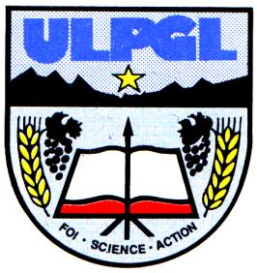
UNIVERSITÉ LIBRE DES PAYS DES GRANDS LACS

FACULTÉ DE SCIENCES ET TECHNOLOGIES

FILIÈRE DES SCIENCES DE L'INGÉNIEUR



BP. 368 GOMA

www.ulpgl.net

CONCEPTION ET RÉALISATION D'UN SYSTÈME DE CONTRÔLE D'ACCÈS ET DE PRISE DE PRÉSENCE UTILISANT LA RECONNAISSANCE FACIALE POUR UN CAMPUS UNIVERSITAIRE : Cas du campus Salomon/ ULPGL Goma

Par **MUNYI RUBAMBURA Charles**

Travail présenté et défendu en vue de l'obtention du Diplôme de Bachelor en Sciences de l'Ingénieur

Mention : **Génie Informatique**

Directeur : **Prof. BARAKA MUSHAGE Olivier**

Encadreur : MSc. Ir. NZANZU VINGI Patrick

ANNÉE ACADÉMIQUE 2023 - 2024

Epigraphe

"La technologie est au service de l'humain quand elle protège, identifie et simplifie."

Dr. Léonard Mbayo

Dédicace

À mes parents, BIRHAHEKA RUBAMBURA Philémon et SHAMAVU Séraphine, pour leur amour inconditionnel.

MUNYI RUBAMBURA Charles

Remerciements

Je tiens à exprimer ma profonde gratitude à toutes les personnes qui ont contribué à la réalisation de ce mémoire.

Tout d'abord, je remercie sincèrement le bon Dieu pour sa protection et sa bienveillance dans notre vie

A mon directeur et à mon encadreur de recherche, BARAKA MUSHAGE Olivier et NZANZU VINGI Patrick pour leurs précieuse guidance, leurs conseils avisés et leurs soutiens constants tout au long de ce travail. Leurs encouragements et Leurs rigueur académique ont été essentiels à l'aboutissement de ce projet.

Je remercie également l'ensemble des membres de la Faculté des Sciences et Technologies de l'ULPGL pour leur disponibilité et leurs enseignements de qualité.

Mes remerciements vont aussi à mes collègues et amis, dont la camaraderie et le soutien moral m'ont aidé à surmonter les moments difficiles. Vos conseils et votre aide ont été d'une grande valeur pour moi.

Je n'oublie pas mes parents, pour leur amour inconditionnel, leur soutien moral et financier, et pour m'avoir toujours encouragé à persévérer dans mes études. Sans vous, rien de tout cela n'aurait été possible.

MUNYI RUBAMBURA Charles

Résumé

Dans le but de répondre à l'inefficacité et à la vulnérabilité des méthodes traditionnelles de contrôle d'accès et de gestion de présence à l'Université Libre des Pays des Grands Lacs (ULPGL), notamment l'usage de registres papier sujets à la fraude, nous avons conçu et réalisé un système intelligent basé sur la reconnaissance faciale. Ce système vise à renforcer la sécurité du campus et à améliorer l'efficacité administrative. Il utilise des technologies avancées telles qu'OpenCV, Face recognition de Dlib pour la détection et la reconnaissance faciale, Python pour le traitement d'images et l'implémentation des algorithmes, ainsi que les microcontrôleurs ESP32 et ESP32-CAM associés à un servo moteur SG90. Les résultats obtenus démontrent le bon fonctionnement du système, avec un accès automatisé fiable, une meilleure identification des individus et une gestion de la présence optimisée.

Mots clés : reconnaissance faciale, contrôle d'accès, prise de présence, ESP32, OpenCV, sécurité, ULPGL, Python, automatisation, Face recognition, Dlib.

Abstract

In order to address the inefficiency and vulnerability of traditional access control and attendance management methods at the Université Libre des Pays des Grands Lacs (ULPGL), particularly the use of paper registers prone to fraud, we designed and developed an intelligent system based on facial recognition. This system aims to enhance campus security and improve administrative efficiency. It leverages advanced technologies such as OpenCV and Dlib's Face_recognition for facial detection and recognition, Python for image processing and algorithm implementation, as well as ESP32 and ESP32-CAM microcontrollers paired with an SG90 servo motor for real-time image capture and access control. The results obtained demonstrate the system's proper functioning, providing reliable automated access, improved individual identification, and optimized attendance management.

Keywords: facial recognition, access control, ESP32, OpenCV, ULPGL, Python, Dlib.

Table des matières

Epigraphe i

Dédicace ii

Remerciements iii

Résumé iv

Abstract v

Table des matières vi

Liste des sigles et abréviations x

Liste des tableaux xi

Liste des figures xii

0. Introduction générale 1

0.1. Contexte et Généralités sur le thème 1

0.2. Identification et formulation du problème 1

0.3. Questions de recherche 2

0.4. Formulation des hypothèses 3

0.5. Justification du choix du sujet et motivations 3

0.5.1 Justification du choix du sujet 3

0.5.2 Motivations 4

0.6. Énoncé des objectifs de recherche 4

0.6.1. L'objectif général 4

0.6.2. Les objectifs opérationnels/spécifiques 4

0.7. Méthodologie et délimitation du travail 5

0.7.1. Méthodologie de la Recherche 5

0.7.2. Délimitation du Travail 6

9.1. Structure du mémoire/ Subdivision du travail 6

Chapitre 1 Revue de la littérature sur les Systèmes de Contrôle d'Accès et la Reconnaissance Faciale 8
1.1 Généralités sur les Systèmes de Contrôle d'Accès 8
1.1.1 Évolution Historique des Systèmes de Contrôle d'Accès 8
1.1.2 Les Composants Clés des Systèmes Modernes de Contrôle d'Accès 10
1.2 La Reconnaissance Faciale 10
1.2.1 Historique 10
1.2.2 Les Principes Fondamentaux de la Reconnaissance Faciale 11
1.2.3 Avantages et Limites des Systèmes de Reconnaissance Faciale 12
1.3 Présentation du Servomoteur SG90 14
1.4 L'ESP32 Cam : Pilier du Système de Reconnaissance Faciale 15
1.5 Comparaison avec les Systèmes Traditionnels de Contrôle d'Accès 16
1.6 Quelques méthodes de détection de visages 17
1.6.1 Méthode du traitement automatique du visage 17
1.6.2 Méthode d'Eigenface 17
1.6.3 Template Matching Methods 17
1.6.4 Méthode d'analyse des points particuliers(facial landmarks) 18
1.6.5 Méthode LDA (Linear discriminant analysis) Fisher 18
1.6.6 Filtre de Haar 18
1.6.7 Methode LBP (Local Binary Patterns) 18
1.7 Conclusion du partielle 19
Chapitre 2 Conception et Développement du système 20
2.1 Introduction 20
2.2 Spécifications Fonctionnelles et Techniques 20
2.2.1 Spécifications Fonctionnelles 20
2.2.2 Spécifications Techniques 21
2.3 Architecture du Système 21

2.4 Présentation des diagrammes 23
2.4.1 Brève description des spécifications fonctionnelles 23
2.4.2 Diagramme de cas d'utilisation 23
2.4.2.1 Présentation des acteurs 23
2.4.2 Diagramme de Séquence 28
2.4.3 Diagramme des Classes 31
2.5 Présentation des technologies 32
2.5.1 L'interface utilisateur 32
2.5.1.2 installation 32
2.5.1.2 Structure du projet Django et Tailwind css 33
2.5.2 La Reconnaissance Faciale 34
2.6 Partie matérielle [20] 40
2.6.1 Vue d'ensemble des composants [24] 40
2.6.1.1 ESP32-CAM 41
2.6.1.2 ESP32 (Contrôleur principal) 42
2.6.1.3 Servo-moteur SG90 43
2.7 Schéma de Câblage principal 45
2.8 Conclusion Partielle 46

Chapitre 3 Réalisation, Simulation et interprétation des résultats 47
3.1 Introduction 47
3.2 Interface du système 47
3.2.1 Connexion (Login) 47
3.2.2 Page d'accueil (Dashboard) 48
3.2.3 La page des utilisateurs 51
La page Pour la reconnaissance faciale 56
3.2.4 56
3.2.5 La page Pour l'historique des Accès 60

3.3 Interprétation des résultats 64
3.4 Conclusion Partielle 66
Conclusion générale 67
Bibliographie 69
Annexe Extrait du code de la fonction pour la reconnaissance faciale 73
Extrait du code pour l'encodage des images 74
Extrait du code une fois le visage reconnue 75
Extrait du code pour l'affichage des informations 76
Extrait du code pour un visage non reconnue 77
Extrait du code pour exporter la présence 78
Extrait du code Arduino pour contrôler l'esp32 et le servo moteur 79
Extrait du code du fichier de connexion entre l'application et le coté matériels 80
Extrait du code du fichier de connexion entre l'application et le coté matériels 82
Extrait du code Arduino pour actionner le servomoteur 83
..... 83

Sigles et abréviations

AES : Advanced Encryption Standard
API : Application Programming Interface
CPU : Central Processing Unit
CSS : Cascading Style Sheets
ESP32 : Microcontrôleur 32 bits développé par Expressif
ESP32-CAM : Version de l'ESP32 intégrant une caméra OV2640
GPIO : General Purpose Input Output
GUI : Graphical User Interface (Interface Graphique Utilisateur)
HOG : Histogram of Oriented Gradients
HTML : Hypertext Markup Language
LBP : Local Binary Patterns
LDA : Linear Discriminant Analysis
PSRAM : Pseudo Static Random Access Memory
PWM : Pulse Width Modulation
RAM : Random Access Memory
SG90 : Servo-moteur utilisé pour le contrôle d'accès
SQL : Structured Query Language
ULPGL : Université Libre des Pays des Grands Lacs

Liste des tableaux

Tableau I:Documentation de cas d'utilisation « S'authentifier » 25
Tableau II:Documentation de cas d'utilisation « ajouter un voyageur » 26
Tableau III:Documentation de cas d'utilisation « Rechercher les historique de la personne » 27
Tableau IV:Tableau de Résultats 64

