putem presupune că oamenii prezintă un interes suficient de ridicat pentru astfel de produse încat soluția propusă să fie viabila. Numerele prezentate mai sus sunt doar pentru dispozitive de tip smartwatch. Mai sunt și brățările de fitness, care reprezintă alt segment de piață, segment care ar putea fi luat în considerare în viitor.

De asemenea, ceasurile sunt un accesoriu cu care oamenii, în general, sunt obisnuiți și pe care îl poartă aproape zilnic. Chiar dacă nu mai este la fel de popular ca în ani precedenți, este usor de acceptat de către o persoană să poarte un astfel de dispozitiv. Am menționat acest punct pentru că un ceas deranjează mai puțin decât o manușă specială pentru detectarea gesturilor.

De asemenea, am cautat detalii despre alte implementari similare și am găsit două variante promițătoare: Serendipity și ViBand. Indiferent de ce au facut alte persoane, concluzia este că se poate.

# Introducere

Am început să găsesc o metodă care înlătură nevoia de a sta în proximitatea laptopului sau de a ține ceva în mâini și care permite unui utilizator să schimbe diapozitivele unei prezentări Power Point. Soluția gasită este o aplicație pentru un smartwatch.

Aceasta aplicație folosește senzorii din dispozitivul disponibil în comerț pentru a detecta gesturi de o anumită finețe. În urma acestei detectări, o comandă este trimisă unei alte aplicații care schimbă diapozitivul.

După această idee de început, am decis că funcționalitatea poate să fie extinsă. Ce se întamplă dacă prezentatorul execută gestul în mod inconstient și schimbă diapozitivul într-un moment inoportun? Răspunsul imediat va fi sa înceteze să mai utilizeze aplicația pentru a nu-i provoca alte inconveniențe în timpul prezentarilor (comportament confirmat de prezentatori profesioniști ăn cadrul unei discuții în care le-am cerut părerea). M-am gândit că aplicația poate fi folosită și la controlul dispozitivelor din casă (becuri, aer conditionat, etc.). Detecțiile false ar putea provoca situații cel puțin neplăcute. Astfel am ajuns la concluzia că trebuie creată o metodă prin care ceasul să știe pozitionarea obiectelor pe care ar trebui sa le afecteze. Ideea s-a conturat în jurul unui sistem de “înregistrare” a locațiilor.

Ideile finale au fost:

* Crearea unei aplicații pentru un ceas care să înregistreze gesturile executate de un utilizator
* Crearea unei interfețe la care ceasul să semnaleze înregistrările, iar mai departe să se execute funcționalitațile dorite de utilizator. Scopul ei este să transmită mai departe comenzile asociate gesturilor
* Condiționarea executării comenzilor de orientarea ceasului

În urmatoarele pagini voi detalia progresul și voi prezenta atât problemele întâmpinate,

cât și soluțiile aplicate. Ca o scurtă descriere a progresului, aplicația se află la stadiul ân care se folosește doar unul din cei 3 senzori aleși pentru identificarea gesturilor (în timp real), iar gesturile identificate sunt executate dintr-o poziție standard. De asemenea, aplicația nu se află pe ceas, ci pe un server. Ceasul are o aplicație care transmite datele pe care le înregistrează senzorii către un server creat cu Spring Boot. Serverul poate să afișeze, să prelucreze datele și să detecteze gesturi. Nu s-au folosit baze de date încă. Datele sunt stocate în fișiere text. Cu toate că este rudimentară, aceasta formă de stocare este suficientă.

Dar înainte de aceste aspecte, voi prezenta și ce au facut cei de la Serendipity și ViBand, doua proiecte similare, dar cu abordări diferite.

# **Capitolul I: Încercări similare**

## **Serendipity**

Acesta este proiectul de la care m-am inspirat. Este un studiu ce are că obiect recunoașterea gesturilor cu degetele folosind un smartwatch comercial. Conform autorilor, este primul studiu pentru recunoașterea gesturilor fine folosind doar senzorii de mișcare ) *[extras din documentul celor de le Serendipity, secțiunea “Abstract”]*

Cele mai importante idei extrase de mine din acest document sunt:

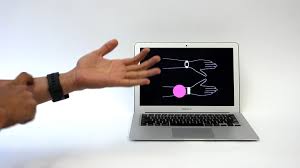
* Conceptul este fezabil
* Sistemul creat de ei necesită un gest de activare
* Utilizatorii au menționat la final că ar prefera, în anumite instanțe, gesturile mai “mari” (e.g. strâns pumnul în loc de ciupit cu două degete) *[extras din documentul celor de le Serendipity, secțiunea “User feedback”]*
* Au folosit mai multe orientări ale ceasului
* Au determinat importanța fiecarui senzor pentru o anumită categorie de gesturi
* Au obținut o precizie de 87% cu o deviație standard de 7,9%
* Clasificatorul cel mai bun este variabil în functie de utilizator
* Gesturile trebuie executate cu o anumită viteză consistentă
* Aplicabilitățile spre care au dorit să se ducă: domeniul medical (studiul unor boli precum Parkinson și interacțiunea cu alte dispozitive)

Având aceste idei extrase, am încercat să văd dacă este vreo metodă de a scapa de acel gest de activare. Cu toate că elimină multe probleme și simplifică treaba clasificatorului, nu este natural. Am dorit să găsesc o cale prin care o persoană poate să execute gesturi în mod natural fară să fie nevoie să se gândească la sistem.

De asemenea, viteza cu care se execută gesturile nu ar trebui să fie o constrângere. Într-adevăr, perioada în care se efectuează gestul nu ar trebui să fie foarte lungă, având în vedere faptul că gesturile sunt fine și o perioadă foarte mare de execuție a unui gest (cu o frecvență redusă a mișcărilor) va face ca acel gest să nu fie recunoscut. În același timp, în funcție de gest, variații în executarea lui sunt admisibile. Din nou, căutam ca gestul să fie cât mai natural.

În concluzie, acest proiect a oferit ideea de bază – recunoașterea unor gesturi fine folosind un smartwatch nealterat din magazin.

## **ViBand**

[](http://www.gierad.com/assets/viband/highres/bb_GestureArm.jpg)

Este un alt proiect care folosește un smartwatch comercial. Creatorii proiectului au observat că se pot identifica gesturi folosind doar accelerometrul, dar frecvența standard de 100Hz pe care o au dispozitivele comerciale pentru a depista orientarea este insuficientă pentru a detecta în mod clar un gest. Au ajuins la concluzia că alterarea kernel-ului sistemului de operare a ceasului pentru a crește frecvența cu care este citit accelerometrul oferă posibilitatea identificării unor gesturi foarte fine.

Diferențele principale între acesta și Serendipity sunt:

* ViBand este mult mai complex
* ViBand folosește doar un senzor, accelerometrul, pe când Serendipity a folosit 5
* Serendipity nu modifică ceasul. ViBand suprescrie driverul IMU al ceasului pentru a mări frecvența cu care un senzor înregistrează datele
* ViBand folosește, de asemenea, și recunoaștere de sunete
* ViBand are un spectru mai mare de recunoaștere a gesturilor și nu numai (detectează activitatea pe care o face o persoană și obiectul pe care îl are în mână – într-un mod limitat, dar are aceasta capacitate)
* ViBand a obținut o precizie de 94,3% cu deviație standard de 4,1 %
* Acuratețea poate fi imbunațită considerabil expunând clasificatorul la o mulțime de date cât mai mare
* Aplicabilitățile explorate: controlul infrastructurii și al sofware-ului, detectarea de obiecte

Dezavantaje:

* Faptul că aplicația suprascrie un driver al sistemului de operare aduce complicații precum încetarea funcționării altor aplicații care depind de acel driver *[informatie luata de pe github-ul celor de la ViBand (referinta bibliografica 3)]*
* Conform <http://www.gierad.com/projects/viband/>, ultimul paragraf: durata de viață a bateriei este afectată

# Contribuții personale

Am început să detectez gesturi folosind un algoritm propriu și datele de la 3 senzori: giroscopul, accelerometrul și accelerația liniară. Am început sa clasific gesturile în categorii și să observ care sunt cele pe care ceasul are cele mai mari șanse să le detecteze.

Am creat o aplicație care permite vizualizarea și corectarea datelor de antrenament, cât și vizualizarea înregistrărilor senzorilor în timp real.

Detectarea gesturilor este mai puțin precisă în acest moment (încă nu e completă analiza datelor, în sensul că se folosește doar un senzor, giroscopul), dar după finalizare, intenționez ca aplicația să detecteze un mulțime de gesturi în mod consistent fără să alterez ceasul în vreun fel. Pe lângă acest aspect, aplicația va lua în calcul atât orientarea ceasului și proximitatea lui față de obiectele de interes, cât și natura gesturilor executate.

De asemenea, unele din aplicațiile finale pe care le propun au avut parte de un mic sondaj în care am colectat feedback-ul persoanelor implicate și am extras atât atât atitudinea lor față de ele, cât și sugestiile lor.

Nici unul dintre proiectele prezentate mai sus nu au încercat sa lege o prezentare Power Point de recunoasterea gesturilor cu ceasul și nici nu s-au raportat la pozitia ceasului fața de obiectele din jur.

# **Capitolul II: Tehnologii utilizate**

## **II.1. Java - Spring Boot**

Spring este un framework peste Java EE menit să faciliteze dezvoltarea de aplicații prin șablonul dependency-injection. Avantajul este dat de decuplarea obiectelor de instantierea dependințelor lor. A apărut că o soluție pe vremea în care Java EE nu era foarte prietenoasa în ceea ce privește dezvoltarea de aplicații enterprise, iar cu timpul a devenit un framework care ajută programatorii să creeze aplicații enterprise foarte rapid.

Spring Boot este încă un nivel de abstractizare peste Spring, a cărui scot este facilitarea

dezvoltării micro-serviciilor. Cu Java EE și Spring, înainte de a dezvolta aplicația este necesară configurarea unui server de aplicații precum Tomcat sau JBoss. Spring Boot vine cu tomcat inclus în arhiva finală, eliminând grija server-ului și permițând unui programator să treacă direct la pasul de dezvoltare. Acesta este motivul principal pentru care am ales acest framework.

Consider că ar trebui să adaug faptul că Spring și Spring Boot nu sunt standarde Java. Acestea au construit peste Java EE și prin urmare pot fi mai lente (deși diferențele, cel puțin din ce am reușit să experimentez până acum, nu sunt sesizabile în contextul acestei lucrări).

### **Motivație**

Având în vedere cele scrise mai sus, putem extrage faptul că am folosit un framework peste Java menit pentru dezvoltarea rapidă a micro-serviciilor. Motivele pentru care am facut această alegere sunt:

* Înainte de a dezvolta o aplicație pentru un smartwatch care să detecteze gesturi am avut nevoie de o aplicație care să mă ajute să colectez date, să le vizualizez și să le prelucrez. La vremea respectivă, limbajul cu care eram cel mai obișnuit era Java (având o experiență de 2 ani de lucru în mediul corporatist, unde am folosit atât Java EE, cât și Spring)
* Având aceasta experiență, am optat pentru Spring Boot datorită ușurinței cu care se putea crea un microserviciu
* Spring Boot ajută și la crearea efectivă a proiectului. Folosind https://start.spring.io/, un programator poate crea un proiect cu toate dependințele necesare folosind această interfață grafică, fară să fie nevoit să adauge manual dependințele în pom.xml (fișier care conține toate dependințele și instrucțiunile de build a unui proiect)
* Spring Boot oferă management-ul versiunior dependințelor. Fiind un framework, Spring Boot are la rândul său dependințe. Pentru a adaugă Spring Boot într-un proiect, una din condiții este ca pom.xml-ul proiectului să moștenească pom-ul Spring Boot. Astfel, se pot aduce dependințe fară a se mai specifica versiunea lor. Creatorii Spring Boot se ocupă de management-ul versiunilor. Acesta poate și avantaj, și dezavantaj
* Fiind un framework foarte popular, există o cantitate suficientă de documentație, materiale de învățare, tutoriale și oameni pasionați un comunitate încât majoritatea piedicilor din timpul dezvoltării sunt soluționate repede
* Configurarea aplicației este facilă, proprietățile ei fiind stocate într-un fișier, iar aplicația citește acest fișier la pornire și se configurează singură

## **II.2. Samsung**

[](https://zdnet4.cbsistatic.com/hub/i/r/2017/01/12/a90de462-7dce-497a-b479-312489de634a/resize/1170x878/b1aa255bc4e5c5648c5754fb3e10e5e4/gear-s3-notification.jpg)

Pentru dezvoltarea acestui proiect am ales ceasul Samsung Gear S3 Classic, motivul fiind faptul că cei ce au făcut proiectul Serendipity au folosit o iterație mai veche a acestei serii. Astfel am dorit să asigur un mediu similar cu cel al lor.

