
Rapport Projet Arduino
SYSTÈME D'ALARME AVEC 'FACE RECOGNIZE'



Réalisé par :

MRABET Abderrahmen

AMEZIANE Nabil

(GROUPE 7)

Encadré par :

OSMANI Aomar

HAMIDI Massinissa

Introduction :

Nous avons eu un projet consistant à utiliser toutes les compétences acquises en matière de l'Internet des objets afin de créer un objet connecté. Pour cela, nous nous sommes tournés vers un projet dont chaque personne (peu importe l'âge, le sexe, etc) peut en avoir l'utilité : un système d'alarme ordinaire qui fonctionne aussi avec la reconnaissance faciale.

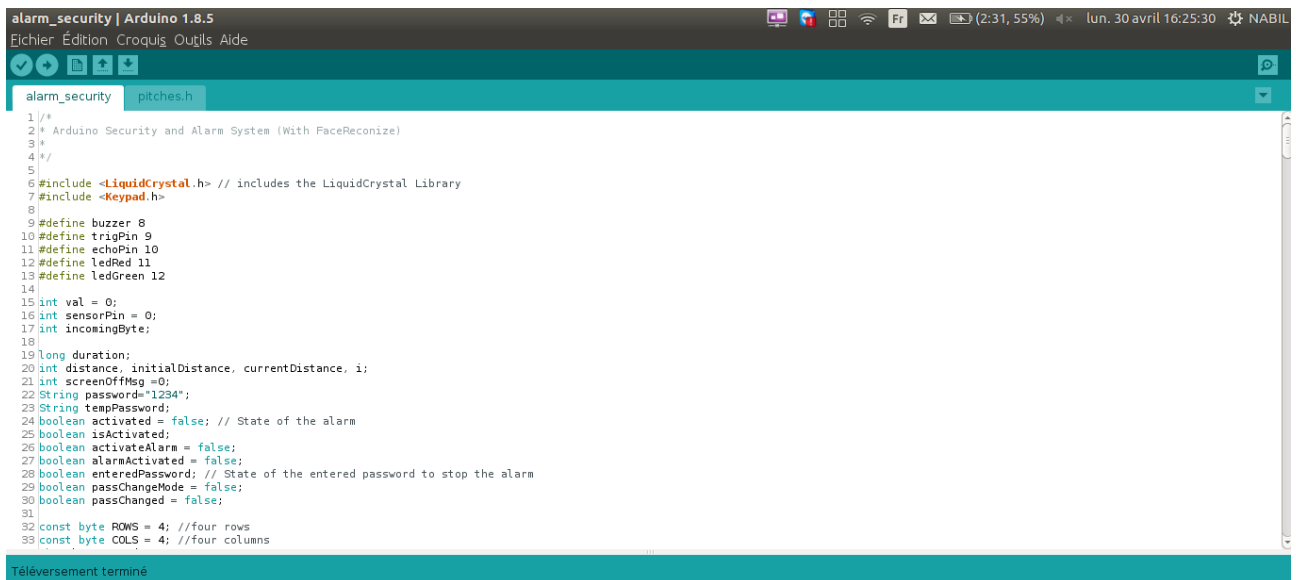
En d'autres termes, nous avons conçu un système d'alarme qui peut être activé/désactivé soit via un clavier numérique en introduisant un mot de passe prédéfini par l'utilisateur, soit via une caméra qui reconnaît un visage enregistré précédemment.

Outils et environnement de développement :

Dans cette partie, nous allons présenter les différents outils qu'on a utilisés lors de la réalisation de ce projet. On peut distinguer deux grandes parties : le Hardware et le Software.