Liste des figures

Figure 1:serrure poignet 9
Figure 2:serrure carénée 9
Figure 3:Principe de fonctionnement de base d'un système de reconnaissance faciale 12
Figure 4 : Architecture generale du systeme 22
Figure 5: Diagramme de cas d'utilisation de super admin et de l'etudiant, personnel 24
Figure 6:Diagramme de Sequence 30
Figure 7:Diagramme de classes 31
Figure 8:Installation django et tailwind css 33
Figure 9:Structure du projet Django et Tailwing css 33
Figure 10 : L'image resultat de la methode des points particulie (Landmark) 36
Figure 11:Installation des dependance pour face_recognitioin 37
Figure 12:Exemple de vecteur facial 38
Figure 13**:**Exemple d'un chiffrement en Python 40
Figure 14:Configuration des broches principale ESP32 Cam 42
Figure 15: Image pour esp32 : 42
Figure 16: Configuration des broches ESP32 Microcontroller 43
Figure 17: L'image du servomoteur : 43
Figure 18:Schema de Cablage principale 45
Figure 19: Interface de connexion 48
Figure 20:La page d'accueil (Dashboard) 49
Figure 21: Page d'accueil avec la partie centrale 50
Figure 22: : Page d'accueil avec changement des statistiques 51
Figure 23: La page pour choisir un utilisateur a enreregistré 52
Figure 24: Champs pour ajouter un etudiant 53
Figure 25: Formulaire ajout etudiant avec le bouton enregistrer 53

Figure 26: Formulaire d'enregistreerment du Personnel 54
Figure 27: Formulaire personnel avec l'image déjà choisie 55
Figure 28: Message de validation du formulaire 55
Figure 29: Page Profils utilisateurs 56
Figure 30: Page pour démarrer la camera 57
Figure 31: Page pour la reconnaissance faciale avec le résultat « Personne reconnue » 58
Figure 32: Message :« Veuillez attendre 60 secondes avant votre prochaine présence » 59
Figure 33 : Message "Acces recusé" pour une personne inconnue 59
Figure 34: Message « Plusieurs visages détectés , veillez isoler une seule personne devant la caméra» 60
Figure 35: Historique des Accès 61
Figure 36: les presence filtrées selon : « La date , utilisateurs, le statuts » 62
Figure 37: le cas pour filtrer un inconnu 62
Figure 38: Fiche de présences selon l'attente , soit « par utilisateur, date, statut » 63
Figure 39: Figure de la fonction pour la raiconnaissance facial 73
Figure 40: Extrait du code pour l'encodage des images 74
Figure 41: Extrait du code une fois le visage reconnue 75
Figure 42: Extrait du code pour l'affichage des informations après reconnaissance 76
Figure 43: Extrait du code pour un visage non reconnue 77
Figure 44: Extrait du code pour exporter la présence 78
Figure 45: Extrait du code arduino pour controler l'ESP32 et le SERVO MOTEUR 79
Figure 46: Extrait du code du fichier de connexion entre l'application et le coté matériels 81
Figure 47: Extrait du code du fichier de connexion entre l'application et le coté matériels 82
Figure 48:Extrait du code Arduino pour actionner le servomoteur 83
Figure 49:Extrait du code Arduino pour actionner le servomoteur 84

0. Introduction générale

0.1. Contexte et Généralités sur le thème

La sécurité et l'efficacité administrative sont des enjeux cruciaux pour les institutions académiques modernes. À l'Université Libre des Pays des Grands Lacs (ULPGL), les méthodes traditionnelles de contrôle d'accès et de gestion de présence, telles que les registres papier et les cartes pour étudiant, présentent plusieurs inconvénients, notamment des risques de fraude, de perte de données et une gestion inefficace des informations.

Pour ce projet, nous utiliserons des technologies avancées pour assurer une détection et une reconnaissance faciale efficaces. OpenCV, une bibliothèque de vision par ordinateur reconnue, sera employée pour la détection de visage et face recognition pour la reconnaissance des visages. Python, grâce à sa flexibilité et sa puissance, facilitera le traitement des images et l'implémentation des algorithmes nécessaires. Enfin, les microcontrôleurs ESP32 et ESP32-CAM joueront un rôle crucial en permettant la capture d'images et le contrôle des accès en temps réel, renforçant ainsi la sécurité et l'efficacité administrative.

0.2. Identification et formulation du problème

La gestion des accès et la prise de présence à l'ULPGL reposent sur des méthodes traditionnelles, telles que les registres papier et les carte d'étudiant, qui sont chronophages et sujettes à diverses formes de fraude. Les principaux problèmes que nous avons pu trouver sont :

1. **Risque de Fraude et d'Erreur** : Les registres papier et les cartes peuvent être facilement manipulés, entraînant des erreurs ou des fraudes
2. **Efficacité et Fiabilité** : Le processus manuel est lent et peu fiable, entraînant des pertes de temps et une inefficacité générale.
3. **Sécurité** : L'absence d'un système de contrôle d'accès augmente les risques d'insécurité sur le campus.

Pour remédier à ces problèmes, il est nécessaire de concevoir et de réaliser un système de contrôle d'accès et de prise de présence utilisant la reconnaissance faciale. Ce système doit :

- Automatiser la vérification de l'identité des étudiants et du personnel pour un accès sécurisé aux locaux.
- Simplifier et accélérer le processus de prise de présence, en éliminant la nécessité de registres papier ou bien de cartes d'étudiants physiques.
- Assurer l'intégrité et la précision des enregistrements de présence, réduisant ainsi les risques de fraude et d'erreur.
- Fournir des mises à jour en temps réel des données de présence, facilitant ainsi la gestion administrative.

0.3. Questions de recherche

Vu le problème présenté, la question principale est : Comment mettre au point un système de contrôle d'accès et de prise de présence utilisant la reconnaissance faciale pour l'ULPGL afin de faciliter la vérification de l'identité des étudiants et du personnel pour un accès sécurisé aux locaux ?

Les questions secondaires incluent:

1. Comment la reconnaissance faciale peut-elle améliorer la sécurité sur le campus de l'ULPGL ?

2. Comment le système de reconnaissance faciale peut-il automatiser la prise de présence de manière fiable et efficace ?
3. Quelle sont les technologies a utilisées pour mettre au point ce système ?

0.4. Formulation des hypothèses

Pour baliser notre travail en nous référant aux préoccupations évoquées ci-dessus, nous avons émis les hypothèses suivantes :

1. **Amélioration de la sécurité sur le campus** : Si un système de reconnaissance faciale était installé sur le campus de l'ULPGL, il pourrait potentiellement améliorer la sécurité en limitant l'accès uniquement aux personnes autorisées, réduisant ainsi les risques d'intrusion et d'usurpation d'identité.
2. **Automatisation de la prise de présence** : Si la reconnaissance faciale était utilisée pour automatiser la prise de présence, elle pourrait améliorer la fiabilité et l'efficacité du suivi des présences, en éliminant les erreurs humaines et les fraudes.
3. **Technologies à utiliser** : Si des technologies avancées telle Open CV de python était intégré au système de reconnaissance faciale et ESP 32 camera pour la capture d'image en temps réel et son microcontrôleur ESP 32, elles pourraient permettre une meilleure précision et vitesse de reconnaissance, rendant le système plus performant et adapté à un environnement universitaire.

0.5. Justification du choix du sujet et motivations

0.5.1 Justification du Choix du Sujet

• **Problème Réel et Actuel** : Les méthodes traditionnelles de gestion des accès et de prise de présence à l'ULPGL sont obsolètes et posent des problèmes d'efficacité et de sécurité. Ce projet vise à moderniser ces processus.

- **Avantages Technologiques** : La reconnaissance faciale offre une solution plus sécurisée et fiable pour l'identification des individus, avec des mises à jour en temps réel des données de présence.
- **Innovation et Modernisation** : En intégrant des technologies avancées, ce projet positionne l'ULPGL à la pointe de la modernisation technologique.

0.5.2 Motivations

- **Amélioration de la Sécurité** : Un système de contrôle d'accès basé sur la reconnaissance faciale réduit les risques d'intrusion et garantit que seules les personnes autorisées peuvent accéder aux installations universitaires.
- **Efficacité et Gain de Temps** : Automatiser le processus de prise de présence permet de gagner un temps précieux et de minimiser les erreurs humaines.
- **Développement Personnel et Académique** : Ce projet offre une opportunité de développer des compétences techniques avancées et de contribuer de manière significative à la communauté académique.
- **Répondre aux Défis de la Modernisation** : Dans un monde digitalisé, il est crucial pour les institutions académiques de s'adapter aux nouvelles technologies.

0.6. Énoncé des objectifs de recherche

0.6.1. L'objectif général

Concevoir et réaliser un système de contrôle d'accès et de prise de présence utilisant la reconnaissance faciale pour l'ULPGL.

0.6.2. Les objectifs opérationnels/spécifiques

1. Évaluer les besoins de l'ULPGL en termes de sécurité et de gestion de la présence.

2. Mettre en place un système de reconnaissance faciale pour contrôler l'accès au campus.
3. Développer un système automatisé de prise de présence basé sur la reconnaissance faciale.
4. Recueillir des informations auprès des parties prenantes, y compris les étudiants, le personnel et les responsables de la sécurité.
5. Développer une architecture système intégrant des modules de reconnaissance faciale basés sur OpenCV.
6. Utiliser les microcontrôleurs ESP32 et ESP32-CAM pour les contrôles d'accès et la capture des images faciales.
7. Programmer les algorithmes de reconnaissance faciale en utilisant Python et OpenCV.
8. Intégrer les microcontrôleurs ESP32 pour la communication et le contrôle des accès.
9. Créer une interface utilisateur pour la gestion et la visualisation des présences.

0.7. Méthodologie et délimitation du travail

0.7.1. Méthodologie de la Recherche

Tout travail scientifique nécessite l'usage des certaines méthodes et techniques pour aboutir à des résultats concluants, nous allons recourir à certaines d'entre elles, que nous énumérons dans les lignes qui suivent :

• **La méthode historique** qui nous permet d'acquérir des connaissances sur l'histoire de l'ULPGL.

