

Third Eye

Project Engineering

Year 4

Sean Kelly

Bachelor of Engineering (Honours) in Software and Electronic Engineering

Galway-Mayo Institute of Technology

2020/2021

Project Graphic (Optional)

**Declaration**

This project is presented in partial fulfilment of the requirements for the degree of Bachelor of Engineering (Honours) in Software and Electronic Engineering at Galway-Mayo Institute of Technology.

This project is my own work, except where otherwise accredited. Where the work of others has been used or incorporated during this project, this is acknowledged and referenced.

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

**Acknowledgements**

Use this section to acknowledge anyone, if you wish to, who might have helped during your project.

**Table of Contents**

[Summary 6](#_Toc72149749)

[Poster 7](#_Toc72149750)

[Introduction 8](#_Toc72149751)

[Project Architecture 9](#_Toc72149752)

[Components 10](#_Toc72149753)

[Fingerprint Module 10](#_Toc72149754)

[1 optical: 10](#_Toc72149755)

[2Capacitive 11](#_Toc72149756)

[Scanner I used 12](#_Toc72149757)

[How fingerprints are stored and enrolled 13](#_Toc72149758)

[Camera 14](#_Toc72149759)

[Software 16](#_Toc72149760)

[Open CV 16](#_Toc72149761)

[My project 17](#_Toc72149762)

[Ethics 18](#_Toc72149763)

[Conclusion 19](#_Toc72149764)

[Appendix 20](#_Toc72149765)

[References 21](#_Toc72149766)

# Summary

The aim of my project is to create a security system to help house and property owners secure their homes as many houses are broken into around the world at any given time.

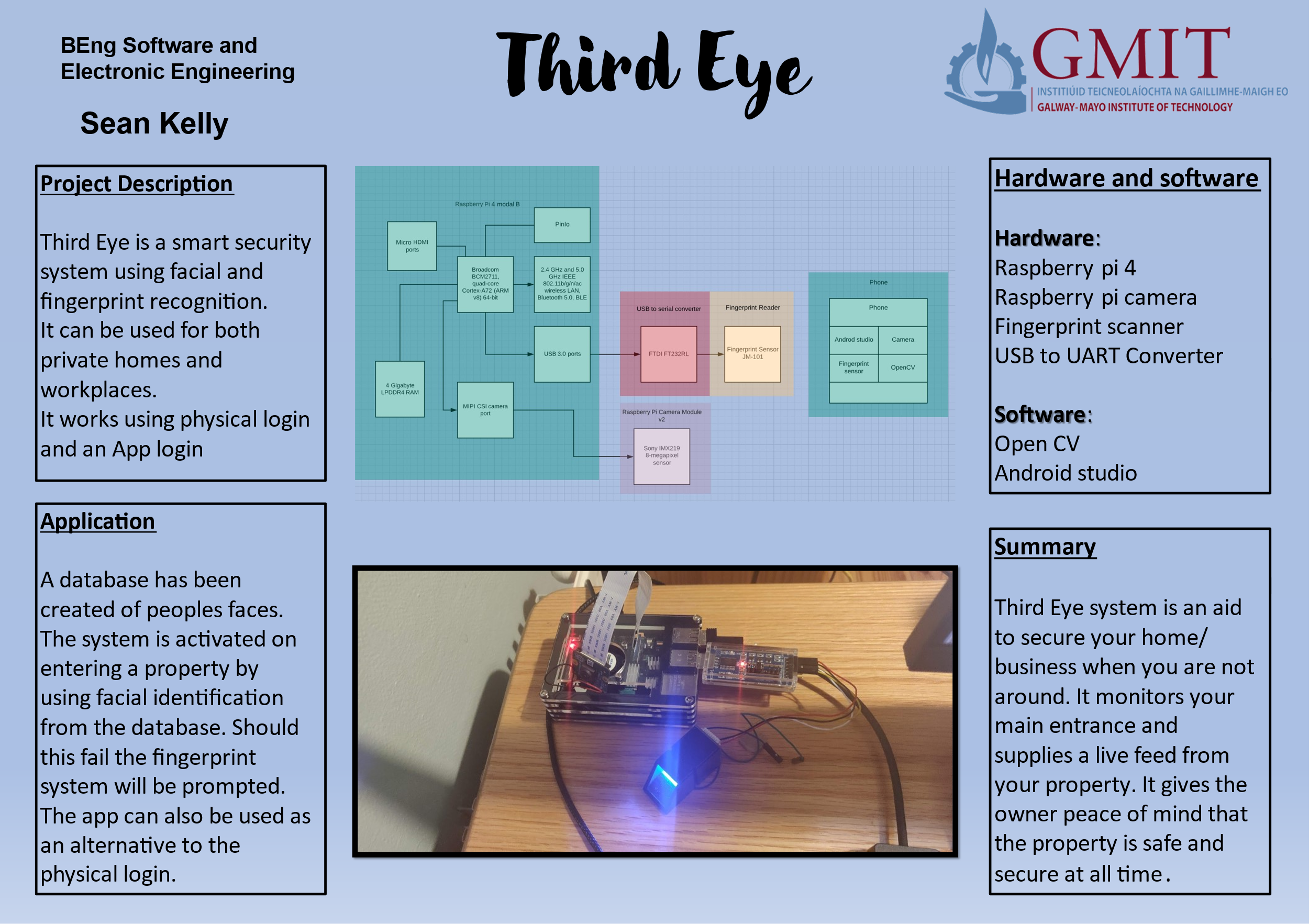
In my project I used the Raspberry Pi with a camera to recognize a person face along with a fingerprint scanner. I attempted to create an android app but was unsuccessful in doing this.

