



UNIVERSITAT  
POLITÈCNICA  
DE VALÈNCIA



UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Higher Polytechnic School of Gandia

Implementation of an emotion detection algorithm for  
therapy applications in children.

End of Degree Project

Bachelor's Degree in Interactive Technologies

AUTHOR: García Andreu, Adrián

Tutor: Pérez Pascual, M<sup>a</sup> Asunción

ACADEMIC YEAR: 2021/2022

## Table of Contents

RESUMEN .....	5
ABSTRACT .....	6
1. INTRODUCTION .....	7
1.1 Research Objectives .....	8
1.1.1 Main objectives .....	8
1.1.2 Secondary objectives.....	8
1.2 Methodology used .....	8
2. STATE OF THE ART.....	9
2.1 Python .....	9
2.2 OpenCV.....	9
2.3 Raspberry Pi .....	9
2.4 Emotion classification .....	10
2.5 OpenCV DNN .....	11
2.6 DeepFace framework .....	12
2.7 CUDA .....	13
3. SYSTEM STRUCTURE.....	15
3.1 From the original idea to the final implemented system.....	15
3.2 Camera nodes.....	16
3.3 Server .....	17
3.4 Web application .....	19
4. PROJECT LIMITS.....	20
4.1 Body pose classification and tracking .....	20
4.2 Eye gazing .....	20
4.3 Light conditions .....	21
5. ALGORITHMS' DESIGN.....	22
5.2 Technologies used to design and develop .....	22
5.2.1 Draw.io .....	22
5.2.2 Gitlab .....	22
5.2.3 Jupyter Notebook.....	22
5.2.4 VisualCode.....	22
5.2.5 Apache.....	22
5.2.6 MariaDB.....	22
5.2.7 SFTP .....	22
5.2.8 SSH.....	22

Implementation of an emotion detection algorithm for therapy applications in children

5.3 Node design.....	23
5.4 Server design .....	24
5.4.1 Face detection and cropping.....	26
5.4.2 Person verification .....	28
5.4.3 Preprocessing .....	29
5.4.4 Emotion analysis.....	30
5.4.5 Display the data.....	31
5.5 Database design .....	32
5.6 Web design.....	33
6. IMPLEMENTATION .....	35
6.1 Node implementation .....	35
6.1.1 Software .....	35
6.2 Server Implementation .....	38
6.2.1 Software .....	38
6.2.2 Others.....	42
6.3 Database implementation.....	42
6.4 Web Implementation .....	43
7. SELENIUM TESTING .....	46
7.1 Access tests .....	46
7.2 Patients tests.....	46
8. CONCLUSION AND FUTURE WORK.....	48
8.1 Conclusion .....	48
8.2 Future work.....	48
Bibliography .....	49

## Figures list

Figure 1: Raspberry Pi model 4, price and some specifications.....	10
Figure 2: Classification performance of TF, OpenCV and PyTorch. Source: [5] .....	11
Figure 3: CUDA Toolkit web site.....	13
Figure 4: System topology.....	15
Figure 5:RGB camera Logitech C920 .....	16
Figure 6: Google cloud pricing.....	18
Figure 7: Azure pricing.....	18
Figure 8: AWS pricing .....	18
Figure 9: Natural illumination conditions 1.....	21
Figure 10: Natural illumination conditions 2.....	21
Figure 11: Node program flow and folder structure .....	23
Figure 12: Server emotion analysis block scheme and analysis program flow scheme.....	25
Figure 13: Server folder structure .....	26
Figure 14: HAAR face detection .....	26
Figure 15: Face detection through DNN .....	27
Figure 16: DNN detection even in difficult angles .....	27
Figure 17: Emotion analysis blocks 1 .....	31
Figure 18: Emotion analysis block 2 .....	31
Figure 19: Data display example .....	32
Figure 20: Database scheme .....	33
Figure 21: Web application pages flow diagram.....	33
Figure 22: Login page wireframe.....	34
Figure 23: Session details wireframe .....	34
Figure 24: Node program UML scheme .....	35
Figure 25: Node JSON config file .....	36
Figure 26: Node, server listener opening socket .....	36
Figure 27: SFTP Server record sending.....	37
Figure 28: Server program UML scheme.....	38
Figure 29: Sentinel, video processing starts.....	39
Figure 30: Record session SQL insertion .....	39
Figure 31: CLAHE .....	40
Figure 32: User verification through DeepFace FaceNet512 .....	41
Figure 33: phpmyadmin MariaDB DBMS management web application .....	43
Figure 34: API REST example.....	43
Figure 35: Web application folder structure .....	45
Figure 36: Access test results .....	46
Figure 37: Patient test results .....	47
Figure 38: Users test results.....	47
Figure 39: Cameras test results.....	47

## Tables list

Table 1: Accuracy comparison between models.....	12
Table 2: Accuracy tests.....	13
Table 3: Test computer specifications .....	19
Table 4: Patient verification at a considerable distance from the camera.....	28
Table 5: CLAHE comparison .....	29

## Acronyms

OpenCV: Open Computer Vision.....	10
HOG: Histogram of Oriented Gradients .....	11
KNN: K-Nearest Neighbor.....	11
NetACT: Networks & Advances in Computational Technologies.....	11
SVM: Support Vector Machines .....	11
DL: Deep Learning .....	12
CNN: Convolutional Neural Network .....	12
CUDA: Compute Unified Device Architecture.....	14
CPU: Central Processing Unit .....	15
GPU: Graphic Processing Unit .....	15
DNN: Deep Neural Network.....	27
MTCNN: Multi-Task Cascaded Convolutional Neural Network.....	27
CLAHE: Contrast Limited Adaptive Histogram Equalization.....	29

## RESUMEN

Este proyecto se centra en el desarrollo de una serie de programas con paradigma cliente/servidor, que tienen como objetivo inferir el estado emocional en niños que se encuentran en terapia. Esto se realizará mediante grabaciones obtenidas en las sesiones con el terapeuta.

Para ello se pretende utilizar, para la grabación de las sesiones, unas cámaras RGB conectadas a ordenadores y un servidor para centralizar el procesado y almacenamiento de la información.

Para poder visualizar la información de una forma que aporte valor a los especialistas, se desarrolla una aplicación web donde se muestran todos los datos obtenidos, y desde donde se puede controlar cuando empiezan, finalizan las grabaciones de las sesiones y otros parámetros de configuración.

En este documento se muestran las aplicaciones desarrolladas, referencias a aplicaciones similares que han sido desarrolladas previamente y conclusiones obtenidas a través de algunos experimentos realizados.

## ABSTRACT

Project based on software developments with client/server paradigm, that have the aim of predict the possible emotional status of children, which are under therapy for emotional or psychological disorders, trying to help them improving the quality of the therapies increasing the amount of information the specialists can obtain in each session.

To do the recording, cameras connected to computers are being used and a server to centralize the heavy processing batches and to store all the necessary data.

To be able to show the data to the specialists worthily a simple web application was developed, in which all the session results are displayed, and the specialists can control when the records start, finish or other parameters.

In this document the developed applications, other similar previously existing and some experiment results will be showed.

## 1. INTRODUCTION

ASD or autism spectrum disorder is a developmental disorder. A disorder which affects the method that the persons use to be able to communicate, the behaviors, and how they learn.

This disorder is always present in a person life, but as it is a developmental disorder, is used to be diagnosed in the firsts years of life, and to be diagnosed sooner is considered to have a significant improvement in the results that can be achieved under treatment [1].

ASD can show several difficulties in a person's life such as having troubles understanding or appreciating the others' emotions, making friends, tolerating changes in a routine, or interpreting abstract ideas in literally. And the ASD is not something minor in rates cause for example in 2018, USA 1 in 44 children of 8 years was diagnosed with ASD, this means around 2.3% of children population, considering the USA population that means a considerable amount of people [2] .

In these disorders as in others therapy can make the difference, not being able to cure this the best way to fight against it is to learn how to deal with it. Therapy helps children to acquire new skills and overcome developmental challenges, to play with similar or even to understand the others' emotions.

Cause of this therapy is so important, and to improve the quality of the therapy sessions is a significative point.

With such a problem the intention is to try to improve the therapy the specialists can give to the patients. Some specialists of the Safor area in Spain, Valencia proposed us to develop a tracking system with the next features.

- Patient recognition
- Body gestures classification
- Emotion detection
- Stereotypies iteratives pattern recognition
- Eye gazing

Between those topics this work will be centered on the emotions detection and the patient recognition, being all the listed topics complementary each other being able to improve the results of the developed system.



Implementation of an emotion detection algorithm for therapy applications in children

## 1.1 Research Objectives

### 1.1.1 Main objectives

The main objective of this research is to develop a system, which can record therapy sessions through one or multiple nodes, send the records to a server, process them and classify the emotions that the patients are feeling on each moment of the record.

Additionally, other main objective is to store the processed results and show them to the specialists in a way that is possible to them to use these data for next sessions as feedback.

### 1.1.2 Secondary objectives

Other interesting objectives are:

- 1) Improve the data processing algorithm in order to achieve efficient data manipulation in the emotion prediction algorithm. Cause the big amount of data generated affects the processing times, and this can be an important point in terms of time and money.
- 2) Achieve the best user experience as possible to improve the way the doctors use the system.
- 3) Do as secure as possible the records sending to the server, due to the patient data is a sensible information.

## 1.2 Methodology used

In this section will be introduced the steps used to develop the project.

- **State of the art:** In this step is where the state of the art and similar works that were found are exposed to be able to know possible useful techniques or tools that can be used for this research and to know what things are possible to achieve and what can't be achieved.
- **System structure:** Based on the previous point information and the requested features, in this point is decided which devices will be necessary and the way they will communicate.
- **Project limits:** With some research done and the system structure decided, the objectives that are able to achieve are decided, and its considered real working space to establish the design rules.
- **Algorithm's design:** Technologies and algorithms test are performed to determine which are the best option and depending on that results a final algorithm flow is decided for the nodes and the server.
- **Implementation:** Development of every single algorithm as programs in the appropriate platforms, for the nodes, server, and the web system.
- **Testing:** Performing tests to know the limits of the web application and how the data are displayed.

Implementation of an emotion detection algorithm for therapy applications in children

## 2. STATE OF THE ART

In this point the state of the art of some technologies used, algorithms, techniques and references to some other similar developed works related with this research will be exposed.

### 2.1 Python

Python is an object-oriented programming language and interpreted, this brings it the capacity to interoperate in most of the popular operative systems, is easy to develop in python compared to other languages and has lots a libraries AI and data manipulation oriented. Cause of that it was chosen as one reliable option to the development.

Other languages that have been considered were R or C. R is popular but does not have as many libraries as Python and it's just focus for statistics and C is no so fast to develop for it and does not have the advantage to be interpreted, and no so many libraries again.

For the project Python 3.5.10 is being used, being Python 3.5.5 or superior version required to be used by DeepFace a framework that will be introduced later.

### 2.2 OpenCV

OpenCV is without a doubt the most popular and extended library in terms of image processing, it simplifies lots of basic operations like change color spaces, filters, morphological operations and more.

The library is open source and as it is so used is always high end, updated and it comes with lots of others useful libraries and models pretrained which is an advantage and is developed to work with Python.

For that reasons OpenCV is being used for this project.

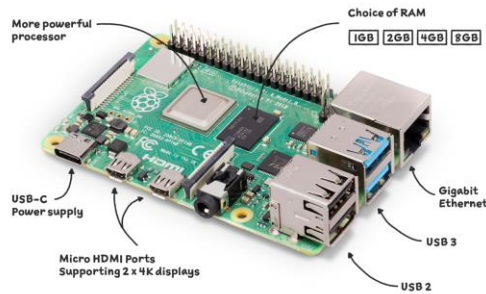
### 2.3 Raspberry Pi

In terms of hardware, for the camera nodes Raspberry Pi is the option that has been chosen, there are enough reasons such as the price, physical space, and the no need of excessive processing power just to record the therapy sessions.

Recently the raspberry Pi prices are increasing, due to the semiconductor's crisis but as much as almost any kind of hardware, even so the prices still being affordable for this purpose. The Figure 1 shows some specifications of the Raspberry Pi in its official website.

## Implementation of an emotion detection algorithm for therapy applications in children

Completely upgraded, re-engineered  
Faster, more powerful



From \$35

You'll recognise the price along with the basic shape and size, so you can simply drop your new Raspberry Pi into your old projects for an upgrade; and as always, we've kept all our software backwards-compatible, so what you create on a Raspberry Pi 4 will work on any older models you own too.