• **La méthode analytique** qui nous permet d'analyser systématiquement les données et les informations en notre disposition.

• **La technique documentaire** pour décortiquer les ouvrages trouvés à la bibliothèque et sur Internet ayant un lien avec notre sujet.

Pour orienter notre travail nous nous sommes focalisés sur quelques points ci-dessous :

0.7.2. Délimitation du Travail

Ce travail se concentre sur la conception et la réalisation d'un système de contrôle d'accès et de prise de présence utilisant la reconnaissance faciale, avec les limites suivantes :

• **Portée Géographique:** Le projet sera déployé et testé uniquement sur le campus Salomon de l'ULPGL.

•**Technologies Utilisées:** Seules les technologies OpenCV et Face recognition pour la reconnaissance faciale, et les microcontrôleurs ESP32, ESP32-CAM et le servo moteur SG90 pour le contrôle d'accès seront utilisées.

• **Données:** Les données de reconnaissance faciale et de présence seront limitées aux étudiants et au personnel dans les salles ou auditoires de l'ULPGL.

9.1. Structure du mémoire/ Subdivision du travail

Le projet sera subdivisé en plusieurs étapes clés pour assurer une réalisation méthodique et structurée :

Chap1 Revue de la littérature

- Analyse des systèmes existants : Exploration des technologies et systèmes de reconnaissance faciale déjà implémentés pour mettre au point ce système

- Identification des besoins de l'ULPGL : Description des enjeux de sécurité et des limites des méthodes traditionnelles actuelles à l'ULPGL.
- Justification du projet : Motivation du choix du système de reconnaissance faciale pour résoudre les problèmes identifiés.

Chap2 Conception et développement du Système :

- Définition des spécifications techniques et fonctionnelles.
- Sélection des technologies et outils nécessaires (OpenCV, Face recognition, ESP32, ESP32-CAM, Python,).

Chap3 Development et Integration:

- Développement des algorithmes de reconnaissance faciale avec OpenCV, Face recognition.
- Programmation des microcontrôleurs ESP32 et ESP32-CAM pour la capture d'images et le contrôle d'accès.
- Intégration des différents modules pour former un système cohérent.

Chapitre 1 Revue de la littérature sur les Systèmes de Contrôle d'Accès et la Reconnaissance Faciale

1.1 Généralités sur les Systèmes de Contrôle d'Accès

Les systèmes de contrôle d'accès sont des solutions de sécurité conçues pour réguler et surveiller les entrées et sorties dans des espaces protégés. Ces systèmes sont utilisés dans une variété de contextes, allant des entreprises et des résidences privées à des lieux très fréquentés comme les aéroports, ainsi qu'à des institutions sensibles telles que les banques et les installations militaires [1].

1.1.1 Évolution Historique des Systèmes de Contrôle d'Accès

Depuis des siècles, les dispositifs de contrôle d'accès ont évolué, passant des clés physiques aux systèmes numériques avancés. Les premiers systèmes mécaniques ont été supplantés par des solutions électroniques, telles que les claviers à code, les cartes magnétiques et, plus récemment, les biométries. L'évolution vers des systèmes plus intelligents et autonomes reflète la nécessité croissante de sécurité et d'efficacité dans la gestion des accès [2].

1. **Serrures Mécaniques Traditionnelles** : Basées sur des clés physiques, elles restent largement utilisées mais sont vulnérables au vol et à la duplication [3].

Les Figure 1 et Figure 2 ci-dessous montrent les serrures mécanique traditionnelles :

Figure 1:serrure poignet [3] [4]

Figure 2:serrure carénée [5]

2. **Systèmes Électroniques** : Introduction des cartes magnétiques et RFID, offrant plus de flexibilité et de contrôle, mais exposées à des risques de piratage.
3. **Reconnaissance Biométrique** : Utilise des caractéristiques biologiques uniques telles que les empreintes digitales, la rétine ou les traits faciaux pour l'identification. Ces systèmes sont plus sûrs car ils sont basés sur des éléments uniques et difficiles à falsifier.

1.1.2 Les Composants Clés des Systèmes Modernes de Contrôle d'Accès

Les systèmes modernes de contrôle d'accès combinent divers composants technologiques pour sécuriser les entrées. Ces composants incluent : [4]

- **Microcontrôleurs et Capteurs** : Utilisés pour la collecte de données et le contrôle des accès.
- **Interfaces Utilisateur (GUI)** : Offrent une interface visuelle pour la gestion des utilisateurs et la configuration des accès.

  Systèmes de Communication : Incluent Wi-Fi, Bluetooth, et Ethernet pour connecter les composants du système et permettre une gestion à distance.

1.2 La Reconnaissance Faciale

La reconnaissance faciale est une technologie biométrique qui identifie et authentifie les personnes en analysant leurs traits faciaux. [5] Elle est de plus en plus populaire dans les systèmes de contrôle d'accès en raison de sa rapidité, de son efficacité, et de sa commodité.

1.2.1 Historique

La reconnaissance faciale automatique est un concept relativement nouveau. Le premier système semi-automatisé de la reconnaissance faciale a été développé dans les années 1960,

il nécessite à l'administrateur de localiser les yeux, les oreilles, le nez et la bouche sur la photo et de saisir les distances calculées et les ratios à un point de référence commun, qui ont ensuite été comparés aux données de référence. [5]

1.2.2 Les Principes Fondamentaux de la Reconnaissance Faciale

La reconnaissance faciale se base sur des algorithmes qui détectent, analysent, et comparent des traits faciaux spécifiques. Ces systèmes capturent une image d'un visage, extraient des caractéristiques (comme la distance entre les yeux, la largeur du nez, la forme de la bouche), et les comparent à une base de données pour vérifier l'identité de l'utilisateur.

Processus de Reconnaissance représenté à la *Figure 3* est décrit comme suit : [6]

1. **Détection du Visage** : Le système localise un visage dans une image ou un flux vidéo en temps réel.
2. **Extraction des Caractéristiques** : Identification et extraction des points clés du visage (yeux, nez, bouche, contour).
3. **Comparaison et Vérification** : Le visage capturé est comparé à une base de données de visages autorisés pour une correspondance.
4. **Décision** : En fonction de la correspondance trouvée, le système accorde ou refuse l'accès

La **Erreur ! Source du renvoi introuvable.** ci-dessous montre le principe de fonctionnement de base d'un système de raiconnaissance faciale :

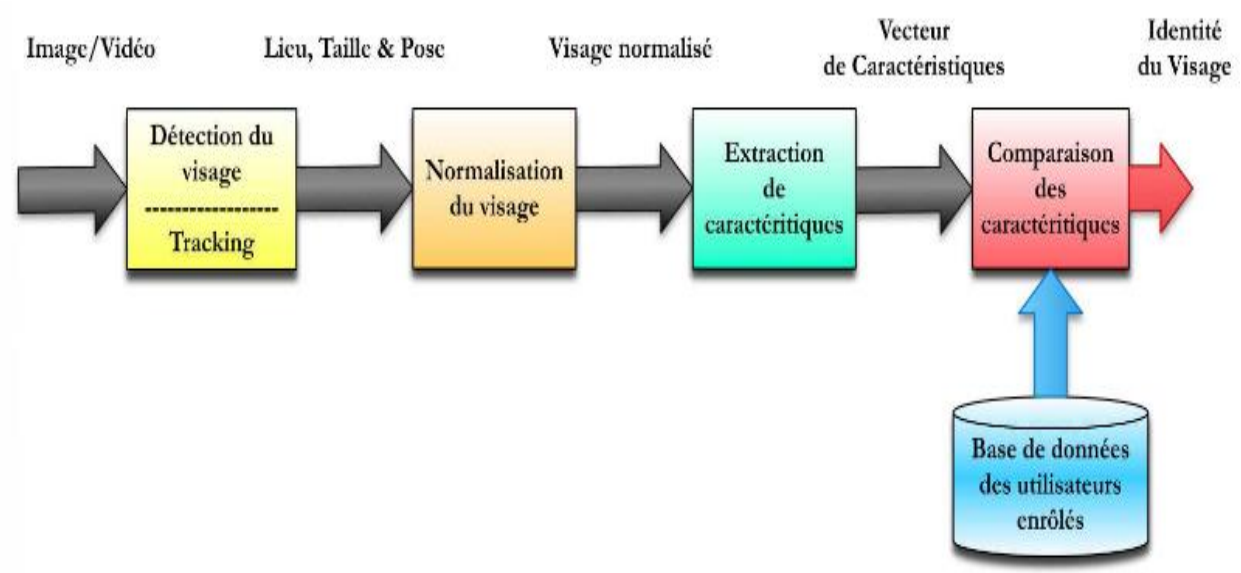


Figure 3:Principe de fonctionnement de base d'un système de reconnaissance faciale [3][19]

1.2.3 Avantages et Limites des Systèmes de Reconnaissance Faciale

La reconnaissance faciale présente de nombreux avantages, notamment une sécurité renforcée et une automatisation simplifiée. Cependant, elle présente également des défis techniques et éthiques.

Avantages : [7]

- **Non-Intrusif** : Contrairement aux empreintes digitales ou aux scans rétiniens, la reconnaissance faciale ne nécessite pas de contact physique, réduisant les risques de contamination et augmentant le confort utilisateur.
- **Rapidité et Efficacité** : Permet une identification en temps réel, accélérant le processus d'accès.
- **Difficile à Contrefaire** : Comparée aux badges ou codes, la reconnaissance faciale utilise des traits uniques, rendant la falsification complexe.

Limites : [3]

- **Influence des variations de la pose :**

Les changements d'orientation et les changements de l'angle d'inclination du visage engendrent de nombreuses modifications d'apparence dans les images collectées. Les rotations en profondeur engendrent l'occultation de certaines parties du visage comme pour les vues de trois-quarts. D'autre part, elles amènent des différences de profondeur qui sont projetées sur le plan 2D de l'image, provoquant des déformations qui font varier la forme globale du visage. Ces déformations qui correspondent à l'étirement de certaines parties du visage et la compression d'autres régions font varier aussi les distances entre les caractéristiques faciales

- **Influence des changements d'éclairage :**

L'intensité et la direction d'éclairage lors de la prise de vue influent énormément sur l'apparence du visage dans l'image. En effet, dans la plupart des applications courantes, des changements dans les conditions d'éclairage sont inévitables, notamment lorsque les vues sont collectées à des heures différentes, en intérieur ou en extérieur. Etant donnée la forme spécifique d'un visage humain, ces variations d'éclairage peuvent y faire apparaître des ombres accentuant ou masquant certaines caractéristiques faciales [6].

