**University of Macau**

**Faculty of Science and Technology**

澳門大學
UNIVERSIDADE DE MACAU
UNIVERSITY OF MACAU

# The Electrocardiogram (ECG) Based Biometric Identification

***by***

**Oscar, KUAN KA MENG, Student No: DB526258**

Final Project Report submitted in partial fulfillment
of the requirements of the Degree of
Bachelor of Science in Computer Science

Project Supervisor

Prof. Liming ZHANG

02 June 2020

# DECLARATION

I sincerely declare that:

1. I and my teammates are the sole authors of this report,
2. All the information contained in this report is certain and correct to the best of my knowledge,
3. I declare that the thesis here submitted is original except for the source materials explicitly acknowledged and that this thesis or parts of this thesis have not been previously submitted for the same degree or for a different degree, and
4. I also acknowledge that I am aware of the Rules on Handling Student Academic Dishonesty and the Regulations of the Student Discipline of the University of Macau.

Signature : ___________________________

Name : Oscar, KUAN KA MENG

Student No. : DB526258

Date : 02 June 2020

# ACKNOWLEDGEMENTS

The author would like to express his utmost gratitude to UM for providing the opportunity to carry out a project as a partial fulfillment of the requirement for the degree of Bachelor of Science.

Throughout this project, the author was very fortunate to receive the guidance and encouragement from his supervisor…

# ABSTRACT

This document serves the final year project of the electrocardiogram (ECG) biometric identification with deep learning living personal identification as the report.

Motivation:

Traditional identification uses account and password, electronic card certificate, which is easy to be stolen and posing. Biometrics is far superior to traditional identification. First, it has higher security. It will not be lost. It is unique, cannot be impersonated, and has better anti-counterfeiting performance. Second, it's convenient. Biometrics do not need to carry the extra things. Third, it is confidential and cannot be copied. Fourth, biological characteristics are stable and will not change. These advantages make it more powerful than traditional identification.

Therefore, we are thinking to build a low-cost biometrics identification method to give a new choice to the identification technical area.

Background:

Biometric identification mainly uses the inherent physiological and behavioural characteristics of the human body to identify individuals. It includes fingerprint, face, iris, handwriting, sound, gait etc. This identification technique combining computer with high-tech methods such as optics, acoustics, biosensors and biometric principles. We introduce the ECG biometric identification because it is new, and keep the tester is alive when it is testing.

Goal:

We'll using Arduino UNO, AD8232 Heartbeat / ECG Collecting Shield as the hardware. Python 3 with Jupyter Notebook or Google Colaboratory as the software and platform. SVM with scikit-learn is the deep learning algorithm to implement the biometric personal identification system. The final goal is implementing the GUI applicaiton by Tkinter with Python 3.

Major function:

- User personal identification with hardware in low-cost and with reasonable accuracy.

Major result:

We collected 3 personal user ECG signal dataset. The result and accuracy is quite reasonable and acceptable.

Major work distribution:

In this project, we are in a group of two, we separate and distribute the work to all members, for:

Oscar, Kuan Ka Meng (DB526258):

- Deep learning training model coding and training
- ECG signal collecting implementation
- Graphical user interface (GUI) implementation.

Elaine, Guo Yan Jia (DB301736):

- Papers and Literatures Viewing
- Deep learning training model coding and training
- Tuning and test the training set

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# CHAPTER 1. INTRODUCTION

Biometric identification mainly uses the inherent physiological and behavioural characteristics of the human body to identify individuals. It includes fingerprint, face, iris, handwriting, sound, gait etc. This identification technique combining computer with high-tech methods such as optics, acoustics, biosensors and biometric principles.

Like fingerprints and irises, everyone's heartbeat is unique. An electrocardiogram (hereinafter referred to as ECG) is a test that records the potentials produced by the atria and ventricles during contraction and relaxation.

With so many biometric technologies, why do we need ECG recognition? The reason for that is different biometric identification technologies have obvious differences in many aspects such as security, accuracy, stability, convenience, recognition speed, cost, etc. Thus, they have different advantages and disadvantages in different application fields. Fingerprint recognition is the most widely used in biometrics currently.

In addition, ECG recognition **can be guaranteed** the tester have heartbeat, which means **the tester alive**.

Currently, fingerprint recognition is dominant in biometrics.

The identification technology based on electrocardiographic signals not only meets the stability, uniqueness and convenience required for fingerprint identification, but also has unique anti-counterfeiting properties. ECG will be affected by factors such as the physical fitness state and mental state of the human body, but these factors will only cause the scaling and deformation of the ECG waveform and will not change its structure, so each person's ECG is still unique.

# CHAPTER 2. LITERATURE SURVEY AND RELATED WORK

Biometric identification mainly uses the inherent physiological and behavioural characteristics of the human body to identify individuals. It includes fingerprint, face, iris, handwriting, sound, gait etc. This identification technique combining computer with high-tech methods such as optics, acoustics, biosensors and biometric principles.

Traditional identification uses account and password, electronic card certificate, which is easy to be stolen and posing. Biometrics is far superior to traditional identification. First, it has higher security. It will not be lost. It is unique, cannot be impersonated, and has better anti-counterfeiting performance. Second, it's convenient. Biometrics do not need to carry the extra things. Third, it is confidential and cannot be copied. Fourth, biological characteristics are stable and will not change. These advantages make it more powerful than traditional identification.

## 2.1 Fingerprint recognition

Biometric technology is used in transfer and withdrawal, payment settlement, remote insurance business and other fields.


***Figure 1: Fingerprints are lines formed by the uneven skin on the fingertips of human fingers. (Capture by [1])***

The lines that protrude on the epidermis of each person's fingerprint, the starting point, ending point, joining point, and bifurcation point of the line are all different.

They are unique and unchanged for life. Fingerprint recognition technology generally uses the overall characteristics of the fingerprint to classify, and then uses local characteristics such as location and orientation to identify the user. In recent years, fingerprint recognition has been widely used, such as mobile phone unlocking, electronic payment, security, etc. It's fast to scan, read and use fingerprints. Also, the device for fingerprint collection is inexpensive. This provides an economic basis for the popularity of fingerprint recognition. However, the fingerprints of relatives are similar. If the accuracy of the algorithm is not high, it may lead to recognition errors. In addition, fingerprints left behind are easily stolen and copied. It may cause some security problem.

## 2.2 Face recognition

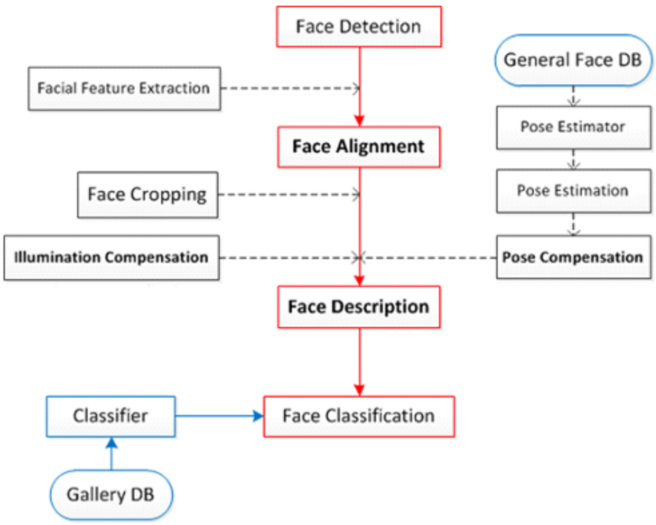



*Figure 2: The flowing diagram of face detection and face recognition (Capture by [2])*

Face recognition refers to how to recognize and understand a face. It is based on the facial feature information of a person, using a video camera or a camera to collect an image or video stream containing the face. Face recognition is one the most popular computer research technology now.

Face recognition problems are mainly divided into face verification and face recognition.

Face verification is a one-to-one comparison. It determines if the two pictures are the same person. Usually applied in face unlock of mobile phone. Face recognition is a one-to-many comparison. Determine whether the person the system is currently seeing is one of the databases. This technology mainly used for suspect tracking, residential access control etc.

Face recognition requires the system to store a large number of different face and identity information in advance, in order to compare the face with the database at runtime.

However, it is difficult for the system to perform face recognition when the light is dim, or the face is obscured, and the side faces. This can lead to failure of face recognition. It has its limitations. In addition, the appearance of the face changes with age.

Face recognition is a widely used biometric approach. Face recognition technology has developed rapidly in recent years and it is more direct, user friendly and convenient compared to other methods. But face recognition systems are vulnerable to spoof attacks made by non-real faces. It is an easy way to spoof face recognition systems by facial pictures such as portrait photographs. A secure system needs Liveness detection in order to guard against such spoofing. In this work, face liveness detection approaches are categorized based on the various type's techniques used for liveness detection. This categorization helps understanding different spoof attacks scenarios and their relation to the developed solutions. A review of the latest works regarding face liveness detection works is presented. The main aim is to provide a simple path for the future development of novel and more secured face liveness detection approach.

Here, liveness detection approaches are categorized based on the type of liveness indicator used to assist the liveness detection of faces. Three main types of indicators were mainly used: motion, texture and life sign.

The most common problems that have been observed in case of many liveness detection techniques are the effects of illumination change, effects of amplified noise on images which damages the texture information. For blinking and movement of eyes-based liveness detection methods, eyes glasses which causes reflection must be considered for future development of liveness detection solutions. Furthermore, the datasets, which play an important role in the performance of liveness detection solutions, must be informative and diverse that mimics the expected application scenarios. Non-interactive video sequences must include interactive sequences where the users perform certain tasks. Future attack datasets must consider attacks like 3D sculpture faces and improved texture information. Our main aim is to give a clear pathway for future development of more secured, user friendly and efficient approaches for face liveness detection.

"An Identity Authentication Method Combining Liveness Detection and Face Recognition" also proposes a liveness detection approach based on infrared radiation (IR) images acquired using a Kinect camera.

IR images from live faces are used as positive samples, while IR images from photos or videos are used as negative samples. The samples above are input into the convolutional neural network (CNN) for training to distinguish live faces and spoof attacks.

After liveness detection, an improved FaceNet will continue to recognize a face and provide the corresponding ID or UNKNOWN output for accurate identity authentication.

The rest of the paper is organized as follows. Section 2 briefly reviews the related works and recent liveness detection methods. Section 3 presents a framework that combines liveness detection and face recognition, and then the proposed liveness detection method based on IR image features and an improved FaceNet model, called IFaceNet, are described.