Figure 1: Raspberry Pi model 4, price and some specifications

The Raspberry Pi model 4 with 4 RAM GB will be used as it has enough power to process the recording and do other needed operations, with 2 RAM GB should be enough to but maybe there can be some lacks and for the difference in the prices the 4GB option is considered a better option. And besides that, the other things needed is a USB port to connect the camera and an Ethernet port to not depend just on Wi-Fi that is less stable in terms of lost data packages.

### 2.4 Emotion classification

The techniques and technologies behind the emotion classification is one extensive researched topic, as there are many possible approaches to a problem like this.

To infer the emotions in a record, the appearing emotions should be classified, which is a multiclass classification problem.

There are some techniques like KNN, Naïve Bayes, SVM or Decision trees which are used in ML to solve this kind of problems, as used in a classification system developed as described in the article published in 2017 in an international conference of NetACT [3]

Another interesting example is from an article in 2013, in which was developed a face recognition system recording videos with a RGB and a depth camera at the same time and extracting features from both images (using HOG) to be classified with a random decision forest model. Those are very feasible options to do a classification system, and computationally not so expensive.

## Implementation of an emotion detection algorithm for therapy applications in children

Besides the mentioned techniques there are DL techniques too. The usage of this techniques is growing up due to several things as the higher accuracy rates, scalability, and adaptability. But to develop and train a DL network needs a lot of data to be trained with, and computationally to achieve this on a reasonable time with a good enough accuracy is really expensive in comparison.

In the article “Deep Learning-Based Drivers Emotion Classification System in Time Series Data for Remote Applications”, is exposed how to address a similar classification problem with a deep learning approach, and to perform this a CNN was developed and trained to classify two emotions (aggressive or normal) [4].

Both types of models’ approaches are feasible to make a classification system and inspired the process pipeline on this research, even though for reasons that will be introduced later, this research will be based on the use of DL techniques.

### 2.5 OpenCV DNN

As mentioned before in the point 2.2 OpenCV is an important library in the field of computer vision, but it includes another technologies to mention, as the DNN module.

This module is used to load deep learning models onto OpenCV from different frameworks, and has features like image classification, object detection, or face detection. In the Figure 2 is shown a difference of more than 0.6 seconds in image classification between OpenCV DNN module and TensorFlow, what means a really good performance for classification in the ambit of DNN technologies [5].

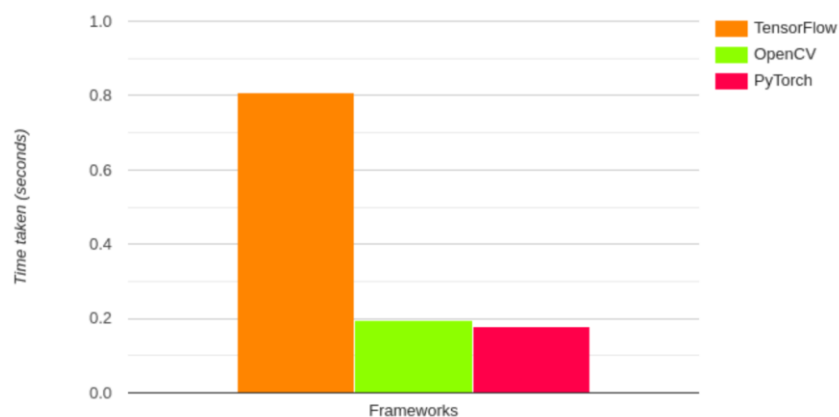


Figure 2: Classification performance of TF, OpenCV and PyTorch. Source: [5]

Implementation of an emotion detection algorithm for therapy applications in children

## 2.6 DeepFace framework

DeepFace is a framework that operates with multiple CNN robust prediction models oriented to detect people in images, face recognition and face analysis [6]. For face recognition, it includes the next state of the art models:

- VGG-Face
- Google FaceNet,
- OpenFace,
- Facebook DeepFace
- DeepID
- ARCFace
- Dlib
- SFace

Google FaceNet512 will be used for face recognition, as it is the model that achieves the highest number of accuracies.

This framework uses in the background technologies like TensorFlow and Keras to work, using the previously mentioned models as other some important detectors like OpenCV, ssd, dlib, mtcnn, retinaface or mediapipe.

With all the technologies implemented, used properly this framework is a really reliable and feasible tool to work with, without the need of create and train a CNN model.

<b>Model</b>	<b>LFW Score</b>
<b>FaceNet512</b>	99.65%
<b>SFace</b>	99.60%
<b>ArcFace</b>	99.41%
<b>Dlib</b>	99.38%
<b>FaceNet</b>	99.20%
<b>VGG-Face</b>	98.78%
<b>Human-Beings</b>	97.53%
<b>OpenFace</b>	93.80%

*Table 1: Accuracy comparison between models [2]*

## Implementation of an emotion detection algorithm for therapy applications in children

Images		
Results	<pre>{'emotion': {'angry': 1.4767108433968332, 'disgust': 5.387763800938491e-06, 'fear': 0.39860226727141546, 'happy': 0.5577469577042435, 'sad': 0.6609684250335947, 'surprise': 0.22442086074774628, 'neutral': 96.68154696712178}, 'dominant_emotion': 'neutral', 'region': {'x': 970, 'y': 249, 'w': 763, 'h': 763}, 'age': 34, 'gender': 'Man', 'race': {'asian': 8.429595688141944e-07, 'indian': 4.230049199804853e-07, 'black': 2.603832731026823e-10, 'white': 99.93758797645569, 'middle eastern': 0.046833124361000955, 'latino hispanic': 0.015578774036839604}, 'dominant_race': 'white'}</pre>	<pre>{'emotion': {'angry': 4.513013607265748e-06, 'disgust': 2.1909376712992394e-14, 'fear': 3.6384597591754275e-10, 'happy': 98.60653877258301, 'sad': 1.210694904330012e-05, 'surprise': 0.00037731458633061266, 'neutral': 1.3930724002420902}, 'dominant_emotion': 'happy', 'region': {'x': 325, 'y': 75, 'w': 419, 'h': 419}, 'age': 24, 'gender': 'Man', 'race': {'asian': 4.395343018407584e-06, 'indian': 1.4951670304865424e-05, 'black': 99.99988675116818, 'white': 1.8270046287922217e-08, 'middle eastern': 9.500534259361315e-08, 'latino hispanic': 9.431461187978349e-05}, 'dominant_race': 'black'}</pre>

Table 2: Accuracy tests

The previous images are images that were used to test the detection capabilities of the framework, as the real time is not a requirement for this project, the models and backends with more accuracy were choose like retinaface, and FaceNet512. In the Table 2, to the left is an image of myself with no clear expression (neutral), and to the right one image with some happy person. The results are clearly accurate, and more if we add some preprocessing to the images.

As mentioned before, this framework will be used for person recognition, to be able to know the patient the system is working on, and besides that to do the emotion recognition.

### 2.7 CUDA

CUDA is a parallel computing platform so popular in the AI field due to its high-performance doing computation that for a CPU would late a lot of time [7]. The Figure 3 shows the web site of the CUDA Toolkit.



#### CUDA Toolkit

Develop, Optimize and Deploy GPU-Accelerated Apps

The NVIDIA® CUDA® Toolkit provides a development environment for creating high performance GPU-accelerated applications. With the CUDA Toolkit, you can develop, optimize, and deploy your applications on GPU-accelerated embedded systems, desktop workstations, enterprise data centers, cloud-based platforms and HPC supercomputers. The toolkit includes GPU-accelerated libraries, debugging and optimization tools, a C/C++ compiler, and a runtime library to deploy your application.

Using built-in capabilities for distributing computations across multi-GPU configurations, scientists and researchers can develop applications that scale from single GPU workstations to cloud installations with thousands of GPUs.

Figure 3: CUDA Toolkit web site

## Implementation of an emotion detection algorithm for therapy applications in children

This technology is owned by Nvidia and to be able to run it a Nvidia GPU card is needed. Is not necessary but strongly recommended to reduce processing times.

In the article “A comparison between CPU and GPU for image classification using Convolutional Neural Networks” some experiments were performed to expose how the GPUs offer a higher speed in processing times over CPUs [8].

Everything was developed to work with CPU processing, but after that CUDA was implemented in the system to reduce the processing times drastically. A test of processing 3 videos of FullHD resolution and 5 seconds of duration were performed, with the result that CPU processing late 14 minutes and GPU processing with CUDA reduced this time to 4 minutes.

Implementation of an emotion detection algorithm for therapy applications in children

### 3. SYSTEM STRUCTURE

In this part it will be exposed how is composed the developed system structure, the hardware used and the communication flow between everything. Explaining the original idea and how it was developed until the final development.

#### 3.1 From the original idea to the final implemented system

The original idea was to use one unique computer integrated with everything as a robot itself.

It would include a camera to perform all the detection attached to a robotic arm to be able to track the children's movements, record the session and process the data to recognize the patient and detect the emotions. That wasn't an impossible approach, but in fact it requires lots of development time and after all, lots of money specially in the robotic arm part, because the tracking depends on a moment prediction algorithm that needs to be near to real time and infer over the control system that would be applied to the arm so the hardware costs could increase considerably.

To perform the records the intended to used camera was a Microsoft Kinect camera. This camera doesn't have a really high resolution but being near the patients that wouldn't be a problem, and the strong points of that camera are the built-in IR and the Depth camera it has. As mentioned in the point 3, there are some similar projects in which this camera was used too.

Being not too much affordable the real time tracking idea as mentioned before, other more feasible approach was to use cameras placed at fixed points of the workspace which can track the maximum amount of the therapy as possible, and if needed more than one camera could be used, trying to minimize the blind spots, decreasing considerably the deployment and development costs. For these reasons, this is the chosen approach in this research to implement the system.

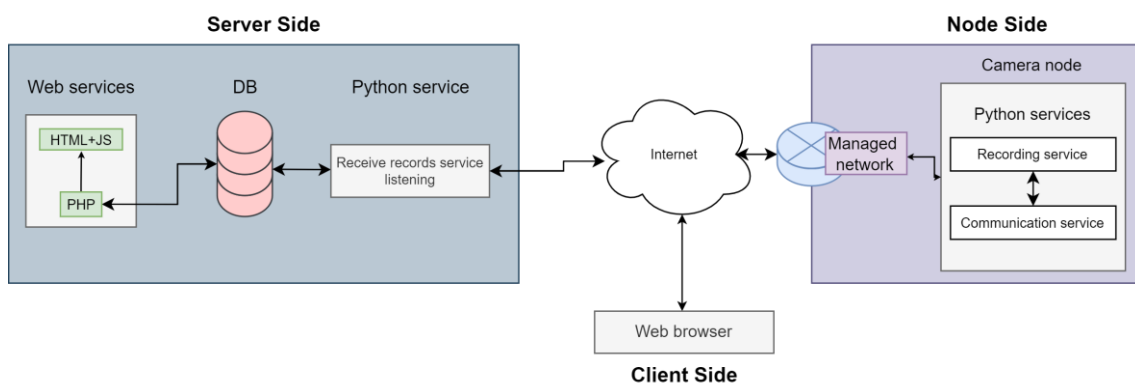


Figure 4: System topology

In the Figure 4, there is a representative idea of the intended implementation of the system and some of the technologies that could be used.



## Implementation of an emotion detection algorithm for therapy applications in children

The server side with the processing software, the web application to display the information and a database to store the necessary data. The node side with the cameras, a recording service, and a communication service to be able to communicate with the server.

Additionally, a user can access the information from a web browser.

### 3.2 Camera nodes

To be able to record the sessions, a little microcontroller or microprocessor could be used, and the Raspberry Pi 4 suites enough for that purpose, so a camera node or a node as will be named from now on in this document, will be a camera attached to a Raspberry Pi and connected to the LAN of the workspace.

Considering that the system does not need to work in real time, because will be used as feedback data for next sessions, there was no need to be processed on the nodes, then all the data will be gathered in the nodes and sent to a server, who will be in charge of doing all the data storage and processing.

With this topology, to use a Kinect camera was considered not feasible, because at medium and long distances, the Kinect is no so effective as the resolution it can offers is 640x480px, so a conventional RGB camera with more resolution was chosen after doing a bit of research to get a better option.

Another reason to clearly discard the use of a Kinect camera was the fact that with this topology more than one node can be used for the system. A Kinect camera is not easy to find in the market actually and will lack considerably the capacity of being a scalable system if more than one could be need.

A good option for the camera is for example the Logitech C920, which has a full HD resolution, the capacity to encode the recordings in H264 (this will affect precio releasing hardware stress when recording), and is not so expensive. The Figure 5 exposes the camera on Amazon web site.