- **Influence des expressions faciales :**

Les visages sont des éléments non rigides. Les expressions faciales véhiculent constamment des émotions. Ainsi donc, combinées avec les déformations induites par la parole, ces expressions peuvent produire des changements d'apparence importants.

- **Influence du vieillissement normal :**

Diverses études indiquent qu'il existerait une modification des capacités de reconnaissance des visages, au cours du vieillissement normal, qui débuterait vers l'âge de 50 ans [7] [8]et s'amplifierait à partir de 70 ans [9].

1.3 Présentation du Servomoteur SG90

Le servomoteur SG90 est un composant essentiel dans les systèmes de contrôle d'accès modernes pour l'automatisation de l'ouverture et de la fermeture des portes. Ce moteur miniature est contrôlé par des signaux électriques pour déplacer un axe avec une grande précision [10] .

1.3.1. Caractéristiques Techniques du Servomoteur SG90

Le SG90 est un servomoteur à rotation limitée qui offre un contrôle précis de la position angulaire sur un intervalle de 180 degrés. Il est couramment utilisé dans les applications robotiques, les modèles réduits, et les systèmes automatisés [10].

Specifications:

- **Angle de Rotation** : 0 à 180 degrés.
- **Tension de Fonctionnement** : 4.8V à 6V.
- **Couple Maximale** : Environ 1.8 kg/cm à 4.8V.
- **Poids** : Environ 9g.
- **Type de Contrôle** : Signal PWM (modulation de largeur d'impulsion). [11]

1.3.2. Intégration dans les Systèmes de Contrôle d'Accès

Le servomoteur SG90 est intégré dans le système de contrôle d'accès pour actionner le verrouillage mécanique en fonction des résultats de la reconnaissance faciale fournis par l'ESP32 Cam.

- **Signal de Contrôle** : Le microcontrôleur envoie un signal PWM au servomoteur SG90, qui se positionne pour déverrouiller ou verrouiller la porte.
- **Interaction avec l'ESP32 Cam** : Une fois la reconnaissance faciale validée, l'ESP32 Cam envoie une commande de rotation au servomoteur SG90, permettant l'ouverture de la porte.

1.4 L'ESP32 Cam : Pilier du Système de Reconnaissance Faciale

L'ESP32 Cam est un module de microcontrôleur équipé d'une caméra, conçu pour la reconnaissance faciale et d'autres applications de vision par ordinateur. Il est au cœur du système de contrôle d'accès, où il traite les images, exécute des algorithmes de reconnaissance faciale, et contrôle le servomoteur. [12]

L'ESP32 Cam combine un microcontrôleur puissant avec une caméra OV2640, capable de capturer des images en haute résolution.

Caractéristiques :

- **Processeur** : Dual-core 32-bit Xtensa LX6 CPU, jusqu’à 240MHz.
- **Mémoire** : 520KB SRAM + 4MB PSRAM.
- **Caméra** : Capteur OV2640, résolution jusqu'à 2 MP.
- **Connectivité**: Wi-Fi 802.11 b/g/n, Bluetooth 4.2.

- **Capacités de Reconnaissance Faciale** : Intégration d'algorithmes tels que Viola-Jones et CNN.

1.4.2. Application dans le Système de Contrôle d'Accès

Le fonctionnement de l'ESP32 dans le système de contrôle d'accès est basé sur un flux de travail simple mais efficace :

1. **Capture d'Image** : L'ESP32 Cam capture en temps réel des images de l'individu
2. **Traitement et Analyse** : Les algorithmes de reconnaissance faciale sont appliqués pour identifier ou vérifier l'individu.
3. **Commande de Contrôle** : En cas de correspondance, une commande est envoyée au microcontrôleur ESP32 afin d'actionner servomoteur SG90 pour déverrouiller la porte. [12]

1.5 Comparaison avec les Systèmes Traditionnels de Contrôle d'Accès

L'utilisation de la reconnaissance faciale couplée au servomoteur SG90 présente des avantages significatifs par rapport aux systèmes traditionnels :

- **Sécurité Améliorée** : Réduit les risques de vol ou de contournement par rapport aux systèmes de badges ou de codes PIN.
- **Automatisation Intégrale** : Grâce au contrôle automatique par la reconnaissance faciale, il n'y a pas besoin d'intervention humaine.
- **Facilité d'Installation et d'Entretien** : Le système est facile à installer avec des composants abordables et nécessite peu de maintenance.

1.6 Quelques méthodes de détection de visages

Aujourd'hui, on utilise plusieurs méthodes de détection et de localisation de visages. Nous pouvons classer ces méthodes de localisation faciale comme suit :

1.6.1 Méthode du traitement automatique du visage

C'est une technologie rudimentaire, elle caractérise les visages par des distances et des proportions entre des points particuliers comme les deux yeux, le nez, et les coins de la bouche. Aussi éprouvé que les autres technologies, le traitement automatique du visage est le plus efficace dans des situations de capture d'image avec peu d'éclairage. [13]

1.6.2 Méthode d'Eigenface

Les « Eigenfaces » furent le premier type de caractérisation utilisé avec succès dans des traitements faciaux tels que la détection et la reconnaissance du visage [14]. Elle utilise une représentation des éléments caractéristiques d'une image de visage à partir d'images modèles en niveau de gris. Des variantes d'Eigenface sont fréquemment utilisées comme base pour d'autres méthodes de reconnaissance [14].

1.6.3 Template Matching Methods

Ces méthodes consistent à créer des modèles caractéristiques d'un visage entier ou de sous-partie de visage comme la bouche et le nez. La détection se fait ensuite par corrélation de ces modèles avec les candidats. Elles peuvent aussi être plus efficaces si on les combine à d'autres méthodes. Elles présentent l'avantage de s'exécuter très rapidement mais elles demandent beaucoup de temps d'entrainement [13].

1.6.4 Méthode d'analyse des points particuliers(facial landmarks)

Elle est la technique d'identification faciale la plus utilisée. Cette dernière se rapproche de Eigenface, mais elle est capable de s'adapter à des changements d'aspect facial (sourire, froncement des sourcils, ...). Les ingénieurs numériques l'utilisent souvent [15]

1.6.5 Méthode LDA (Linear discriminant analysis) Fisher

Elle fait partie des techniques d'analyse discriminante prédictive. Il s'agit d'expliquer et de prédire l'appartenance d'un individu à une classe (groupe) prédéfinie à partir de ses caractéristiques mesurées à l'aide de variables prédictives [16].

Cette méthode consiste à créer un détecteur fort en formant un ensemble de classifieurs faibles pour les caractéristiques locales trouvées dans des positions spécifiques du visage

1.6.6 Filtre de Haar

La détection du visage dans ce filtre est réalisée par un filtre multi-échelles de Haar. Les propriétés d'un visage sont décrites dans un fichier XML. Elles ne sont pas choisies au hasard et reposent sur un échantillon de quelques centaines d'images tests [16].

1.6.7 Methode LBP (Local Binary Patterns)

Les LBP **(Modèles binaires locaux)** ont également été utilisés pour la détection du visage ou ce dernier est subdivisé en sous–régions carrées de taille égale sur lesquelles sont calculées les caractéristiques LBP. Les vecteurs obtenus sont ensuite concaténés pour obtenir le vecteur de caractéristiques final. [16]

1.7 Conclusion du partielle

Ce chapitre a présenté une revue approfondie des technologies de contrôle d'accès et de reconnaissance faciale, en mettant l'accent sur l'intégration du servomoteur SG90 et de l'ESP32-Cam. Ces technologies offrent une solution innovante et efficace pour les applications de sécurité, combinant automatisation, précision et coût abordable.

Dans le prochain chapitre, nous nous focaliserons sur la conception et la mise en œuvre pratique du système. Cela inclura des schémas de circuit, des algorithmes de reconnaissance faciale, ainsi que des exemples de programmation pour le servomoteur SG90.

Chapitre 2
Conception et Développement du système

2.1 Introduction

Ce chapitre présente la façon dont nous avons conçu et développé le système. Il met en avant les choix technologiques, l'architecture générale, les étapes clés du développement, ainsi que quelques diagrammes UML.

Nous détaillons ici la création du système de contrôle d'accès et de prise de présence basé sur la reconnaissance faciale. Ce type de projet demande une bonne planification, le choix de technologies adaptées et un développement structuré, pour garantir sa fiabilité et son efficacité.

2.2 Spécifications Fonctionnelles et Techniques

2.2.1 Spécifications Fonctionnelles

Le système doit répondre aux besoins de l'ULPGL en assurant les fonctionnalités suivantes :

- **Authentification des individus** : Permettre la reconnaissance faciale des étudiants et du personnel.
- **Contrôle d'accès sécurisé** : Autoriser ou refuser l'accès en fonction de l'identité vérifiée.
- **Enregistrement des présences** : Stocker l'heure de reconnaissance soit l'entrée en temps réel dans une base de données.
- **Interface de gestion** : Fournir une interface utilisateur intuitive pour la gestion des accès et des historiques.

2.2.2 Spécifications Techniques

Les technologies utilisées pour la mise en œuvre du système incluent :

- **Reconnaissance faciale** : la bibliothèque face_recognition en Python. [18]
- **Microcontrôleurs** : ESP32-CAM pour la capture d'images et ESP32 le microcontrôleur pour le contrôle des accès via un servomoteur SG90. [19]
- **Base de données** : SQLITE pour stocker les informations des utilisateurs et leurs présences.
- **Interface utilisateur** : Développée en Django (Python) avec un frontend en HTML, Tailwind css et JavaScript. [20]
- **Arduino** un environnement pour gérer les composants électroniques pour l'accès des étudiant ou personnel. [19]

2.3 Architecture du Système

L'architecture du système repose sur trois principaux composants [20] :

1. **Module de capture et traitement d'image** : Captation des images via l'ESP32-CAM, prétraitement avec OpenCV et extraction des caractéristiques faciales.
2. **Serveur de traitement et base de données** : Comparaison des visages détectés avec ceux enregistrés, décision d'accès, et enregistrement des présences.