Detaliile cele mai importante legate de acest ceas sunt:

* Oferă acces la accelerometru – determină mișcarea ceasului într-o directie
* Oferă acces la giroscop – adică acces la rotația ceasului, ajutând la determinarea orientării ceasului (în combinație cu accelerometrul)
* Oferă acces la accelerația liniară – îți spune accelerația într-o anumită direcție
* Nu are magnetometru, ceea ce necesită ca orientarea ceasului în functie de punctele cardinale să fie calculată artificial (soluția gasită de mine este ca atunci când aplicația este pornită, utilizatorul trebuie să stea cu fața spre nord. Încă nu a fost implementată această parte)
* Are posibilități de Wi-Fi și Bluetooth
* Folosește ca sistem de operare Tizen 3.3

### **II.2.1 Tizen OS**

Este un sistem de operare open source bazat pe Linux și folosit de Samsung pentru partea de IoT. Este sustinut atat de Intel, cât și de Fundația Linux și Samsung. La momentul scrierii acestui document, ultima versiune oficială lansată de Samsung este 4.0. Caracteristicile cele mai relevante pentru acest context sunt:

* Cei de la samsung oferă un IDE pentru dezvoltarea aplicațiilor – Tizen Studio
* IDE-ul menționat este o iterație peste Eclipse, un IDE open-source. Iterația de la Samsung vine cu o serie de aplicații ajutătoare. Cele pe care le-am folosit sunt:
  + Emulator Manager – simulează un dispozitiv la alegerea utilizatorului. L-am folosit pentru a vedea cum se modifică senzorii unui device și pentru a dezvolta aplicația de colectare de date pentru ceas (în loc să folosesc ceasul fizic pentru a o testa, am folosit un emulator)
  + Certificate Manager – facilitează crearea de certificate samsung. Aceste certificate sunt necesare pentru a putea instala aplicații pe un dispozitiv Tizen
  + Device Manager – acesta actionează ca un intermediar între IDE și dispozitiv (fie el fizic sau virtual) și permite accesul la sistemul de fisiere
* Sistemul de operare oferă posibilitatea dezvoltării de aplicații atât în C (limbajul nativ), cât și în tehnologii web (HTML, CSS, JS). Ca observație, este de preferat sa fie folosit C pentru că oferă acces la mai multe facilități ale sistemului de operare și este mai rapid. Motivele pentru care am ales sa fac o aplicație web in loc de una nativă sunt:
  + Am avut nevoie sa folosesc websocket-uri. Nu știam cum să fac acest lucru în C, așa că am ales JavaScript pentru faptul că eram mult mai familiar cu el și știam să fac atat request-uri http, cât și să folosesc web socket-uri
  + A fost mai usor să fac interfata grafică – simpla și suficientă pentru acest proiect

**Accessory SDK**

Un API care permite dezvoltarea de aplicații pentru dispozitive samsung cu Tizen OS. Oferă acces la funcționalități precum Debug, instalarea de aplicații și acces la sistemul de fișiere.

**Comunicare**

Pentru stabilirea conexiunii între ceas și aplicația Java am folosit urmatoarele protocoale:

* HTTP
* WebSocket

Ca bibliotecă am folosit Stomp.js și messaging din Spring. Nu am folosit bluetooth pentru că ar fi prelungit perioada necesară dezvoltării. De asemenea, acest lucru ar fi însemnat că aplicația cu care comunică ar trebui să fie pe un dispozitiv cu acces la bluetooth. Într-adevăr, este mai robust decât semnalul Wi-Fi-ului (observație făcută în timpul folosirii ceasului prin spațiul de desfășurare a experimentelor). Formatul ales pentru mesaje este JSON, iar ca bibliotecă adiționale voi menționa doar jQuery. În rest am folosit javascript pur pentru a scrie aplicația pentru ceas și aplicația de vizualizare a datelor. Pentru server am folosit Spring Boot, care oferă endpoint-uri REST și Web Socket-uri.

Doresc să amintesc aici faptul că această forma de comunicare (mai ales prin Web Sockets) este temporară. După ce aplicația de detectare a gesturilor vă fi mutată pe ceas, singurul rost pentru o comunicare cu un server vă fi să execute comenzi în momentul detectării unui gest. Ea a fost necesară pentru adunarea de date și pentru experimentarea cu datele înregistrate de senzori.

#### **Aplicațiile Web pentru un sistem Tizen**

Tizen oferă posibilitatea dezvoltatorilor de aplicații să utilizeze tehnologii de Web (HTML, CSS, JavaScript) pentru a facilita dezvoltarea aplicațiilor pentru cei care nu sunt obișnuiți cu folosirea limbajului C.

Avantajul este dat de ușurință cu care se poate crea o interfață pentru aplicație folosind tehnologiile Web. De asemnea, numărul dezvoltatorilor de aplicații Web este mare (conform statisticilor obținute de un soldaj - https://insights.stackoverflow.com/survey/2016#most-popular-technologies-per-occupation) și, prin urmare, Samsung a făcut sistemul său foarte accesibil.

Dezavantajul este reprezentat de faptul că aplicația vă fi mai lentă. API-ul de Web este un nivel în plus peste nivelul nativ și, prin urmare, nu vă fi la fel de eficient ca o aplicație nativă. Un alt dezavantaj îl reprezintă accesul la sistemul de operare. Aplicațiile native au mai mult acces la functionalitatile sistemului de operare și permit dezvoltarea unor aplicații puțin mai complexe. De menționat este faptul că diferența nu este atât de mare.

Am optat pentru a folosi aceast API pentru a putea crea o interfață grafică cât mai repede și să fie cât mai ușor de modificat. Aplicațiile grafice în C sunt mai voluminoase că număr de linii de cod și prin urmare ar fi întârziat procesul de dezvoltare. Și atât timp cât nu a fost nevoie să accesez mai mult de trei senzori în prima faza, diferența de eficienta între o aplicație web și una nativa este neglijabilă.

Un alt motiv a fost necesitatea utilizării web socket-urilor, lucru pe care nu știam să îl fac în C. Am început cu o aplicație nativă, însă ar fi durat prea mult să fac ceea ce am realizat cu API-ul Web, așa că am optat să renunț la o aplicație nativă până când nu trec algoritmul de transformare și recunoaștere pe ceas.

În consecință, în prima fază aplicația a fost scrisă folosind API-ul Web, urmând că ulterior atât aplicația de colectare de date, cât și cea de recunoaștere de gesturi să fie scrise în C sau GoLang (limbaj construit pe C, de la Google, eficient în procesarea pe mai multe fire de execuție folosind doar un singur fir și rutinele sale). **Capitolul III: Arhitectura sistemului**

Sistemul are momentan 2 module:

* Aplicația pentru smartwatch de colectare a datelor
* Aplicația pentru înregistrare, prelucrare și analiză a datelor

Sistemul a fost gândit în prima fază să colecteze informațiile necesare vizualizării și prelucrării datelor trimise de către senzori. Astfel s-a create următorul sistem de comunicare:

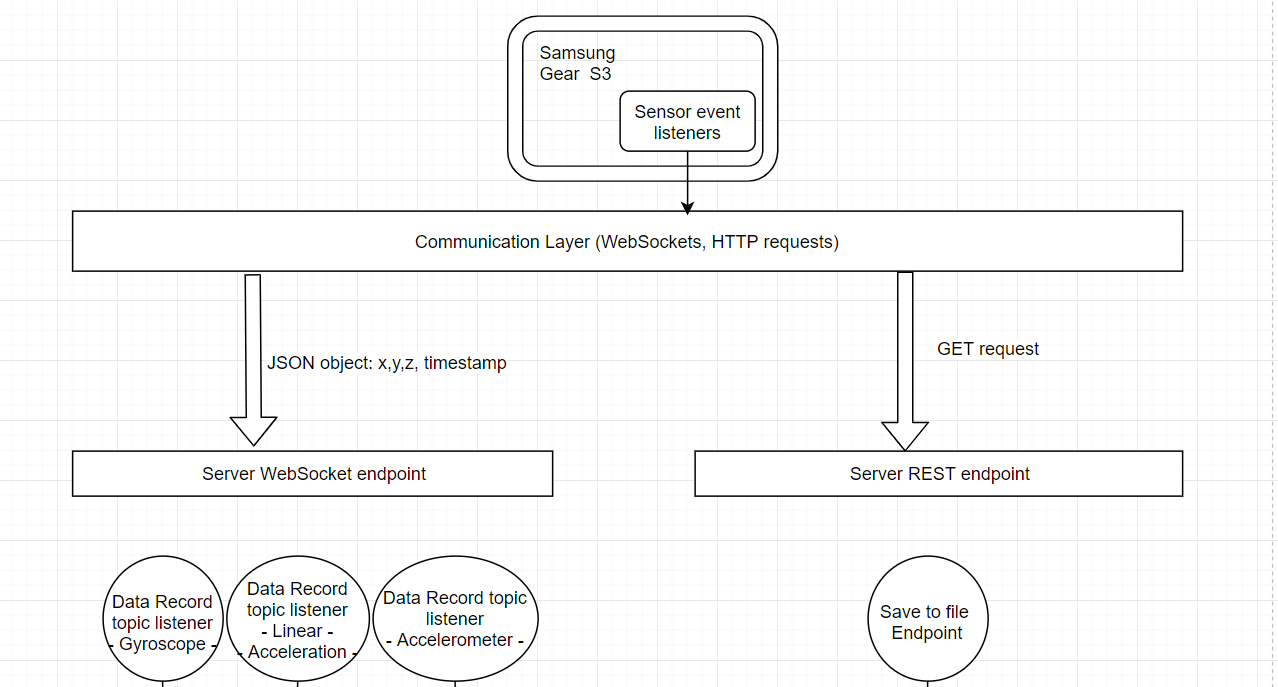


Figura 1 - Segmentul de comunicare între ceas și server

La fiecare înregistrare a unui senzor datele sunt împachetate într-un string json, separate prin virgulă și trimise către server. Endpoint-ul HTTP este pentru a compartimenta datele adunate în gesturi.

Datele înregistrate sunt stocate în memorie în prima fază și în același timp trimise mai departe către front-end

.

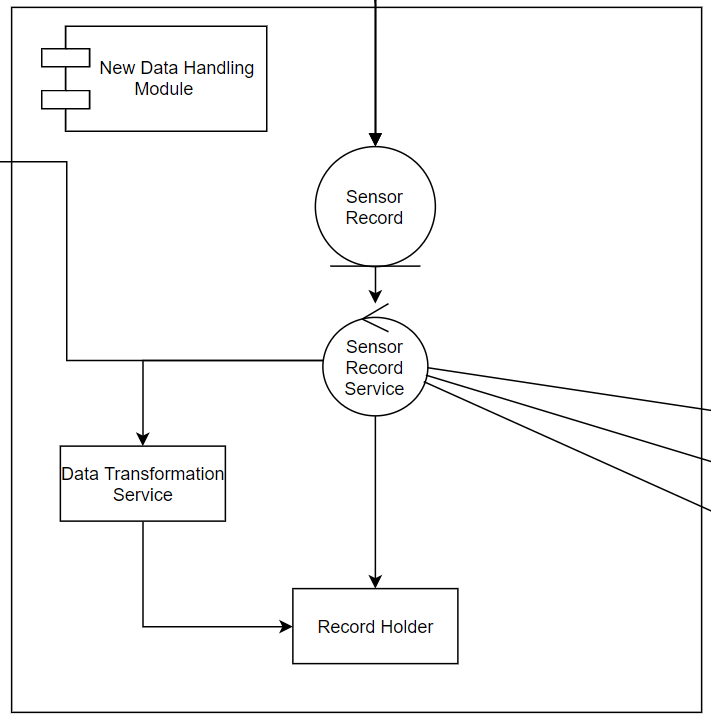


Figura 2 - Segmentul de gestionare a datelor venite în timp real

Figura 2 reprezintă modulul care se ocupă cu tratarea datelor noi. Aplicația are 2 fluxuri pentru aceste date.