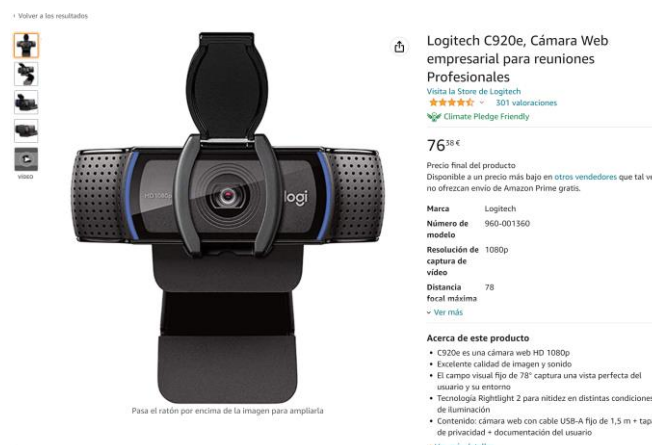


Figure 5:RGB camera Logitech C920

### 3.3 Server

For the tests a personal laptop was used as a centralized server to process everything, but in a real environment, a more powerful server will be a good idea and considering that it will need some maintenance and a 24/7 accessibility to the resources, externalization is a good option.

The main requirements for the server are:

- It should be able to execute Python and install all the needed libraries
- Support of cron jobs or scheduled tasks
- Can store multimedia files
- Have web and DB services

Just searching about which platforms can offer these capabilities that satisfies the needed requirements, the most convenient service would be a cloud virtual machine service, which enables the administrator to have almost full control in the machine.

The analyzed service providers shows that the most established services in the market are Google Cloud Platform, Azure VM and Amazon Web Services (AWS). All the research is based in a standard specifications machine that satisfies the previous needs, being the minimum requirements considered the next ones:

- Processor: 8 cores
- RAM Memory: 16 GB
- Storage capacity: 500 GB
- Location: As nearest as possible to the deployment area (Spain)

With all this in mind, we can say that the most determining factor was the price, as this project is not developed thinking on big companies it should be as affordable as possible and then scalable. So, the price was considering the decisive factor.

And as it's showed in the next figures (Figure 6, Figure 7, Figure 8), for the similar features the less unexpensive provider at this time would be AWS, considering that Azure doesn't offer 500 GB of storage just 150, 300 or 600, being less flexible in the machine you can get. And considering as AWS has the greatest number of services including lots of AI services which could be a considerable option to have in mind facing the future, AWS would be the preferred option.

## Implementation of an emotion detection algorithm for therapy applications in children

The screenshot shows a pricing estimator interface for Google Cloud. It is titled "Estimate" and is divided into two main sections: "Compute Engine" and "Persistent Disk (Accompanying)".

**Compute Engine Section:**

- Quantity: 1 x
- Region: Madrid
- 730 total hours per month
- Provisioning model: Regular
- Instance type: e2-standard-4 (USD 139.91)
- Operating System / Software: Free
- Estimated Component Cost: USD 139.91 per 1 month**

**Persistent Disk (Accompanying) Section:**

- 1 x boot disk
- Product accompanying: Compute Engine
- Zonal SSD PD: 500 GiB (USD 85.00)
- USD 85.00**

**Total Estimated Cost: USD 224.91 per 1 month**

At the bottom, there are three buttons: "EMAIL", "SAVE", and "DOWNLOAD\*". The currency is set to "USD - US Dollar".

Figure 6: Google cloud pricing

D4d v5	4	16 GiB	150 GiB	\$171,5500/mes	\$103,9009/mes ~39% savings	\$69,2624/mes ~59% savings	\$72,5452/mes ~57% savings	+
D8d v5	8	32 GiB	300 GiB	\$336,5300/mes	\$201,2391/mes ~40% savings	\$131,9621/mes ~60% savings	\$138,5212/mes ~58% savings	+
D16d v5	16	64 GiB	600 GiB	\$666,4900/mes	\$395,9009/mes ~40% savings	\$257,3469/mes ~61% savings	\$270,4723/mes ~59% savings	+

Figure 7: Azure pricing

The screenshot shows the AWS Pricing Calculator interface. It is titled "My Estimate" and includes several action buttons: "Duplicate", "Delete", "Move to", "Create group", "Add support", and "Add service".

Below the buttons is a search bar labeled "Find resources".

The main table displays the following information:

Service Name	Upfront cost	Monthly cost	Description	Region	Config Summary
Amazon EC2	0.00 USD	196.55 USD	-	Europe (Paris)	Operating syste...

At the bottom, there is an "Acknowledgement" section with a small text box containing details: "Operating system (Linux), Quantity (1), Pricing strategy (EC2 Instance Savings Plans 1 Year No Upfront), Storage amount (500 GB), Instance type (t4g.2xlarge)".

Figure 8: AWS pricing

If there is an intention of implement the system with the CUDA acceleration there is also the need to check for a GPU card.

Implementation of an emotion detection algorithm for therapy applications in children

All the needed packages and software were deployed over AWS with its free tier plan and no problems were found, but due to security policies over the testing network, and knowing the computing power of the AWS free tier, for the tests a regular computer in the local network has been used as a server, having the specifications shown in the Table 3:

<b>Specification</b>	<b>Model/Quantity</b>
<b>CPU</b>	Intel core i7 8650u
<b>RAM Memory</b>	8 GB DDR3
<b>Storage Capacity</b>	SSD 320 GB
<b>GPU</b>	Nvidia GeForce GTX 1050
<b>Networking</b>	Gigabit Ethernet

*Table 3: Test computer specifications*

### 3.4 Web application

To be able to display all the information gathered by the nodes and processed by the server, an application was needed.

The most useful application to achieve that purpose would be a web application, letting the therapists have access to the data anytime and anywhere, discarding phone or desktop application choice.

The objective of the web in the project is to have a proper way of show the information to the therapists, so they can evaluate the sessions with the information that offers to them the emotion analysis, but not just that, the web application could be used to start and stop the recordings of each session at the users will.

As showed in the Figure 4, profiting the server used to process all the data, can be used too as a web server decreasing the deployment costs. To make the web application the idea was to use HTML and JavaScript for the frontend and PHP for the backend as they are pretty standardized technologies and there are no special needs for the web application as long as the information displayed is clear.

Besides these to store the information properly a database would be needed too to exchange information between the nodes and web as an intermediary.

So, the web application seemed to be a really feasible solution to display the information, having access to that information at any time and any place there is a computer and the same way to control the record system.

Implementation of an emotion detection algorithm for therapy applications in children

## 4. PROJECT LIMITS

In this section will be explained the project limitations and considerations that have been done due to expenses, data, and time terms.

### 4.1 Body pose classification and tracking

One interesting feature is the possibility to classify a specific types of repetitive body movements or behaviors, or as it's known stereotypies. For example, legs crossing, tiptoe or throwing and object repeatedly.

These behaviors could be classified, but the resources needed to do this are considerably high to be included now, so it can be considered as a future feature to think about. Not being static scenarios as would be expressions or objects detection, a possible approach could be time series classification techniques for example.

But the classification of the stereotypies itself would need a trained model which requires first of all the data to be trained and with so specific poses. To get that data and a considerable quantity of it, is not feasible now.

The body language expressed through pose offers a complementary information in the emotions understanding, but the main focus in the emotions understanding are the facial expressions [9], is not considered feasible now but still being a considerable feature to have in mind.

### 4.2 Eye gazing

Other interesting feature was the capability to measure the eye deviation when the patients are interacting with the specialists to detect some kind of behavior that relies on those deviations.

The therapy sessions are developed in an open space, and when the children interact with the environment they have absolutely freedom, and they are in unpredictable constantly movement.

The eye gazing is feasible when the subject to be analyzed is stopped frontally to the detection node, or the node tracks the user constantly to have the same angle between the camera lens and the target eye to know exactly if the look direction changes or not. Meanwhile the patients could be in constant movement and the need to have the highest angle of view as possible the nodes should be placed in the room corners and to detect the eyes from that distance would need a really high-resolution camera as a 4K camera at least which would increase the needed space required to store the data, and considerably the processing time and will increase the specifications of the server to be used.

For those reasons the eye gazing is other interesting feature but will not be included in the current release.

# Implementation of an emotion detection algorithm for therapy applications in children

## 4.3 Light conditions

In image processing, one of the most critically conditions that affects the results is the fact to have a good illumination.

In this case the workspace in which will be deployed the system have not a controlled illumination system to have a precise light control, but it has a strong natural illumination in the schedule the specialists usually work, and the space is notably well light conditioned daily as is shown in the next figures (Figure 9 and Figure 10).

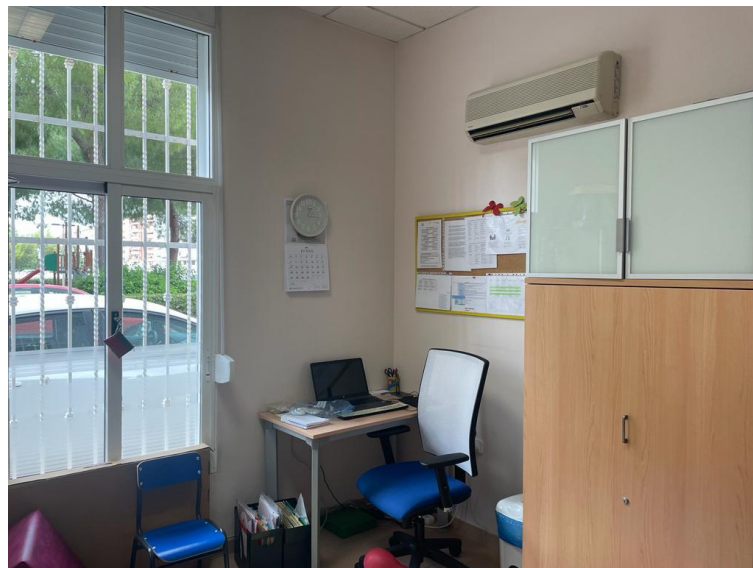


Figure 9: Natural illumination conditions 1

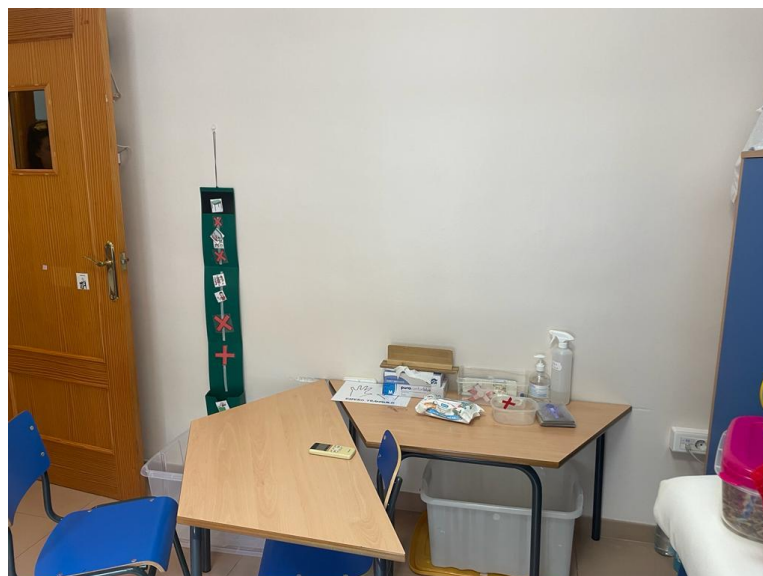


Figure 10: Natural illumination conditions 2

Implementation of an emotion detection algorithm for therapy applications in children

## 5. ALGORITHMS' DESIGN

In this part will be explained each part of the application design, like schemes, technologies, and communication flow between nodes, everything according to the point 3 of the document, the system structure.

### 5.2 Technologies used to design and develop

#### 5.2.1 Draw.io

Is an online application really useful to create schemes, diagrams, and others, it is used mainly to create the flow schemes and other diagrams used in this paper as the architecture scheme.

#### 5.2.2 Gitlab

Online repository to use git, the most used version control system, so useful to have a good workflow without hesitate to have mistaken and come back to a previous state of the project.

#### 5.2.3 Jupyter Notebook

Jupyter notebook is a framework to develop and test applications in a closed environment and with the facilities to have a visual environment to check results and other capabilities. It was used in the design steps of the project due to its facility to do tests and show results.

#### 5.2.4 VisualCode

Lightweight IDE of Microsoft, it has all the options and features needed with simplicity and a clean UI, perfect to develop multipurpose applications with different technologies and have multiple instances open at the same time.