My system is based on letting people into a property by using facial and fingerprint recognition. In my project I used Raspberry Pi running open CV and a camera as well as a fingerprint scanner. When starting out I first got the fingerprint scanner to pick up a fingerprint of a person which wanted to gain access to the house. I used open CV as I had used it before and felt happy using it.

I was able to accomplish the synchronization between the fingerprint and the facial recognition in my work.

The main conclusion is the aim of project was met but it can be further developed and enhanced with a longer time scale.

# Poster



# Introduction

I decided on this project because two properties in Ireland are broken into every hour as the owners do not have a security system installed.

The main components used was a camera and a fingerprint reader, which was controlled using the Raspberry Pi. The camera was used at it can pick up facial features and movements in an area rather than a motion sensor. There is a live feed, that runs and looks for the face of the user and record a picture of it also incorporates fingerprint reader.

In this project I used several hardware and software frameworks. The main feature of my project is that the user is registered on the system beforehand. When the person arrives to the property, they can use either the facial recognition or the fingerprint reader to gain entry.

# Project Architecture

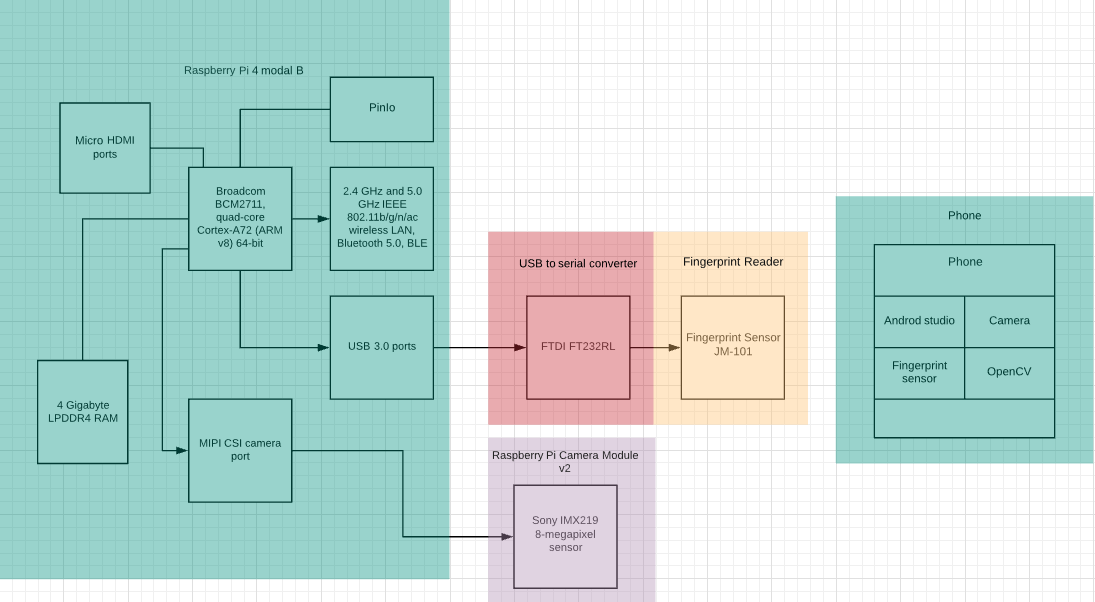


Figure 4‑1 Architecture Diagram

As the diagram above illustrates the Raspberry Pi is the primary controller together with the camera and fingerprint reader. I interfaced the fingerprint reader over the USB port using the USB to serial converter. The raspberry Pi bord came with a Mobile Industry Processor Interface (MIPI) and a Camera Serial Interface type 2 (SCI-2).