Software :

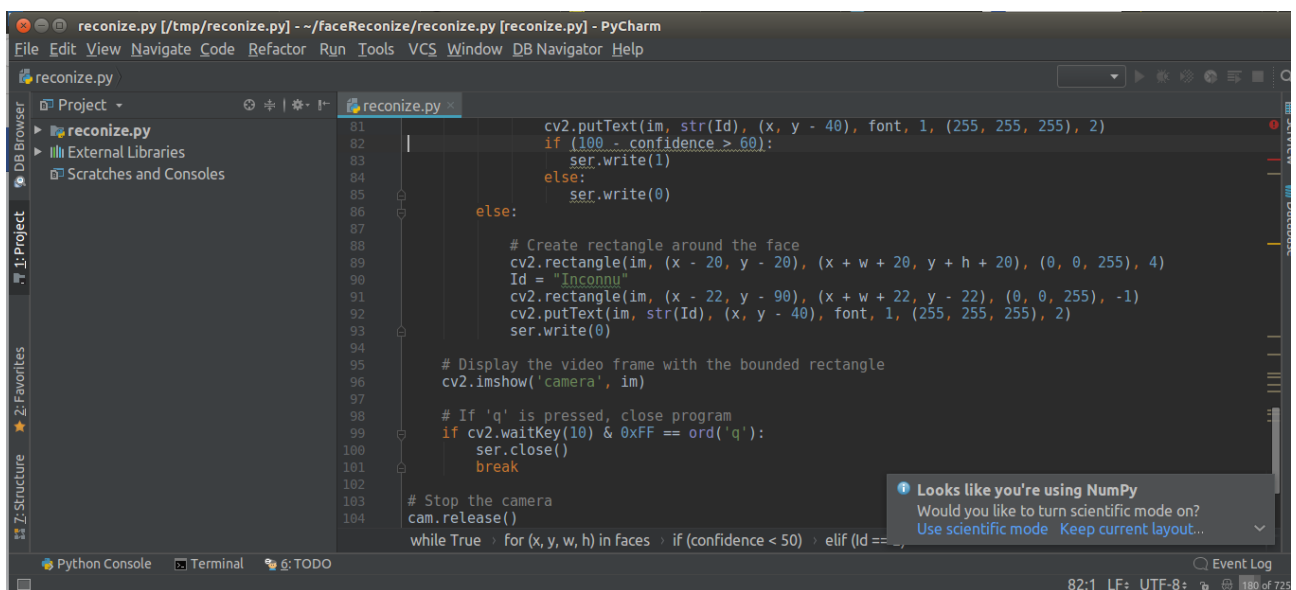
- IDE Arduino (version 1.8.5): Cette application propose tout ce qui est nécessaire pour éditer un programme, vérifier sa syntaxe, puis le téléverser dans une carte Arduino, c'est-à-dire programmer le microcontrôleur de la carte. Voici à quoi ressemble une interface Arduino :



```
1 /**
2  * Arduino Security and Alarm System (With FaceReconize)
3  *
4  */
5
6 #include <LiquidCrystal.h> // includes the LiquidCrystal Library
7 #include <Keypad.h>
8
9 #define buzzer 8
10 #define trigPin 9
11 #define echoPin 10
12 #define ledRed 11
13 #define ledGreen 12
14
15 int val = 0;
16 int sensorPin = 0;
17 int incomingByte;
18
19 long duration;
20 int distance, initialDistance, currentDistance, i;
21 int screenOffMsg = 0;
22 String password = "1234";
23 String tempPassword;
24 boolean activated = false; // State of the alarm
25 boolean isActivated;
26 boolean activateAlarm = false;
27 boolean alarmActivated = false;
28 boolean enteredPassword; // State of the entered password to stop the alarm
29 boolean passChangeMode = false;
30 boolean passChanged = false;
31
32 const byte ROWS = 4; //four rows
33 const byte COLS = 4; //four columns
```