#### 5.2.5 Apache

Web service, used to run the web system, is the most common web service around the world and because of that is so easy to use and to configure. No special feature was needed, so Apache suits enough for this project.

#### 5.2.6 MariaDB

DBMS service, so used around the world in terms of relational database services, for the same reason as the Apache service, there is no special needed for the Database, so MariaDB suits perfectly.

#### 5.2.7 SFTP

Service with the purpose of send data and files securely through the internet, establishing a secure channel between two terminals. It is used to send the session records to the server due to the high sensibility of the information gathered.

#### 5.2.8 SSH

In a final deployment, SSH could be used to communicate with the server, for example with the mentioned in the point 3, AWS. Is a service that enables the user to establish a secure connection between two terminals to send commands.

# Implementation of an emotion detection algorithm for therapy applications in children

## 5.3 Node design

The main purpose of the nodes is to record the sessions on the user demand, so different options were considered as controlling the nodes manually, use a specific computer on the local network to control when the record starts or ends, or just record everything without stopping.

At the end, the most feasible and comfortable option was to communicate with the nodes through the internet.

To be able to do that, the idea was to use a program that will be listening for “START” or “STOP” commands arrives and when start command reaches the nodes, they will start to record the session and in parallel keep listening for commands, until a stop command is received, as is presented in the Figure 11.

In the moment a stop command was received, the record file will be saved locally in the node with a unique name that contains the current date, time, patient id, camera id and be sent through SFTP to the server. The nodes are designed to just have the program root folder and another folder to store the records. After that the node will continue listening for commands.

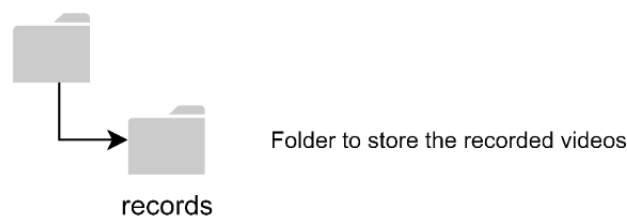
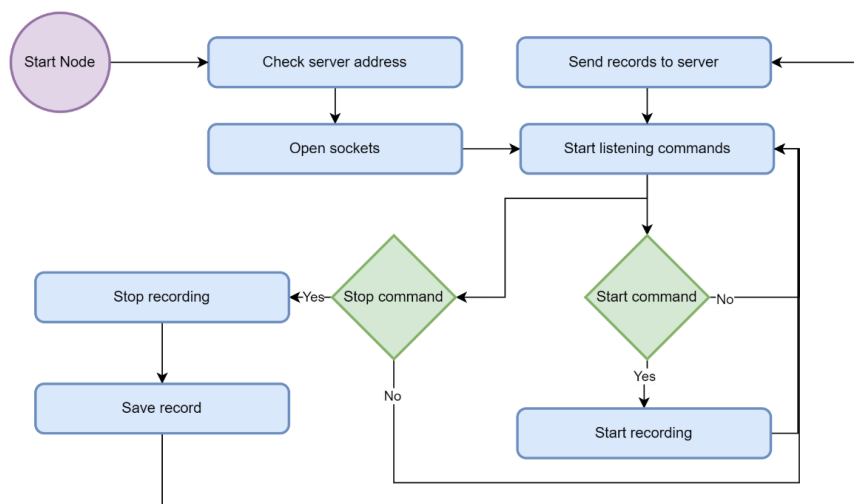


Figure 11: Node program flow and folder structure



Implementation of an emotion detection algorithm for therapy applications in children

#### 5.4 Server design

For the server algorithm the folder structure is an important thing to consider, and each part of the algorithm will be explained separately to comprehend how it was developed.

The steps are, obtain the video frames, and attempt to detect faces in that frame. If there are faces, these faces will be cropped, and verified to know what patient appears on that image. If the patient which appears is the right one, then that image will be preprocessed and sent to the analysis model to infer the emotions. After that just the results will be sent to the server database and displayed on the web application.

Figure 12 shows the different steps of the analysis software developed, the software flow and the server folder structure.

# Implementation of an emotion detection algorithm for therapy applications in children

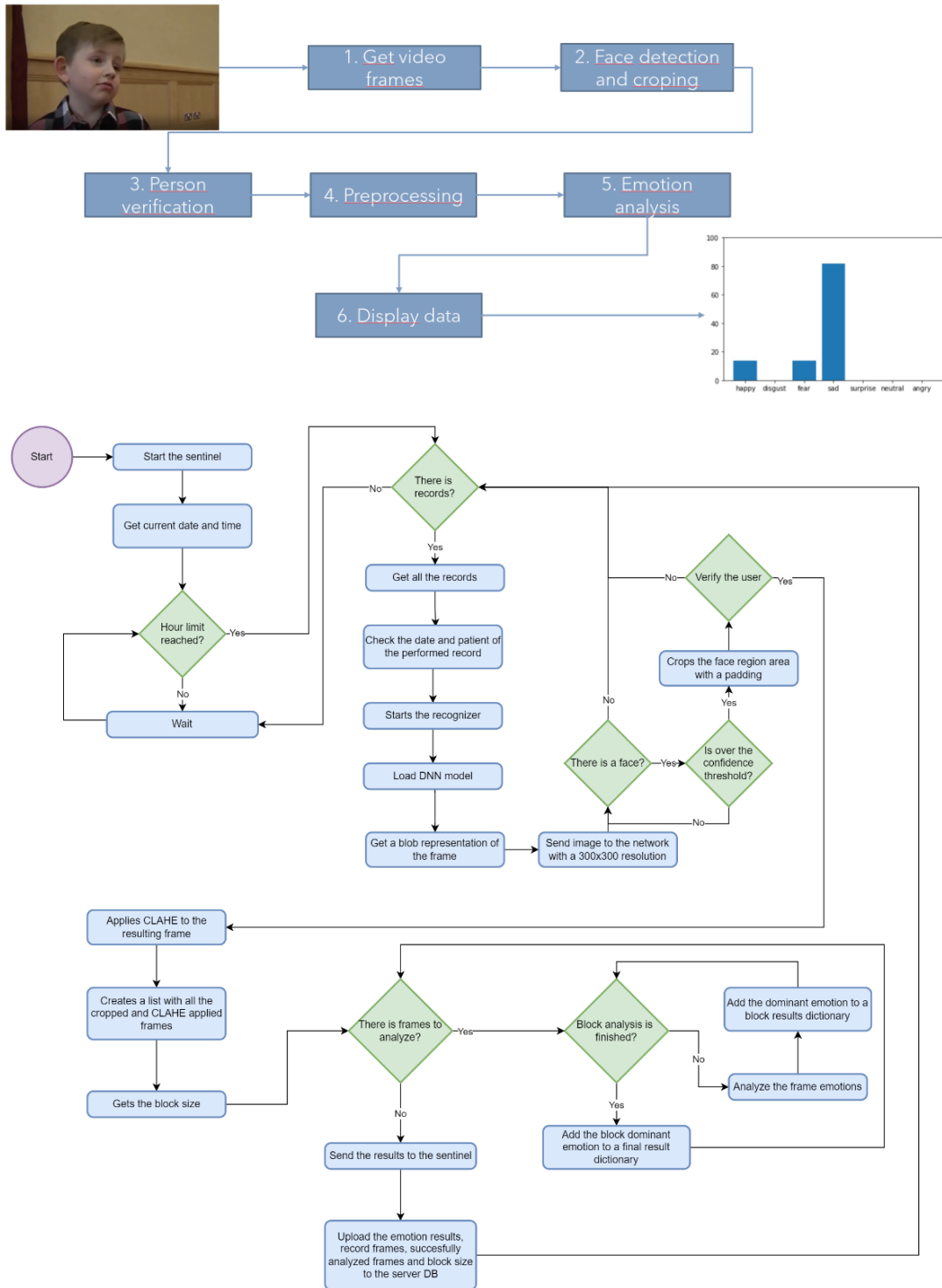


Figure 12: Server emotion analysis block scheme and analysis program flow scheme

## Implementation of an emotion detection algorithm for therapy applications in children

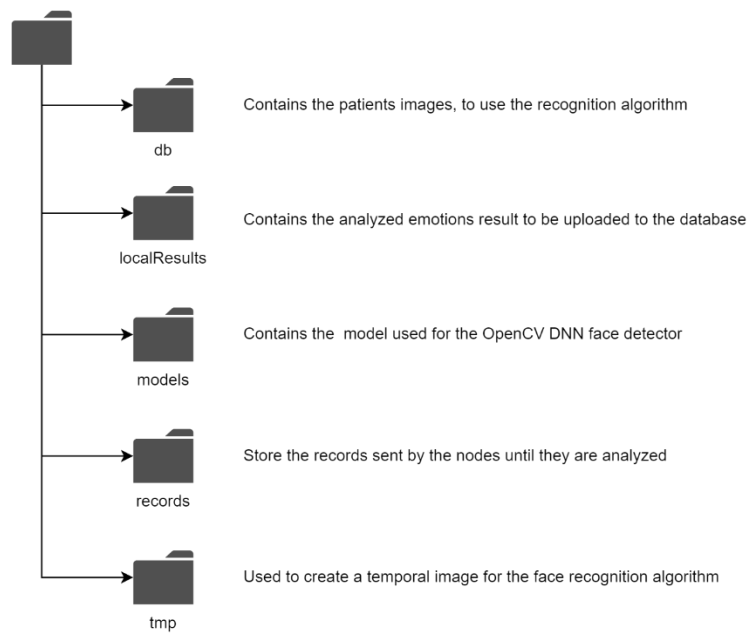


Figure 13: Server folder structure

### 5.4.1 Face detection and cropping

For the analysis the first step is to achieve the best possible result and reduce computational costs. So, to reduce the analysis region and to work with just a region of interest (ROI) is the strategy used, working just with the face area.

To achieve this the first approach used was to use a Haar recognition model, which were satisfactory being so fast in the detection, but depending on distance and other factor the Haar model can get several false positive and works better with faces of the same size and angle, for example just with frontal faces. And for this project purpose this can be a real problem.

The Figure 14 shows a Haar model face detection giving false positive results.

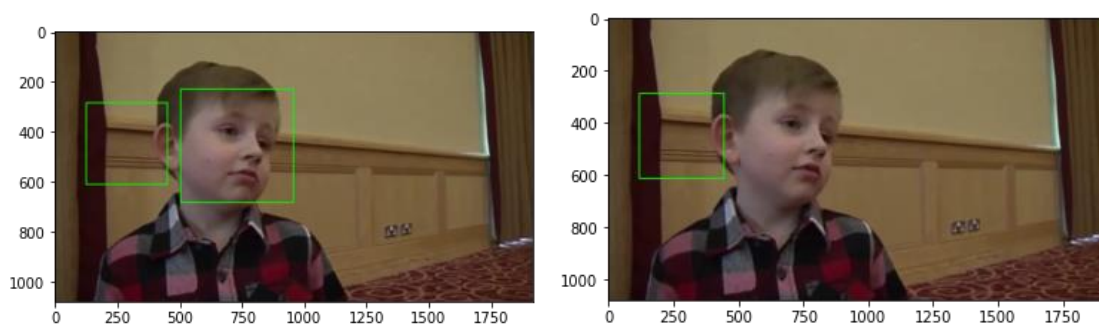


Figure 14: HAAR face detection

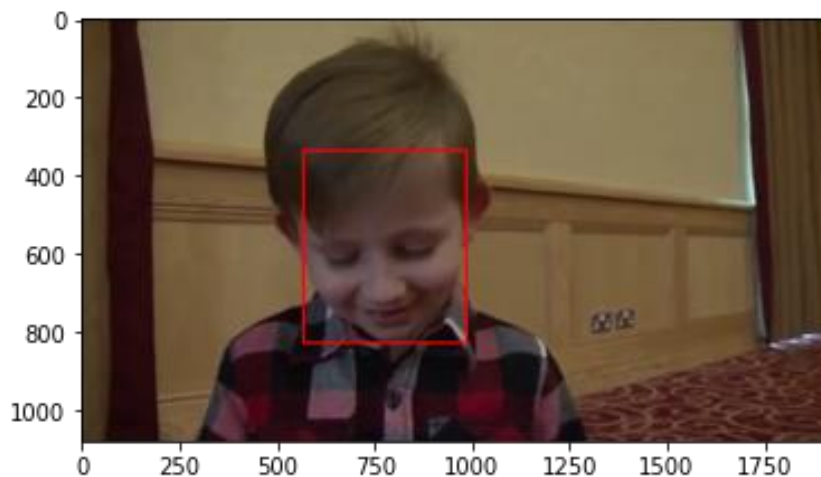
## Implementation of an emotion detection algorithm for therapy applications in children

After that, other models were considered like MTCNN, Dlib or DNN, being the DNN the election for the project.