## 2.3 Iris recognition

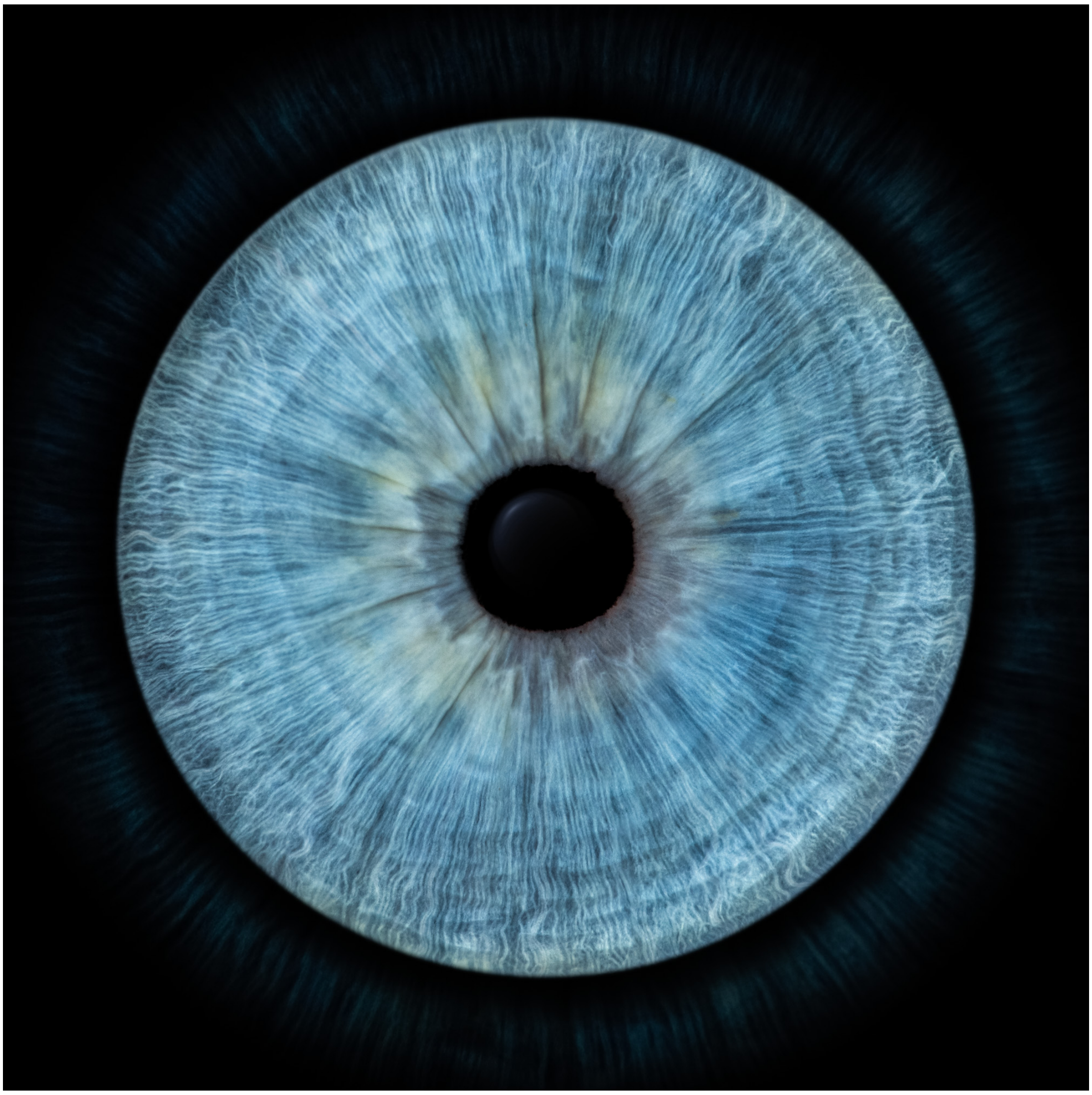


***Figure 3: Iris (Capture by [3])***

Iris recognition technology is based on the identification of the iris in the eye. The iris is a coloured circle that surrounds the pupil. It contains many interlaced spots, filaments, crowns, stripes, crypts and other detailed features. The iris determines the colour of our eyes, and the iris on each of us is unique. Even identical twins have different irises. So, it can used for identification.

Compared with fingerprint recognition and face recognition in biometrics, iris recognition starts late. However, iris recognition is more accurate, and the rate of misrecognition can be as low as one part per million. It is currently the most accurate biometric identification technology.

However, the equipment cost of iris recognition is high and cannot be widely promoted.

In addition to fingerprint recognition, iris recognition, and face recognition in the field of biometrics, ECG recognition has now become a new research direction.

## 2.4 Voiceprint recognition

*Figure 4: Voiceprint recognition (Capture by [4])*

Voiceprint recognition, also called speaker recognition or voice authentication, it analyzes a person's voice to verify their identity. Voiceprint refers to the spectrum of sound waves that carry speech information in human speech. Like fingerprints, voiceprints have unique biological characteristics. It is not only specific but also relatively stable. Everyone has a unique voiceprint. Although the computer can't judge a person's identity by one word at present, through a large amount of training speech data, the system can learn to intelligently learn your voiceprint. It can determine your identity when you say several words.

This identification technology can be divided into two types of content-related and content-independent technologies. "Content-related" refers to the system's assumption that users only say that the system prompts the content or a small range of permitted content, and "content-agnostic" does not limit what users say content.

Compared to other forms of authentication, speech recognition has several key advantages. Today's mobile phones are basically equipped with a microphone, and identity authentication can be performed on the mobile phone. The microphone can be integrated into other devices, such as cars and home appliances. It is cost-effective. It is convenient and familiar for most users.

Some disadvantages are as follow, it is not as accurate as other biometrics (such as facial recognition). The background noise will affect the quality of the sample and then the matching performance. So, it is not ideal in noisy or public places.

## 2.5 Technical issues use to implement electrocardiogram (ECG) recognition

We use:

- Hardware:

  Arduino UNO (with USB cable)

  AD8232 Heart Beat / ECG Collecting Shield (with sensors / electrode)

- Software Platform:

  Python 3 with Jupyter Notebook or Google Colaboratory

- Algorithm:

  SVM with scikit-learn

- Graphical User Interface (GUI) appliation:

  Tkinter with Python 3

# CHAPTER 3. OVERALL SYSTEM DESIGN

## 3.1 Abstract system workflow

In this project, we needed to use the ECG Collecting Kit (Arduino UNO + AD8232 ECG Collecting Shield) to get the user's ECG signal. After, we process the signals and training the model. In this chapter, we are going to tell you what we needed to do in the overall views.

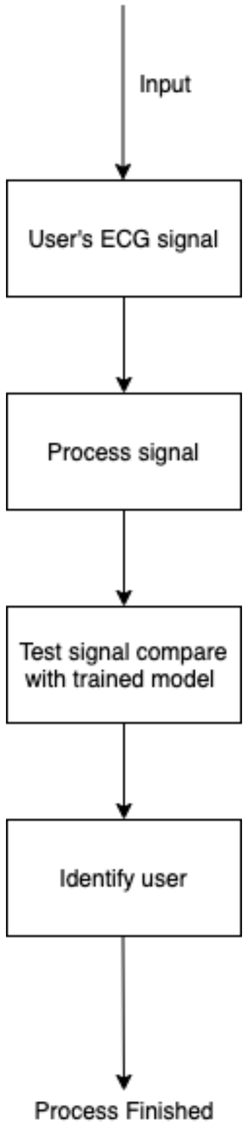



***Figure 5: Abstract system workflow diagram***

The abstract system workflow of our project as shown as Figure 5. First of all, we need to set up the ECG Collecting Kit and paste the sensor on the user's body, the kit will detect and collect the ECG signal of the users. Then, the system processes the signal by convert and trim the signal to the appropriate format. The converted signal will be tested and compare with the trained deep learning model. Finally, we can identify the user by its comparing accuracy.

## 3.2 Overall system workflow

After showing the abstract workflow of our system, we introduce the overall workflow. First, our system is an identification system, we need the user to connect with the ECG Collecting Kit to make sure the tester alive. Therefore, we will break down the overall workflow into two parts: 1) How the user can run through the identification process with our system. 2) What we need to do behind when a new user appears.

### 3.2.1 User Routine of using the system

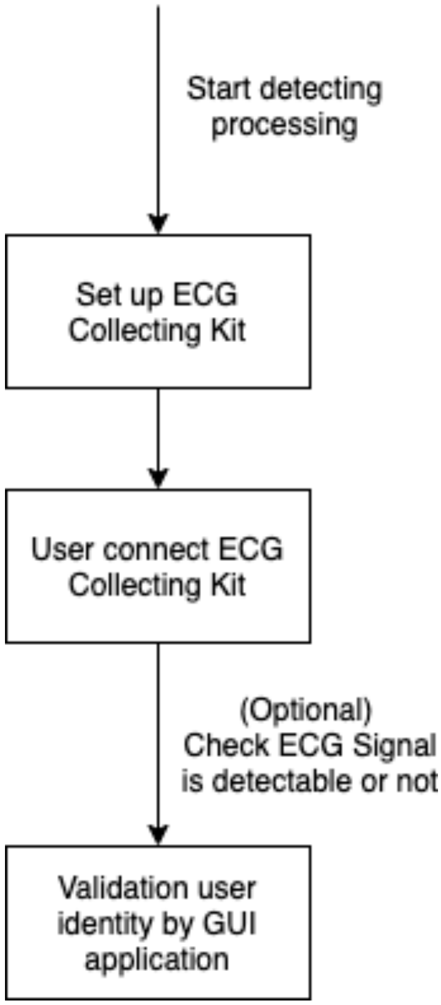
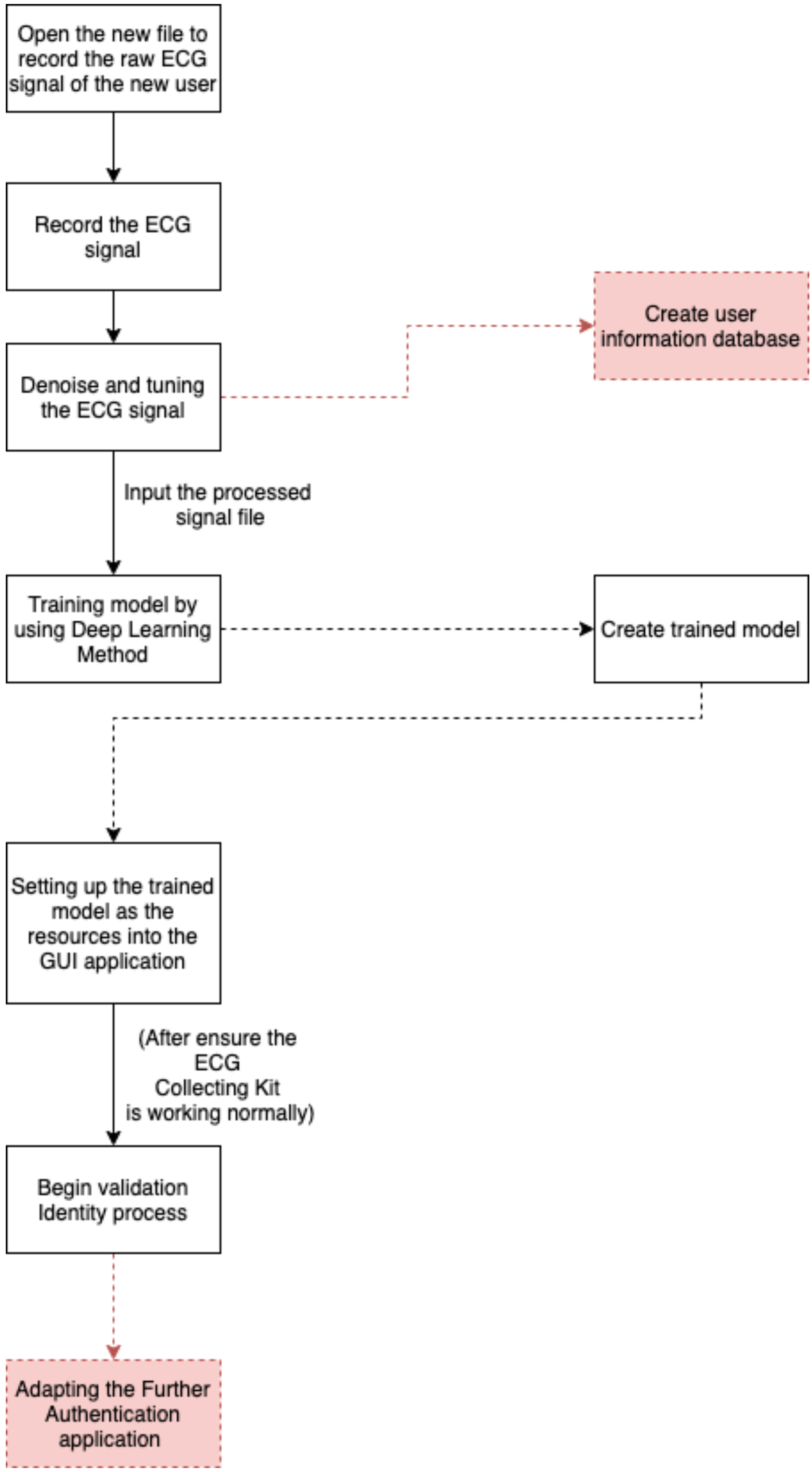



***Figure 6: The routine of the user***

In Figure 6, it shows the easy routine of the user when they want to use our ECG biometric identification system. After finishing the set up and collect the ECG Collecting Kit, then it can the validation identification process can be started. In optional, before the validation process begin, we can check the ECG signal is detectable or not, and the shape of ECG signal is normal or not.