Téléversement terminé

- PyCharm : c'est un environnement de développement intégré (IDE) utilisé dans la programmation informatique, spécifiquement pour le langage Python.



```
81 cv2.putText(im, str(Id), (x, y - 40), font, 1, (255, 255, 255), 2)
82 if (100 - confidence > 60):
83     ser.write(1)
84 else:
85     ser.write(0)
86
87 # Create rectangle around the face
88 cv2.rectangle(im, (x - 20, y - 20), (x + w + 20, y + h + 20), (0, 0, 255), 4)
89 Id = "Inconnu"
90 cv2.rectangle(im, (x - 22, y - 90), (x + w + 22, y - 22), (0, 0, 255), -1)
91 cv2.putText(im, str(Id), (x, y - 40), font, 1, (255, 255, 255), 2)
92 ser.write(0)
93
94 # Display the video frame with the bounded rectangle
95 cv2.imshow('camera', im)
96
97 # If 'q' is pressed, close program
98 if cv2.waitKey(10) & 0xFF == ord('q'):
99     ser.close()
100     break
101
102 # Stop the camera
103 cam.release()
104 while True: for (x, y, w, h) in faces: if (confidence < 50): elif (Id == ...
```

Looks like you're using NumPy
Would you like to turn scientific mode on?
Use scientific mode Keep current layout...

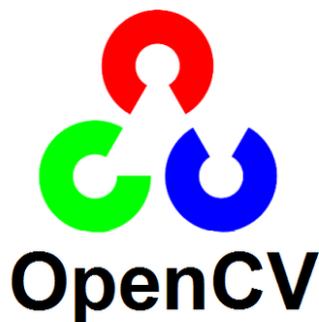
- Langues et bibliothèques utilisés : Lors de la réalisation de ce travail, nous avons utilisés deux langages de programmation : Python3 et C++.

La partie Python est utilisée pour assurer la fonctionnalité de la reconnaissance faciale. Celle-ci a été implémentée en utilisant la bibliothèque OpenCV. Tandis que la partie C++

(Arduino) est utilisée afin de mettre en place le matériel utilisé et de lier les différentes composantes. Les bibliothèques utilisées dans cette partie sont Keypad.h qui gère le clavier en le simulant en une matrice, et LiquidCrystal.h qui gère l'écran LED.



- OpenCV (Open Computer Vision) : est une bibliothèque graphique libre, initialement développée par Intel, spécialisée dans le traitement d'images en temps réel.

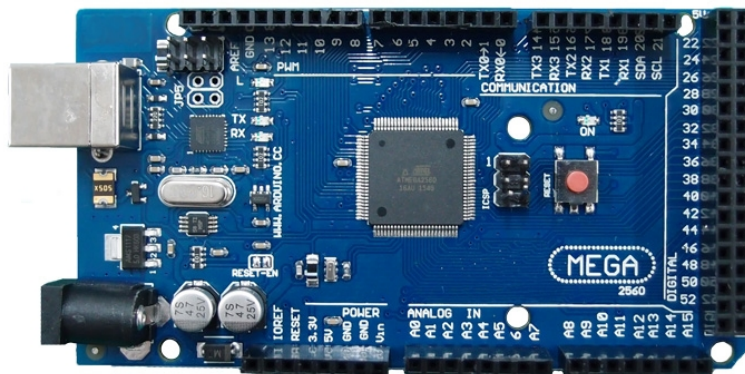


- LiquidCrystal : Cette bibliothèque permet à une carte Arduino de contrôler les affichages LiquidCrystal (LCD) basés sur le chipset Hitachi HD44780 (ou un chipset compatible), qui se trouve sur la plupart des LCD à base de texte.