Primul flux reprezintă vizualizarea și salvarea datelor. Pe măsură ce datele sunt primite, acestea sunt transmise mai departe către partea de front-end a aplicației pentru a fi afișate. De asemenea, sunt stocate în memorie până când se primește cererea de salvare, moment în care se crează fișierele în care este stocată mulțimea de date. Fiecare senzor are propriul său folder.

Al doilea flux transformă datele primite (pentru evaluarea lor) și determină dacă un gest este prezent. De asemenea, aceste date pot fi trimise mai departe către front-end.

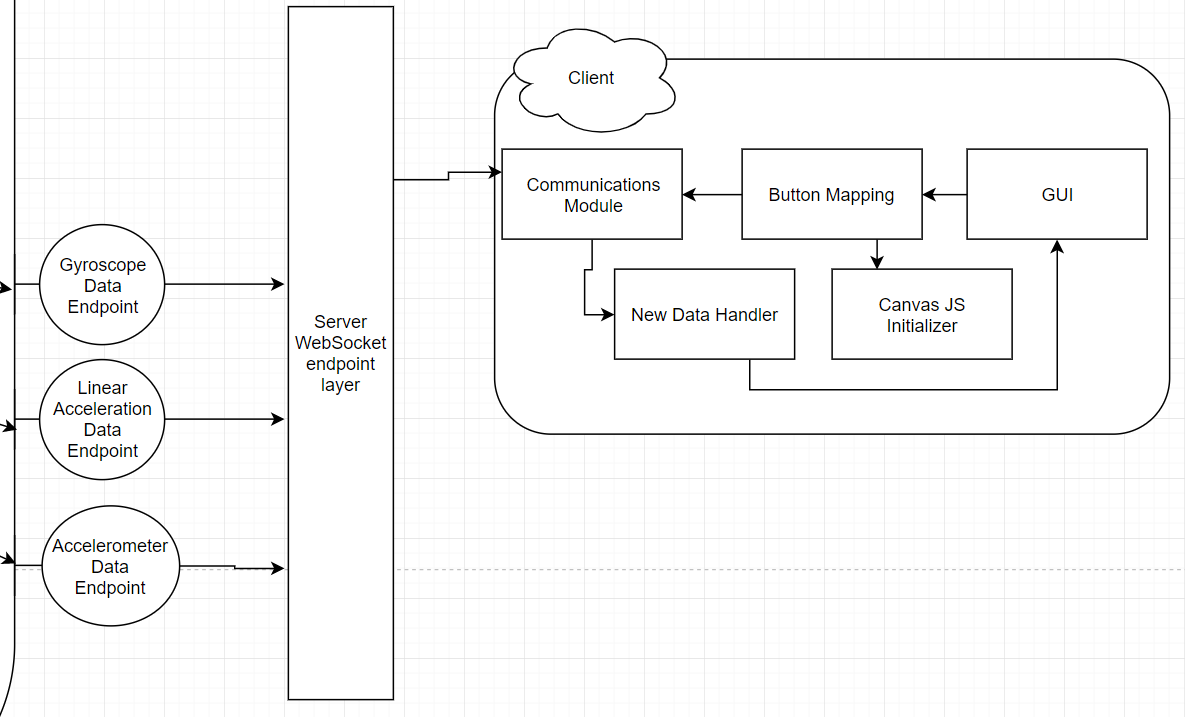


Figura 3 - Segmentul de comunicare cu front-end-ul

|  |  |
| --- | --- |
| Figura 4 - Segmentul de salvare a datelor noi | După ce se primește cererea de salvare în fișier, server-ul transferă datele din memorie în 3 fișiere separate după sezor. Salvarea se face în folder-ul Raw. |

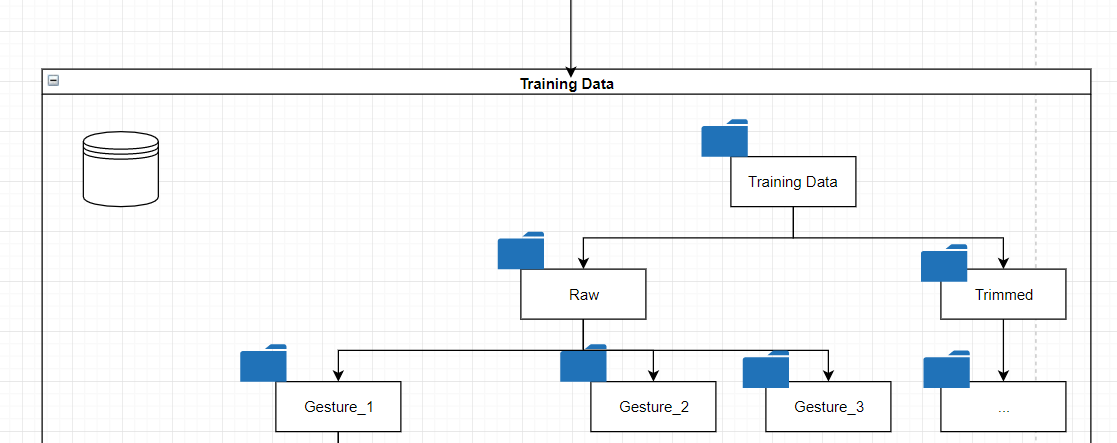


Figura 5 - Structura de fișiere pentru mulțimile de date

Folder-ul Raw reprezintă datele neprelucrate și salvate, reprezentând gesturi care pot conține date în plus și care ar putea încurca antrenarea clasificatorului.

După ce datele au fost colectate, s-a implementat logica pentru corectarea lor:

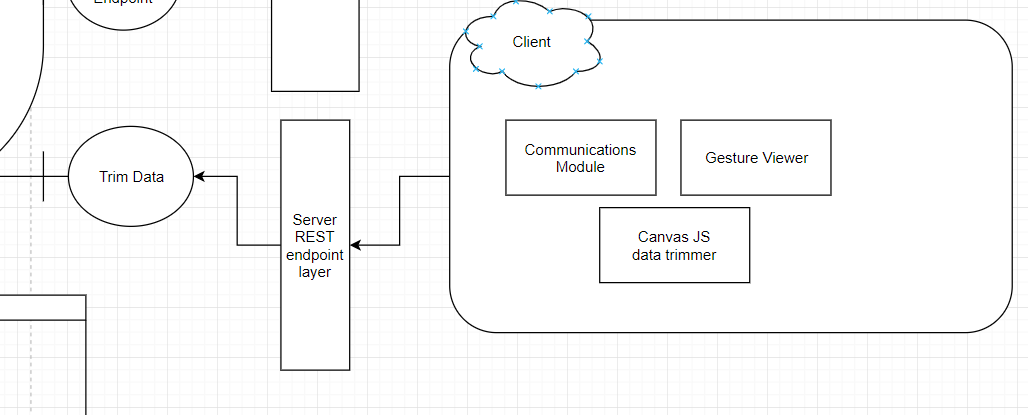


Figura 6 - Structura de corectare a datelor

Corectarea se facea în front-end, după care se trimitea intervalul de date corecte

serverului. Acesta ajustează fișierele inițiale și salvează o copie noua, corectată, în folder-ul Trimmed.

S-au folosit fișiere și nu o bază de date pentru că simplifica infrastructura și era mai ușor să transfer aplicația pe diferite stații de lucru. Baza de date vă fi folosită în viitor.

Într-un final au fost implementate modulele de antrenare și analiză a datelor în timp real. Gesturile sunt antrenate folosind fișierele din folder-ul Trimmed și pe baza datelor se generaza câte o matrice de recunoaștere pentru fiecare. Datele noi care vin de la sensozi sunt trecute prin modulul de analiza, iar rezultatul este comparat cu matricele de recunoaștere.

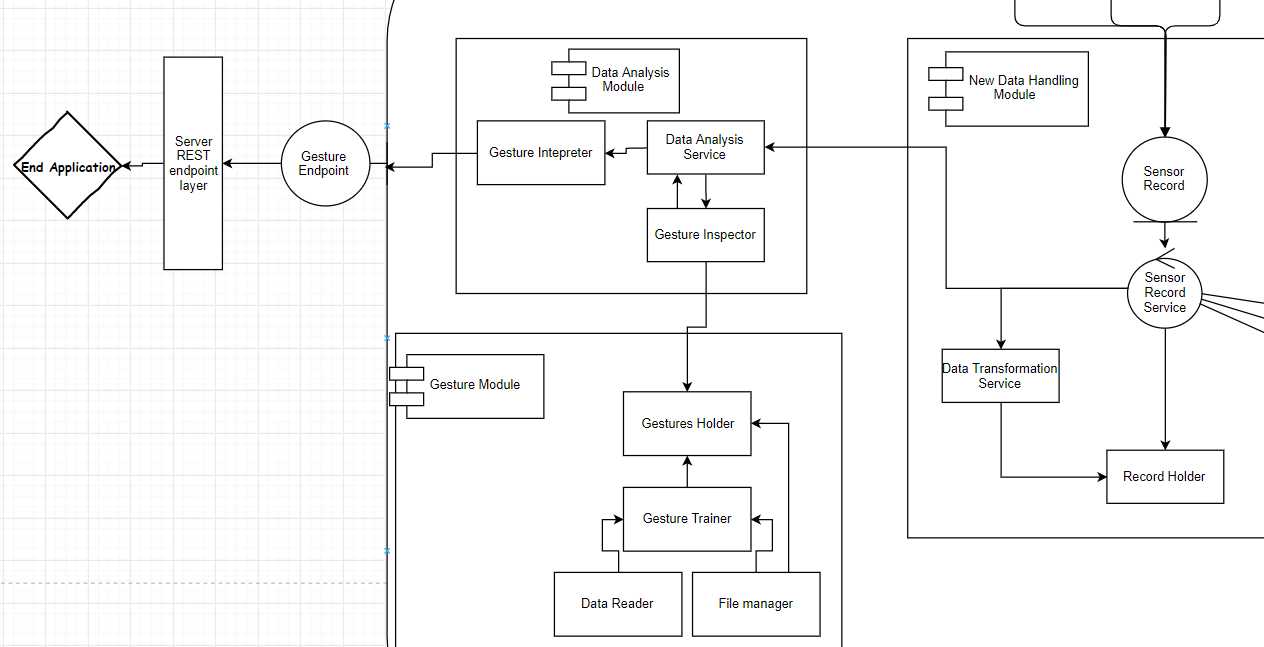


Figura 7 - Segmentul care interpretează datele și decide dacă sunt gesturi și viitorul endpoint folosit pentru comunicarea cu alte aplicații

Acest segment conține partea de antrenare și partea de interpretare a noilor date. Aceasta parte poate fi împărțită astfel:

* Partea de antrenare
* Partea de transfomare a datelor
* Partea de interpretare a datelor
* Partea de comunicare cu alte aplicații auxiliare

Antrenarea are loc la pornirea serverului și crează câte o matrice de semne pentru fiecare gest pentru care s-au colectat date. Aceste matrici sunt folosite pentru a putea identifica gesturile în timp ce sunt transmise date.

Partea de transformare a datelor ia înregistrările senzorilor și calculează distanța lor față

de pozițiile neutre ale senzorilor. În cazul în care depășesc o anumită distanță, acestea sunt transformate în 1 sau -1, iar în caz contrar 0. Acest lucru permite vizualizarea evoluției graficului rezultat în urma unui număr aleator de valori (ascendența – “+” sau descendența – “-”).

Partea de interpretare analizează mulțimile de date care au întrerupt șirul de valori 0. Extrag, precum la antrenament, evoluția graficelor fiecărei axe ale unui senzor și crează matricea de identificare. Aceasta este comparată cu matricile stocate în memorie, urmând a se determina daca un gest a fost identificat sau nu.

Partea de comunicare cu alte aplicații auxiliare nu este relevanta momentan, dar odată ce gesturile vor fi recunsocute într-un mod consistent, fiecărui gest i se va atribui o comandă. Acea comandă se va transmite aplicației auxiliare, care va reacționa în funcție de destinația ei (ex. schimbarea diapozitivului unei prezentări ppt).