For general purpose detection applications DNN offers the best results and is considerably faster compared to Dlib and MTCNN, and besides that it seems to be the best option compared with the previous libraries in terms of illumination changes adaption, face occlusion management and tracking the face in movement [10], being this last feature one really considerable for this project, as is expected the children could be in continuous movement, as is presented in the next two figures (Figure 15 and Figure 16).



*Figure 15: Face detection through DNN*



*Figure 16: DNN detection even in difficult angles*

Implementation of an emotion detection algorithm for therapy applications in children

#### 5.4.2 Person verification

With the frames' faces cropped as ROG the next step is to verify the person that appears on that ROG.

To be able to do that each patient should have at least one frontal photography which will be stored on the server in the "db" as showed in previously.

The DeepFace framework allows to us to verify one image with other one doing a comparison. So, the patient image is used to verify that the cropped face is the from the patient that is supposed to be. This is so important because the therapist can appear in the recording and some method to perform an identification of the tracked patient was needed.

The Table 4 exposes a face verification example.

Patient image	Recorded image	Verified face
		
<pre>Out[24]: {'verified': True, 'distance': 0.5943954252817916, 'threshold': 1.04, 'model': 'Facenet512', 'detector_backend': 'opencv', 'similarity_metric': 'euclidean_l2'}</pre>		

Table 4: Patient verification at a considerable distance from the camera

There are other possible approaches to achieve this, as marking the user with some IR object like the motion tracking suits used to make films, or some object with similar features.

But considering that the main target of the therapy sessions is to analyze the children's behaviors in the most natural possible environment and that they are considerably young, if is possible it's preferred not to use a very invasive method to perform the detection.

Because of that software solutions were preferred, and the DeepFace framework offers a good solution for this.

## Implementation of an emotion detection algorithm for therapy applications in children

### 5.4.3 Preprocessing

With every face cropped and verified there are possibilities that due to shadows, lights or just for the facial shape that the real expression is not able to be detected correctly by the framework.

To solve that problem, after several tests to perform a contrast improvement to the frames was decided as preprocessing before to start the emotion analysis.

CLAHE technic [11] was chosen for this task, because it can improve the contrast based on the image, equalizing the image histogram considering some limits to not loose information saturating contrast with too many dark zones or light zones.

An example of the improvement respects an original image and a preprocessed one is showed in the Table 5.


Image	Results
 <p data-bbox="427 1122 596 1151">Original image</p>	<pre data-bbox="884 864 1262 1151"> {'emotion': {'angry': 16.84383451938629, 'disgust': 0.028964155353605747, 'fear': 29.28096652030945, 'happy': 0.1397483516484499, 'sad': 53.101712465286255, 'surprise': 0.054557976545765996, 'neutral': 0.550217833699703}, 'dominant_emotion': 'sad', 'region': {'x': 992, 'y': 276, 'w': 719, 'h': 719}, 'age': 35, 'gender': 'Man', 'race': {'asian': 0.0003708075149017907, 'indian': 0.00035090076653126074, 'black': 1.686872431253843e-06, 'white': 98.55312696415943, 'middle eastern': 0.9866417973722099, 'latino hispanic': 0.4595122108317739}, 'dominant_race': 'white'} </pre>
 <p data-bbox="357 1420 667 1449">CLAHE preprocessed image</p>	<pre data-bbox="884 1162 1262 1449"> {'emotion': {'angry': 1.4767108433968332, 'disgust': 5.387763800938491e-06, 'fear': 0.39860226727141546, 'happy': 0.5577469577042435, 'sad': 0.6609684250335947, 'surprise': 0.22442086074774628, 'neutral': 96.68154696712178}, 'dominant_emotion': 'neutral', 'region': {'x': 970, 'y': 249, 'w': 763, 'h': 763}, 'age': 34, 'gender': 'Man', 'race': {'asian': 8.429595688141944e-07, 'indian': 4.230049199804853e-07, 'black': 2.603832731026823e-10, 'white': 99.93758797645569, 'middle eastern': 0.046833124361000955, 'latino hispanic': 0.015578774036839604}, 'dominant_race': 'white'} </pre>

Table 5: CLAHE comparison

Besides that, other preprocessing techniques were tested as high pass filtering to increase the edges and try to accentuate the differences in the face and do easier the emotion detection.

But the results show that the improvement is too poor to gain accuracy or the image losses too much its naturality obtaining the opposite effect, that the model didn't recognize properly the emotions or the person itself.

So, it is possible add more preprocessing steps, but the CLAHE is considered good enough for the actual project needs and it's preferred to don't unneeded steps to fasten the process.

Implementation of an emotion detection algorithm for therapy applications in children

#### 5.4.4 Emotion analysis

For the emotion analysis it is used a vector with all the cropped, verified, and preprocessed faces. The difference between the number of verified faces and the number of the recorded frames will be used to measure the detection accuracy of the system on a specific record.

To classify the emotion the framework follows an internal pipeline in which first of all there is an attempt to detect the face using some of the built-in detectors as OpenCV, SSD, Dlib, MTCN or Retinaface and aligns the face the have a 0 degrees angle rotation. But these detectors sometimes don't detect properly faces that appears in some frames. To improve that was other reason because in the previous steps a face detection is performed first through OpenCV DNN module offering higher accuracies.

When the face detection and angle correction is performed and internal preprocessing occurs where the image is color space becomes greyscale, its resized and normalized. These steps are important because the CNN will expect a fixed image size the same way than normalizing the values between 0 and 1.

With the internally preprocessed image, is sent to the prediction model to get a result of the appearing emotions. The prediction model itself is a CNN model built in Keras, with that model an already trained weights for that model are loaded and a prediction is done within 7 possible classes as listed next:

1. Happy
2. Disgust
3. Fear
4. Sad
5. Surprise
6. Neutral
7. Angry

The prediction will return the 7 classes with a value between 0 and 100, being the summatory of every class 100, this way is how the dominant emotion is known.

For this process a block size should be decided, it is not a good idea to analyze each frame independently as it is assumed that a human being cannot express an emotional change in less than one second or even in more.

For that reason, an approach is to perform the emotions analysis by blocks. Having an important relevance to choose a good compromise between a large block size that allows to us to reduce the abnormal values and a small block size that doesn't lose valuable emotions for the result.

These emotions will be stored meanwhile a block is being analyzed, and when the block analysis is finished, the dominant emotion will be chosen as the real emotion of that block.

When the analysis is finished, all the values will be uploaded to the database, with the total number of frames, the analyzed frames and the block size used, and the record will be sent to a folder which is accessible to the web application.

## Implementation of an emotion detection algorithm for therapy applications in children

```

Action: race: 100% | 4/4 [00:11<00:00, 2.76s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.40s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.38s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.40s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.37s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.36s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.33s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.39s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.47s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.43s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.42s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.41s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.41s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.48s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.44s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.42s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.46s/it]
Action: race: 100% | 4/4 [00:10<00:00, 2.57s/it]
Action: race: 100% | 4/4 [00:10<00:00, 2.51s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.50s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.46s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.42s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.44s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.44s/it]
Action: race: 100% | 4/4 [00:10<00:00, 2.72s/it]
Action: race: 100% | 4/4 [00:10<00:00, 2.53s/it]
Action: race: 100% | 4/4 [00:10<00:00, 2.52s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.50s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.44s/it]

Current block: {'happy': 0, 'disgust': 0, 'fear': 17, 'sad': 5, 'surprise': 0, 'neutral': 8, 'angry': 0}
TOTAL: {'happy': 0, 'disgust': 0, 'fear': 1, 'sad': 0, 'surprise': 0, 'neutral': 0, 'angry': 0}

```

Figure 17: Emotion analysis blocks 1

```

Action: race: 100% | 4/4 [00:09<00:00, 2.41s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.41s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.44s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.41s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.44s/it]
Action: race: 100% | 4/4 [00:13<00:00, 3.34s/it]
Action: race: 100% | 4/4 [00:11<00:00, 2.98s/it]
Action: race: 100% | 4/4 [00:10<00:00, 2.66s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.49s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.43s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.44s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.42s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.40s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.42s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.42s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.43s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.42s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.43s/it]
Action: race: 100% | 4/4 [00:11<00:00, 2.78s/it]
Action: race: 100% | 4/4 [00:10<00:00, 2.70s/it]
Action: race: 100% | 4/4 [00:10<00:00, 2.56s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.43s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.50s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.42s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.41s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.45s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.41s/it]
Action: race: 100% | 4/4 [00:10<00:00, 2.64s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.45s/it]
Action: race: 100% | 4/4 [00:09<00:00, 2.41s/it]

Current block: {'happy': 4, 'disgust': 0, 'fear': 0, 'sad': 22, 'surprise': 0, 'neutral': 4, 'angry': 0}
TOTAL: {'happy': 0, 'disgust': 0, 'fear': 1, 'sad': 1, 'surprise': 0, 'neutral': 0, 'angry': 0}

```

Figure 18: Emotion analysis block 2

Some examples of tests realized over Jupyter notebooks are shown in the previous two figures (Figure 17 and Figure 18).

Those are the results of the analysis of a record and summarizing the first and the second block. It's appreciated that, for the total counting, only the emotions that most occurrences had in each block are the resulting ones.

### 5.4.5 Display the data

For the data display the best method found is to use plots, being the most useful plot a round chart.

Also, bar charts were used, but is strange that more than 3 emotions appear in a record, so the rest of the emotions will be always zero. And a round chart allows to display just the appearing emotions being clearer from the UX perspective. The Figure 19 shows an example of the emotional results in a round chart.



## Implementation of an emotion detection algorithm for therapy applications in children

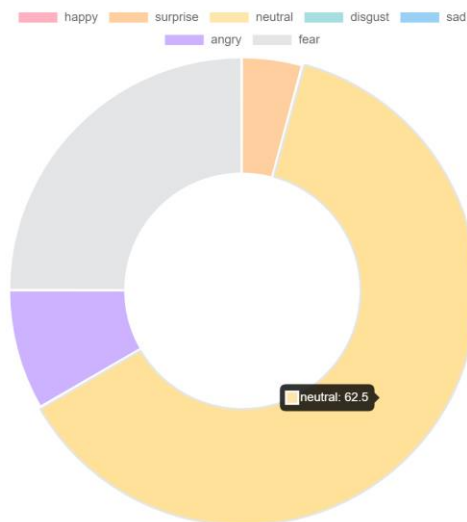


Figure 19: Data display example

Besides that, as each record has its own duration and block size, it's really difficult to have an idea of how to measure the progress of a patient.

To avoid that problem a normalization was performed between 0 and 100 previously to display the data on the web application.

### 5.5 Database design

For the database a relational MariaDB database has been chosen, and the database itself will contain 5 tables, 1 for users and another for the patients, 1 table for the cameras, 1 table for the sessions and the last one to store the sessions results.

And it was preferred to store the sessions and the session results separately, because in case of using more than one camera node, the results will be stored in independent registers, being able to do the processing to display the data after it is stored in the database and not before, this way there won't be data losses and the algorithm can be changed if desired. Figure 20 shows the database scheme.

## Implementation of an emotion detection algorithm for therapy applications in children

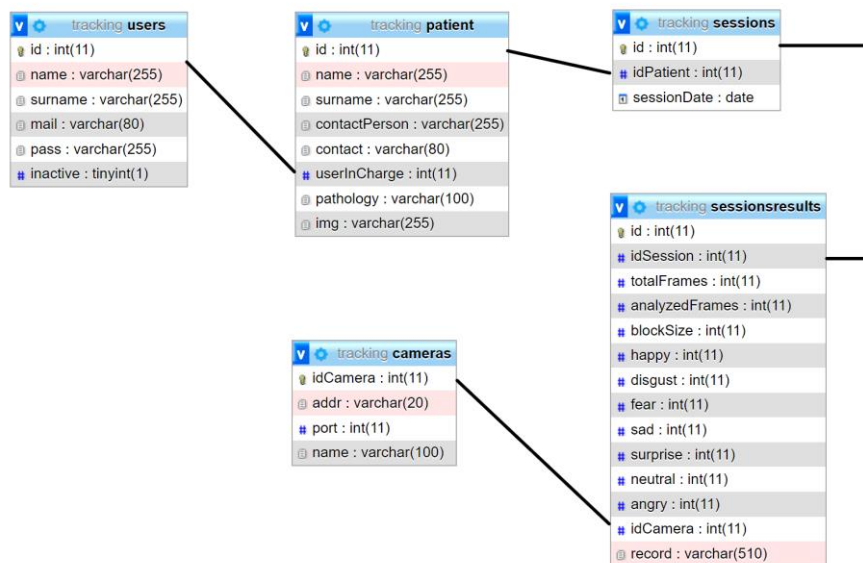


Figure 20: Database scheme

### 5.6 Web design

For the web application, an Apache server was chosen, because is really easy to deploy, free and well known.