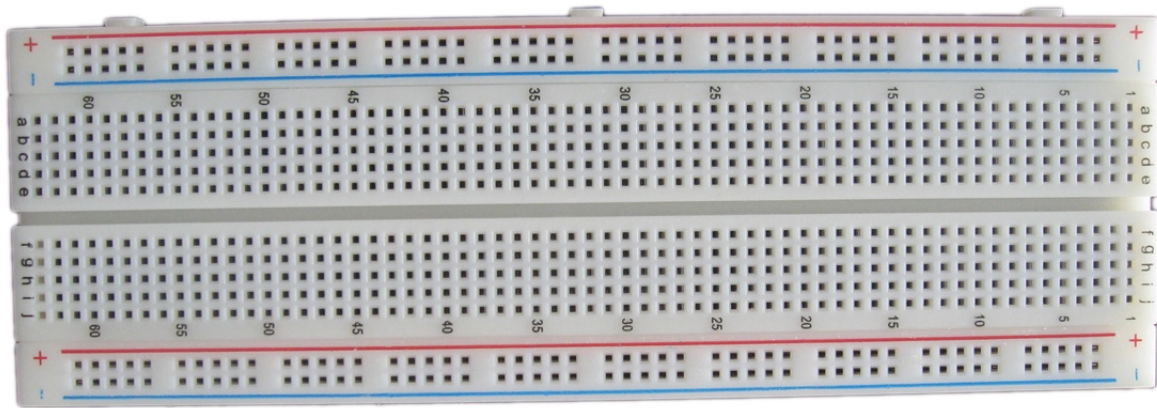
- Keypad : La bibliothèque Keypad permet à votre Arduino de lire un clavier de type matriciel.

Hardware :

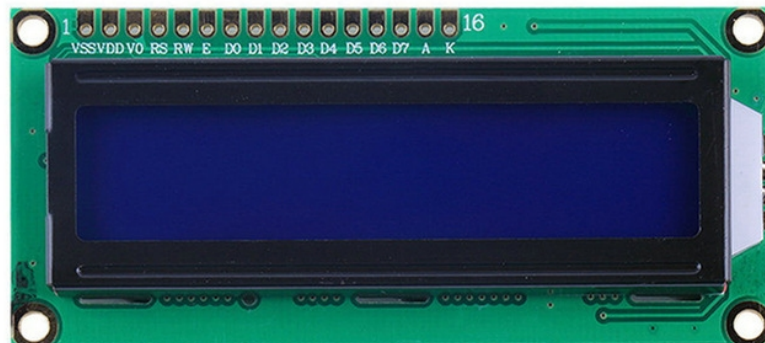
- ◆ **Carte Arduino Mega2560:** C'est une carte à microcontrôleur basée sur un ATmega2560. Cette carte dispose de 54 broches numériques d'entrées/sorties, 16 entrées analogiques, une connexion USB, un connecteur d'alimentation jack et d'un bouton de réinitialisation (reset).



- ◆ **Breadboard - 830 tie in points :** Elle comporte quatre lignes de bus couvrant chacune la longueur de la carte, et 63 rangées de broches, ce qui permet d'avoir jusqu'à neuf circuits intégrés DIP à 14 broches ou sept circuits DIP à 16 broches.



- ◆ Écran LCD 1602: L'écran qui sert à afficher les informations.



- ♦ Clavier à membrane 4x4 : Pour permettre à l'utilisateur d'interagir avec le système.



- ♦ capteur à ultrasons : Qui sert à détecter les passages à proximité.



- ♦ Buzzer passive: Il s'agit d'un haut parleur miniature à qui il faut fournir le signal audio à diffuser.



- ◆ **Potentiomètre:** Une résistance réglable qui est utilisée pour protéger l'écran.