The enable the connection of the Raspberry Pi camera to the main Broadcom processor on the board (BCM2835). The CSI connector consists of two smaller interfaces. The first interface is to transfer data and clock signal from the camera to the processor in one direction only. The second interface consists of SCL/SDA lines which is a link which runs both ways.

Development platform and tools

Machine learning development system was developed only on the Raspberry Pi using open CV and Python. The app was written in Python on Android studio.

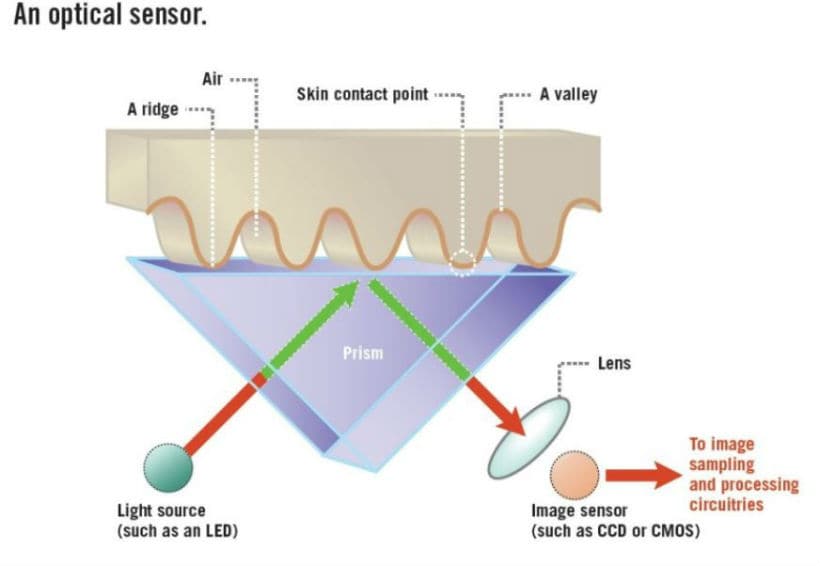
# Components

This section goes into detail and interface related to the two main hardware components of the system, mainly the fingerprint sensor and camera module.

## Fingerprint Module

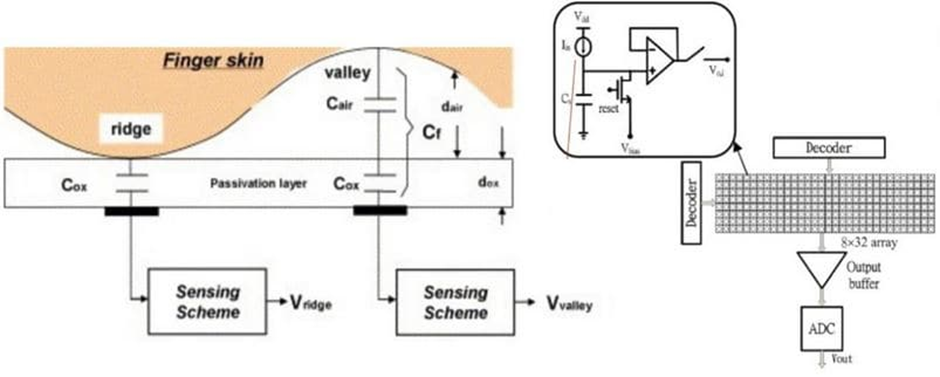
### 1 optical:

optical works by shining a light on a finger, it then takes a digital picture of the ridges and valleys of the finger and saves the image as binary.



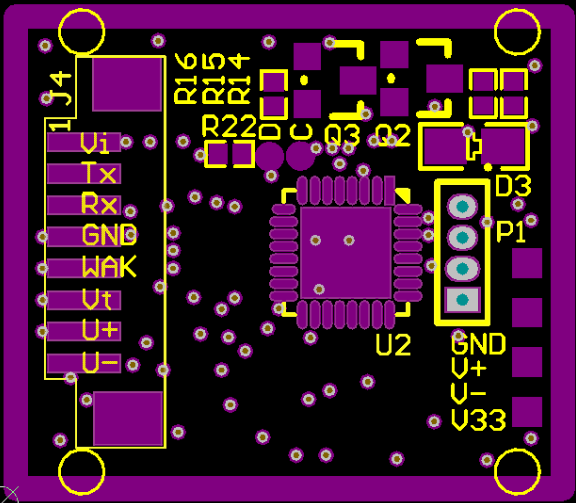
### Capacitive

Capacitive works by using an electrical field to scan the finger and the ridges get picked up, but the valleys do not. It uses a capacitor to capture the data. When finished it uses an analogue to digitally convert into binary.