### 3.2.2 Our overall workflow when a new user appears




(Note: Dotted line for future implementation)

*Figure 7: The workflow when a new user arrive*

In Figure 7, the things we needed to do when a new user arrive, first our developer or system administrator needed to open a new file to record the raw ECG signal of the new user. Then, we can record the ECG signal by ECG Collecting Kit. By the practice of the collecting experience, there are always some noise will be collected during the recording step. Therefore, we needed a step to denoise and tune the raw ECG signal. As in the figure shown, for the future implementation, creating the user information database is necessary.

After the signal is finely tuned, we put all the signal of all users into the deep learning training program to create the trained model after the deep learning process. During the learning process, we'll monitoring the F1-score / accuracy indicator to make sure the training is relatively accurate.

Finally, after the validation identity process, it will show the selected user is clear the identity test. Also, our system can be further implement an API to connect with the other system or software.

To be aware that, because of the time and resources are limited and the COVID-19 virus outbreak, some ideas we didn't have time to implement, for future implementation and work, we marked as dotted line in Figure 7, such as building the user information database, and develop the API after the identification process, and have the more powerful hardware resources to make the training time less and the accuracy higher.

## 3.3 Detail system workflow

After the overall views of our system, we are going to explain what we need to do in each section, which is corresponding to the overall workflow.

### 3.3.1 ECG Signal collecting workflow

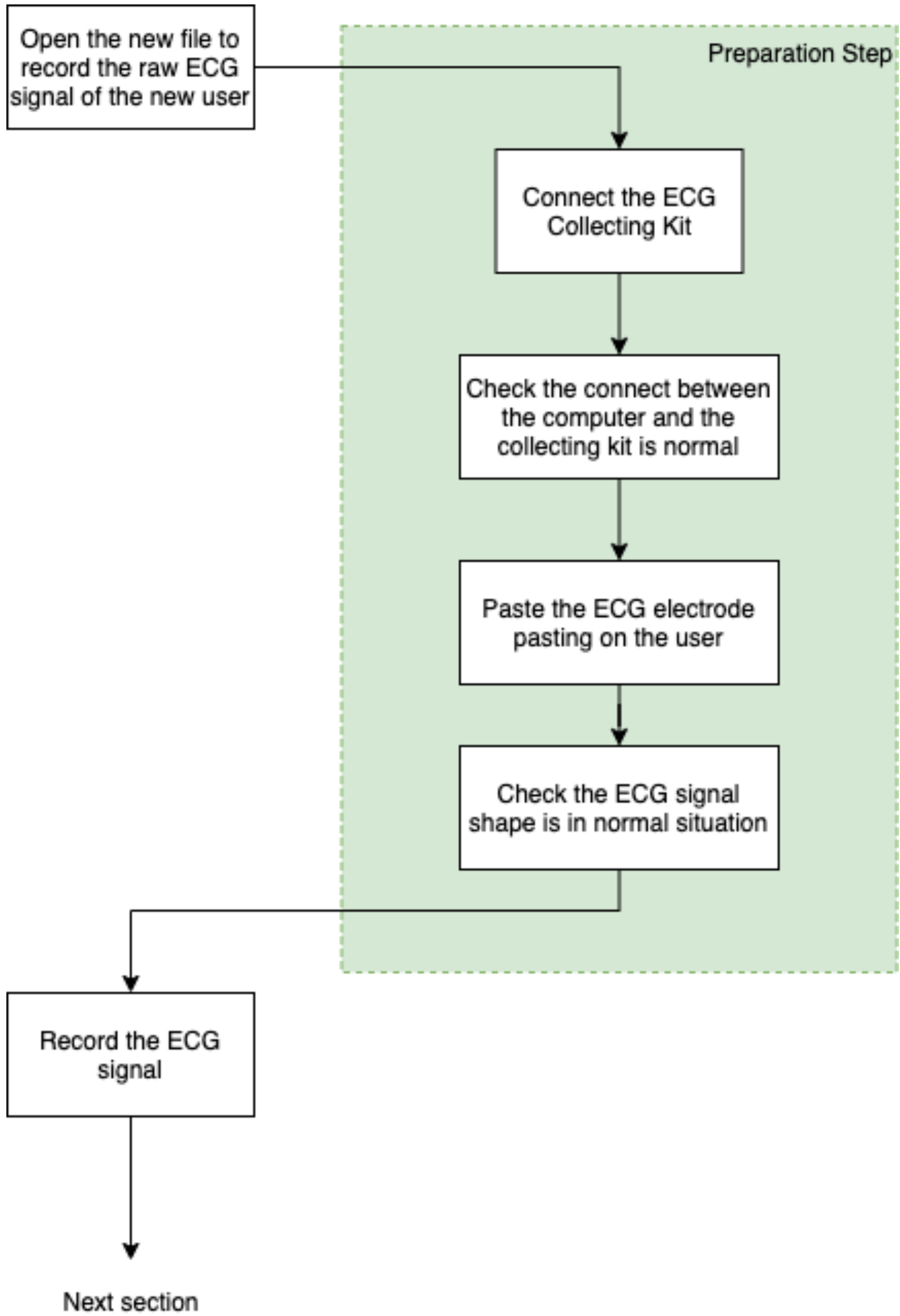



***Figure 8: The workflow of record the ECG signal***

From the Figure 8, we can see that this is the preparation step of the record the ECG signal. It is done by the system developer or administrator manually.

For the preparing step of check the ECG signal shape is in normal situation, we can use the advantage by using Arduino IDE monitor to see the shape is normal or not. We will have more discussion of the implementation part of all function and work. Therefore, let us to see the next part.

### 3.3.2 Connection with ECG Collecting Kit workflow

This is an important part of our system. If the connect model cannot work normally, everything else are useless. Therefore, we say how to connect the ECG Collecting Kit is very essential. For the ECG Collecting Kit, as we mentioned before, it components with Arduino UNO, and AD8232 ECG Collecting Shield. The following is the workflow to show how to connect the ECG Collecting Kit.

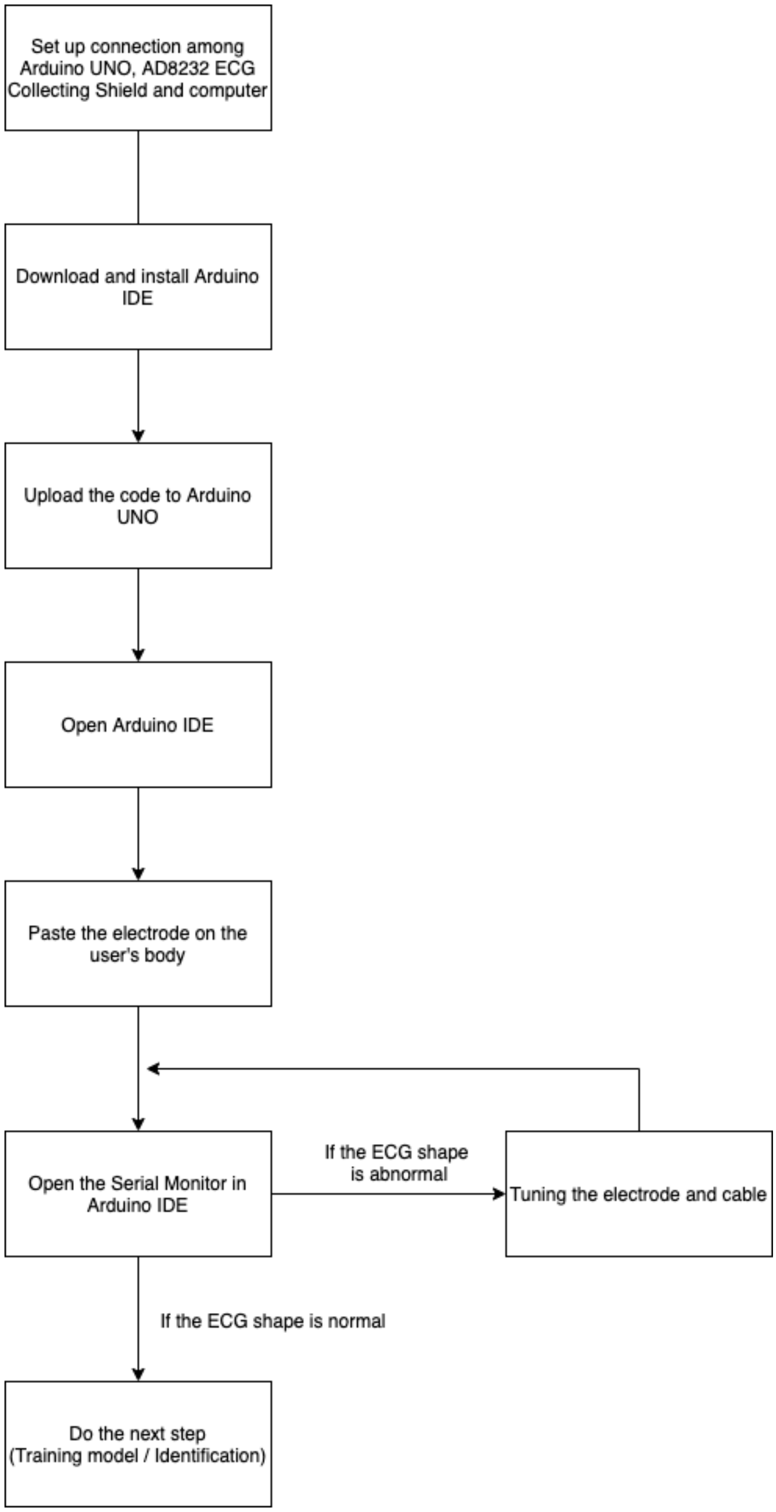



***Figure 9: The workflow of connection with ECG Collecting Kit***

In Figure 9, we can see we needed to download the Arduino IDE. After the connection between ECG Collecting Kit and computer is working, and also uploaded the code to Arduino UNO, we can paste the electrode on the user's body. In order to implement the ECG collecting function, the codes will discuss in the following chapter. The next step is the training model process.