The web application was designed to be used easily, displaying all the needed information as much clear as possible and in the minor quantity of pages than is possible.

The Figure 21 will show a diagram that represents the web pages flow for a user.

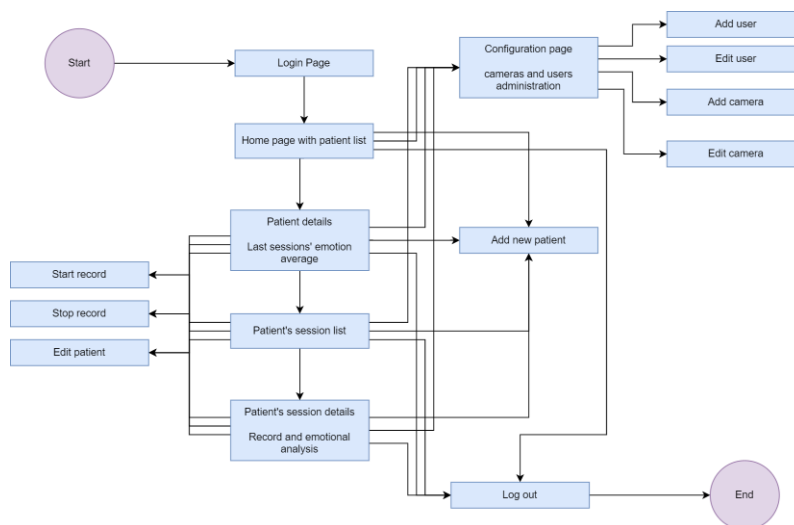


Figure 21: Web application pages flow diagram

## Implementation of an emotion detection algorithm for therapy applications in children

The web application consists of six web pages, and the user can navigate between everything with the help of a head navigation bar. As the information should be clear, every element is displayed with light colors, good contrasts, and wide spaces, as for the majority of the medical applications.

The web application has been designed to have a login system too, even if a user and password system itself was not required for this system, is considered so convenient, because having access to that amount of sensible data without any kind of restriction or protection could be a great risk for the patients.

In terms of UX design, everything is explained in detail in the attachment's document, being the main features of the web application the mentioned before. The next figures (Figure 22 and Figure 23) shows some of the wireframes with the element disposition alongside the screen.

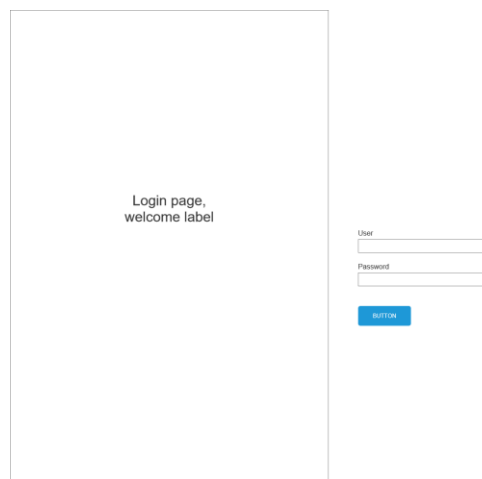


Figure 22: Login page wireframe

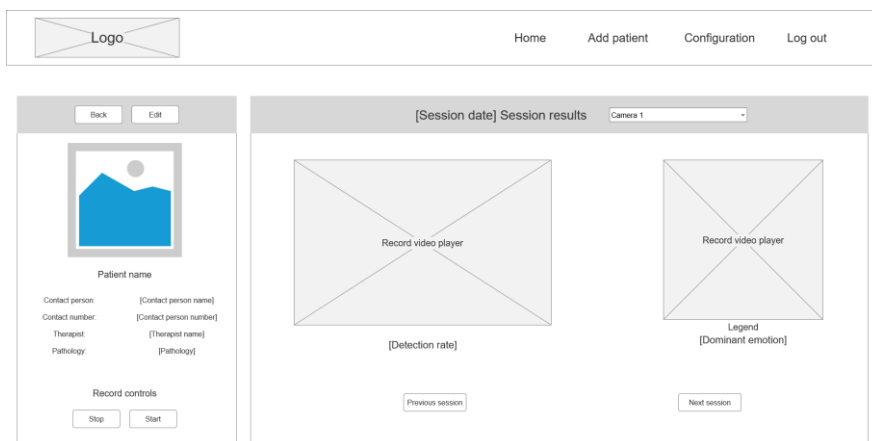


Figure 23: Session details wireframe

## 6. IMPLEMENTATION

In this part will be explained the whole system was implemented. Also, some of the project difficulties during the development will be exposed. Everything considering the real needs of the project and the existent limitations.

To access the developed software, it can be found in this public git repository [here](#).

### 6.1 Node implementation

#### 6.1.1 Software

Here will be represented all the files in the node program that are essential to understand the behavior. Next to this text Figure 24 shows an UML scheme about the node program.

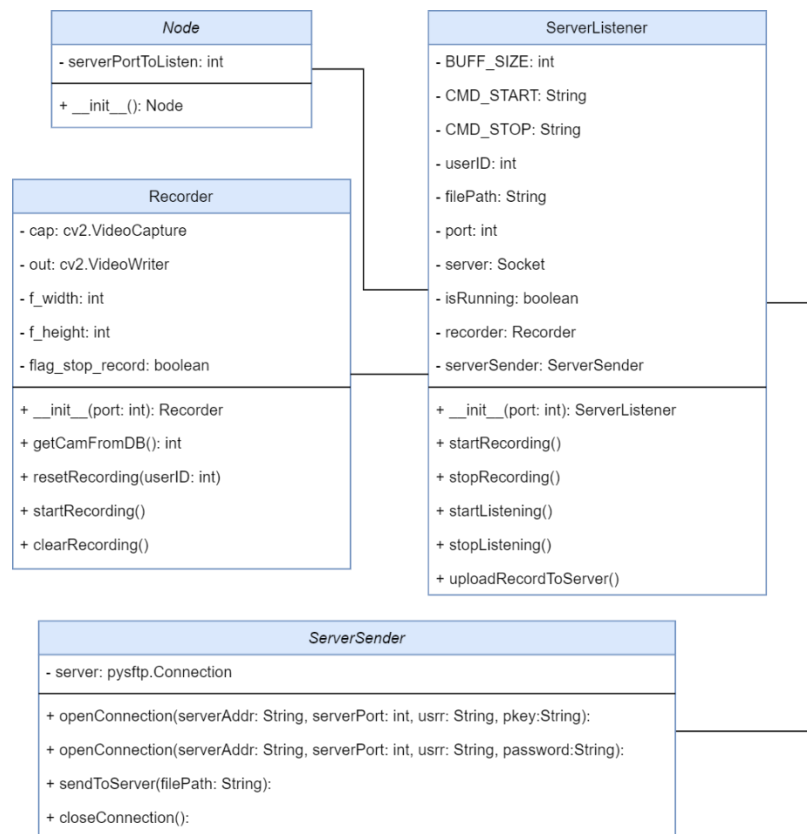


Figure 24: Node program UML scheme

##### 6.1.1.1 Node

The Node class is used as starting point, executing this, everything will run automatically,

Is a way of centralizing everything that is needed. Essentially loads the configuration from a JSON file to know the server address and port and create a ServerListener to start listening.

## Implementation of an emotion detection algorithm for therapy applications in children

### 6.1.1.2 ServerListener

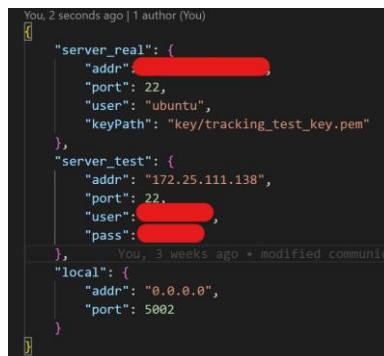
This is the most important class in the node, as it represents the features to listen continuously commands through the network to start or stop the recordings.

The problem needed to solve here was the communication between the nodes and the server, because the nodes couldn't be always recording but should be always listening.

The solution that used was to open a socket connection through a port, this port will be set up in the previously mentioned JSON. This way if there were multiple nodes, just change to change the port number in the configuration file is needed.

To use everything as mentioned it is strongly recommended that the system is installed in a network that can be managed, to allow the input traffic to reach the camera nodes addresses and ports. Without this, the system could be used too but just over a local network.

The figures down this paragraph (Figure 25 and Figure 26) show a node JSON configuration file and how a socket is created and starts listening for the server.



```
You, 2 seconds ago | 1 author (You)
{
  "server_real": {
    "addr": "172.25.111.138",
    "port": 22,
    "user": "ubuntu",
    "keyPath": "key/tracking_test_key.pem"
  },
  "server_test": {
    "addr": "172.25.111.138",
    "port": 22,
    "user": "ubuntu",
    "pass": "12345678"
  },
  "local": {
    "addr": "0.0.0.0",
    "port": 5002
  }
}
```

Figure 25: Node JSON config file



```
self.server = socket.socket()
self.server.bind(("0.0.0.0", self.port)) # Listens for server commands by the port in config file
self.server.listen(5)
print(f"[*]Nodes server listening..")
```

Figure 26: Node, server listener opening socket

When the socket is open and listening, a loop will start waiting for any expected commands ("START" or "STOP").

In case that start command is received a record will start to being recorded in parallel, without stop the loop that listens for commands to be able to stop the record in any moment.

If a stop command is received, the currently running record will stop, be saved, and uploaded to the server.

As mentioned before the upload to the server will be held via SFTP protocol. Previously the data was sent via socket connection, similar to the commands but as the data is considered sensible data another securer solution was needed, and SFTP is a good solution.

## Implementation of an emotion detection algorithm for therapy applications in children

Have SFTP service installed, and running is a requirement to run this system.

```
# Performs an SFTP Secure connection with the server addressed on
# the config file and sends the current recording file
def uploadRecordToServer(self):
    serverSender = ServerSender()

    f = open("config.json", 'r')
    fJSON = json.load(f)

    #serverParams = fJSON["server_real"]
    serverParams = fJSON["server_test"]

    addr = serverParams["addr"]
    port = serverParams["port"]
    user = serverParams["user"]
    #keyPath = serverParams["keyPath"]
    password = serverParams["pass"]

    serverSender.openConnection(addr, int(port), user, password=password)
    serverSender.sendToServer(self.filePath) # Sends the data and closes the connection
```

Figure 27: SFTP Server record sending

The Figure 27 shows how a file is sent to the server through SFTP service.

### 6.1.1.3 Recorder

The Recorder class is the responsible of managing the records, when it's created the basic record configuration parameters are set up, and when the method startRecording is called, in a loop the system will capture as a record everything from the connected camera to the computer, this way any kind of computer with a camera can be used as a node, allowing to be modified in function of the needs.

When the record is stopped all the variables of the record are cleared and set up again to be ready for the next record.

At the moment the record configuration parameters are set up, the path and record file name are chosen too. The name of the file is so important as it will represent the exact date, time, and the patient for which the record is supposed to be.

### 6.1.1.4 ServerSender

This class is used to send files via the SFTP connection to the server, and manages the connection opening, closing, and sending.

When the sendToServer method is called this already sends the file to the appropriate folder in the server to be able to read the records correctly.

# Implementation of an emotion detection algorithm for therapy applications in children

## 6.2 Server Implementation

### 6.2.1 Software

In this point, the server program and how it was implemented will be explained. The Figure 28 will show an UML scheme of this program.