### Scanner I used

The JM-101 fingerprint module is an optical type of sensor. Optical are more widely available and a cheaper scanner in comparison to the capacitive. As well as being a cheaper version it can hold over 162 individual fingerprints within the sensor. Also, it is small, has a very simple interface and can be used accurately by both wet and dry hands.

The fingerprint module connects to my project through serial communication. Serial communication is used for short distance communication mainly for embedded systems.



In addition, I used a USB to UART converter to send the data from the fingerprint reader to the Raspberry Pi as this way was physically more robust. The USB to UART converter presents itself as a serial port to the Raspberry Pi and behaves similarly.



Figure VCC= red, GND=black, TXD = White RXD = yellow

### How fingerprints are stored and enrolled

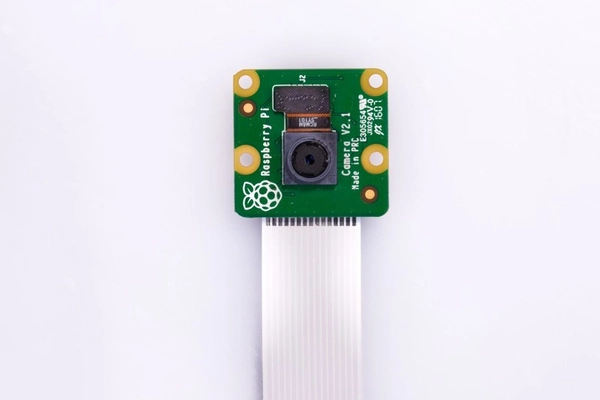
A fingerprint is stored in a binary format by the ridges representing 1 and valleys at 0 in the database.The database is located on the fingerprint reader to keep it on a closed system. It can hold up to 163 fingerprints.

A fingerprint is enrolled by putting the finger on the scanner two times, the first time is to take a picture of the finger and the second is to compare the second image to the first by ridges and valleys of the finger on the data base.

“””add more””””

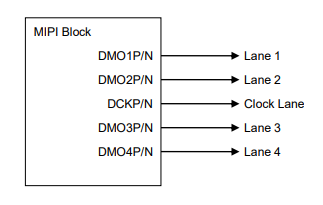
## Camera

The camera that I used is the right Raspberry Pi Camera Module v2. The reason I chose it was because it is compatible with the Raspberry Pi and uses a ribbon cable connected to the Raspberry Pi. Its high-definition camera uses a Sony IMX219 8-megapixel sensor. It’s very high definition and captures a very clear picture which is essential for my system. The one thing I liked about the camera was that it was very easy to use and to set up as well as a large variety of libraries to use. When I initially was researching for this project, I was between two different models, however time and time again the model v2 came out as being the best for my project.



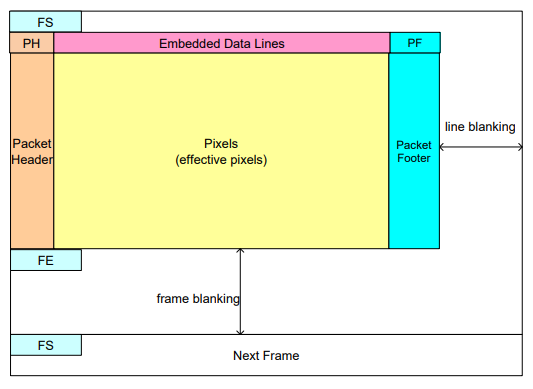
The CSI interface and MIPI bus

Data for the camera is transferred over a MIPI bus using the CSI-2 interface standard. A high-level block diagram of the MIPI bus is shown below.