### 3.3.3 Training model workflow

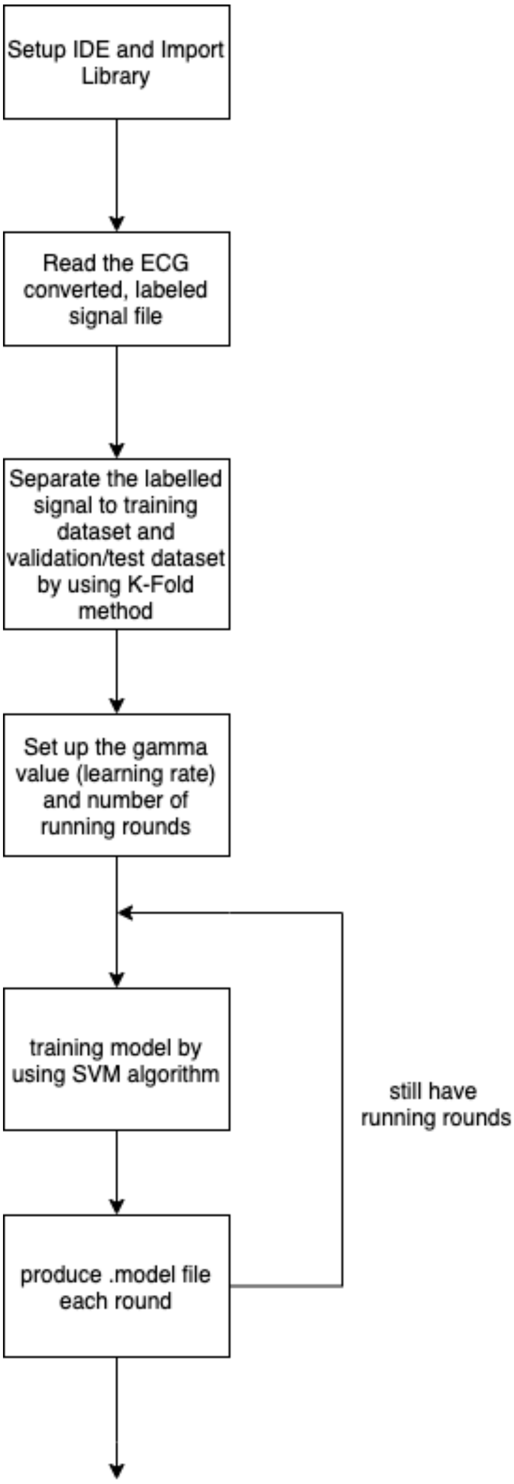



*Figure 10: The workflow of training model*

In Figure 10, we can see the workflow of training model, first we set up the IDE and important the library. We used the Jupyter Notebook and Google Colaboratory as the IDE and using Python 3 as the main programming language. Then, the ECG converted signal process is implemented by developer and system administrator. We will discuss the detailed implementation in the following chapters.

As the following steps, it is implemented by Python 3 code. We used the K-Fold method to separate the labelled signal records into training dataset and validation/testing dataset. Then, we set the gamma, also called learning rate and number of training rounds. Next, we run the training model by using SVM algorithm. We will also discuss the detailed information in the following chapters.

In the final part of the training model section, each training round will produce a .model file. This .model file we will use it as the key of biometric identification part.

### 3.3.4 Identification process workflow

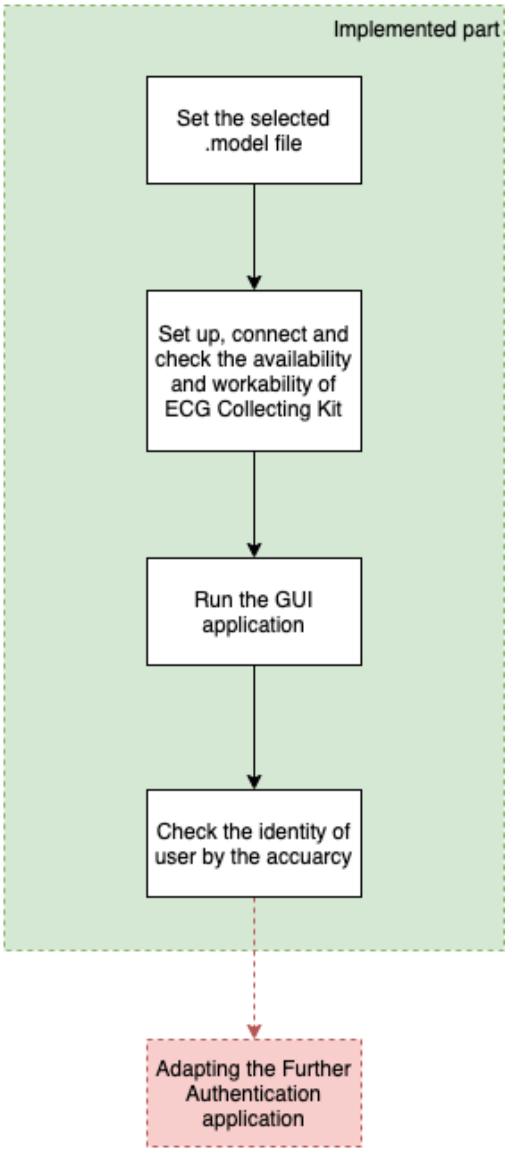



*Figure 11: The workflow of identification process*

In Figure 11, we can notice about the implemented part. As the previous section mentioned, we needed to set up the selected .model file into the system setting. Then, the connecting ECG Collecting Kit is required and the step is same as before.

The key part of this section is running the GUI application, it is implemented by Python 3 with Tkinter library. We will check the accuracy of the ECG signal segment to determine the identity of user. The detailed part will discuss in the following chapters.

### 3.3.5 Interface workflow

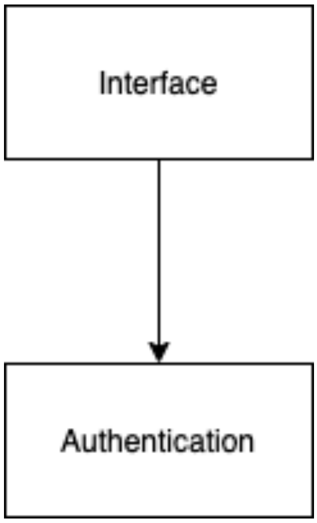



*Figure 12: The interface of our identification system*

The last thing is the interface of our identification system. Our interface is pretty simple, as shown in Figure 12. The interface contains only one authentication part. It likes the login page, but using the ECG biometric identification method. It is also the key part of our whole system.

To the end of 0. The next chapter will discuss our implementation part. From CHAPTER 4, the content with my groupmate will be different because we are working in different part with each other. We will only write the part which belongs to our job.

# CHAPTER 4. IMPLEMENTATION OF THE ECG CLASSIFICATION SYSTEM

From this chapter, we will write our own part of the project in the group. Here are my parts:

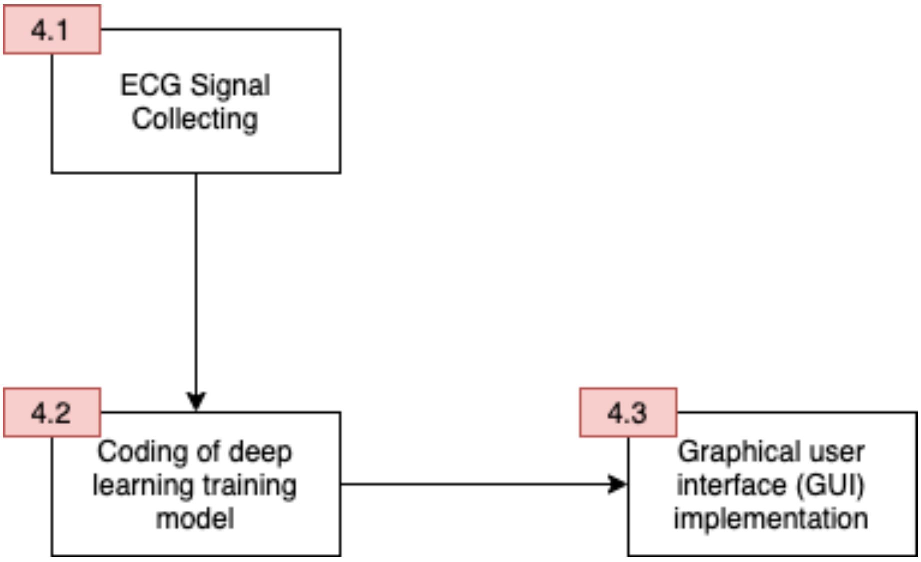



*Figure 13: My work in our project*

In 0, we have a briefly introduction about each part of our work, In this chapter we will have a implementation description and method.

In this project, we select Arduino UNO and AD8232 ECG Collecting Shield as the hardware, Arduino IDE as the platform and C++ as the programming language to collect the ECG signal.

The training model section, we select Jupyter Notebook and Google Colaboratory as the platform, Python 3 as the main programming language.

Because of our GUI identification program also using Python 3 (with Tkinter) as the programming language, the OS compatibility is good. Our GUI identification program can be run in Windows, Linux and also Mac OS.

This is some information about my work. Let us begin in detailed:

## 4.1 Implementation of ECG Signal Collecting

Based on Figure 13, we firstly needed to collect the ECG signal. We needed two hardware:

1) Arduino UNO


*Figure 14: Arduino UNO R3 (Capture by [9])*

2) AD8232 ECG Collecting Shield

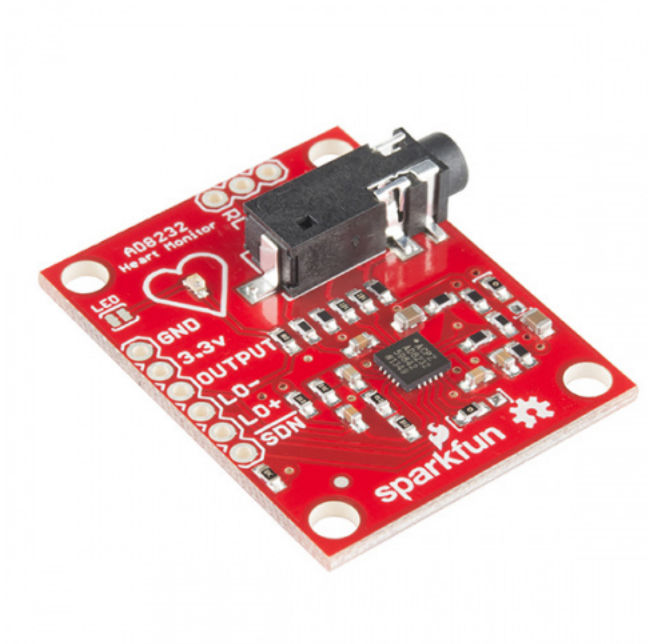



*Figure 15: AD8232 ECG Collecting Shield (Capture by [10])*

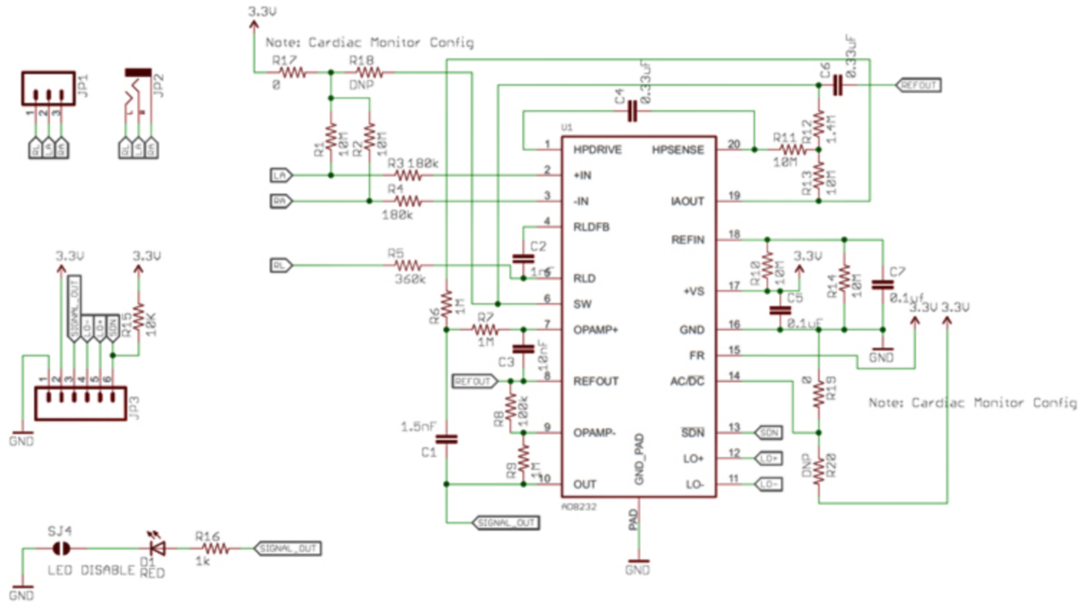



*Figure 16: The schematics of AD8232 ECG detecting module (Capture by [11])*

Here is the technical specification (see Table 1) and the pins usage of AD8232 module (see Table 2):

*Table 1: technical specification of AD8232 module*

| | |
|---|---|
| *Input Voltage (recommended)* | DC 3.3V |
| *Output* | Analog output |
| *Plots* | RA, LA, RL (3 pins, by using 2.54 pins or headphone holder) |

***Table 2: Usage of pins of AD8232 module***

| | |
|---|---|
| *GND* | Ground |
| *3.3V* | DC Voltage in |
| *OUTPUT* | The output of operating amplifier, it outputs the fully adjusted ECG signals. |
| *SDN* | Cut the input control. SDN drives to low level of voltage, change the module into low-power consumption mode. |

The following is the wiring diagram that describe the connection of Arduino UNO and AD8232 ECG detecting module.