**Capitolul IV: Implementare**

## **IV.1. Pașii pentru implementare**

În realizarea acestei lucrări mi-am propus următoarea serie de pași pentru a ajunge la rezultatul dorit:

* Găsirea ceasului potrivit
* Crearea unei aplicații pentru un smartwatch cu Tizen OS
* Crearea unei conexiuni între server și ceas
* Transmiterea datelor de către ceas în timp real
* Afișarea datelor în timp real
* Stocarea datelor înregistrate
* Afișarea datelor stocate
* Prelucrarea datelor stocate
* Stabilirea criteriilor pentru determinarea unui gest
* Determinarea algoritmului
* Antrenarea clasificatorului
* Testarea clasificatorului folosind date reale
* Legarea gesturilor de anumite acțiuni

Momentan singurul pas care nu a fost îndeplinit este ultimul, iar antrenarea și testarea mai necesită lucru.

### **Crearea unei aplicații pentru un smartwatch cu Tizen OS**

Am început prin a crea o aplicație pentru ceas în care am pus două butoane care nu făceau nimic la momentul respectiv. Doar voiam să văd că am putut instala o aplicație făcută de mine pe acel ceas.

**Probleme**:

* Au fost necesare 2 zile pentru a genera în mod corespunzator certificatul pentru că nu realizasem că certificatul nu este universal, ci este atașat unei anumite versiuni de sistem de operare

### **Crearea unei conexiuni între server și ceas**

Am avut de ales între a folosi bluetooth-ul și websocket-uri. Cu toate că o conexiune bluetooth ar fi mult mai stabila, am ales websocket-uri din următoarele considerente:

* O conexiune bluetooth ar fi însemnat să fac și o aplicație de mobil, arie în care încă nu am experiență și ar fi prelungit timpul de dezvoltare
* O conexiune bluetooth ar fi însemnat să fac o aplicație nativă pentru ceas, ceea ce ar fi prelungit timpul de dezvoltare atât din cauza faptului că aș fi folosit C, cât și a faptului că nu am experiență cu folosirea bluetooth-ului

Astfel am creat o aplicație care transmite date cu o frecvență configurabilă.



Figura 8 - Aplicația de colectare de date de pe ceas reprezentată pe un emulator

Aceasta este interfața aplicației reprezentată pe un emulator de ceas.

Butoanele:

* Start -> inițiază conexiunea cu serverul, începe să citească senzorii și să transmită datele către server
* Stop -> încheie citirea senzorilor și transmiterea datelor
* Save -> transmite serverului printr-un request HTTP să salveze datele acumulate

Accelerometrul l-am descoperit mai tărziu, întrucât modul de accesare al acestuia diferă de ceilalți 2 senzori.

Accelerometru:



Giroscop și acclerație liniara:

*Codul de mai sus se află în aplicația pentru ceas, iar modul de acces la senzori l-am luat de pe siteul oficial.*

Am decis că pentru a putea crea o secvență de informații care să mă ajute să identific un gest aveam nevoie de:

* Timpul la care au fost înregistrate datele de la senzor
* Axele x,y și z ale fiecărui senzor

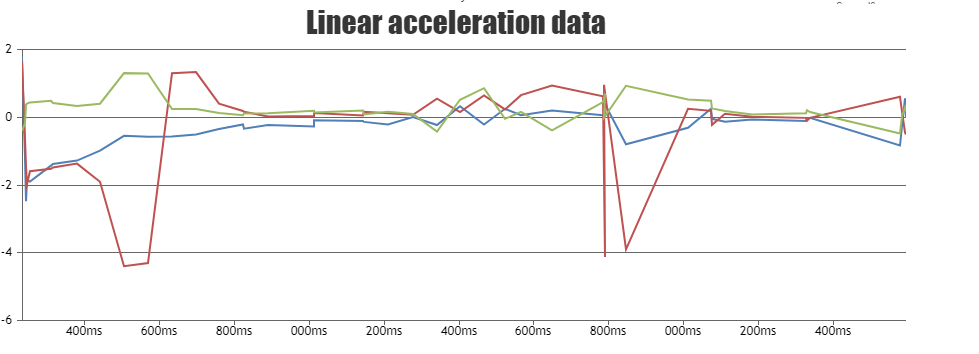


Figura 9 - Exemplu de comportament al senzorilor

### **Observații**

* Conexiunea Wi-Fi este foarte susceptibilă întreruperilor. Spun foarte pentru că în timp ce testam, aveam un scaun cu cadru metalic între mine și router. De fiecare data când coboram mâna, iar o parte a scaunului era în calea ceasului, conexiunea se pierdea.
* În funcție de configurația locului unde se folosește această aplicație, ceasul poate avea probleme în a se conecta și a rămâne conectat. Acest lucru a fost observat în momentul în care am încercat să lucrez într-o cameră care avea probleme cu conexiunea Wi-Fi în anumite puncte
* Accelerometrul oferă mai multe înregistrări decât giroscopul și accelearia liniară
* Giroscopul și accelarția liniară furnizează același număr de înregistrări

### **Concluzii**

* Am implementat o aplicație pentru ceas care transmite date în timp real prin Wi-Fi folosind API-ul de web al Tizen OS
* Am putut aduna datele de antrenament pentru aplicație

### **Crearea unei aplicații pentru vizualizarea, prelucrarea și analiza datelor**

Am avut nevoie de un server care să primească date de la ceas și să le afișeze pentru a putea găsi o metodă să le interpretez. Astfel am folosit Spring Boot pentru a crea o serie de endpoint-uri la care să se contecteze ceasul și să transmită datele.

Următorul pas a fost să creez o interfață care să îmi permită să vizualizez datele:

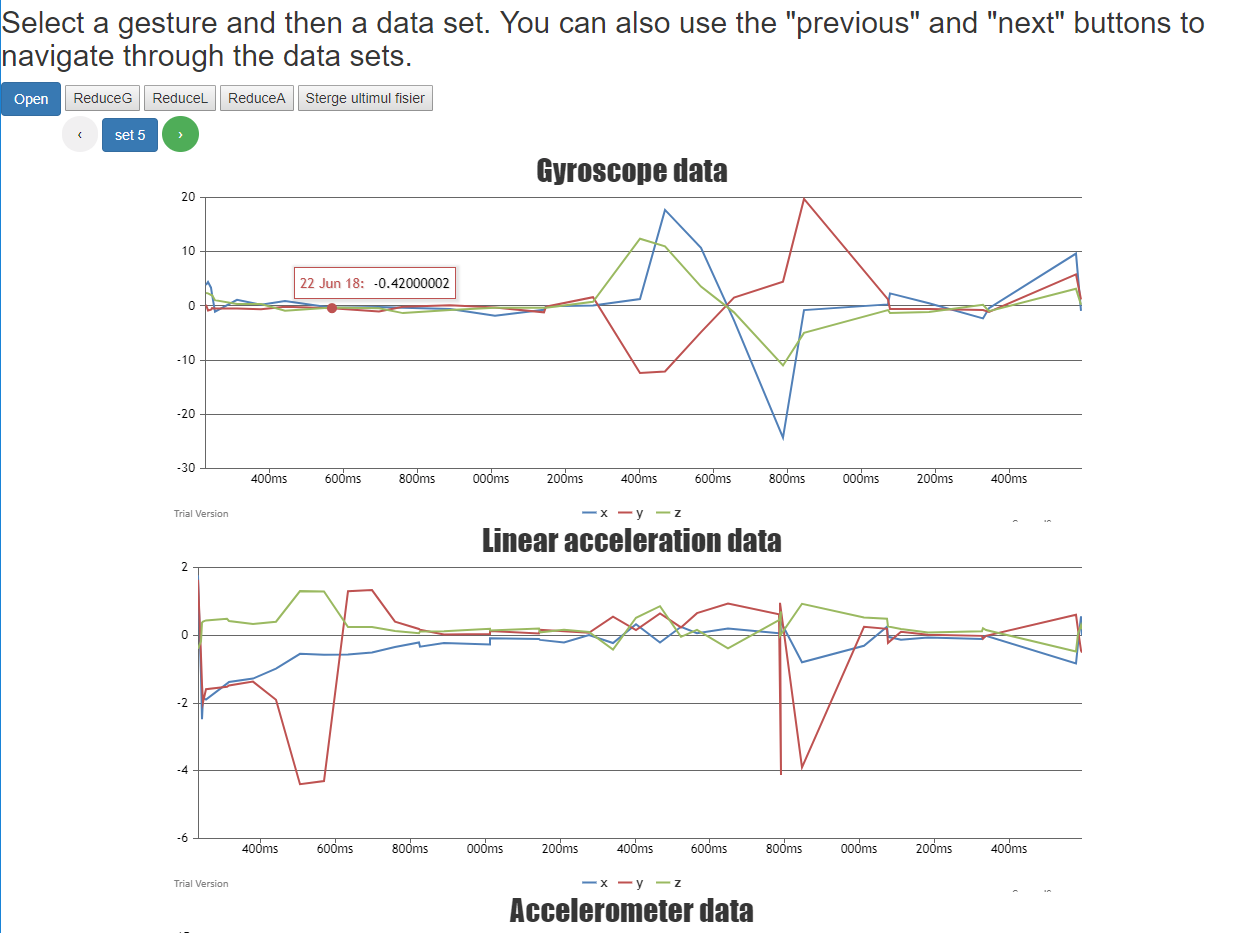


Figura 10 - Front-end-ul pentru prelucrarea datelor

Pentru asta am folosit:

* Stomp pentru comunicare
* Bootstrap pentru partea estetica
* Canvas.js pentru partea de reprezentare a datelor.

Au fost mai multe biblioteci candidat pentru reprezentarea datelor, însa CanvasJS permite selectarea unui anumit interval de date, aspect ce m-a ajutat foarte mult în partea de procesare a datelor.

### **Salvarea datelor**

Am decis să salvez datele în fișiere în loc de baza de date din următoarele motive:

* Nu era un avantaj clar în a folosi o baza de date la momentul respectiv
* Aplicația putea fi pusă pe altă stație de lucru fară să fie necesară instalarea sistemului de baze de date

Datele trimise de ceas sunt salvate în folder-ul “Raw”, semnificând date neprelucrate, care pot conține date necorespunzătoare la captele mulțimii de date.

Folder-ul “Trimmed” reprezintă mulțimile de date ale caror date eronate au fost eliminate.