3. **Interface utilisateur** : Affichage des informations, gestion des utilisateurs et consultation des historiques.

Le schéma de la Figure 4 illustre l'architecture générale du système :

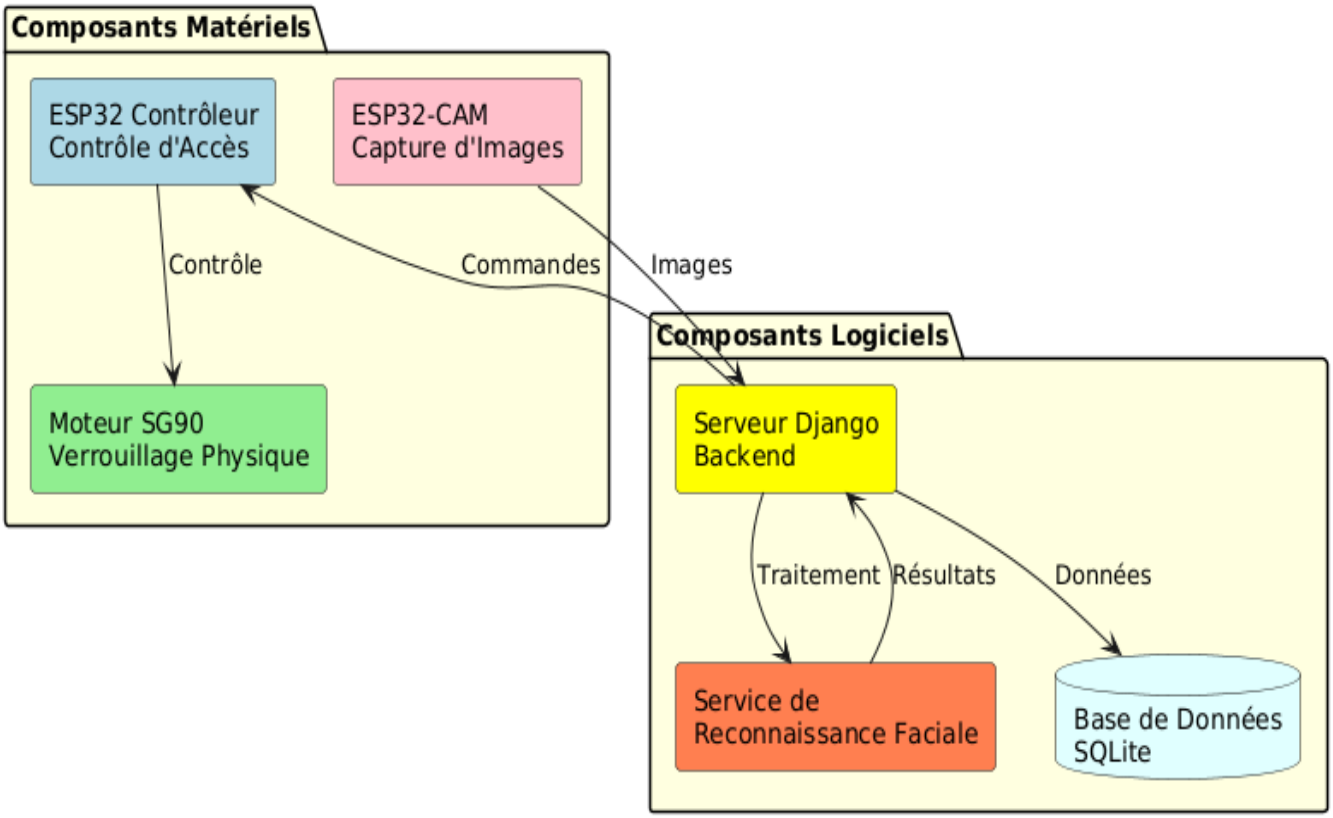


Figure 4 : Architecture generale du systeme

2.4 Présentation des diagrammes

2.4.1 Brève description des spécifications fonctionnelles

Le Système doit offrir à l'utilisateur la possibilité de :

- Enregistrer les informations des étudiants et du personnel de l'ULPGL ;
- Valider ou refuser l'accès des étudiants ou personnels dans des locaux de l'ULPGL ;
- Voir l'historique des accès des étudiants et personnels ;
- Vérifier la similarité d'un étudiant ou personnel voyageur déjà enregistré dans la base de données ;
- Afficher les informations de la personne une fois il y a reconnaissance on envois de message « accès autoriser » sinon « accès refuser (inconnu) » ;
- Enregistrer un nouvel utilisateur ;

2.4.2 Diagramme de cas d'utilisation

2.4.2.1 Présentation des acteurs

- **Le super Admin :** c'est un acteur principal, cet utilisateur représente un utilisateur inscrit par le système qui peut exécuter les fonctionnalités du système.
- **L'étudiant ou personnel :** est un acteur qui interagit avec le système pour certaines fonctionnalités (lors de la capture de l'image par exemple).

2.4.2.2 Présentation des cas d'utilisations de tout le système

La Figure 5 représente les cas d'utilisation du super admin ou employer de l'ULPGL :

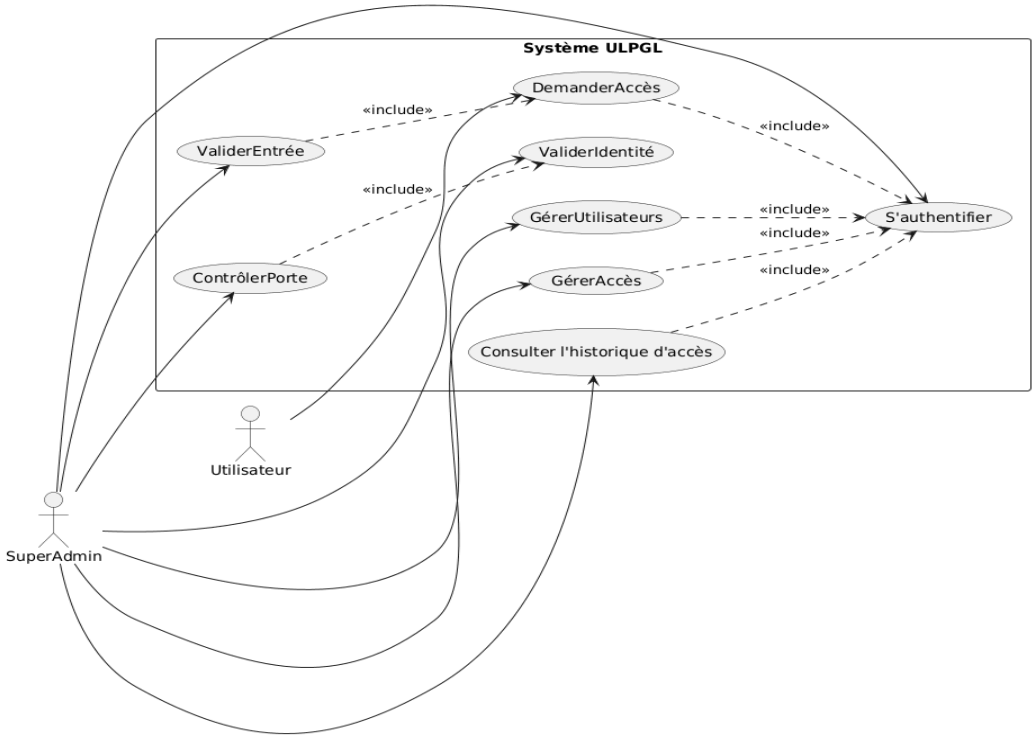


Figure 5: Diagramme de cas d'utilisation de super admin et de l'etudiant, personnel

2.4.2.3 Documentation des cas d'utilisations

Dans cette partie nous présentons les différents tableaux descriptifs de cas d'utilisation.

- **Cas d'utilisation s'authentifier :**

L'admin de l'ULPGL pour pouvoir accéder à d'autres interfaces (qui sont des fonctionnalités) doit s'authentifier ;

- **Documentation de cas d'utilisation « S'authentifier »**

Tableau I:Documentation de cas d'utilisation « S'authentifier »

ID	1
Description brève du CU	Connexion au système
Acteur principal	Le super admin
Événements déclencheur	Valider le lien du système dans le navigateur ou après s'être déconnecté.
Précondition	L'acteur doit avoir un compte dans le système
Scénario principal	0. L'acteur demande l'accès au système 1. Le système affiche le formulaire d'authentification 2. L'acteur est invité à compléter et valider le formulaire d'authentification. 3. Le système vérifie les données saisies par l'acteur 4. Le système informe que l'on est authentifié et affiche l'interface du système (le Dashboard)
Scénario alternatif	4.a. Les informations d'authentifications ne sont pas valides, le système en informe l'acteur, qu'il doit mettre le mot de passe ou nom d'utilisateur correct.

- **Documentation de cas d'utilisation « Créer un étudiant ou personnel »**

Tableau II:Documentation de cas d'utilisation « ajouter un voyageur »

Cas d'utilisation	Ajouter un étudiant ou un personnel
ID	2
Description brève du système	L'acteur enregistre les infos des étudiants et personnels dans le système
Acteur principal	Le super admin ou l'admin,
Acteur secondaire	Les étudiants et personnels
Événements déclencheur	Cliquer sur le bouton ajouter un voyageur
Précondition	L'acteur doit être authentifié
Scénario principal	1. Le système affiche l'interface de l'ajout utilisateurs (un formulaire) 2. L'acteur est invité à compléter et valide le formulaire enregistrer dans le système le formulaire d'ajout utilisateur. 3. Le système vérifie les données saisies par l'acteur. 4. Les informations des utilisateurs (étudiants, personnels) sont enregistrées et le système vérifie la similarité entre la photo du nouvel utilisateur et les photos de personnes suspectes se trouvant la base de données. 5. Le système affiche résultat de la vérification

Scénario alternatif	4.a. Les informations d'ajout utilisateurs ne sont pas valides, le système en informe l'acteur.
Post condition	Augmentation du nombre d'utilisateur dans le système

- **Documentation de cas d'utilisation « Rechercher l'historique des présences »**

Tableau III:Documentation de cas d'utilisation « Rechercher les historique de la personne »

Cas d'utilisation	Rechercher l'historique des présence par étudiant ou personnel
ID	4
Description brève du système	Rechercher le l'utilisateur(étudiant ou personne)
Acteur principal	L'admin,
Acteur secondaire	Etudiants ou personnel
Précondition	L'acteur doit être authentifié
Scénario principal	1. Le système affiche l'interface relative 2. L'acteur voit les historiques de présence et est invité à saisir le nom de la personne pour voir son historique. 3. Le système revoit les informations concernant la personne retrouvée.
Scénario alternatif	3.a. Le système ne trouve pas de personne avec le nom entré par l'acteur, personne non retrouver

2.4.2 Diagramme de Séquence

Les diagrammes de séquences sont les représentations graphiques des interactions entre les acteurs et le système selon un ordre chronologique dans la formulation UML. Ce diagramme permet de montrer les interactions d'objets dans le cadre d'un scénario d'un diagramme de cas d'utilisation.