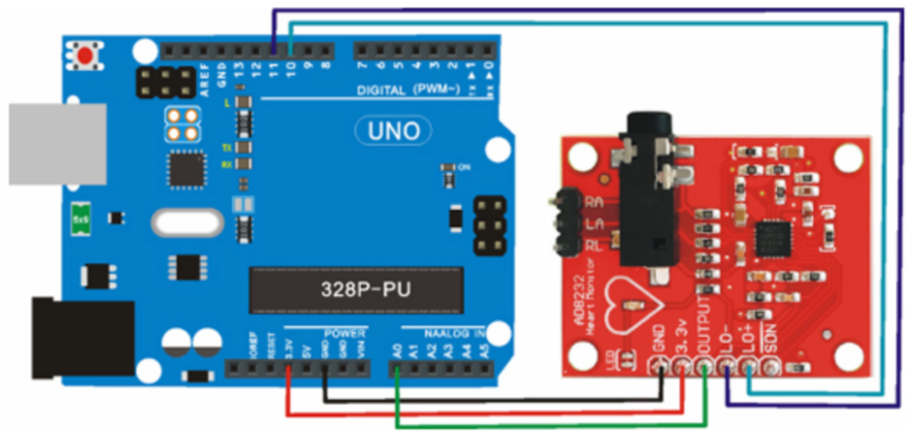



***Figure 17: The wiring of Arduino UNO and AD8232 ECG detecting module (Capture by [12])***

1. Arduino IDE and coding
   The Arduino software IDE is a java-based written cross-platform software application. Arduino has its own language called "sketch", which likes C/C++.

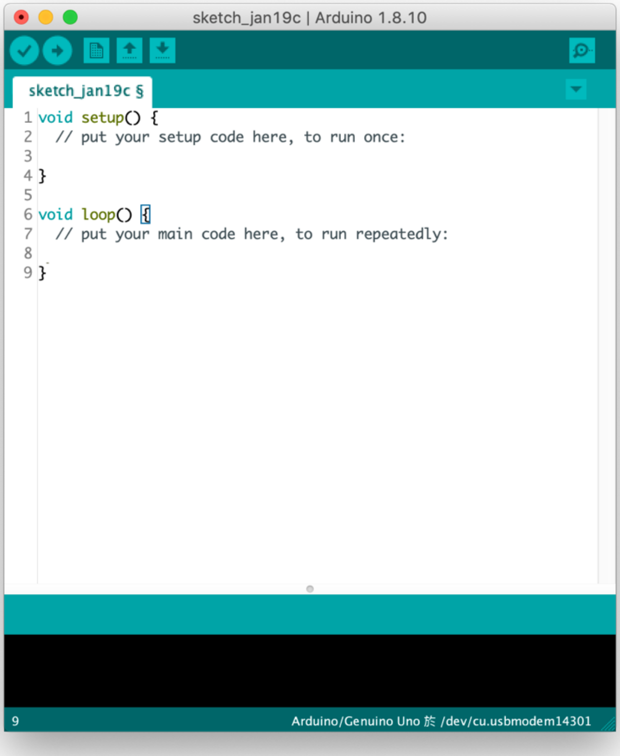



*Figure 18: The user interface of Arduino IDE*

The following is the Arduino *sketch* code that we used:

```
void setup() {
// initialize the serial communication:
Serial.begin(9600);
pinMode(10, INPUT); // Setup for leads off detection LO +
pinMode(11, INPUT); // Setup for leads off detection LO -

}

void loop() {

if((digitalRead(10) == 1)||(digitalRead(11) == 1)){
Serial.println('!');
}
else{
// send the value of analog input 0:
Serial.println(analogRead(A0));
}
//Wait for a bit to keep serial data from saturating
delay(1);
}
```

*Figure 19: the Arduino code (Capture by [13])*

After the coding, we verify the above code and upload to the Arduino UNO board.

2. Overserving the output ECG signals

The following shows the real setup image of the ECG equipment (see Figure 20, Figure 21) and the output ECG signals (see Figure 22) we get:

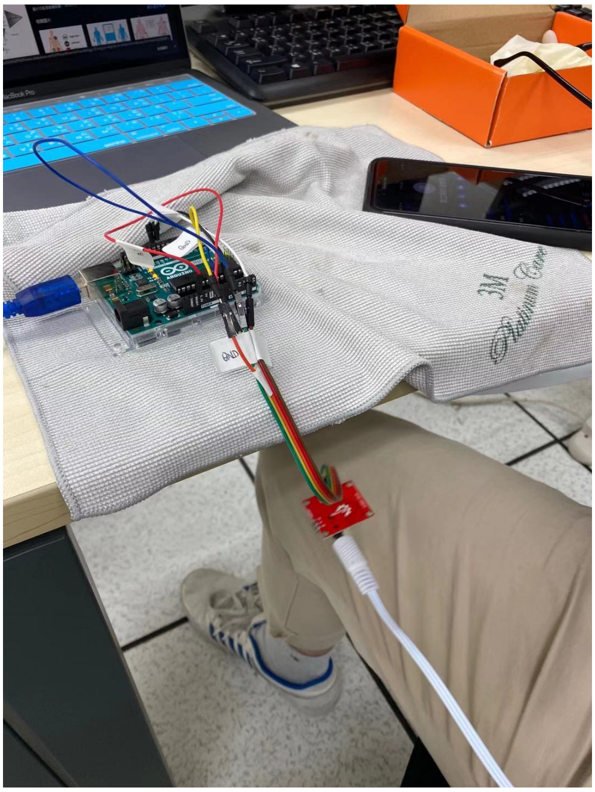


***Figure 20: The real setup of the ECG equipment***

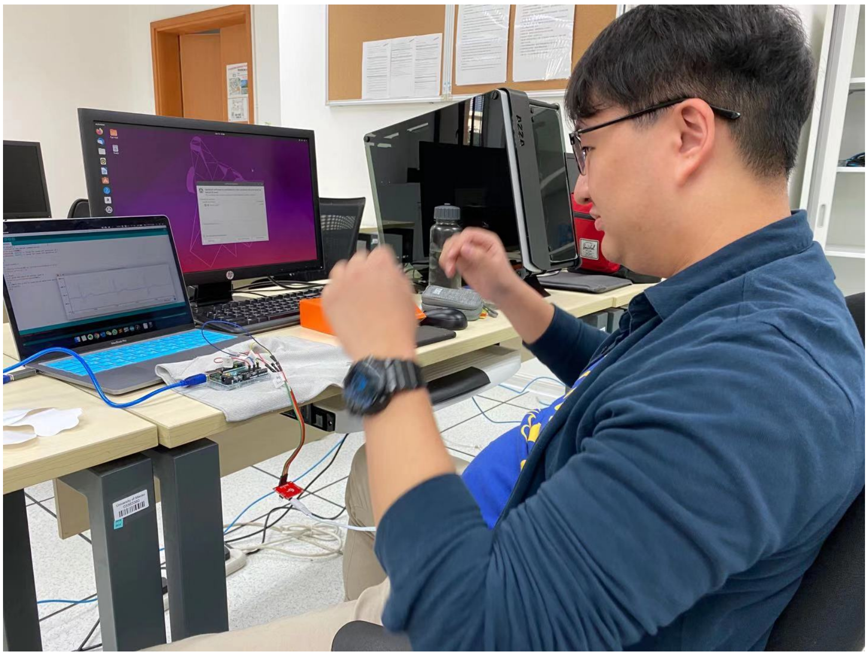


***Figure 21: The testing experiment of ECG equipment to collect the ECG signal***

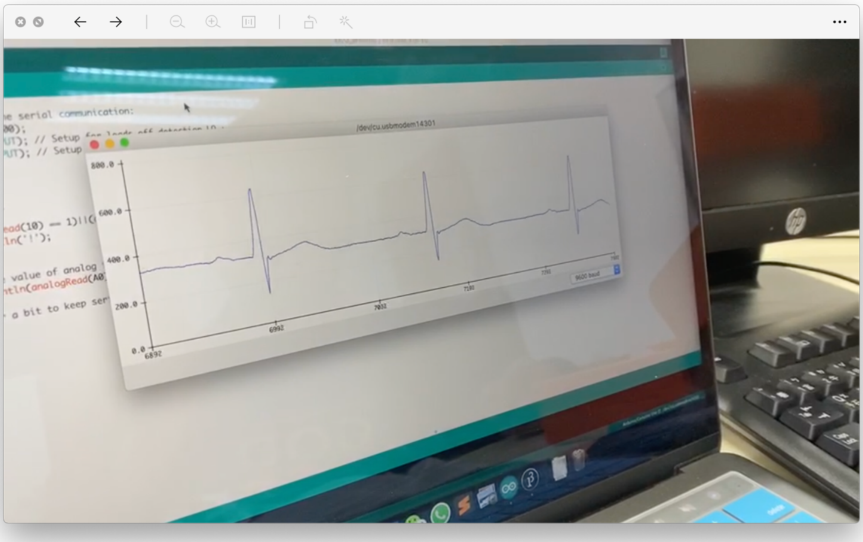



***Figure 22: The testing experiment of ECG equipment to collect the ECG signal***

## 4.2 Coding of deep learning training model

In the developing step we used the ECG database, ECG-ID Database in Physionet [14] to develop the Python 3 coding. For the dataset, we need to convert the .dat files to .csv files to make the dataset can be work in training process.

Here is the code I modified by *Abhishek Patil*'s code to the convert process:

```
"""
Code to convert all .dat files (ECG signals) in a folder to CSV format
@author: Abhishek Patil
@modified by: Oscar, Kuan Ka Meng
"""

import wfdb #WaveForm-Database package. A library of tools for reading, writing,
and processing WFDB signals and annotations.
import pandas as pd
import numpy as np
import glob2 as glob
import os

def create_dataset(person_id):
    db_folder = "ECGID_DB/"
    dataset_folder = "ECGID_DB_DATASET/"

    if (person_id < 10):
        person_path = "0" + str(person_id)
    else:
        person_path = str(person_id)

    person_folder = "Person_" + person_path + "/"


    dat_files=glob.glob(db_folder  +  person_folder  +  '*.dat')  #Get  list  of
all .dat files in the current folder
    df=pd.DataFrame(data=dat_files)
    df.to_csv("files_list_"+person_path+".csv",index=False,header=None)  #Write
the list to a CSV file
    files=pd.read_csv("files_list_"+person_path+".csv",header=None)
    print(dat_files)

    for i in range(0,len(files)):
        recordname=str(dat_files[i])
        temp=recordname.split('.')
        temp2=temp[0].split('/')
        recordname = temp2[2]

        print(recordname)

        record = wfdb.rdsamp(db_folder + person_folder + recordname) # rdsamp()
returns the signal as a numpy array
        record=np.asarray(record[0])
        path=dataset_folder + str(person_id) + "/" + recordname + ".csv"
        print(path)

        dataset_path = dataset_folder + str(person_id) + "/"

        if not os.path.exists(dataset_path):
            os.mkdir(dataset_path)

        np.savetxt(path,record,delimiter=",") #Writing the CSV for each record
        print("Files done: %s/%s"% (i+1,len(files)))

    print("\n [ " + person_folder + "] All files done!\n")

def main():
    for x in range(1,91):
        create_dataset(person_id=x)

if __name__ == "__main__":
    main()
```

***Figure 23: The full code of dat2csv.ipynb***

In Figure 23, The full code of *dat2csv.ipynb*. It gives the function of conversion is shown. We use libraries [15] from Python: wfdb, pandas, numpy, glob2 and os.