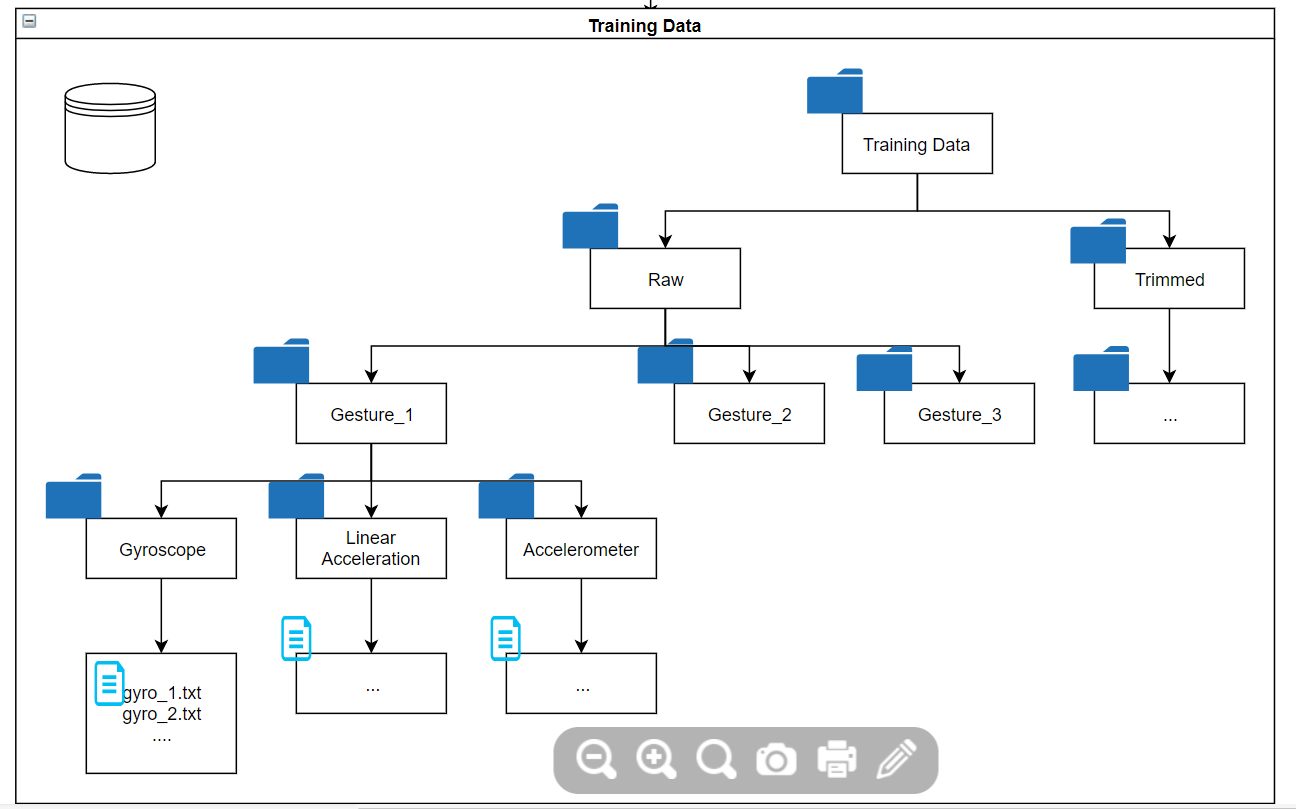
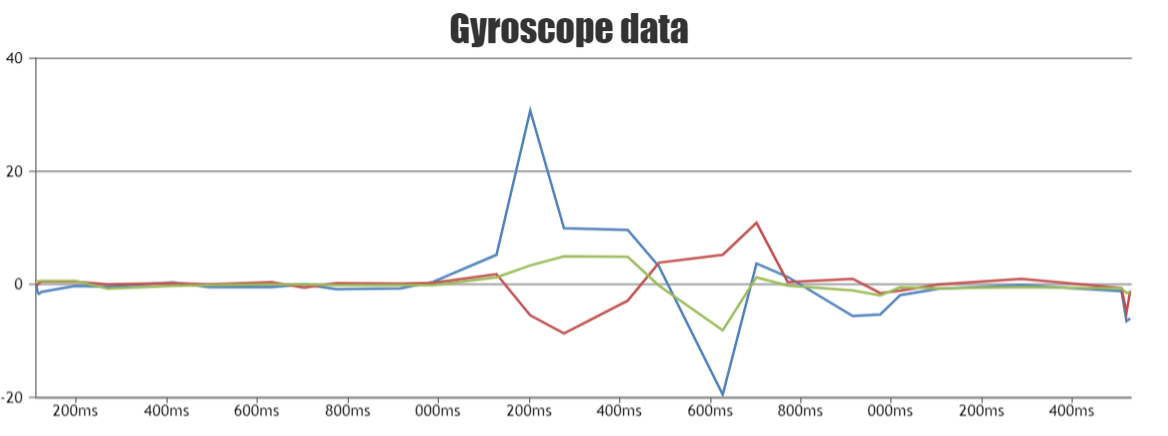


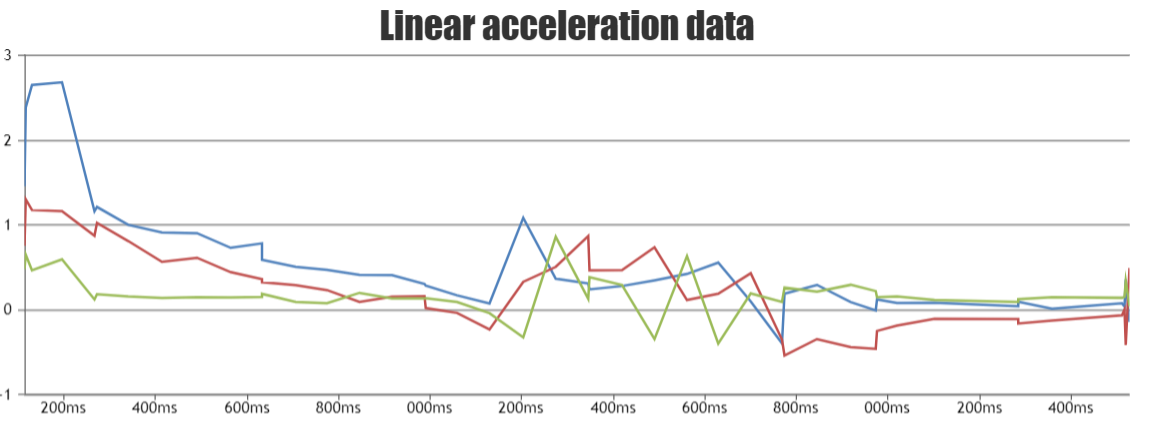
Figura 11 - Structura de foldere și fișiere pentru mulțimile de date de antrenament

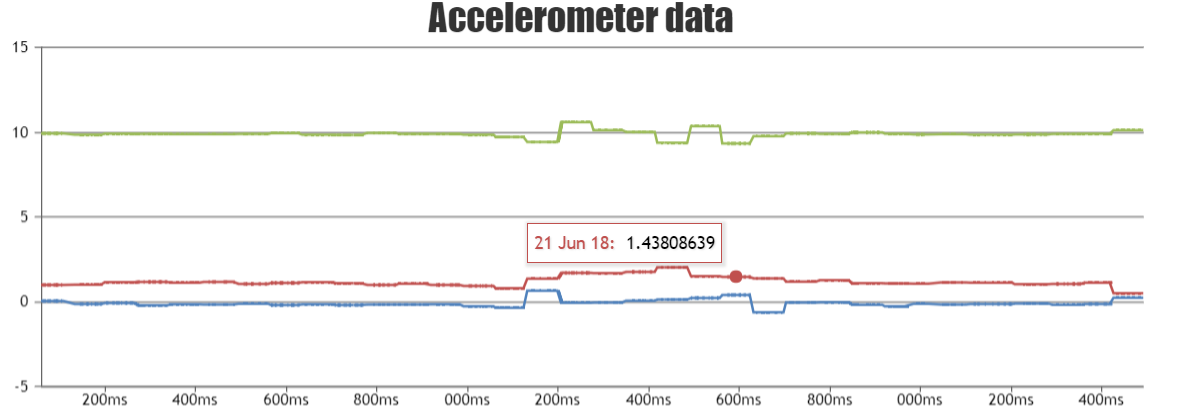
### **Vizualizarea și prelucrarea datelor salvate**

Următorul pas a fost să înregistrez o cantitate rezonabilă de date (pentru faza incipientă a procesului de analiza). Am decis că 100 repetări ale fiecărui gest sunt sufieciente. Gesturile înregistrate sunt: grab, open, point, rub fingers, slap, snap fingers.

***Exemplu – gestul open (deschidere completă a palmei)***







După faza de înregistrare a urmat prelucrarea datelor. Nu am putut găsi o metodă de prelucrare automată la momentul respectiv, așa că fișierele au fost prelucrate manual, unul cate unul, folosind o altă pagină a aplicației – cea de prelucrare a datelor.

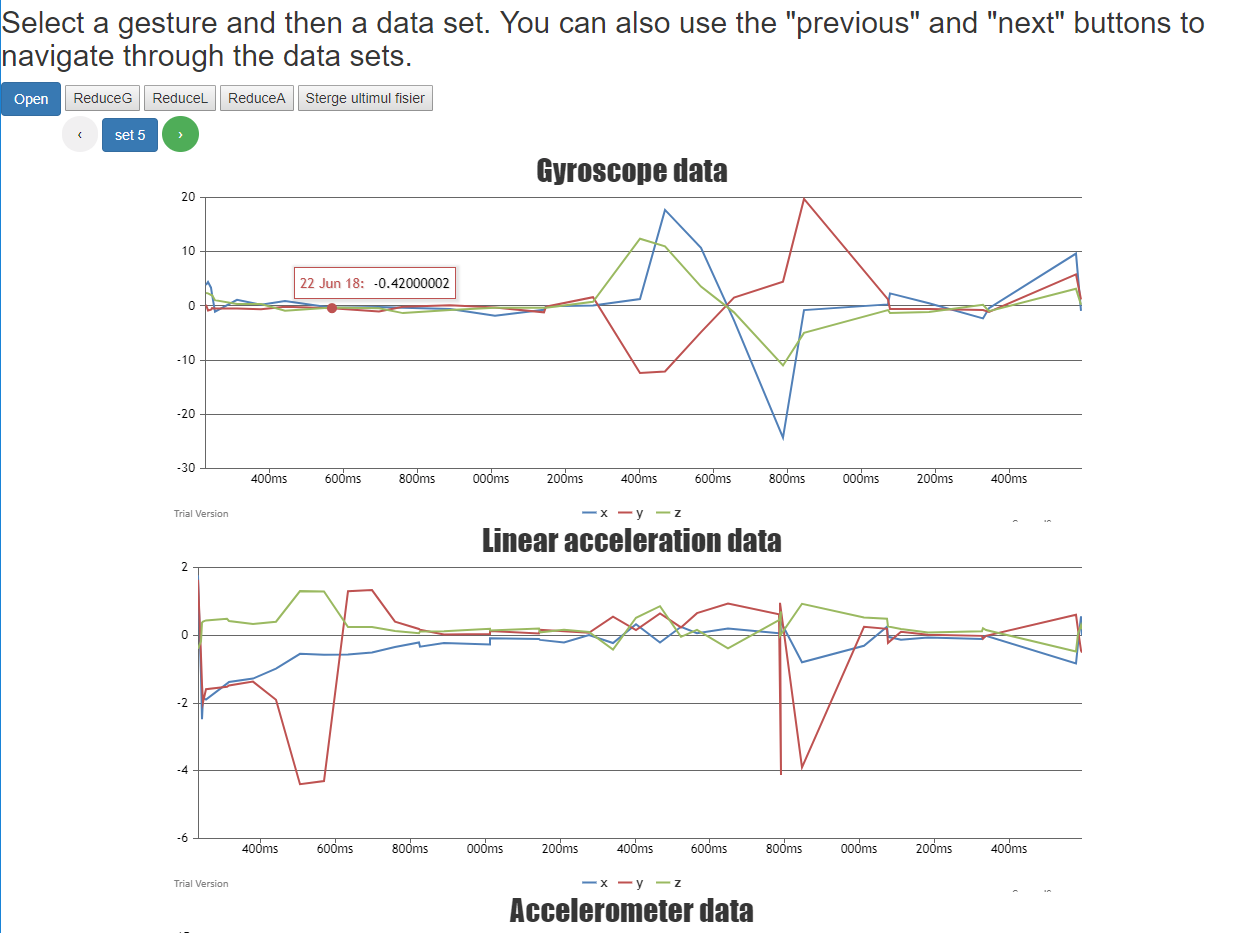


Figura 12 - Interfașa pentru prelucrarea datelor

Aplicația permite selectarea gestului și a fișierelor de prelucrat. Se poate selecta folosind mouse-ul intervalul de date care reprezintă un gest.

### **Criterii de recunoaștere și algoritmul**

Observând datele, am stabilit faptul că un gest poate fi recunoscut în funcție de oscilația graficului fiecărei axe ale senzorilor. Astfel am obținut 9 indicatori după care mă puteam ghida.

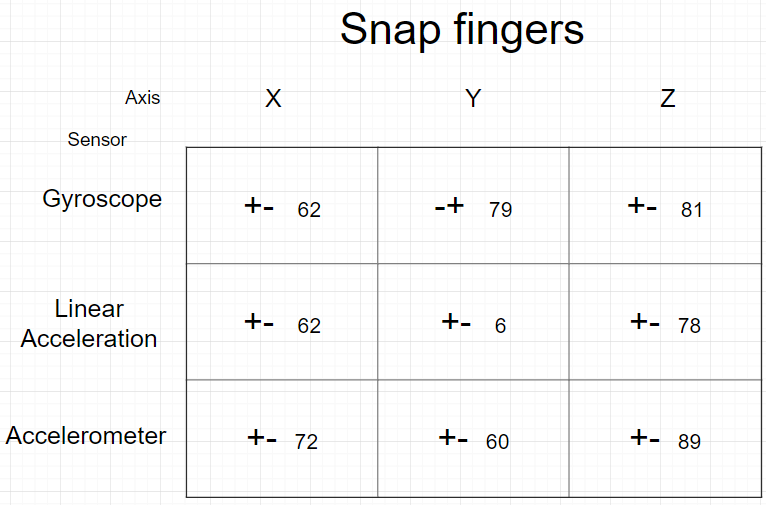


Figura 13 - Matrice de recunoaștere pentru pocnirea degetelor

**Problema**

Datele au fost înregistrate dintr-o anumite poziție, în care mâna stătea pe loc, apoi executa gestul, apoi iar stătea nemișcată. Chiar și în cazul în care mâna era nemișcată, senzorii tot înregistrau “zgomot”. Am avut nevoie de o metodă de a selecta datele care erau înregistrări semnificative.