Alors la figure ci-dessous montres le diagramme de séquence de notre système :

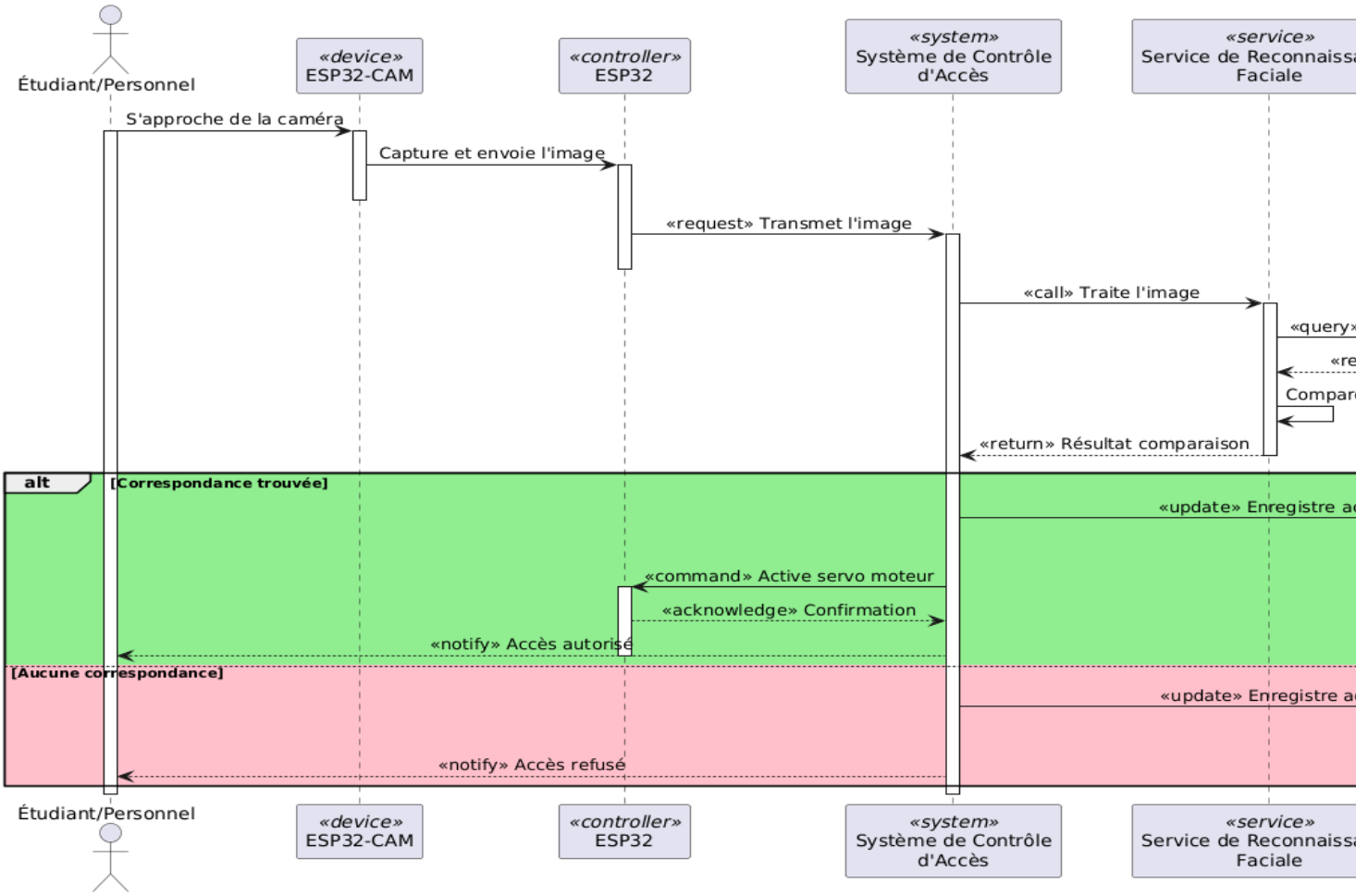


Figure 6:Diagramme de Sequence

2.4.3 Diagramme des Classes

Le diagramme de cas d'utilisation permet de visualiser un système à travers les yeux de ses utilisateurs, tandis que le diagramme de classes en dévoile l'organisation interne. Il aide à comprendre comment les différents objets du système interagissent pour donner vie aux fonctionnalités attendues.
Une classe, c'est un peu comme un modèle : elle définit la structure d'un objet en regroupant ses caractéristiques (appelées attributs) et les actions qu'il peut effectuer (les méthodes). Les attributs reflètent l'état de l'objet, tandis que les méthodes définissent ce qu'il est capable de faire.

Voici, en image, le diagramme de classes du système :

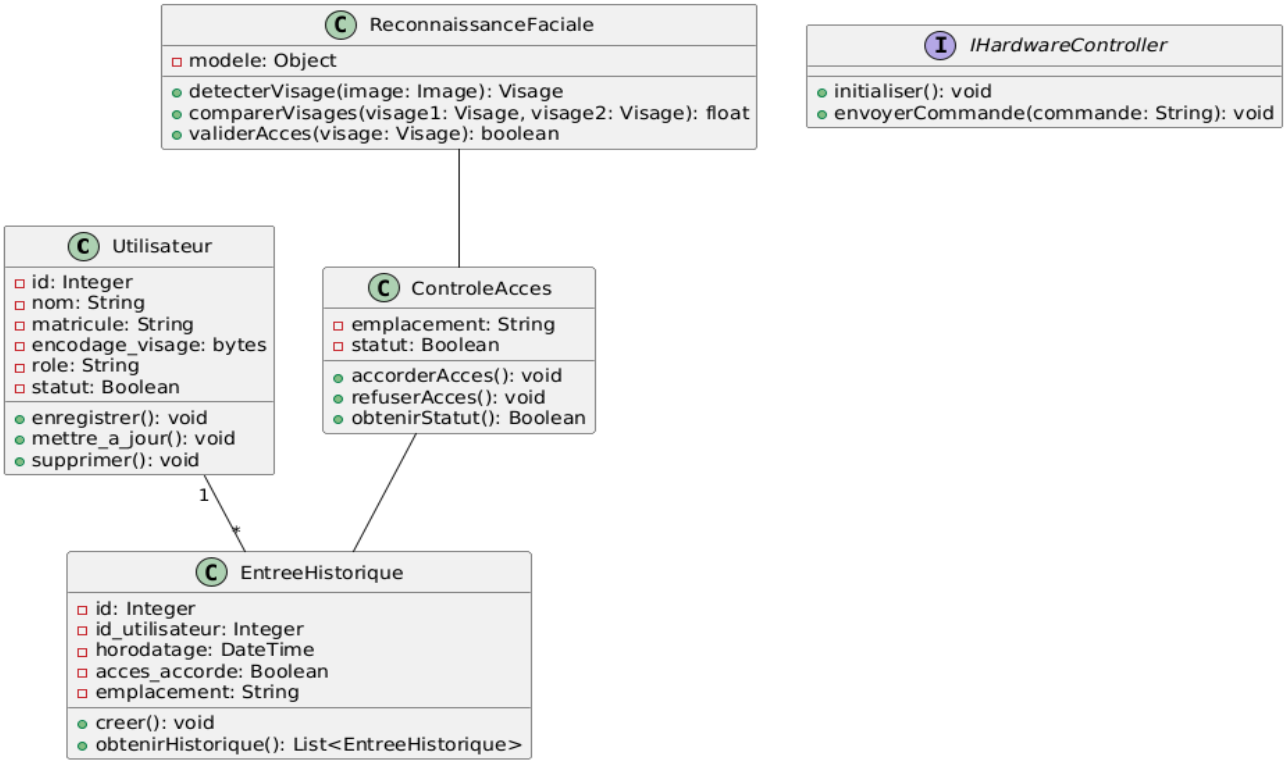


Figure 7:Diagramme de classes

2.5 Présentation des technologies

Notre système est composé de trois grandes parties :

- **La partie interface (front)**
- **La partie qui gère la reconnaissance faciale**
- **La partie serveur (backend)**

2.5.1 L'interface utilisateur

L'interface utilisateur joue un rôle clé dans l'expérience des utilisateurs. Nous avons choisi **Django Templates**, un moteur de templates intégré à Django, associé à **Tailwind CSS**, un framework moderne facilitant le design responsif et épuré. Grâce à cette combinaison, nous pouvons concevoir des interfaces dynamiques, ergonomiques et élégantes, tout en séparant la logique métier du code de présentation.

2.5.1.1 Avantages

Les avantages que l'utilisation de Django offre sont : [22]

- **Intégration fluide avec Django** pour une cohérence entre le front et le backend.
- **Système de templates modulaire** permettant la réutilisation des composants.
- **Tailwind CSS facilite un design rapide et personnalisé** avec une approche utility-first.
- **Séparation claire des responsabilités**, garantissant une meilleure maintenabilité.
- **Performance et sécurité renforcées**, grâce au rendu côté serveur et à la protection CSRF intégrée.
- **Communauté active**, assurant un support et des mises à jour régulières.