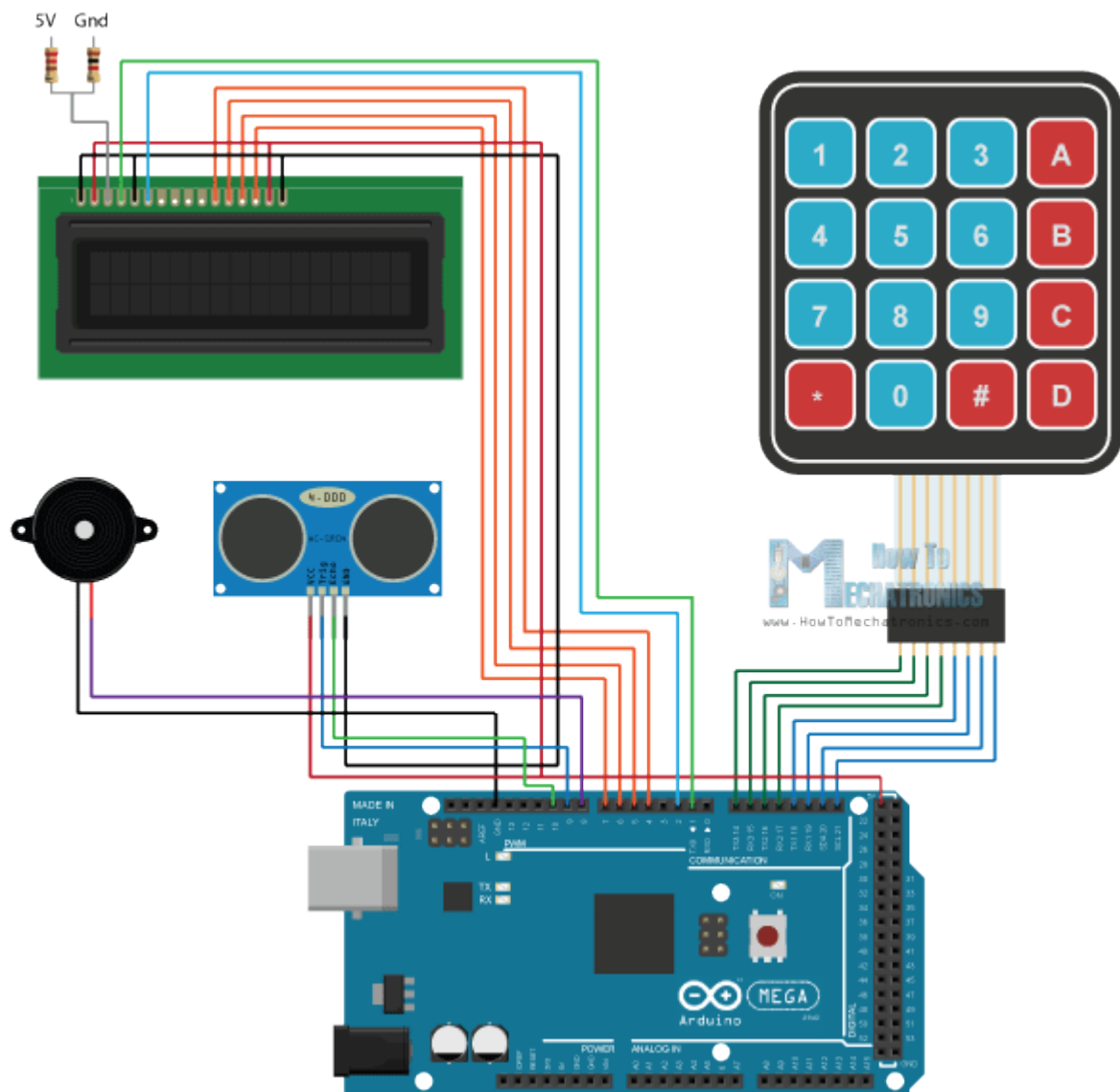


- ◆ **Deux Led's:** Une verte pour indiquer l'état du système (alarme activée ou désactivée) et une rouge qui s'allume quand une intrusion est détectée.