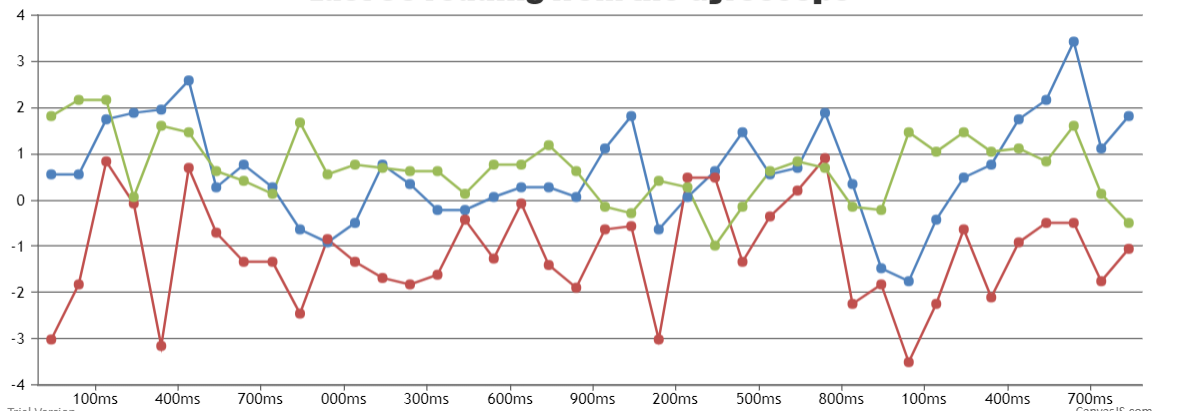


Figura 14 - Înregistrarile giroscopului când mâna stă pe loc

**Soluție**

Am pus un prag pentru a determina detele semnificative.

**Problema**

Amplitutdinea pentru un gest poate să fie diferită de a altui gest, dar în același sens. Acest lucru poate duce la confuzie în ceea ce privește recunoașterea gesturilor.

**Soluție**

Folosirea mai multor praguri și integrarea lor în algoritmul de recunoaștere.

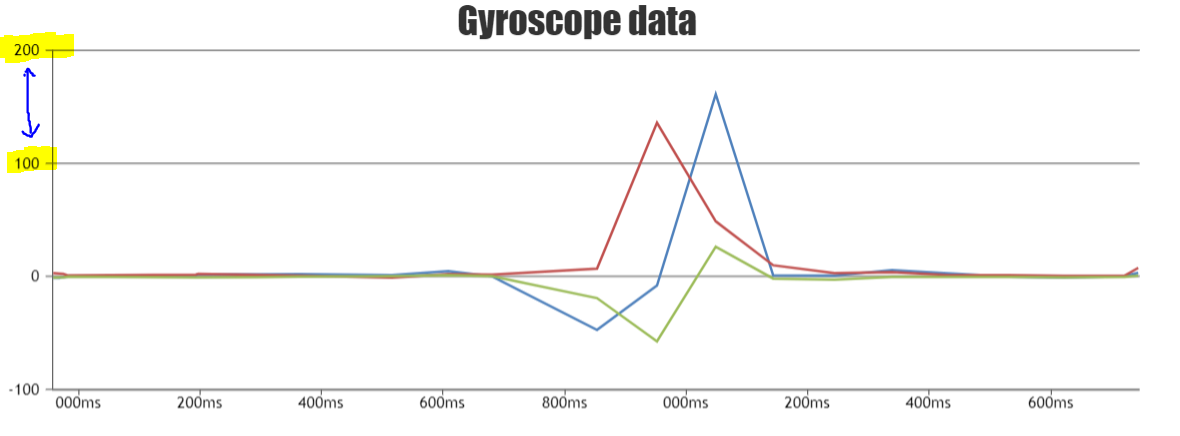


Figura 15 - Înregistrare a giroscopului și valori la care poate ajunge

După ce s-au adaugat pragurile, algoritmul vede datele în felul urmator:

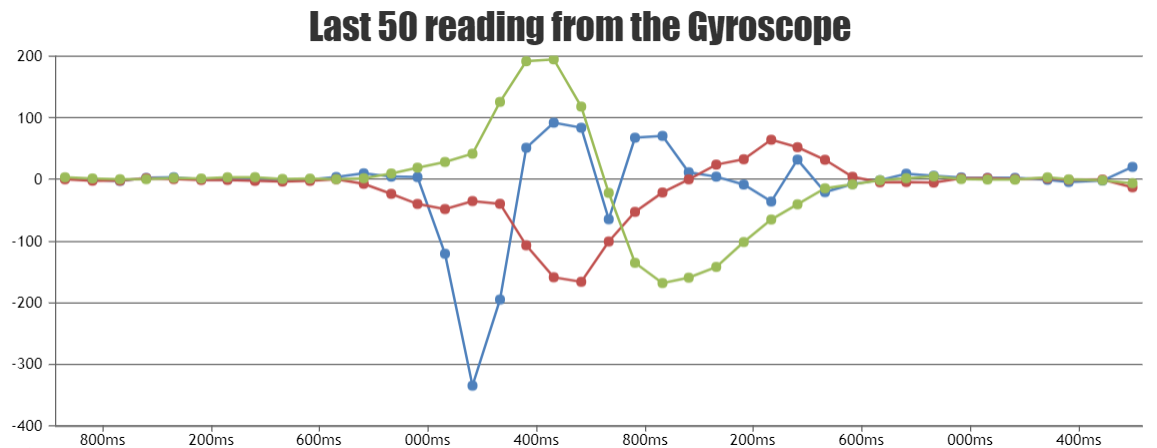


Figura 16 - Date primite de la senzor

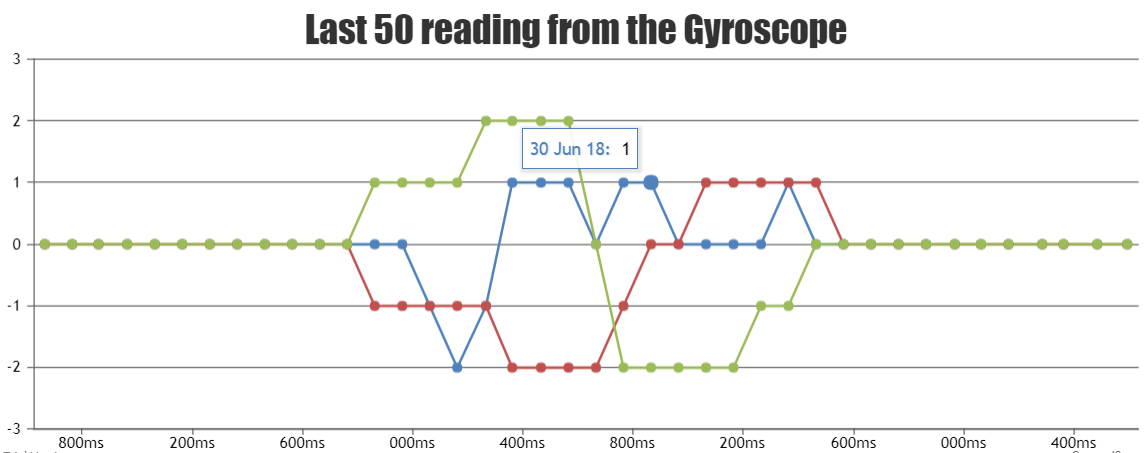


Figura 17 - Date vazute de algoritmul de recunoastere

### **Algoritmul de antrenare**

Algoritmul ia fiecare mulțime de date și crează o matrice de semne de forma “+-“, unde “+” reprezintă o modificare notabilă a senzorului pozitivă, iar “-“ o modificare negativă.

Matricea finală conține cele mai des întâlnite succesiuni de amplitudini alături de procentul de influență, care reprezintă de cate ori a fost întâlnită această succesiune din cele 100 de fișiere. Dacă acest procent este mare (>87%), putem spune cu o oarecare certitudine că acea succesiune pentru axa senzorului este caracteristică acelui semn.

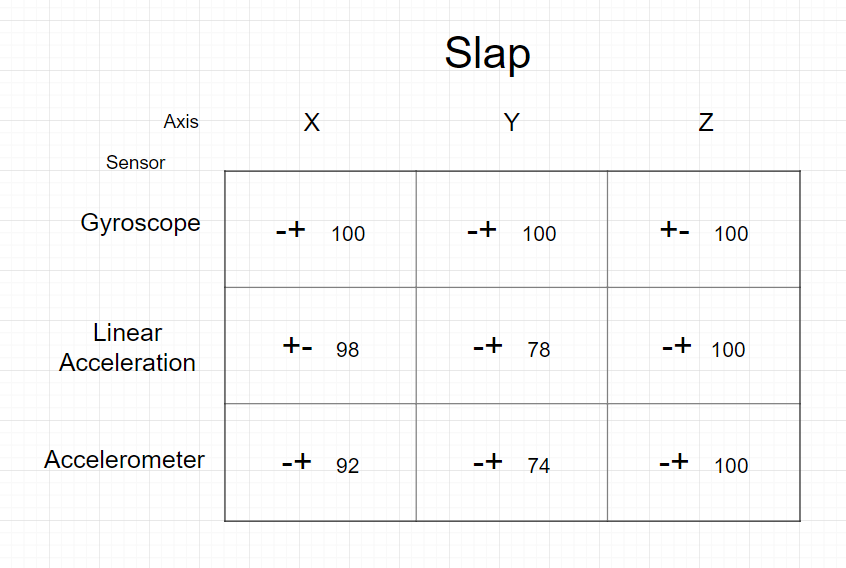


Figura 18 - Matricea de recunoaștere a gestului “lovire cu dosul palmei în aer”

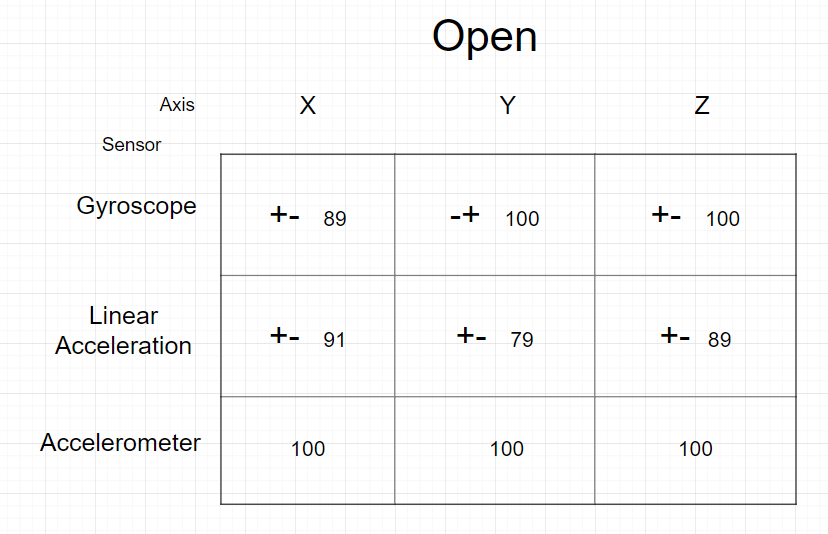


Figura 19 - Matricea de recunoaștere a gestului “deschidere completa a mâinii”

Aceste două gesturi se pot diferenția numai din giroscop, spre exemplu, întrucat primul are “-+”, iar cel de-al doilea “+-“.

### **Recunoașterea în timp real (aproape)**

Datele transmise de ceas sunt transformate și analizate. Un gest începe dacă nu se mai înregistrează 0-uri (momentan algoritmul cere că mâna să înceapă un gest dintr-o poziție statică). După ce se detectează faptul că șirul de 0-uri este întrerupt, datele sunt salvate în memorie până când se detectează o nouă secvență de 0-uri. După ce se încheie înregistrarea, mulțime de date este supus tehnicilor din algoritmul de antrenare și se formează matricea de recunoaștere. După acest pas se compară această matrice cu toate matricele existente în sistem și se determină daca există vreo potrivire. Sunt luate în considerare semnele care au acel procent peste 87%, restul nefiind certe și astfel nu ne putem baza pe ele.