2.5.1.2 installation

La Figure 8 ci-dessous représente l'installation Django et Tailwind css :

```
# Installation de Django
pip install django

# Installation de Tailwind CSS
pip install django-tailwind

# Ajout de 'tailwind' dans INSTALLED_APPS
python manage.py tailwind init

# Installation des dépendances npm
python manage.py tailwind install
```

Figure 8:Installation django et tailwind css

2.5.1.2 Structure du projet Django et Tailwind css

La Figure 9 ci-dessous représente la structure du projet Django Tailwind css :

```
project/
├── manage.py
├── project/
│   ├── __init__.py
│   ├── settings.py
│   ├── urls.py
│   └── wsgi.py
├── templates/
│   ├── base.html
│   └── components/
├── static/
│   ├── css/
│   └── js/
└── theme/
    └── static_src/
        └── tailwind.config.js
```

Figure 9:Structure du projet Django et Tailwing css

2.5.2 La Reconnaissance Faciale

2.5.2.1. Présentation de la bibliothèque *face_recognition*

La reconnaissance faciale dans notre système repose sur la bibliothèque `face recognition` de Python, basée sur dlib. Cette bibliothèque propose une solution précise et efficace pour la détection et la reconnaissance des visages, en exploitant des réseaux de neurones profonds pour extraire les caractéristiques faciales. [24]

2.5.2.1.1 Composants principaux

- **Détection des visages**
  - Utilisation de HOG (*Histogram of Oriented Gradients*) pour une détection rapide

    La méthode **HOG (Histogram of Oriented Gradients)** est une technique utilisée pour détecter les visages en analysant la **répartition des gradients d'intensité** dans une image. Cette approche est basée sur le fait que les formes et contours des objets (comme un visage) peuvent être représentés par des gradients d'intensité plutôt que par des couleurs spécifiques.

  - Capacité d'identifier plusieurs visages dans une même image
- **Extraction des caractéristiques faciales**
  - Identification de 68 points de repère par visage (yeux, nez, bouche, contours)
  - Localisation précise des traits pour améliorer l'alignement et la normalisation des visages

❖ **Encodage des visages**

- Transformation de chaque visage en un vecteur unique de 128 dimensions
- Utilisation d'un réseau de neurones pré-entraîné pour générer cette empreinte numérique
- Compression des informations faciales en une représentation compacte et efficace

❖ Méthode de reconnaissance Faciale utiliser

Parmi les méthodes que nous avons listées dans le chapitre précèdent, notre système se rapproche le plus de la **"Méthode d'analyse des points particuliers" (facial landmarks)** car :

- Il identifie des points clés du visage
- Il s'adapte aux changements d'expression
- Il est robuste aux variations d'éclairage et d'angle

La Figure 10 ci-dessous montre resultat de la methode des points particulie (Landmark) :

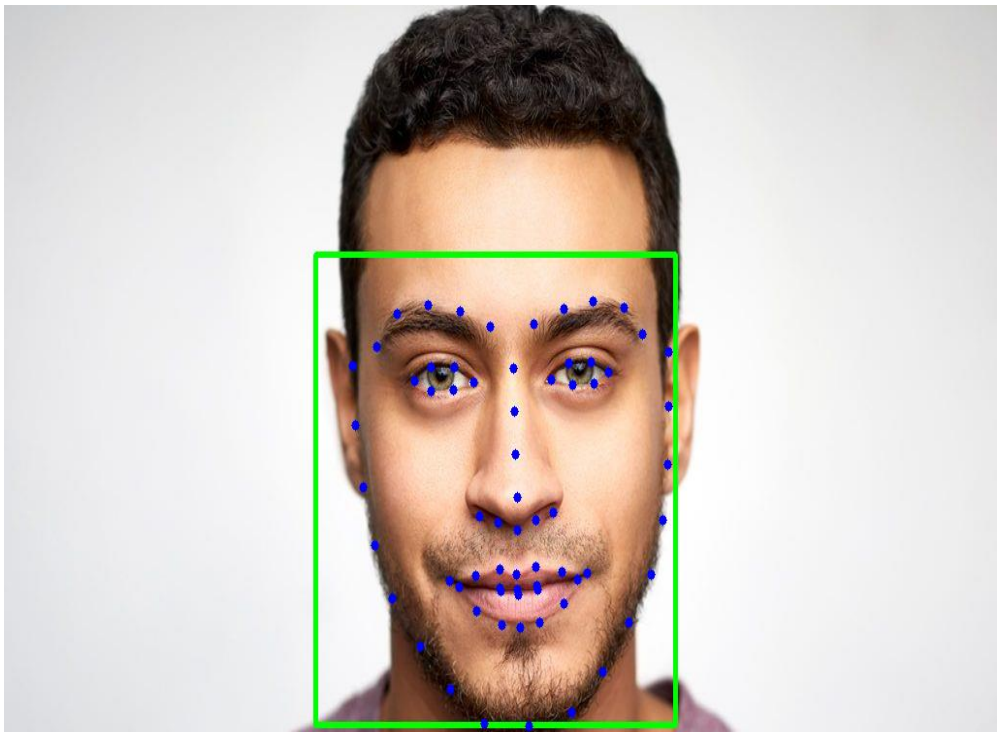


Figure 10 : L'image resultat de la methode des points particulie (Landmark) [26]

2.5.2.1.2 Installation et prérequis

Pour utiliser *face_recognition*, il faut d'abord installer quelques dépendances :

- Cmake
- Dlib
- Face_recognition

```
# Installation des dépendances système
pip install cmake
pip install dlib

# Installation de la bibliothèque face_recognition
pip install face_recognition
```

Figure 11:Installation des dependance pour face_recognitioin

Configuration requise :

- Python 3.6
- *Dlib*
- *Numpy*
- *PIL* (Python Imaging Library)

2.5.2.1.3 Intégration avec Django

Dans notre application, nous avons intégré *face_recognition* au backend Django comme suit :

2.5.2.1.3.1 Performance et optimisation

❖ **Traitement en temps réel**

- Optimisation des performances pour une reconnaissance rapide
- Utilisation du CPU pour accélérer le processus (si disponible)
- Mise en cache des visages fréquemment utilisés pour éviter des calculs inutiles

❖ **Précision et fiabilité**

- Haut taux de reconnaissance dans des conditions d'éclairage et d'angles variés
- Ajustement possible du seuil de reconnaissance pour réduire les faux positifs
- Algorithmes robustes aux variations d'expression faciale

2.5.2.1.3.2 Sécurité et protection des données

❖ **Stockage sécurisé**

- Les empreintes faciales sont stockées sous forme de vecteurs chiffrés
- Sécurisation des données sensibles par des mécanismes de chiffrement avancés

2.5.2.1.4 Qu'est-ce qu'un vecteur en reconnaissance faciale ?

Un **vecteur** est simplement une liste de nombres qui représente un visage de manière unique. Avec la bibliothèque *face_recognition*, chaque visage est converti en un **vecteur de** 128 dimensions. Ce vecteur contient les caractéristiques faciales importantes extraites par un réseau de neurones. [25]

Exemple de vecteur facial illustré par la Figure 12 **:**

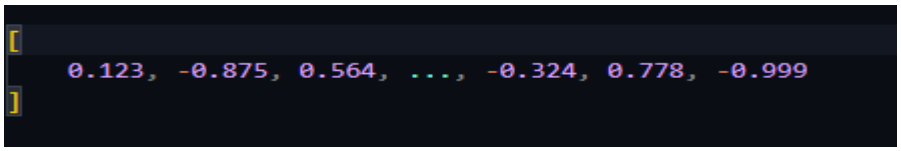


Figure 12:Exemple de vecteur facial

Chaque nombre du vecteur représente une caractéristique du visage, comme la distance entre les yeux, la forme du nez, ou l'orientation de la bouche. Deux visages similaires auront des vecteurs proches, tandis que deux visages différents auront des vecteurs très éloignés.

2.5.2.1.5 Pourquoi chiffrer ces vecteurs ?

Même si ces vecteurs ne sont pas directement des images, ils contiennent **des** informations **biométriques sensibles**. Pour éviter qu'ils soient volés ou utilisés à mauvais escient, ils peuvent être **chiffrés** avant d'être stockés dans une base de données.

2.5.2.1.6 Comment chiffre-t-on un vecteur facial ?

On peut utiliser un **algorithme de chiffrement** pour transformer le vecteur en une version illisible pour toute personne non autorisée. Voici deux méthodes courantes :

- **AES (Advanced Encryption Standard)**
  Un chiffrement symétrique qui protège les données tout en permettant leur utilisation par des systèmes autorisés.

- **Hachage (SHA-256,Bcrypt,etc.)**
  Contrairement au chiffrement réversible, le hachage convertit le vecteur en une empreinte irréversible, utile pour vérifier une identité sans stocker directement les données sensibles.

Exemple d'un chiffrement en Python à la **Figure 13**:

```python
# Exemple de vecteur encodé sous forme de chaîne
face_vector = "[0.123, -0.875, 0.564, ..., -0.324, 0.778, -0.999]".encode()

# Chiffrement du vecteur
encrypted_vector = cipher_suite.encrypt(face_vector)
print("Vecteur chiffré :", encrypted_vector)

# Déchiffrement si nécessaire
decrypted_vector = cipher_suite.decrypt(encrypted_vector)
print("Vecteur déchiffré :", decrypted_vector.decode())
```

Figure 13:Exemple d'un chiffrement en Python

- Avantages du chiffrement des vecteurs faciaux :

  - Sécurité : Protège les données biométriques contre le vol ou l'utilisation non autorisée.
  - Confidentialité : Même si quelqu'un accède à la base de données, il ne pourra pas comprendre ou utiliser les données.
  - Conformité : Respect des réglementations comme le RGPD (Europe) ou la CCPA (États-Unis) qui exigent une protection des données biométriques.

2.6 Partie matérielle [20]

2.6.1 Vue d'ensemble des composants [24]

Notre système de contrôle d'accès repose sur trois composants matériels essentiels pour assurer une performance fluide et une gestion efficace de l'accès :

- **ESP32-CAM** : utilisé pour capturer des images et reconnaître les visages.

- **ESP32** : qui sert de contrôleur principal pour coordonner l'ensemble du système.
- **Servomoteur SG90** : pour contrôler physiquement l'accès en ouvrant ou fermant la porte.