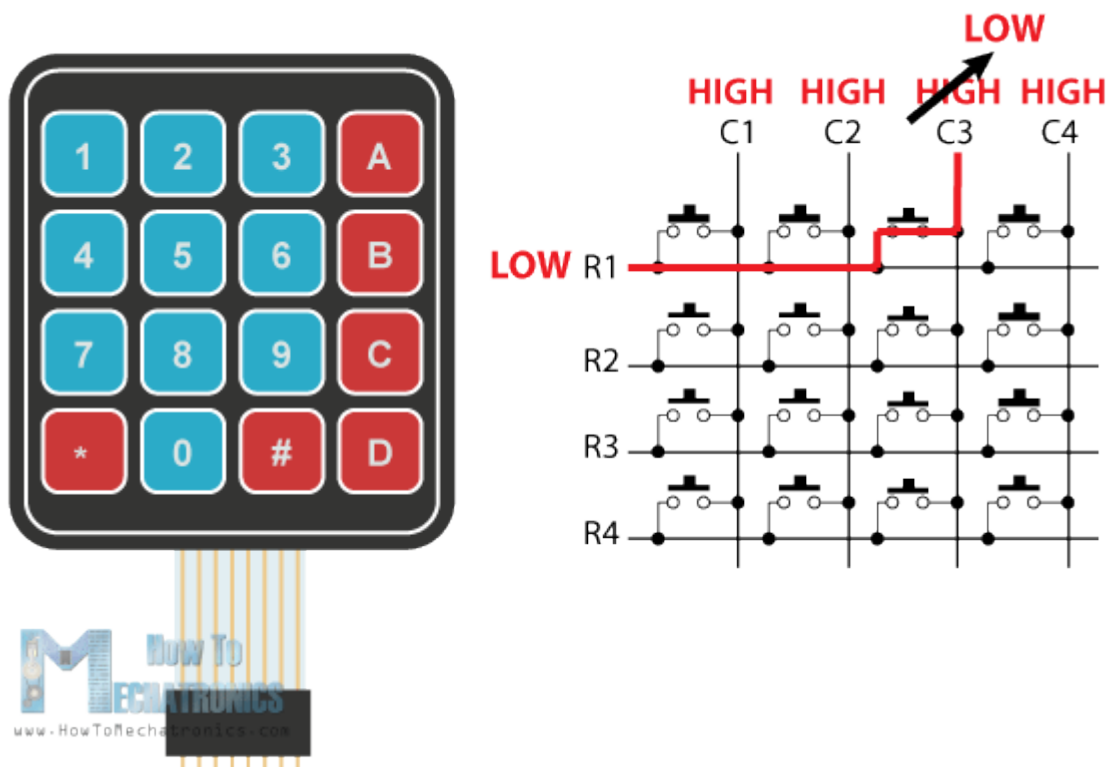


Réalisation:

Afin de mettre le système en marche, nous avons réalisé les branchements suivants :



pour le buzzer, nous avons besoin d'une seule broche. Le clavier 4 × 4 a 8 broches, 4 d'entre eux sont pour les lignes et 4 d'entre eux pour les colonnes du clavier.



Ensuite, pour faire fonctionner tout ça, il faut implémenter le code Arduino. Celui ci doit inclure les deux bibliothèques que nous avons définies (LiquidCrystal et Keypad).

D'autre part, et pour assurer la reconnaissance faciale, nous avons fait appel à la bibliothèque OpenCV que nous avons implémentée en Python. Pour cette partie, nous avons trois fichiers de code Python qui utilisent plusieurs algorithmes de Machine Learning.

Le premier 'dataset.py' permet de collecter plusieurs photos à partir de la caméra définissant un visage. Le deuxième 'trainee.py' permet d'entraîner les modèles de photos, c'est à dire convertir les photos prises en un modèle de visage.

Finalement le fichier 'reconize.py' permet d'identifier un visage en comparant l'input de la caméra à des modèles que nous avons déjà générés en format YAML.

On remarque donc, qu'il y a une certaine communication qui se fait entre le code python et Arduino (quand on identifie un visage, l'alarme doit se désactiver). Cette communication est assurée par le port série qu'on initialise dans les deux parties.

Conclusion:

Au début du cours et suite à la recommandation de nos professeurs, nous nous avons fixé comme objectif la réalisation de la partie de la reconnaissance faciale qui ne nous a pas été forcément facile vu le manque de connaissances en Arduino et en Machine Learning. Néanmoins, nous avons réussi à réaliser un prototype qui fonctionne parfaitement.

Ce projet nous a permis non seulement de nous initier à l'internet des objets et à la prise en main de l'Arduino, mais aussi à l'électronique en général.