## 

## **Concluzii**

* Algoritmul folosește momentan doar giroscopul (din motive de test în ceea ce privește funcționarea lui)
* Se pot înregistra și prelucra date folosind aplicația
* Stocarea datelor se face prin fișiere la momentul actual
* Rata de identificări greșite poate să fie cauzată și de pierderea de date în momentul în care are loc transmisia – datele de antrenament au fost culese într-un mediu “perfect”, lângă router. De asemenea, daca mișcarea este foarte rapidă și violentă, datele pot fi transmise aleatoriu (fară a respecta ordinea lor cronologică). Astfel, ar fi mai avantajos că recunoașterea gesturilor să fie trecută de pe server pe ceas
* Găsirea documentației adecvate legate de crearea aplicațiilor pentru un sistem Tizen, mai ales legat de accesul la senzori nu este facilă
* În momentul actual, fară să fi testat cu toți cei 3 senzori și cu o calibrare posibil greșită, gesturile fine (din degete) sunt greu de înregistrat. Cel pe care doresc să îl identific în mod cert este ridicarea degetului arătător. Problema este că pentru a avea o înregistrare, mișcarea trebuie să fie foarte energică, lucru care nu este natural și obositor
* Dintre toți senzorii, accelerometrul ar putea fi cel mai viabil pentru gesturile fine, care nu necesită mișcarea mâinii. Giroscopul și accelerația liniară ajută la gesturile mai ample
* O posibilă modificare a frecvenței cu care se înregistrează datele va fi necesară

## **Păreri externe**

Pe parcursul dezvoltării aplicației, am prezentat această idee în diverse contexte pentru a afla opinia unor potențiali utilizatori. Până acum am primit feedback pozitiv. În următoarele rânduri voi descrie recepția ideii în fiecare context. Menționez înainte de asta că nu am susținut că vin cu ceva inovator, în sensul că nu s-a mai făcut așa ceva. Tuturor celor ce le-am prezentat ideea li s-a menționat faptul că astfel de inițiative deja există.

#### **Vorbitor în public profesionist**

Foarte încântat de ideea de a-și elibera mâna și de a adaugă elementul de teatralitate prezentărilor sale.

#### **Vorbitor în public la conferințe tehnice**

Mi-a oferit un feedback realist despre cu s-ar comporta un prezentator în cazul în care aplicația nu detectează gesturile în mod corespunzător și, folosind informațiile pe care i le-am furnizat, a spus că mănușile existante pentru controlarea diverselor prezentări 2d și 3d sunt incomode și nu sunt ușor de acceptat, iar soluția cu ceasul este una foarte viabilă. De asemenea, dacă ceasul nu este foarte receptiv la gesturi foarte subtile, atunci aplicabilitatea aplicație se poate găsi și în alte contexte, unde gesturile pot fi mai ample, chiar și în cadrul prezentărilor.

#### **Companie de training-uri tehnice**

A fost o prezentare scurtă. Au fost foarte încântați de idee și au spus că ar putea beneficia de pe urma unei asfel de soluții.

#### **InnovationLabs 2019**

Am participat la ediția din primăvara anului 2019 a InnovationLabs, accelerator de start-up-uri. În cadrul acestui eveniment am prezentat ideea. Recepția a fost pozitivă și am mers până la etapa finală pentru Iași. Nu am continuat din motive presonale.

#### **Proiect european – BonAntreprenor**

M-am înscris în acest proiect cu finanțare europeană folosind această idee. A fost acceptat și am primit finanțarea.

## **Ce urmează**

### Calibrarea programului

Pentru o recunoaștere mai consistenta și mai precisă a gesturilor, modificarea pragurilor și introducerea lor pentru diferențierea gesturilor cu grafice similiare (secvențe ascendente și descendente identice), dar amplitudini diferite este necesară.

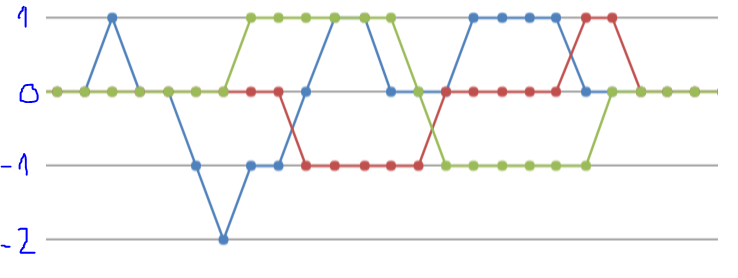
* 

Figura 20 - exemplu amplitudini diferite

### Testarea cu toți cei 3 senzori activi

Momentan s-a folosit doar giroscopul pentru recunoașterea gesturilor. Experimentele au arătat că este util în a identifica gesturi care implică o mișcare mai mare (atât în spatiu, cât și a parților mâinii implicate). Având în vedere că cei de la ViBand au folosit accelerometrul pentru a identifica majoritatea gesturilor (probabil au folosit și alti senzori, dar nu pot confirma acest lucru), un studiu mai amănunțit al acestui senzor este de dorit. De asemenea, adunarea datelor a fost scrisa în JavaScript. Este posibil că limbajul C să ofere mai mult acces la senzori și la calibrarea lor.

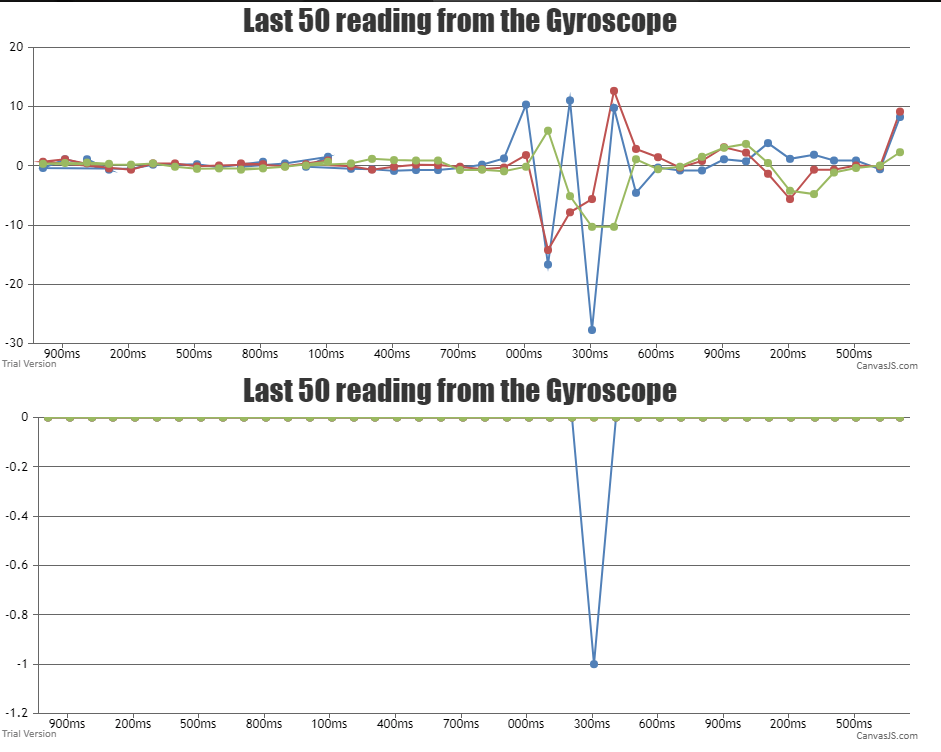


Figura 21 - Înregistrarea giroscopului pentru ridicarea degetului aratator

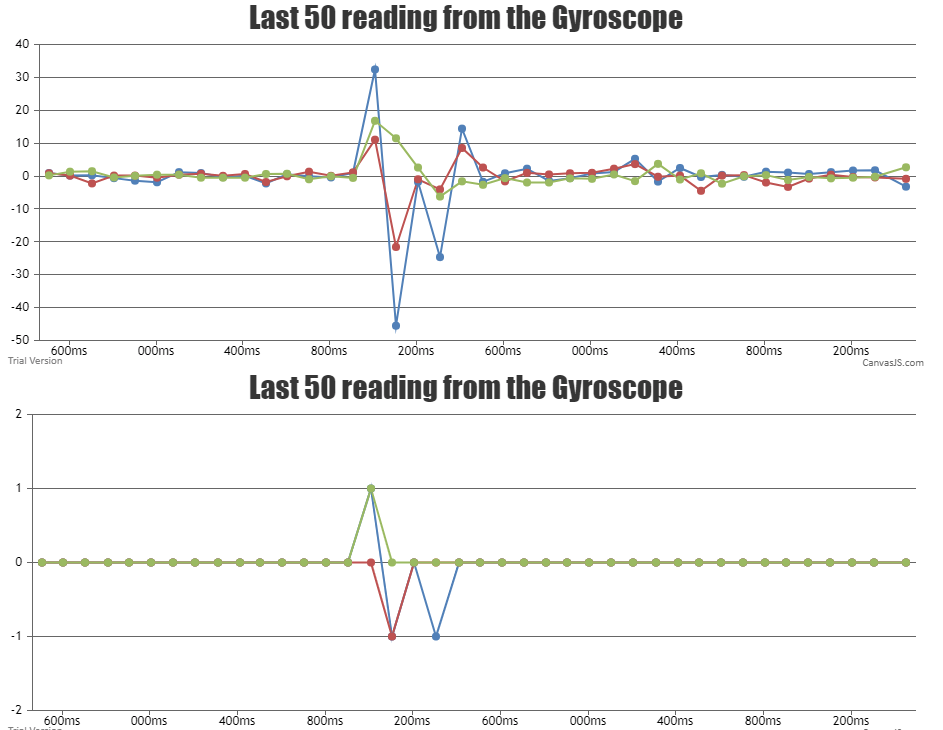


Figura 22 - Înregistrarea giroscopului pentru deschiderea palmei

### Determinarea celui mai relevant senzor în recunoașterea de gesturi în timp ce o persoană execută activități normale de zi cu zi

Senzorii înregistrează schimbări atunci când o persoană își mișcă mâna. Însă unii sunt mult mai sensibili la anumite mișcări decât alții. Astfel, este probabil că un senzor să fie mult mai influent atunci când vine vorba de detectarea unui gest.

### Evitarea detectărilor false

Detectarea gesturilor în momentul în care utilizatorul nu execută gestul respectiv sau pur și simplu își mișcă brațul este o problemă la moementul actual. Pentru a evita acest lucru, propun două soluții.

În primul rând, dacă ceasul nu este îndreptat spre țintă, atunci gestul trebuie ignorat.

În al doilea rând, într-un final, toate mișcările mâinii sunt gesturi. S-ar putea construit un model care să recunoască gesturile simple (generale, precum ridicatul mâinii, mișcatul stangă-dreapta, etc.) și gesturile mai complexe (deschis palma, arătat cu degetul, ținut în mana, etc). Cei de la ViBand au reușit ceva similar, însă ei au modificat kernel-ul ceasului. Este interesant de investigat dacă există vreo metodă de recunoaștere fară modificarea ceasului.

### Crearea unui mod de procesare a datelor automat și folosirea mai multor canditați pentru adunarea de date și testarea sistemului

O limitare a sistemului curent poate fi cantitatea de date disponibilă (100 de colecții de antrenament pentru fiecare gest). De asemenea, mulțimile au fost înregistrate de către o singură persoană (autorul acestei lucrări) și prin urmare nu putem ști cum se vă comporta în mâna altei persoane. Atât cei de la Serendipity, cât și cei de la ViBand au folosit un număr de persoane variate pentru testele lor. Procesarea volumului actual de date a consumat mult timp și este posibil să fi fost afectată de o pierdere de capacitate de concentrare. Un sistem automat de procesare a noilor mulțimi de date ar îmbunătăți radical timpul necesar pregătirii datelor.

O posibilă soluție ar fi chiar sistemul de transformare curent al datelor. Având în vedere că datele sunt stocate în memorie doar dacă se întrerupe șirul de înregistrări nule, atunci s-ar putea sări pasul în care se înregistrează date neprelucrate. Totuși, datele neprelucrate sunt utile în a vizualiza mai bine diferențele și, posibil, pentru antrenare, într-o anumită măsură.