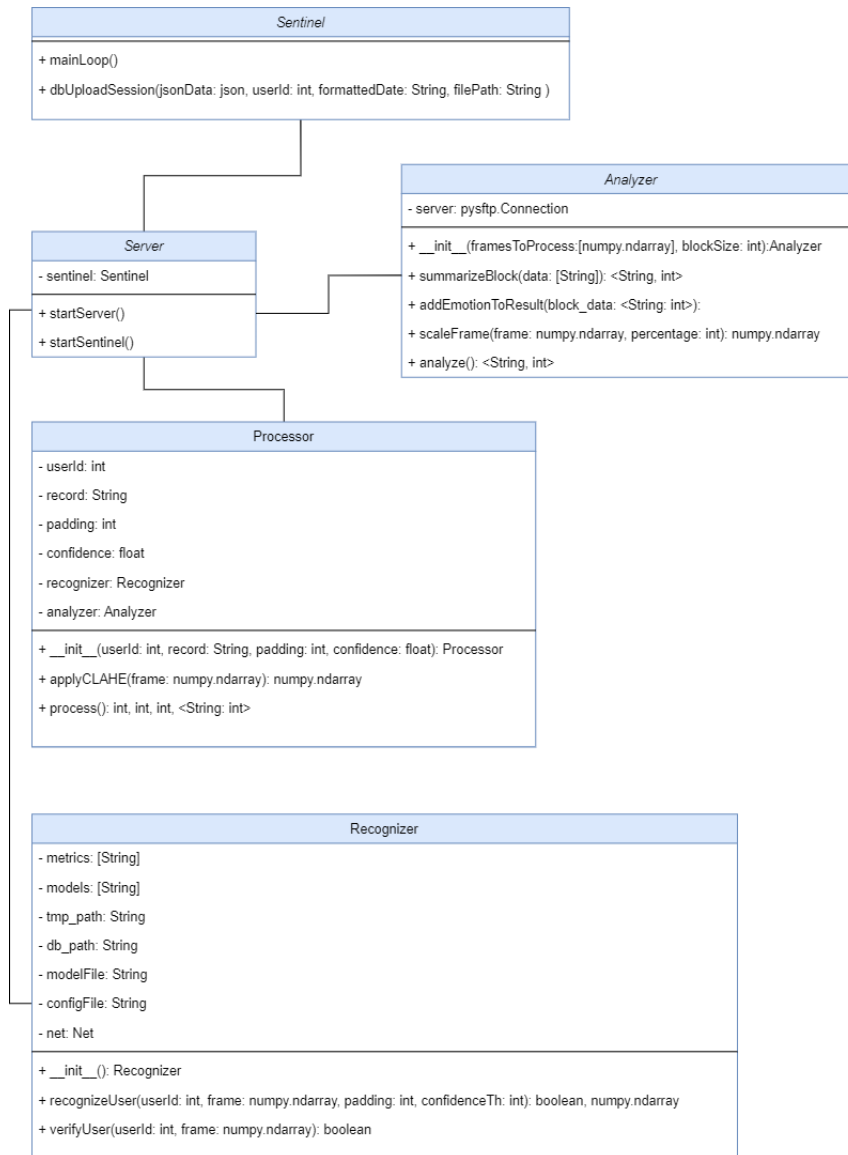


Figure 28: Server program UML scheme

#### 6.2.1.1 Server

The server class as the same way that happens in the nodes, is a class that works as starting point to executing everything that is needed by the system to work.

## Implementation of an emotion detection algorithm for therapy applications in children

When this class is running it's created an instance of the Sentinel.

### 6.2.1.2 Sentinel

The most important class of the Server side, first of all It's assumed that the therapy sessions won't be held 24 hours/day, so when it's late night, it will be considered as a safe time to start doing all the processing analysis.

The first thing the sentinel does is to check the current date and time and enters in a loop. In this loop if is not considered late nighttime yet, it will continue waiting, and when the established time is reached, it will start the processing procedure. Figure 29 shows how a loop iterates through the found files and send them to the Processor.

```
while(True):
    if(now.hour>= limit_hour and not processed): # Search all the videos and process it
        files = glob.glob(recordsPath+"/*.avi")

        for file in files:
            recordData = file.split("_")
            userId = recordData[9]

            file = file.replace("\\", "/")
            print("Start "+file+" processing!")

            fileNameArray = file.split("/")
            fileName = fileNameArray[len(fileNameArray)-1]

            # Process the video
            processor = Processor(userId, file,80,0.4)
            totalFrames, analyzedFrames, blockSize, results = processor.process()
```

Figure 29: Sentinel, video processing starts

The processing procedure consists in reading the folder that contains all the current day recordings, for each record the Sentinel will call the rest of the components eventually when needed, as the Processor.

The obtained results they will be uploaded to the MariaDB database, creating a session for the current day and a session result register attached to that session, this happens to be able to store multiple results per session in case of using more than one node.

```
# Connect with DB
cnx = mysql.connector.connect(user='root', password='', host='127.0.0.1', database='tracking')

print("Successfully connected")
mysqlCursor = cnx.cursor()

sql = "INSERT INTO sessions(id, idPatient, sessionDate) VALUES (%s, %s, %s)"
values = ("NULL", userId, formattedDate)

mysqlCursor.execute(sql, values)
cnx.commit()

recordUploaded = mysqlCursor.rowcount
print("1 session created: "+str(recordUploaded))

if(recordUploaded>0): # Add the details of the session

    sqlB = "INSERT INTO sessionsResults(idSession, totalFrames, analyzedFrames, blockSize, happy, disgust, fear, sad, surprise, neutral, angry, idCamera)"
    sqlB = sqlB + "VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s)"

    valuesB = (mysqlCursor.lastrowid, jsonData["totalFrames"], jsonData["analyzedFrames"], jsonData["blockSize"], jsonData["happy"],
    jsonData["disgust"], jsonData["fear"], jsonData["sad"], jsonData["surprise"], jsonData["neutral"],
    jsonData["angry"], jsonData["idCamera"])

    mysqlCursor.execute(sqlB, valuesB)
    cnx.commit()
    recordUploaded = mysqlCursor.rowcount
    if(recordUploaded<1): print("Insertion FAILED!!")

cnx.close()
```

Figure 30: Record session SQL insertion



## Implementation of an emotion detection algorithm for therapy applications in children

The Figure 30 shows how a session is included in the database.

After that the Sentinel will continue waiting until the next day. There is no need to run twice the server application.

### 6.2.1.3 Processor

The processor is the class that operates as an intermediary between the Recognizer and the Analyzer. Applies all the needed preprocessing techniques and return the results to the sentinel when everything is finished.

The processor reads the video record frame by frame and invokes the recognition method of the Recognizer, expecting to have a valid result, and if that is the case then the preprocessing techniques are applied.

In the current version, just the CLAHE preprocessing technique is used as mentioned in the point 4 of the document, when it's applied a list with all the successfully preprocessed frames is created and then sent to the Analyzer.

For the CLAHE equalization, the images are converted from BGR to LAB color space, as with a color space that doesn't have a separate channel for the luminance or similar apply this type of technique could result not satisfactory and in a loose of information. After the operation the resulting image is returned to BGR color space, as show in the Figure 31.

```
# Apply CLAHE equalization to improve contrast
# for emotion detection improves the result drastically
def applyCLAHE(self, frame):
    lab= cv2.cvtColor(frame, cv2.COLOR_BGR2LAB)
    l_channel, a, b = cv2.split(lab)

    clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8,8))
    cl = clahe.apply(l_channel) # Applying CLAHE to L-channel

    limg = cv2.merge((cl,a,b)) # merge the CLAHE enhanced L-channel with the a and b channel

    enhanced_img = cv2.cvtColor(limg, cv2.COLOR_LAB2BGR) # Return to the original color space

    return(enhanced_img)
```

Figure 31: CLAHE

### 6.2.1.4 Recognizer

The Recognizer is the part where using the registered patient images will be tried to get if the desired patient appears in that frame with the OpenCV DNN module, if that's the case then a cropping will be made in the patient face surrounding area, to get a better image for the emotion analysis.

The process consists of loading the network which is public on OpenCV GitHub repository and pass the model layers and weights as arguments.

## Implementation of an emotion detection algorithm for therapy applications in children

After that the original image is loaded following the DNN documentation the image is resized to 300x300 and a mean subtraction is applied to the values (104, 177, 123), being BGR the best result obtaining color space for this detector. (In the mean subtraction there is not a clear conviction if for the green value the best option is 177 or 117, so 177 offers in the realized test the best results, then it was chosen for the subtraction mean that is used).

The image is passed to the model, and it returns as a result the detection confidence, and the coordinates of the detected faces scaled down between 0 and 1, to be able to resize them to the original image resolution.

When the confidence is checked, at least 40% is considered the minimal accepted value, then the image is cropped adding some padding to the cropping because just the face seems like sometimes lose some valuable information from the corners of the hair, ears or similar.

Then with the results, the cropped face image is passed to the method `verifyUser`, which will use the `verify` method of the `DeepFace` framework to know if the user appears or not in the image.

For the detector configuration the `FaceNet512` model was used and to check the results the metric used was `Euclidean L2` (there are `cosine` and `Euclidean` too as possible metrics) [12].

The Figure 32 shows how an image is verified with a database existing image.

```
def verifyUser(self, userId, frame):  
  
    cv2.imwrite(self.tmp_path+'frame_tmp.jpg', frame)  
    path_ad = self.tmp_path+"frame_tmp.jpg"  
  
    df = DeepFace.verify(img1_path = path_ad, img2_path = 'db/'+str(userId)+'.jpg',  
                        model_name = self.models[2], distance_metric = self.metrics[2], enforce_detection = False)  
  
    if(df['verified']): return(True)  
    else: return(False)
```

Figure 32: User verification through `DeepFace FaceNet512`

One thing to mention is that the verification function can't operate without a stored image, because of that each frame that will be verified will be stored temporarily in a folder called "tmp" as shown in the Figure 13.

### 6.2.1.5 Analyzer

The `Analyzer` is the class that performs the emotional analysis to a list of frames. The list of frames used will be the list with preprocessed and cropped faces, and the result will be uploaded to a `MariaDB` database.

This process can take a considerable amount of time, depending on the computer, image size, and video duration, because of that the used data to test are really short videos. Each frame can take around 3 or 4 seconds to be analyzed with the test computer, and each real video second are around 30 frames, so it can spend a long time to do all the processing, this should be considered.

Implementation of an emotion detection algorithm for therapy applications in children

The analyze process consist of determine the size of the analysis block (now the frame rate is used, which means the block will be 1 second on any used camera), create two dictionaries to store the total analyzed emotion counting and another to store the dominant emotions in each block until the entire block is analyzed.

In a loop a block will be analyzed with the DeepFace analyze method, using the model of “retinaface” [13], and the dominant emotion will be stacked on the current block dictionary until the block ends.

When the block is finished the emotion which more occurrences have will increase in one unit in the global dictionary.

When all the blocks analysis are finished, all the record analysis process is finished.

## 6.2.2 Others

### 6.2.2.1 Image database

To be able to perform the patient verification it’s needed a database with one image per each patient that can appear on the records, this database consists of a folder with all the images (Figure 13).

The name of the images will represent the patient id number they have in the database when they are registered on the database.

### 6.2.2.2 temporal folder

The temporal folder is a folder created because the verification process works with disk stored images, for that purpose when the verification is performed, the current frame is stored temporally on that folder and this image is the one that is used to verify if that patient exists in the patient’s image database (Figure 13).

### 6.2.2.3 Records folder

A record folder is needed to store all the recording sessions. It contains one folder per day that a session exists and inside that folder the respective session recordings (figure 13).

When the analysis is completed, the records are erased from this directory to save space.

## 6.3 Database implementation

For the database implementation as mentioned before, MariaDB service was used, it was installed in the default port 3306, through the service manager known as XAMPP [14].

To manage faster the database the management was perform through a the “phpmyadmin” web application, which allows through a web service manage all the DBMS databases. The next Figure 33 shows how is the application.