Either using ECG-ID database or collecting dataset by ourselves, we needed to label all the records. Then, I modify another program to label the dataset.

```
import os
import csv
dataset_path = '/content/drive/My Drive/FYP_ECG_ID/coding/ECGID_DB_DATASET/'
person_id = '3'
rec_id = '3'
temp_data = []
train_X = []
train_Y = []

with open(dataset_path + person_id + '/rec_' + rec_id + '.csv', newline='') as
csvfile:
    spamreader = csv.reader(csvfile, delimiter=' ', quotechar='|')
    for row in spamreader:
        temp_data.append(', '.join(row))

    for i in range(len(temp_data)):
        temp = temp_data[i].split(',')
        train_X.append(temp[0])
output_path = '/content/drive/My Drive/FYP_ECG_ID/coding/LABELED_DATASET/'
with open(output_path + 'person_id_' + person_id + '_' + rec_id + '.csv', 'w+',
newline='') as csvfile:
    spamwriter = csv.writer(csvfile, delimiter=' ',
                            quotechar='|', quoting=csv.QUOTE_MINIMAL)

    for i in range(len(temp_data)):
        temp = train_X[i]
        spamwriter.writerow([temp, person_id])
```

***Figure 24: The full code of label_dataset.ipynb***

For the main training program (*ECG_Prediction_SVM.ipynb*), we also use some essential libraries of Python. Here is the code snipper

```
import numpy as np
import os
import matplotlib.pyplot as plt
from sklearn import svm
from sklearn.model_selection import KFold
import joblib
from sklearn.metrics import accuracy_score
from sklearn.metrics import matthews_corrcoef
from sklearn.metrics import roc_auc_score
from sklearn.metrics import classification_report
import datetime
```

***Figure 25: code snipper: libraries of main training program***

From Figure 25, we use numpy, matplotlib, sklearn, joblib and datetime. It can all find in PyPI [15].

Then, we see the driven program of the main training program.

```
if __name__ == '__main__':
    path = '/content/drive/My Drive/FYP_ECG_ID/coding/LABELED_DATASET/'
    outfile = 'seq_encoded.txt'
    outdir = 'KFold'
    a = []
    # encode the original dataset
    window_Mat = seq_encoding(path,outfile)

    # split the feature matrix into N fold
    train_all, test_all = splitDataSetbyKFold(window_Mat, 10, outdir)

    print("Generate dataset end and cross validation start")

    clf = svm.SVC(C=1, kernel='rbf', gamma=0.2, probability=True)
    curdir = '/content/drive/My Drive/FYP_ECG_ID/coding'
    clfname = 'SVM'

    crossValidation(clf, clfname, curdir, train_all, test_all)
```

***Figure 26: code snipper: driven program***

In Figure 26, there are some parameter we need to notice. First is the number of training rounds. In the setting, we set 10 rounds as usual. (We also set 100 in experimental training). Second, we focus on the *gamma* value, also known as *learning rate*. The following shown the learning rate we tested:

***Table 3: Tested learning rate***

| ***Gamma value*** | ***Accuracy*** |
|---|---|
| ***0.0001*** | ***0.336*** |
| ***0.001*** | ***0.40*** |
| ***0.01*** | ***0.58*** |
| ***0.1*** | ***0.61*** |
| ***0.5*** | ***0.624*** |
| ***0.6*** | ***0.55*** |
| ***0.7*** | ***0.54*** |
| ***0.8*** | ***0.55*** |

From the practice, we notice that the range from 0.1 to 0.5 is the relatively good range of the accuracy. No matter we take the value form the range of 0.1 to 0.5, the accuracy just change a bit. In the final setting, we choose 0.2 as the *gamma* value.

```
Generate dataset end and cross validation start
----- Round  0  -----
Start Time:  2020-05-21 16:32:51.437781
 training begin...
 training end.
 test begin.
              precision    recall  f1-score   support

           1   0.524452  0.525714  0.525083      3325
           2   0.790276  0.676919  0.729219      3674
           3   0.632395  0.713195  0.670369      4062

    accuracy                       0.644788     11061
   macro avg   0.649041  0.638610  0.641557     11061
weighted avg   0.652388  0.644788  0.646242     11061

 test end.
1748.0
0.0
0.0
0.0
Wrote results to output.data...EOF...
End Time:  2020-05-21 17:09:09.290636
---------------

----- Round  1  -----
Start Time:  2020-05-21 17:09:09.293146
 training begin...
 training end.
 test begin.
              precision    recall  f1-score   support

           1   0.526155  0.514816  0.520424      3341
           2   0.801849  0.694422  0.744279      3747
           3   0.618870  0.708281  0.660563      3973

    accuracy                       0.645150     11061
```

*Figure 27: sample output of the training model program*

In Figure 27, we can see the sample output of the training model. We mainly focus on the accuracy part.

## 4.3 Graphical User Interface (GUI) Implementation

For the GUI application, we used the Tkinter in PyPI [15] as the main library of implementation.

Here is the full code of the GUI implementation

```
from tkinter import ttk
import tkinter as tk
import math
import numpy as np
from sklearn import svm
```

```
import joblib
import serial

def ECG_identify():
    model_dir = "/Users/oscarkuan/coding/fyp_ecgid/ML_Model/"
    filename = "train_SVM1.model"
    detect_length = 14

    X_test = np.arange(detect_length).reshape(detect_length,1)
    Y_test = np.arange(detect_length).reshape(detect_length)

    for i in range(0,detect_length):
        Y_test[i] = person_id

    arduino = serial.Serial('/dev/cu.usbmodem1412201', 9600, timeout=.1)

    i = 0
    while (i<detect_length):
        data = arduino.readline()[:-2] #the last bit gets rid of the new-line 
chars
        if (data and data.decode("utf-8") != '!'):
            testdata = data.decode("utf-8")
            print(testdata)
            X_test[i, 0] = testdata
            i = i + 1

    loaded_model = joblib.load(model_dir + filename)
    result = loaded_model.score(X_test, Y_test)
    print(result)

    acc.set(result)

window = tk.Tk()
window.title('ECG Signal Authenticator 心電圖訊號驗證器')
window.geometry('800x350')
window.configure(background='white')

acc = tk.StringVar()
person_id = 1

header_label = tk.Label(window, text='ECG Signal Authenticator 心電圖訊號驗證器')
header_label.pack()

ecgid_frame = tk.Frame(window)
ecgid_frame.pack(side=tk.TOP)

person_label = tk.Label(ecgid_frame, text='Person ID#: ')
person_label.pack()
personid_label = tk.Label(ecgid_frame, text='1')
personid_label.pack()


ecgid_label = tk.Label(ecgid_frame, text='Accuracy: ')
ecgid_label.pack(side=tk.TOP)
acc_label = tk.Label(ecgid_frame, textvariable=acc)
acc_label.pack()

blank_label = tk.Label(ecgid_frame, text='')
blank_label.pack(side=tk.TOP)

calculate_btn = ttk.Button(window, text='Auth 立即驗證', command=ECG_identify)
calculate_btn.pack(side=tk.TOP)


window.mainloop()
```

***Figure 28: The full code of dat2csv.ipynb***

The coding is quite straight forward, we will not discuss it in detail.

In practice, we will change the serial USB port of Arduino UNO and the identity id with related to the corresponding trained model dataset.

Finally, we will show the screenshot of the GUI application.


***Figure 29: Screenshot of the GUI application***

This is the complete workflow of my own part. The next chapter will be talk more about the system quality and run through the result.

# CHAPTER 5. SYSTEM QUALITY AND RESULT

## 5.1 ECG Signal Collecting

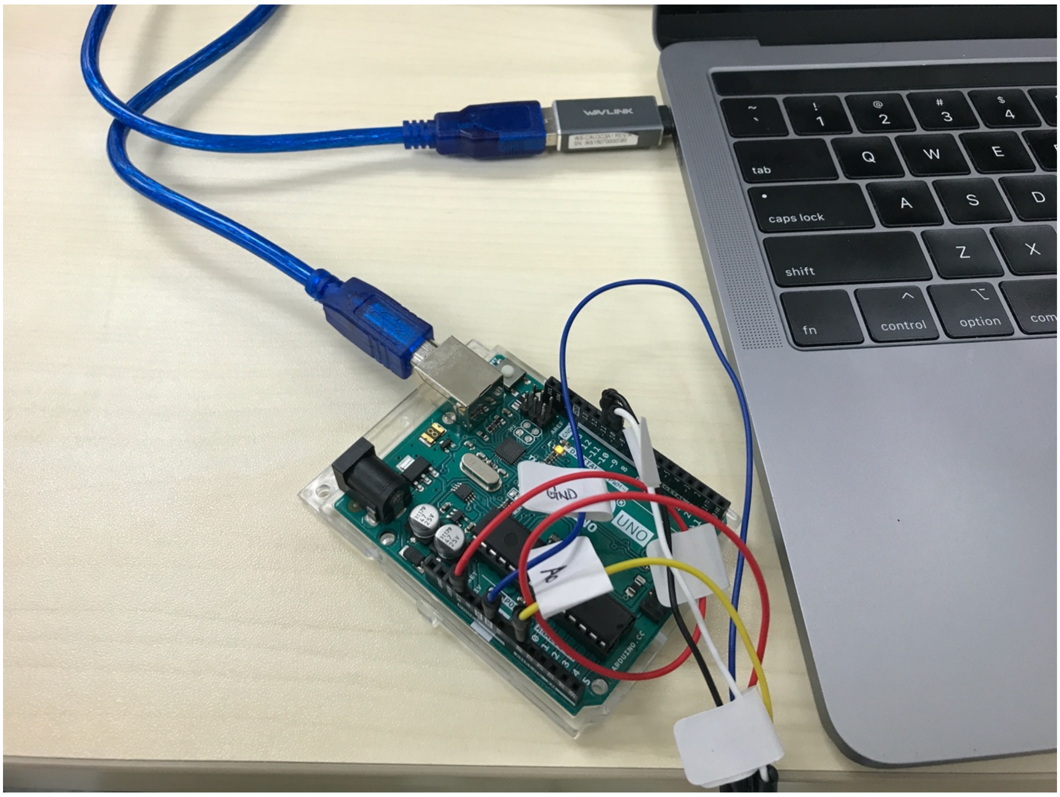


*Figure 30: Arduino UNO with AD8232 ECG collecting shield connected to MacBook Pro*

Then, we pasted the electrodes on our body then we testing to collect the signal. The following is the typical electrodes / sensors placement on our body.