### Imbunatatirea vizuală a aplicației

Momentan aplicația are un aspect foarte simplu, menit să fie practic. În timpul dezvoltării am încercat să utilizez VueJS și Bootstrap pentru a crea o consolă de administrare care să facitliteze controlul aplicației. Acest lucru dura prea mult, așa că din constrângeri de timp am amânat această parte a proiectului. Cu toate acestea, consider că merită investit timp în crearea acestei interfețe pentru a putea avea un control mai mare asupra aplicației. Cu timpul, vor fi și alte date care fi afișate pe pagini. De asemenea, în timpul dezvoltării am încercat să pun aplicația în Azure pentru a putea avea acces la ea de oriunde cu ceasul. Din nou, din constrângeri de timp am preferat să continui cu forma actuală.

Astfel, având în vedere cele menționate mai sus, consider că ar ajuta că front-end-ul să fie scris în VueJS și pus pe un server accesibil de pe internet.

### Transferarea algoritmului pe ceas folosind ori C, ori GoLang

În stadiul curent, aplicația de pe ceas doar trimite date către server. Datorită consumului de baterie, conectivitatii limitate și susceptibile întreruperilor și incertidunii prezentei unei rețele Wi-Fi adecvate, consider că înregistrarea datelor și detectarea gesturilor ar fi avantajată de mutarea acestui segment pe ceas. Pentru a implementa aceste funcționalități, propun două variantede limbaj: C și GoLang.

C vine cu avantajul că este limbajul nativ al sistemului de operare Tizen și are acces la mai multe facilitați.

GoLang, un limbaj de la Google, este foarte eficient în ceea ce privește tratarea mai multor fire de excutie numite rutine. Folosind aceste rutine, GoLang simulează mai multe fire de execuție folosind unul singur – lucru care ar putea fi de folos pe un dispozitiv cu abilitați computationale limitate. De asemenea, din câte am reușit să citesc, GoLang poate apela funcții C, lucru care ar putea face ca o aplicație scrisă în GoLang să se poată comporta că o aplicație nativă.

### Găsirea unei formule pentru determinarea orientării ceasului

Așa cum am menționat pe parcursul lucrării, orientarea ceasului ar putea ajuta la evitarea erorilor și la controlul mai multor dispozitive cu același gest, fară a exista confuzie (ex. pocnirea degetelor pentru a schimba diapozitivele și pentru a închide becul).

Având în vedere lipsa magnetometrului (responsabil pentru determinarea punctelor cardinale), soluția pe care o propun este că în momentul în care utilizatorul dorește să înceapă să folosească Mimic, acesta să se orienteze spre nord și să apese un buton care să pornească un calculator de orientare care să stabilească mișcarea ceasului și orientarea lui pe tot parcusul utilizării.

De asemenea, un buton pentru stabilirea locației obiectelor de interes ar putea fi util pentru a menține o “harta” a obiectelor din jur. Ceasul dispune de GPS și prin urmare această propunere este realizabilă. Una dintre limitări, în schimb, este faptul că a la momentul actual nu cunosc precizia GPS-ului. În schimb, pedometrul pare să fie suficient de capabil.

### Crearea unei aplicatii care sa “reactioneze” la gesturile inregistrate

Aplicația Mimic a fost concepută cu scopul de a oferi baza pentru augmentarea experienței utilizatorului în viață de zi cu zi. În momentul în care am început această lucrare am avut următoarele aplicabilități în minte:

* Controlarea unei prezentări Power Point sau Google Presentations
* Crearea unui joc folosind WebAR (bibliotecă pentru integrarea realității augmentate într-un browser web precum Google Chrome) în care în funcție de gest, utilizatorul invoca diverse vrăji și poate să se lupte cu alți utilizatori (precum duelurile din Harry Potter)
* Controlarea dispozitivelor SMART din casă

Pe parcurs, un urma interacțiunii cu persoane cărora le-am prezentat ideea, am stabilit următoarele aplicabilități:

* Controlul hologramelor, a prezentărilor și a elementelor de realitate augmentată folosind gesturi mai ample decât, spre exemplu, ridicarea arătătorului
* Aplicație care simulează o orchestră și cântă în funcție de dirijor

## Concluzie pentru functionalitățile viitoare

Rezultatele actuale sunt suficient de bune încât să fie viabilă o investiție suplimentară de timp în această aplicație datorită posibilităților de aplicabilitate care pot augmenta experiență unei persoane în iteractiunea cu mediul înconjurător.

De asemenea, este un subiect interesant de studiu și de la care se pot extrage numeroase concluzii și ipoteze. Chiar dacă forma finală nu va fi un ceas, stabilirea contextului în care un dispozitiv sub forma unei brățări poate să detecteze gestica mâinilor și să permită interacțiunea cu alte sisteme (precum un televizor) poate reprezenta un pas important în ceea ce privește dezvoltarea de viitoare dispozitive.

O altă idee este de a contacta o companie producătoare de astfel de dispozitive în vederea colaborării pentru a putea identifica contextul menționat mai sus.

Și, nu în ultimul rând, studiile actuale au fost și sunt desfășurate cu tehnologia și limitările sale de astăzi. În viitor există posibilitatea să apară componente hardware de dimensiuni foarte mici dedicate pentru funcționalități precum recunoașterea de gesturi. Un exemplu este unitatea specială de procesare de rețele neuronale din Huawei Mate 10 Pro.

# Bibliografie

### **Cărți:**

1. Brink Henrik, Richards Joseph, Fetherolf Mark (2017) - Real World Machine Learning, Editura Manning
2. Urma Raoul-Gabriel, Fusco Mario, Mycroft Alan (2015) – Java 8 in Action, Editura Manning
3. Raoul-Gabriel Urma, Mario Fusco, Alan Mycroft (2019) – Modern Java in Action, Editura Manning
4. Walls Craig (2019) – Spring in Action Fifth Edition, editura Manning

### **Siteuri:**

#### Surse principale de informații

1. <http://www.gierad.com/projects/viband/>
2. <https://developer.tizen.org/>
3. <https://www.andrew.cmu.edu/user/ingenia/papers/wen2016serendipity.pdf>
4. <https://developer.tizen.org/forums/web-application-development/accessing-sensor-data>

#### Alte surse utile

1. <https://pdfs.semanticscholar.org/450c/a19932fcef1ca6d0442cbf52fec38fb9d1e5.pdf>
2. <http://cs.nju.edu.cn/wujx/>
3. <https://machinelearningmastery.com/inspirational-applications-deep-learning/>
4. <https://papers.nips.cc/paper/1087-using-feedforward-neural-networks-to-monitor-alertness-from-changes-in-eeg-correlation-and-coherence.pdf>
5. <https://deeplearning4j.org/visualization>
6. <http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0188733>
7. <https://aibusiness.com/small-businesses-ai/>
8. <http://www.psych.utoronto.ca/users/reingold/courses/ai/cache/neural3.html>
9. <https://blog.slavv.com/37-reasons-why-your-neural-network-is-not-working-4020854bd607>
10. <https://www.semanticscholar.org/paper/Using-Feedforward-Neural-Networks-to-Monitor-from-Makeig-Jung/423031001ff973a4244b5aa89df2bfce2d612fd9>
11. <https://www.tandfonline.com/doi/full/10.1080/00051144.2017.1343328>
12. <https://www.tandfonline.com/doi/abs/10.1080/002075499191148>
13. <https://machinelearningmastery.com/improve-deep-learning-performance/>
14. <https://www.technologyreview.com/s/609710/neural-networks-are-learning-what-to-remember-and-what-to-forget/>
15. <https://www.sciencedirect.com/science/article/pii/S1474667017425286>
16. <https://pdfs.semanticscholar.org/b0be/7e7f9e46f4b411b7eb9ca3e99bd94b552f7b.pdf>
17. <https://towardsdatascience.com/how-to-build-a-real-time-hand-detector-using-neural-networks-ssd-on-tensorflow-d6bac0e4b2ce>
18. <https://scholar.google.ro/scholar?q=best+neural+network+for+monitoring+changes&hl=en&as_sdt=0&as_vis=1&oi=scholart>
19. <https://www.sciencedirect.com/science/article/pii/S2210832717301485>
20. <http://cs229.stanford.edu/proj2014/Ben%20Ulmer,%20Matt%20Fernandez,%20Predicting%20Soccer%20Results%20in%20the%20English%20Premier%20League.pdf>
21. <https://www.uruit.com/blog/2018/02/16/soccer-and-machine-learning-tutorial/>
22. <https://www.uruit.com/blog/2017/10/03/introduction-machine-learning-ebook/>
23. <https://doctorspin.me/digital-strategy/machine-learning/>
24. <https://www.wsj.com/video/startup-predicts-soccer-results-with-90-accuracy/AD51ECE6-2082-4E97-830D-22BC151A798C.html>
25. <https://dashee87.github.io/football/python/predicting-football-results-with-statistical-modelling/>
26. <https://github.com/dmitryvodop/football-results-prediction-ml>
27. <https://github.com/neaorin/PredictTheWorldCup>
28. <https://www.quora.com/What-is-the-simplest-method-to-predict-result-of-a-soccer-match>
29. <https://elitedatascience.com/learn-statistics-for-data-science>
30. <https://developer.tizen.org/ko/forums/general-support/configuring-gyroscope-sensor-sampling-ratefrequency?langswitch=ko>
31. <https://developer.tizen.org/ko/forums/web-application-development/capturing-devicemotion-events-when-screen-switched-off.?langswitch=ko>
32. <https://developer.tizen.org/dev-guide/2.4/org.tizen.tutorials/html/web/w3c/device/device_orientation_tutorial_w.htm>
33. <https://developer.tizen.org/ko/forums/general-support/accelerometer-data-devicemotion-event-handler-not-working-properly?langswitch=ko>
34. <https://www.lifewire.com/pros-and-cons-of-native-apps-and-mobile-web-apps-2373173>
35. <https://www.quora.com/What-is-the-advantage-of-using-Tizen-rather-than-Android-as-an-OS>
36. <http://www.trustedreviews.com/opinion/samsung-tizen-os-features-2915424>
37. <https://start.spring.io/>
38. <https://spring.io/projects/spring-boot>
39. <https://www.educba.com/java-ee-vs-spring/>
40. <https://www.tutorialspoint.com/spring_boot/spring_boot_introduction.htm>
41. <https://www.bmc.com/blogs/microservices-vs-soa-whats-difference/>
42. <https://www.springboottutorial.com/spring-boot-with-embedded-servers-tomcat-jetty>
43. <https://www5.epsondevice.com/en/information/technical_info/gyro/>
44. <https://developer.android.com/guide/topics/sensors/sensors_motion#sensors-motion-significant>
45. <https://gizmodo.com/all-the-sensors-in-your-smartphone-and-how-they-work-1797121002>
46. <https://www.samsung.com/ro/wearables/gear-s3-classic/>
47. <https://www.lifewire.com/what-is-tizen-4580383>
48. <https://www.tizen.org/about>
49. <https://www.tizen.org/>
50. <https://developer.samsung.com/galaxy/accessory>
51. <http://www.gierad.com/assets/viband/highres/bb_GestureArm.jpg>
52. <https://image-us.samsung.com/SamsungUS/home/mobile/wearables/pdp/sm-r770nzsaxar/gallery/sm770-feature1-120716.jpg?$product-details-jpg$>
53. <https://zdnet4.cbsistatic.com/hub/i/r/2017/01/12/a90de462-7dce-497a-b479-312489de634a/resize/1170x878/b1aa255bc4e5c5648c5754fb3e10e5e4/gear-s3-notification.jpg>
54. <https://developer.tizen.org/forums/web-application-development/accessing-sensor-data>
55. <https://www.tizen.org/feature/sensor.accelerometer>
56. <https://developer.samsung.com/z/develop/getting-certificates>
57. <https://bugs.tizen.org/browse/TW-59>
58. <https://www.statista.com/statistics/538237/global-smartwatch-unit-sales/>
59. <https://www.statista.com/chart/15035/worldwide-smartwatch-shipments/>
60. <https://www.digitaltrends.com/mobile/smartwatch-sales-rose-in-2018-npd-report/>
61. <https://9to5mac.com/2019/02/12/smartwatch-sales-in-2018-apple/>
62. <https://www.techradar.com/news/apple-watch-was-half-of-all-smartwatch-sales-in-2018-but-its-grip-is-loosening>

# ANEXE

1. Documentul de la Serendipity in format PDF
2. Documentul de ViBand in format PDF
3. Arhitectura generală a aplicației Mimic - https://drive.google.com/file/d/1bx0YfvvJPN7JsWm24B-4sp7JJk5MnRiQ/view?usp=sharing 