## Implementation of an emotion detection algorithm for therapy applications in children

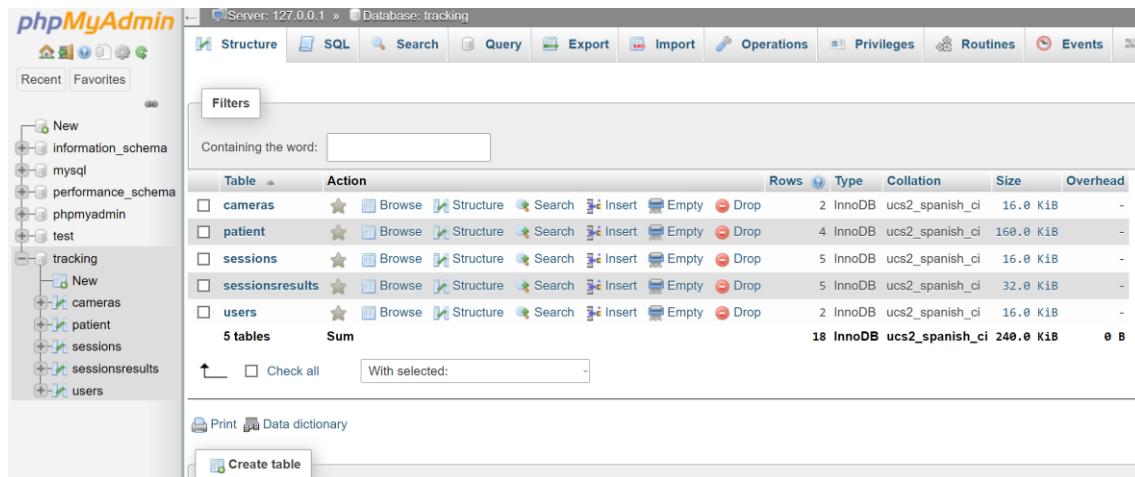


Figure 33: phpmyadmin MariaDB DBMS management web application

### 6.4 Web Implementation

As the database service, the web service is also managed by the service manager XAMPP. XAMPP is installed in the system as a service too if the computer runs a Linux system, which means that running just the XAMPP service will enable or disable the two services if desired.

For the web implementation there are the client side and the server side, as for the client HTML and JavaScript were chosen and for the server-side PHP as main languages.

JavaScript is mainly used to create requests to the php server side which will be the responsible of reading data from the database and offer the correspondent output. With that output the JavaScript will alter the HTML to create a dynamic web application.

The paradigm used to do the communication between the client side and the PHP with the database was the creation of a REST API to manage every request the same way and having the possibility to be easily scalable. Mainly all the operations will be CRUD operation, in the next figure (Figure 34) an example of the REST API is shown.

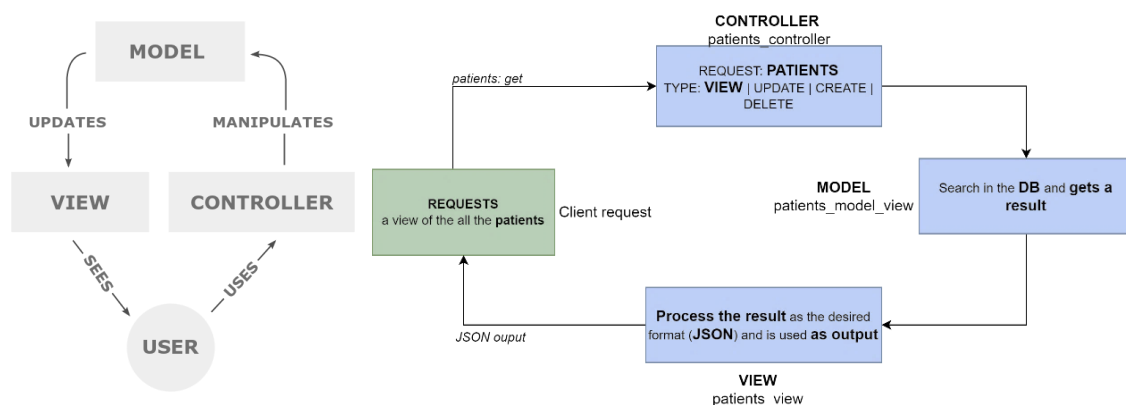


Figure 34: API REST example

## Implementation of an emotion detection algorithm for therapy applications in children

The REST API was implemented through MVC pattern to have a bigger and easier control of the number of files generated and to let a faster deployment of new requests.

The actual request for the developed API is listed next.

1. Camera
2. Camera list
3. Login
4. Logout
5. Log session
6. Patient
7. Patient image
8. Patient list
9. Recording
10. Session
11. Session cameras
12. Session next and previous
13. Session results
14. Session results average
15. Start Record
16. Stop Record
17. User
18. User list

The user session will be fully held on the server side, avoiding the use of cookies as it's not necessary.

One of the serious issues in the web application is the method of manage the recordings, because the user can be notified that a record is in process but as the records depends on the specific selected patient, closing the web application or moving to other page could be a problem to know if there is still active the recording system or what user is the one supposedly being tracked.

To solve this problem the server will be notified when a record is started and when is finished, this way it doesn't matters where is the user in the web navigation or what he does, the web application won't let the user start another record without finished the current one and will be no problems knowing what user is being tracked. In case the user extinguishes the session, if there is a record going on, it will be finished and saved on the server as any other.

For the client side of the web application there are six html pages, that are listed next

1. Index
2. Home
3. Patient
4. Patient sessions
5. Session details
6. Configuration

## Implementation of an emotion detection algorithm for therapy applications in children

Every page listed before have its own associated stylesheet file and JavaScript file to add the appropriate styles and features to work properly and make requests to the previously mentioned REST API.

To have a clear idea of the web application the Figure 35 shows the application folder structure.

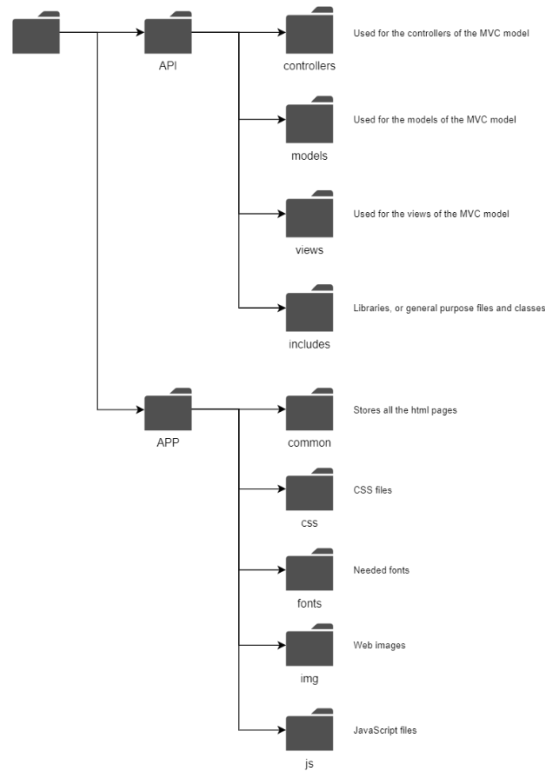


Figure 35: Web application folder structure

## 7. SELENIUM TESTING

To prevent as much as possible fails in the system when is being used, some unitary tests has been performed through a third-party software called Selenium [15], which was used as a library for some python scripts that takes the control of the browser to perform specific actions and check the results.

### 7.1 Access tests

Three tests were performed, two to check the usage of incorrect credentials and correct credentials to log in, and another test to try to have access to the system without being logged in. Every test ran with successful result as it can be seen in the Figure 36.

```
PS C:\Users\Theide1\Downloads\TFG downloads\tests> python .\testLogin.py

Non authorized access test succed
.
Login fail test succed
.
Login success test succed
.
-----
Ran 3 tests in 23.176s

OK
PS C:\Users\Theide1\Downloads\TFG downloads\tests> |
```

Figure 36: Access test results

### 7.2 Patients tests

For the patients users and cameras six tests were performed, two for patients, two for users and two for tests that checks the patients, users and cameras creation and edition, checking that all the fields are populated and for the patient that the patient's image is successfully upload to the server.

As the next figures (Figure 37, Figure 38 and Figure 39) show, every test was performed successfully.

## Implementation of an emotion detection algorithm for therapy applications in children

```
PS C:\Users\Theidel\Downloads\TFG downloads\tests> python .\testPatient.py
Patient edit test succed
.
Patient creation test succed
.
-----
Ran 2 tests in 30.766s

OK
PS C:\Users\Theidel\Downloads\TFG downloads\tests> █
```

Figure 37: Patient test results

```
PS C:\Users\Theidel\Downloads\TFG downloads\tests> python .\testUser.py
User creation test succed
.
User edit test succed
.
-----
Ran 2 tests in 24.915s

OK
PS C:\Users\Theidel\Downloads\TFG downloads\tests> █
```

Figure 38: Users test results

```
PS C:\Users\Theidel\Downloads\TFG downloads\tests> python .\testCamera.py
Camera creation test succed
.
Camera edit test succed
.
-----
Ran 2 tests in 24.911s

OK
PS C:\Users\Theidel\Downloads\TFG downloads\tests> █
```

Figure 39: Cameras test results



## 8. CONCLUSION AND FUTURE WORK

### 8.1 Conclusion

The conclusion of my work is that I am happy to be able to develop a full system that compounds (even if is a little part) hardware, algorithmics, computer vision, a bit of networking and web application as one unique product. This let me value more the potential of team work as specially my bachelor tries to do.

Besides that, know that is a good base to increase the work on this field as is the health care, especially in the ambit of mental health and oriented to children which I personally consider is a fundamental part on the future society.

And is satisfactory to know that the main needs of the project can be achieved through this research, knowing there is many works to do and there are other things that can be improved too, so I feel like the more time I invest more things can be achieved in this field particularly in this project.

### 8.2 Future work

As future work, there are many possible things for this project, starting for improving the developed features to increase the failure tolerance as it can be considerably improved.

The most notorious points for a future work would be to achieve other points that were discarded for this research due to the lack of infrastructure or time. For example, the eye gazing detection is a good point to be integrated in the system, through another type of camera, methodology or environment and it can provide really profitable information to the therapist.

The body pose classification feature is another one really interesting option to consider, cause besides the face being the main way of express emotion for a human being, the body is an important part to consider. In particular the stereotypies movements classifications are one really interesting work to be achieved and not an easy task.

With the mentioned improvements and the already developed features the web application could be really improved to the point of offering a great amount of valuable data to the therapists and increase considerably the therapy results for this ASD diagnosed children.

## Bibliography

- [1] "National Institute of Mental health," [Online]. Available: <https://www.nimh.nih.gov/health/topics/autism-spectrum-disorders-asd>.
- [2] "Centers for Disease Control and Prevention," [Online]. Available: <https://www.cdc.gov/ncbddd/autism/data.html>.
- [3] J. J and T. Mathew, "Facial Expression Recognition and Emotion Classification System for Sentiment Analysis," in *International Conference on Networks & Advances in Computational Technologies*, Trivandrum, 2017.
- [4] N. Rizwan Ali, A. Muhammad, R. Abdul, R. Ateeq Ur, L. Woong-Kee and P. Anand, "Deep Learning-Based Drivers Emotion Classification System in Time Series Data for Remote Applications," *mdpi*, 2020.
- [5] S. Mallick, "Deep Learning with OpenCV DNN Module: A Definitive Guide," [Online]. Available: <https://learnopencv.com/deep-learning-with-opencvs-dnn-module-a-definitive-guide/#what-is-opencv-dnn-module>.
- [6] S. Ilkin Serengil, "deepface," [Online]. Available: <https://github.com/serengil/deepface>.
- [7] "Cuda Toolkit," Nvidia, [Online]. Available: <https://developer.nvidia.com/cuda-toolkit>.
- [8] J. Anusha and P. P, "A comparison between CPU and GPU for image classification using Convolutional Neural Networks," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2015.
- [9] R. Sivalingam, A. Cherian, J. Fasching, N. Walczak, N. Bird, V. Morellas, B. Murpyh, K. Cullen, K. Lim, G. Sapiro and N. Papanikolopoulos, "A Multi-Sensor Visual Tracking System for Behavior Monitoring of At-Risk Children," in *IEEE International Conference on Robotics and Automation*, Saint Paul, 2012.
- [10] V. Agarwal, "Face Detection Models: Which to Use and Why?," [Online]. Available: <https://towardsdatascience.com/face-detection-models-which-to-use-and-why-d263e82c302c>.
- [11] "Adaptive histogram equalization," Wikipedia, [Online]. Available: [https://en.wikipedia.org/wiki/Adaptive\\_histogram\\_equalization#Contrast\\_Limited\\_AHE](https://en.wikipedia.org/wiki/Adaptive_histogram_equalization#Contrast_Limited_AHE).
- [12] S. Ilkin Serengil, "https://www.youtube.com," 24 6 2020. [Online]. Available: [https://www.youtube.com/watch?v=i\\_MOwvhbLdl](https://www.youtube.com/watch?v=i_MOwvhbLdl).
- [13] S. Ilkin Serengil, "RetinaFace," [Online]. Available: <https://github.com/serengil/retinaface>.

Implementation of an emotion detection algorithm for therapy applications in children

[14] K. Oswald Seidler and K. Vogelgesang, "XAMPP," BitRock, [Online]. Available: <https://www.apachefriends.org/es/index.html>.

[15] "Selenium," [Online]. Available: <https://www.selenium.dev/>.