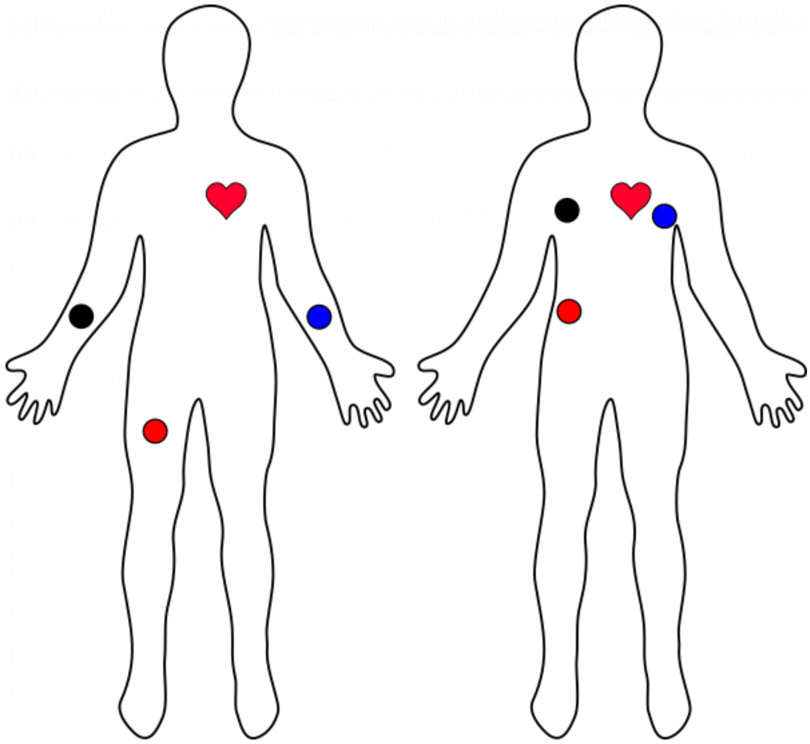


*Figure 31: Typical sensors placement (Capture by [10])*

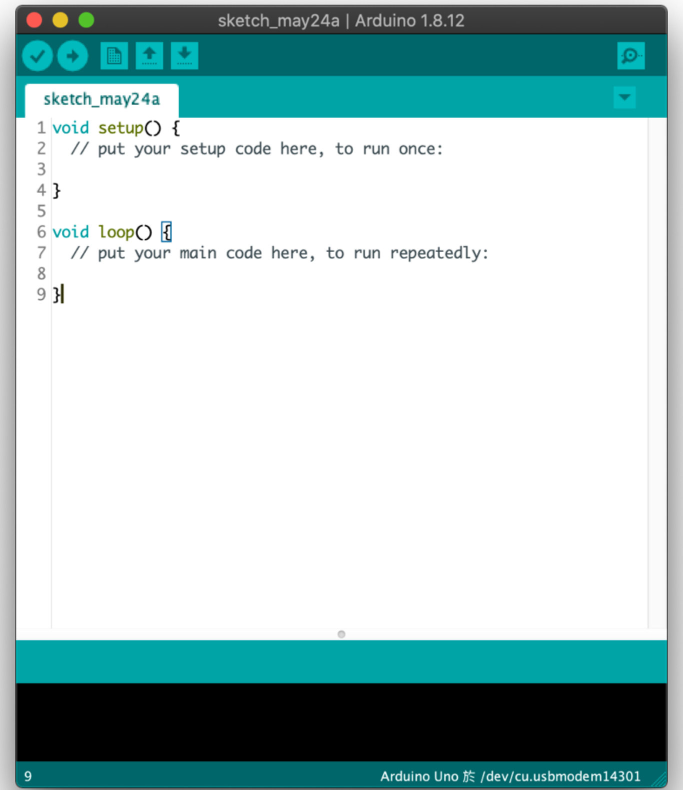



*Figure 32: Arduino IDE GUI*

Then, open the serial plotter in Arduino IDE, we can get the ECG signal plot:

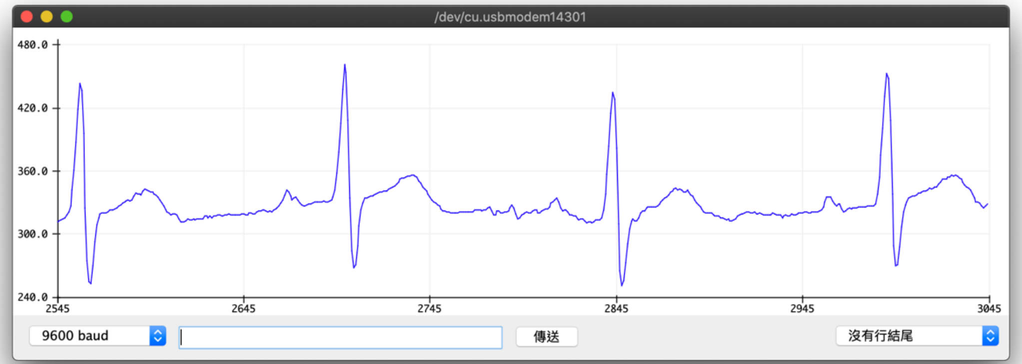



***Figure 33: Arduino IDE: serial plotter***

And we can see the signal in numeric form in the serial monitor.


***Figure 34: Arduino IDE: serial monitor***

We can get the numeric data without the timestamp to collecting the ECG signal in our system. This is the part of signal collecting. The next section will introduce the another key part – User identification with GUI application by collected ECG signal.

## 5.2 User Identification

In this section, we used the collected instantly ECG signal to do the user identification. First, I and my classmate Steven is pasted of the sensors. Here is the photos to show the process during the user identification part:

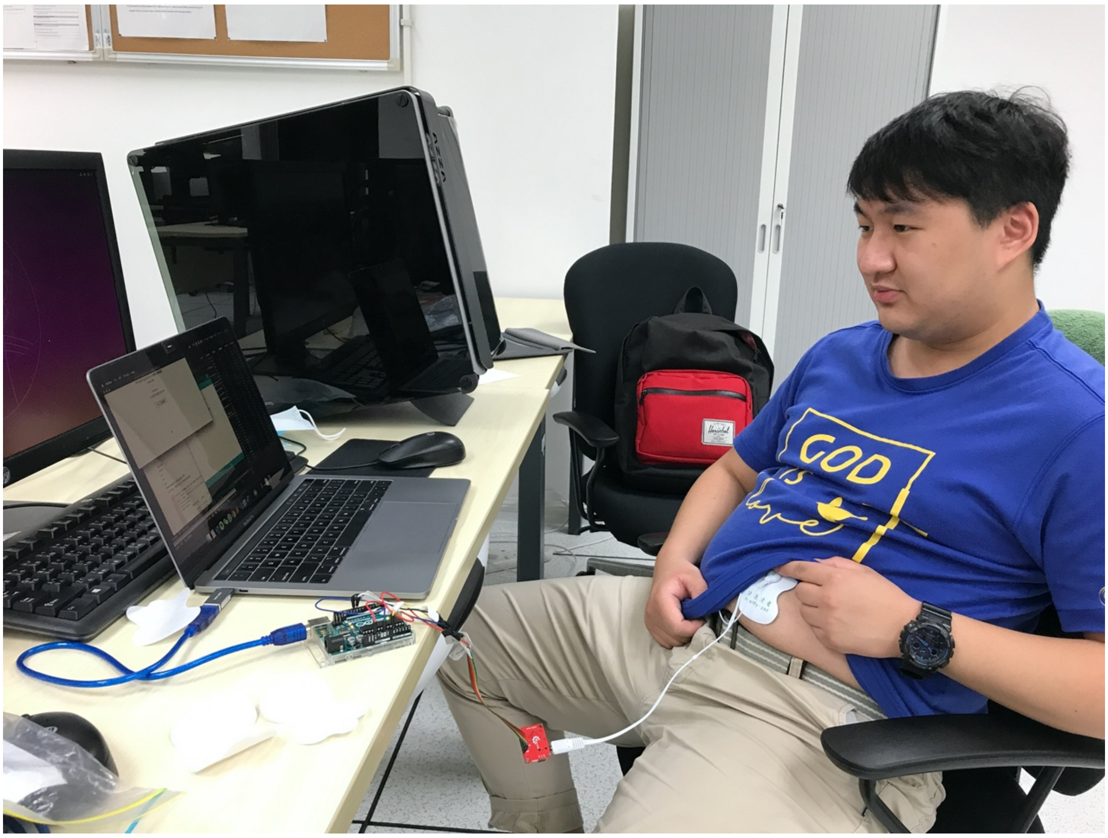


***Figure 35: The testing identification process of me***

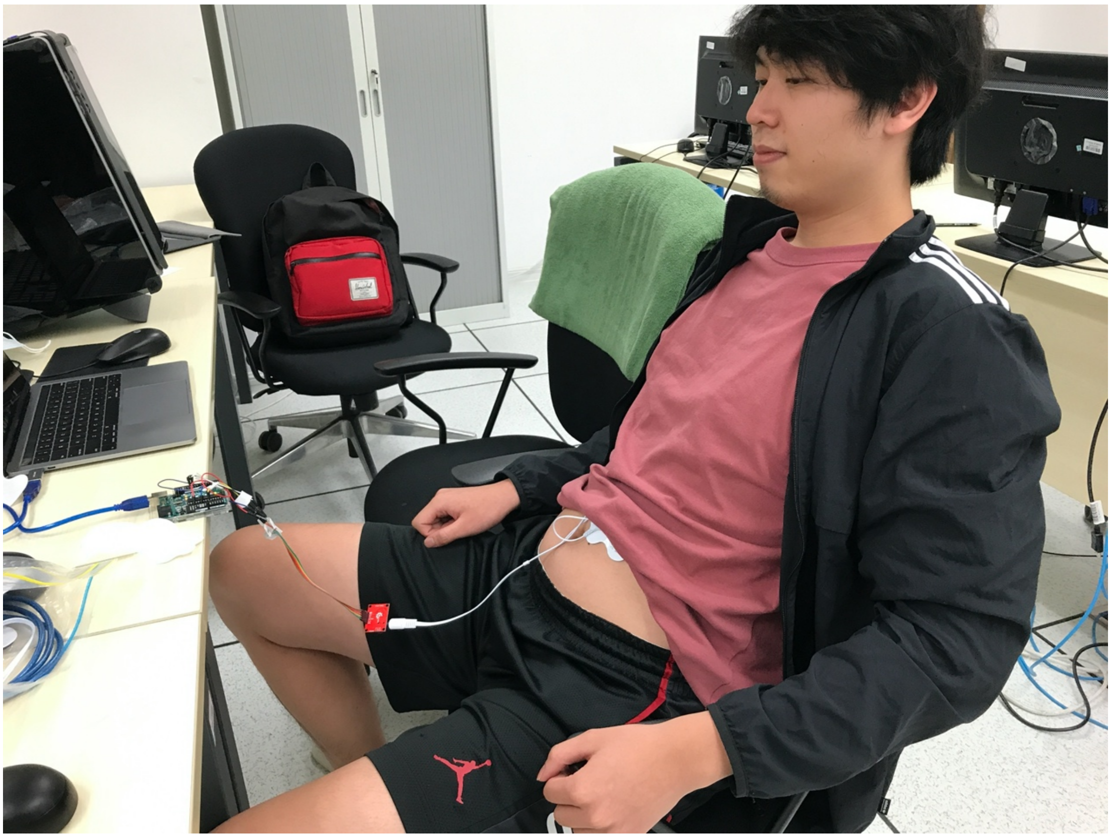


***Figure 36: The testing identification process of my classmate, Steven***

Previously, I train up the model with me and Steven, the user id is 1 and 2 respectively.

Here is the example,


*Figure 37: The testing identification GUI - 1*


*Figure 38: The testing identification GUI - 2*


*Figure 39: The testing identification GUI – 3*


*Figure 40: The testing identification GUI – 4*

In the Figure 37 – Figure 40, they are the GUI application of the user identification system. For GUI, it is a simple user interface.

For the result of identification, it is relatively sensible and correct to distinguish the user identity. However, sometimes the accuracy is not too clear. It may train the model again and modify the threshold.

## 5.3 Experiment result and discussion

In the experiment, we totally collected 3 person's ECG signal data. Here is the table to summarise the information, including user ID (label), corresponding number of data and accuracy (f1-score) of the datasets.