In the CSI-2 interface, one bit of data is transmitted by a pair of differential signals. In the transmitter of CSI-2 interface, differential digital signals of data or clock are converted to differential current signals. Outputs of data and clock come from CSI-2 output pins (DMO1P/DMO1N, DMO2P/DMO2N, DCKP/DCKN). A pair of DMO1P/DMO1N is called Lane1 data and a pair of DMO2P/DMO2N is called Lane2 data. Also, clock signals come from CSI-2 output pins, DCKP/DCKN. Maximum output data rate is 912 Mbps/lane. In addition, the CSI-2 standard allows for a backchannel communications which is used in the control registers contained within the sensor itself. The registers control things like exposure and auto white balance.

Image data from the sensor is delivered serially over the MIPI bus and is decoded at the receiver side. The frame structure of a CSI frame is shown below



For our application, the effective pixel area is configured to be 1080 x 720, and this data is delivered to the Raspberry PI in YUV 420 format, in order the have reduce bandwidth. The software then captures the images and stores them to database contained on the Raspberry Pi.

# Software

The software that I used was OpenCV as well as Android studio. The OpenCV was used for the face recognition as they have a large selection of Algeria to choose from to train a bot to recognize a face.

## Open CV



OpenCV is a computer visual machine learning library that has over 2500 different libraries that hold algorithms that help detect different visuals namely faces and movements. I obviously focused, mainly within the facial and human action libraries and found them very useful and user friendly. OpenCV is hugely popular and thus has lots if helpful and useful tips and information on how to properly utilize and get the best out of it. I used OpenCV mainly through Python and understand it can be used also through Java and C++.

# My project

My code is broken into 3 threads.

* Master controller
* Start video capture.
* Start fingerprint searcher.

## Master controller

This code is the main code in my project.

## Video capture

## Fingerprint searcher

# Ethics

Include a short section on ethical considerations in your project or in the field of study of your project.

Ethics would be considered a big concern for my project as I am gathering the users faces and fingerprints. As this data is held on the system, but if comprised in any way this would be a problem for the users. If this happens the user’s fingerprint could be used to log into their bank account without their knowledge.

“”””come back”””

# Conclusion

Write a short conclusion. What is the outcome of the project? Perhaps you have a product prototype, or some results, or a demonstratable system.

Do not use your conclusion to tell the reader what you might have done if you had more time, but keep it focussed on what you actually have done. You can mention future opportunities for further development of the work, but keep this part short.

# Appendix

# References

|  |  |
| --- | --- |
| [1] | H. Kinsley, "Reinforcement Learning," PythonProgramming, [Online]. Available: https://pythonprogramming.net/q-learning-reinforcement-learning-python-tutorial/. [Accessed 02 02 2021]. |
| [2] | [Online]. |
| [3] | MakeSigns, "Scientic Posters Tutorial," [Online]. Available: https://www.makesigns.com/tutorials/scientific-poster-parts.aspx. [Accessed 09 02 2021]. |
| [4] | Arduino. [Online]. Available: https://www.arduino.cc/. [Accessed 09 02 2021]. |
|  |  |
|  |  |

[x] J. C. Russ and F. Brent Neal. *The Image Processing Handbook*. 7th ed. Boca Raton FL: CRC Press, 2017.

[x] S. Lippman, J. Lajoie and B.E. Moo. “Classes” in *C++ Primer*. 5th ed. Massachusetts: Addison Wesley, 2013.

[x] IEEE Signal Processing Society. “Signal Processing for 5G”, *YouTube*, 2019. [Online]. Available: <https://www.youtube.com/watch?v=uca6X4Ykcmg&list=PLcZOnmyqlalacL9YqkhyufLQGIW_C78Os&index=6>. Accessed: Feb 2, 2021.

[x] Digilent. “Basys 3 Reference Manual”, *Digilent Reference*. [Online]. Available: <https://reference.digilentinc.com/basys3/refmanual>. Accessed: Feb 2, 2021.

[x] P. J. Ashenden. *Digital Design (Verilog): An Embedded Systems Approach Using Verilog*. Burlington: Morgan Kaufmann, 2007.

[x] M. Lynch. “Discrete Fourier Transform (DFT)”, Lecture, Digital Signal Processing, Galway-Mayo Institute of Technology, Galway, 2020.

[x] MRC Centre for Global infectious Disease Analysis. (2021). *Covid-Sim*. [Source Code]. Available: <https://github.com/mrc-ide/covid-sim>. Accessed: Feb 2, 2021.

[x] OpenCV. “Face Detection Using Haar Cascades”, *OpenCV-Python Tutorials*. [Online]. Available: <https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_objdetect/py_face_detection/py_face_detection.html#face-detection>. Accessed: Feb 2, 2021.