***Table 4: Collected datasets information in the experiment***

| **User ID (label)** | Username | Number of data | Accuracy (f1-score) |
|---|---|---|---|
| **1** | Oscar | 33682 | 0.52 |
| **2** | Steven | 37372 | 0.74 |
| **3** | Yumi | 39556 | 0.66 |

We trained about 10 rounds and get the model best accuracy round. The total accuracy is 0.65. (which is the average and reasonable accuracy)

# CHAPTER 6. ETHICS AND PROFESSIONAL CONDUCT

Here we emphasize, for ethics and professional conduct, all our related works, research, and code have been marked by reference.

In our project, we have used Arduino UNO R3 (buy from TaoBao), AD8232 ECG collecting shield (buy from TaoBao), Arduino IDE 1.8.12 (free, open-source), Jupyter Notebook (free), Google Colaboratory (free), Library from PyPI: jupyter, wfdb, pandas, numpy, glob2, tensorflow, matplotlib, math3, scikit-learn, joblib, pyserial, tkinter (all free, support with Python 3).

For our project, we firstly trained with ECG-ID database in Physionet (free). Then, collected 3 personal ECG signal data from me and our classmates. For future implementation, we will collect more personal ECG signal dataset to train the model with no discrimination.

Further the main goals of our project give a new biometric identification implementation, to help everyone have a new choice of identification method. We hope that this idea can pay a positive contribute to society and if we have chance to do some future implementation, we will try our best to achieve the best quality. Hope one day this idea and more complete implementation or product in the real society.

# CHAPTER 7. CONCLUSION AND FEELING

## 7.1 Feeling

Before to make the conclusion of the project, I wanted to share my feeling during finishing this project.

At the first beginning of this project, we didn't know what topic we should choose. We contact with some professors and get some opinion. After we discuss, we chose this ECG biometric identification project as our topic of final year project.

There is another team doing the related topic of ECG signal. They are also supervised by Prof. Liming Zhang. Actually, we can say we are the big team to cover and support each other. In the big team, I take a role as the coordinator of our team and a representative to contact with Prof. Zhang, our assistant Tonia, and lab technical assistant William. It is an extra work for me, but it is also worth to me. I can take more time to know more about my teammates and know how to discuss the academic and technical issue with my supervisor and assistants.

The unfortunate thing is the COVID-19 outbreak came this year. The impact is around all over the world. Of course, our plan to implement the idea of our final year project is also be affected. I will say we really try our best to finish training the model without a powerful GPU computer. The reason for that is we cannot enter the lab in campus during early period of outbreak. We try several methods to run the Python deep learning code, including running with our own MacBook Pro. Finally, we get a relatively good solution. It is running with Google Colaboratory with GPU, and it is free in charge. It helps us to finish the training step and the whole project.

As finishing this project means most of the potential graduated students need to face and step into the real society and have a new career life. That is an unknown world and situation for me and us. It is a little bit scare with this situation during the society facing COVID-19. The trend of economic and financial of the society is not clear. But anyway, I will keep the best hope to face the coming new career life.

Lastly, I am very glad to finish the project together with my teammate – Elaine, and the group members of the related topic – Steven and Yumi. They are all my good partners and we learnt together how to solve the difficulties by doing the project. During this period, I know more about them and I think our friendship increases.

After sharing my feeling, I wanted to conclude about the project and discuss about the future implementation of our project.

## 7.2 Conclusion and Summary

In the beginning, I want to say thank you to my teammate – Elaine, the group members of the related topic – Steven and Yumi. Again, I want to say they are my very good partners. Also, I also want to thank our supervisor Prof. Liming Zhang, she takes care us very much and gives a lot of advices for us. Finally, I want to say thank you to our assistant Tonia, and lab technical assistant William. They give us a lot of technical advices and supports which smoothing our progression of the project.

By finishing this project, I use some knowledge of Arduino which I learnt before. In addition, I can combine what we learnt in the final year project. I think this is a good reason to prove I can have an opportunity to show what I learnt.

The new thing for me to learn in this project is deep learning and implement to solve the real-world problem. We step-by-step to implement the Python code of training model. Then, change the parameter and test it again and again. Finally, we get a relatively reasonable and good result for the model.

The final part is how to integrate the ECG Collecting Kit with model. Finally, I think a method is implement a GUI application to check the accuracy of the user identity with their user ID. It is simple, but really works.

In this project, I not only learn the academic knowledge, but also the soft skills, like problem-solving skills, collaborating skills, leading skills and communication skills. Again, thank you for all and hope them all good for the future life.

## 7.3 Future work

In this step for our project, it is not a stable version and the user dataset is not too much. Because of the COVID-19 outbreak and the time and resources are limited, we still have so many things to tune and implement to make our project more perfect.

Here, we will list some of the future work:

1. Build the user database

   For the real usage of the identification, the user information database is required. In addition, the management of the personal ECG signal should be securely storage. This is also a point why we needed to build the user database

2. Adapting the authentication application (API)

   After the identification process finished, the application should be sent out the result. With the authentication application, we need to build an API (Application Programming Interface). The API can help us to adapt to different application.

3. Tuning and trained model

   As we are using the SVM algorithm, we can use another training algorithm to train the model to get a higher accuracy. We also can tune the learning rate and other parameter to get the same goal.

# CHAPTER 8. OTHER TOPIC

The ECG area have some other topic, which one of them is implemented by my related teammates. Therefore, I want to talk briefly about this topic.

They are implemented the ECG diagnosis also by deep learning method.

There are several ECG databases to implement the diagnosis. The one that they use is PTB Diagnostic ECG Database [15] in Physionet.

Here is the sample information of ECG diagnosis of Brugada Syndrome.

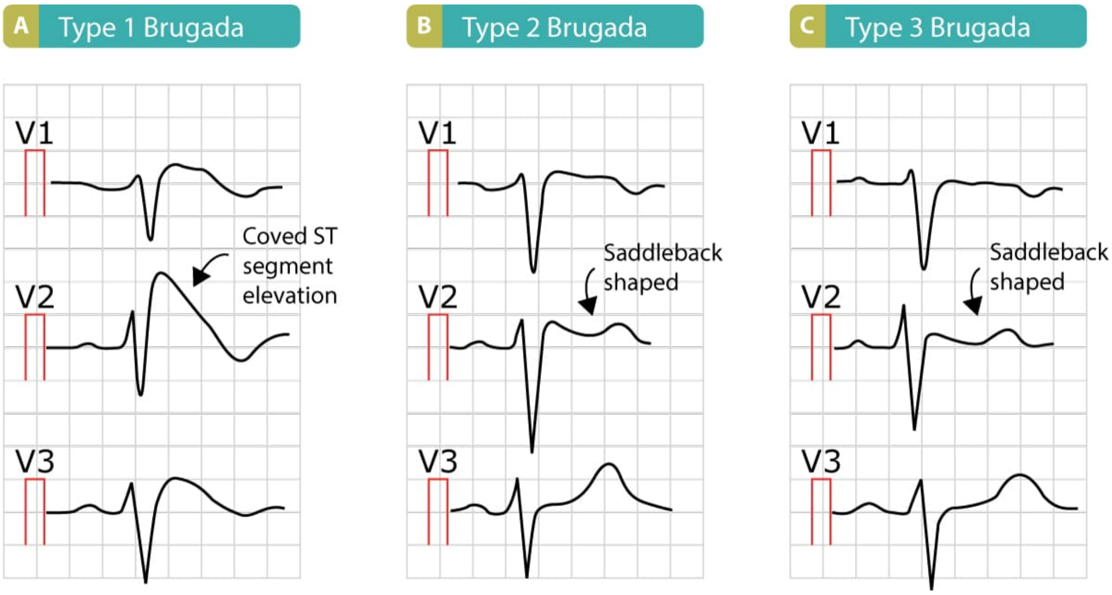



*Figure 41: Information of ECG diagnosis of Brugada Syndrome (Capture by [17])*

# CHAPTER 9. REFERENCES

[1] Elena Abrazhevich, “Vector - Fingerprint and magnifying glass above, man's silhouette with personal information inside. Fingerprint as source of information about person”. [Online]. Available:

https://www.123rf.com/photo_123674330_stock-vector-fingerprint-and-magnifying-glass-above-man-s-silhouette-with-personal-information-inside-fingerprint.html

[2] “CSDN: Face Recognition”. [Online]. Available:

https://blog.csdn.net/sunshine2050_CSDN/article/details/74376063

[3] “IRIS PHOTO”. [Online]. Available:

http://irisphoto.art/

[4] “tumblr.com: tagged/isuniverse”. [Online]. Available:

https://68.media.tumblr.com/f870dc77da5b6e96004102d698769788/tumblr_noakrx7dp51sfotnbo1_500.jpg

[5] Saptarshi Chakraborty and Dhrubajyoti Das, “AN OVERVIEW OF FACE LIVENESS DETECTION”. [Online]. Available:

https://arxiv.org/pdf/1405.2227.pdf

[6] Stephen Balaban Lambda Labs "Deep learning and face recognition: the state of the art." 2015. [Online]. Available:

https://arxiv.org/pdf/1902.03524.pdf

[7] Prof Y Vijaya Lata, "Facial recognition using eigenfaces by PCA".2009. [Online]. Available: https://www.researchgate.net/publication/228748710_Facial_recognition_using_eigenfaces_by_PCA

[8] Yann Lecun, "Gradient-Based Learning Applied to Document Recognition". 1998. [Online]. Available: http://yann.lecun.com/exdb/publis/pdf/lecun-98.pdf

[9] “Amazon: Arduino UNO R3 [A000066]”, 2020, [Online]. Retrieved from:

https://www.amazon.com/Arduino-A000066-ARDUINO-UNO-R3/dp/B008GRTSV6

[10] “AD8232 Heart Rate Monitor Hookup Guide”, 2020, [Online]. Retrieved from:

https://learn.sparkfun.com/tutorials/ad8232-heart-rate-monitor-hookup-guide/

[11] “SparkFun Single Lead Heart Rate Monitor - AD8232”, 2020, [Online]. Retrieved from:

https://easyeda.com/SparkFun/SparkFun_Single_Lead_Heart_Rate_Monitor_AD8232-74YSasKqI

[12] “AD8232 Heart Rate Monitor – GI Electronic”, 2020, [Online]. Retrieved from:

https://www.gie.com.my/shop.php?action=sensors/others/ad8232

[13] “GitHub: sparkfun/ Heart_Rate_Display_Arduino.ino”, 2020, [Online]. Retrieved from:

https://github.com/sparkfun/AD8232_Heart_Rate_Monitor/blob/master/Software/Heart_Rate_Display_Arduino/Heart_Rate_Display_Arduino.ino

[14] “Physionet: ECG-ID Database”, 2014, [Online]. Retrieved from:

https://physionet.org/content/ecgiddb/1.0.0/

[15] “The Python Package Index (PyPI)”. 2020, [Online]. Retrieved from:

https://pypi.org/

[16] “Physionet: PTB Diagnostic ECG Database”, 2004, [Online]. Retrieved from:

https://physionet.org/content/ptbdb/1.0.0/

[17] “Grepmed: ECG representation of Bruguda Syndrome”. 2018, [Online]. Retrieved from:

https://www.grepmed.com/images/3354

# CHAPTER 10. APPENDIX

For implement our project, we need:

1 Computer (with Windows/Linux/Mac OS)

2 Arduino UNO R3 (with USB cable)

3 AD8232 Heart Beat / ECG Collecting Shield (with sensors / electrodes)

4 Arduino IDE 1.8.12 (free, open-source)

5 Jupyter Notebook (free) or Google Colaboratory (free)

6 Library from PyPI (all free, support with Python 3).:

 6.1 jupyter,

 6.2 wfdb,

 6.3 pandas,

 6.4 numpy,

 6.5 glob2,

 6.6 tensorflow,

 6.7 matplotlib,

 6.8 math3,

 6.9 scikit-learn,

 6.10 joblib,

 6.11 pyserial,

 6.12 tkinter

END OF REPORT