2.6.1.1 ESP32-CAM

- **Processeur** : ESP32-S (dual-core) jusqu'à 240 MHz
- **Mémoire** : 4MB PSRAM
- **Caméra** : 2MP OV2640 avec LED flash et un angle de vue de 60°.
- **Connectivité** : Wi-Fi et Bluetooth
- **Alimentation** : 5V
- **Broches** principales :
  - GPIO 0 : Flash LED
  - GPIO 12, 13, 14, 15 : VSPI (communication)
  - GPIO 16, 17 : RX/TX série

Configuration des broches :

```yaml
GPIO 0  : Flash LED
GPIO 12 : VSPI MISO
GPIO 13 : VSPI MOSI
GPIO 14 : VSPI CLK
GPIO 15 : VSPI CS
GPIO 16 : RX2
GPIO 17 : TX2
```

Figure 14:Configuration des broches principale ESP32 Cam

2.6.1.2 ESP32 (Contrôleur principal)

La Figure 15 ci-dessous montre un ESP32 :

Figure 15: Image pour esp32 : [29]

Voici les spécifications techniques du ESP32 :

- **Processeur** : Dual-core Xtensa® 32-bit LX6
- **Fréquence** : jusqu'à 240 MHz
- **Mémoire** : 520KB SRAM
- **Connectivité**: Wi-Fi 802.11 b/g/n + Bluetooth 4.2

- **GPIO** : 38 broches programmables

Dans notre système, le ESP32 doit jouer de Contrôle du servo-moteur c'est à dire gérer l'ouverture et la fermeture de l'accès.

Pour son utilisation, les broches ESP32 sont configurées comme montré sur la Figure 16 :

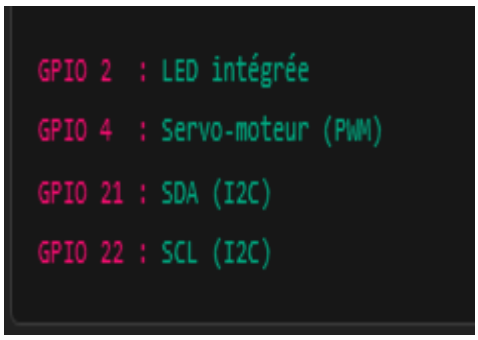


Figure 16: Configuration des broches ESP32 Microcontroller

2.6.1.3 Servo-moteur SG90

La Figure 17 ci-dessous montre un ESP32 :

Figure 17: L'image du servomoteur : [30]

Spécifications techniques :

- **Tension de fonctionnement** : 4.8V-6V
- **Couple** : 1.8 kg/cm
- **Vitesse** : 0.1s/60°
- **Angle de rotation** : 180°
- **Poids** : 9g
- **Signal PWM** : 50Hz (période 20ms)

La Figure 17 montre le servo moteur SG90 pour notre prototypage . En parlant des faits réels, Le servomoteur MG996R est idéal pour le contrôle d'accès, car il peut actionner des portes pesant plus de 10 kg grâce à son couple élevé (il peut exercer une force importante pour faire tourner ou déplacer un mécanisme), des engrenages métalliques et il peut actionner des verrous ou mécanismes d'ouverture de porte plus exigeants qu'un simple micro servo SG90

2.7 Schéma de Câblage principal

Les principaux composants sont connectés sur le ESP32 comme montré sur le schéma de Figure 18 :

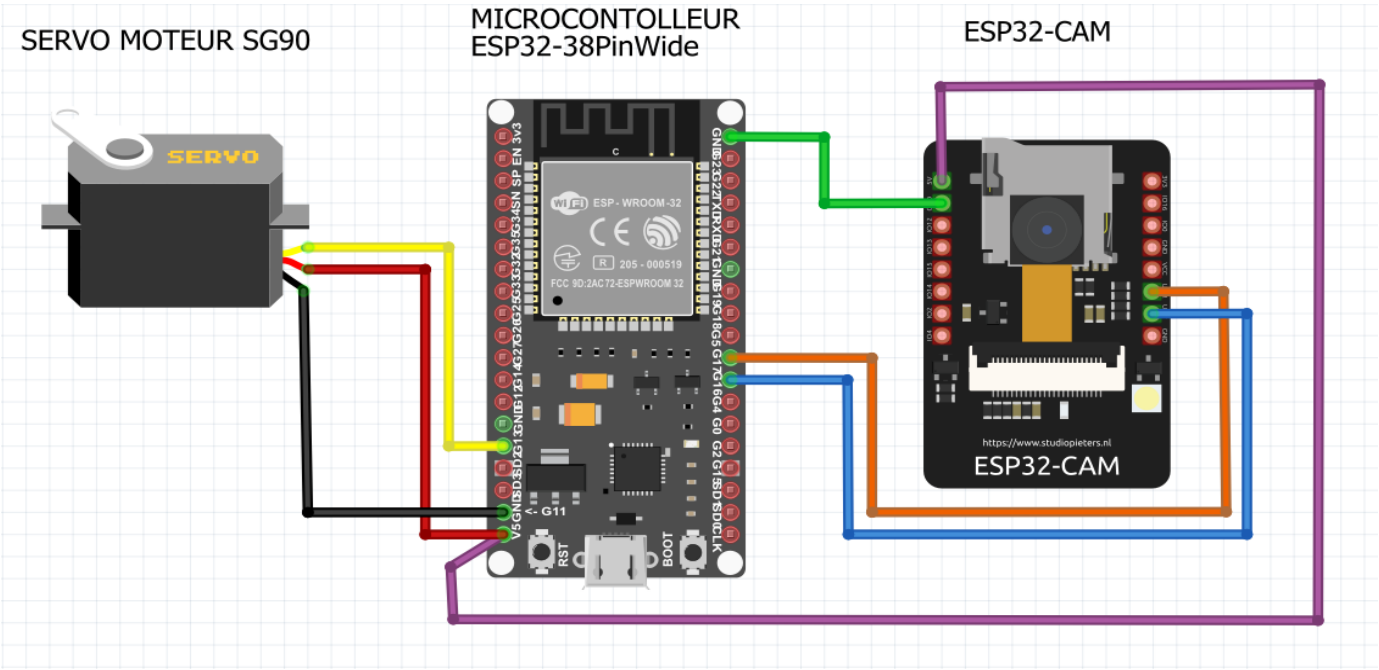


Figure 18:Schema de Cablage principale

2.8 Conclusion Partielle

Dans ce chapitre, nous avons abordé les aspects conceptuels et détaillé les technologies utilisées dans le système. En ce qui concerne la conception, nous avons présenté plusieurs diagrammes, notamment le diagramme de cas d'utilisation, le diagramme de séquence, et le diagramme de classes.
La section suivante a été dédiée à la présentation des technologies, où nous avons expliqué les outils et les technologies intégrés dans notre système. Dans le prochain chapitre, nous entrerons dans le détail du système lui-même et des résultats des tests réalisés.

Chapitre 3 Réalisation, Simulation et interprétation des résultats

3.1 Introduction

Dans cette phase, nous avons concrétisé notre vision en développant un système complet de contrôle d'accès et de prise de présence par reconnaissance faciale. Le système se compose d'un module de capture d'images avec l'ESP32-CAM, d'un module de traitement sur ESP32 pour commander un servomoteur SG90, et d'une interface utilisateur intuitive conçue avec Django et Tailwind CSS. L'objectif est de valider le fonctionnement de l'ensemble et de démontrer la robustesse et la fiabilité de notre solution en conditions réelles.

3.2 Interface du système

3.2.1 Connexion (Login)

Grâce à cette page l'admin se connecte au système. Pour cela l'admin remplit le formulaire montré à la Figure 19 (en donnant son adresse, son nom d'utilisateur et mot de passe) puis il doit appuyer sur le bouton se connecter. Si le nom d'utilisateur et le mot de passe sont valides, (le client) est authentifié et il est redirigé à la page d'accueil.

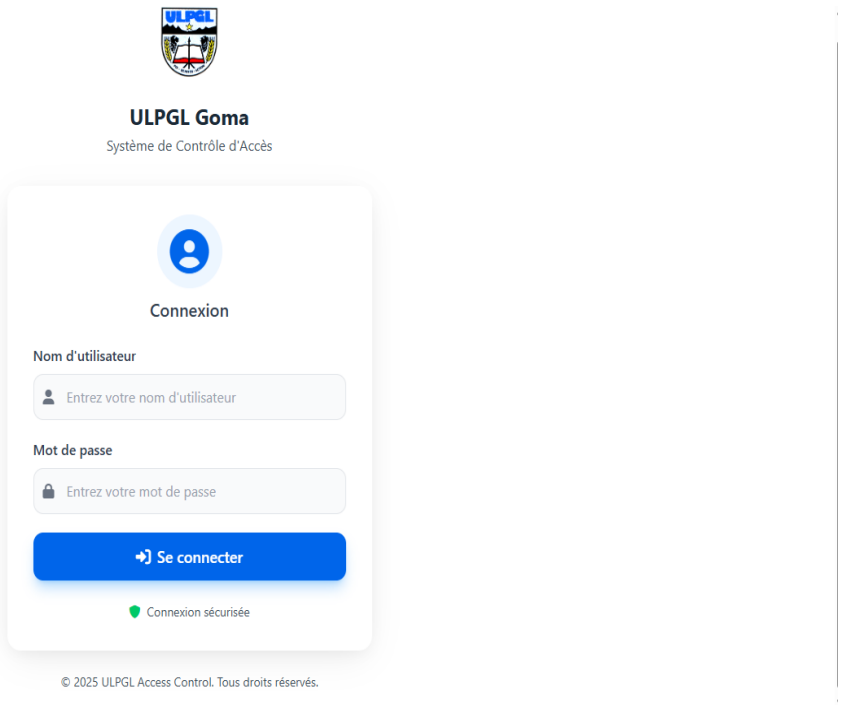


Figure 19: Interface de connexion

3.2.2 Page d'accueil (Dashboard)

Après connexion, l'utilisateur est redirigé vers la page d'accueil montrée à la Figure 20, qui offre un aperçu global des données présentes dans la base de données.

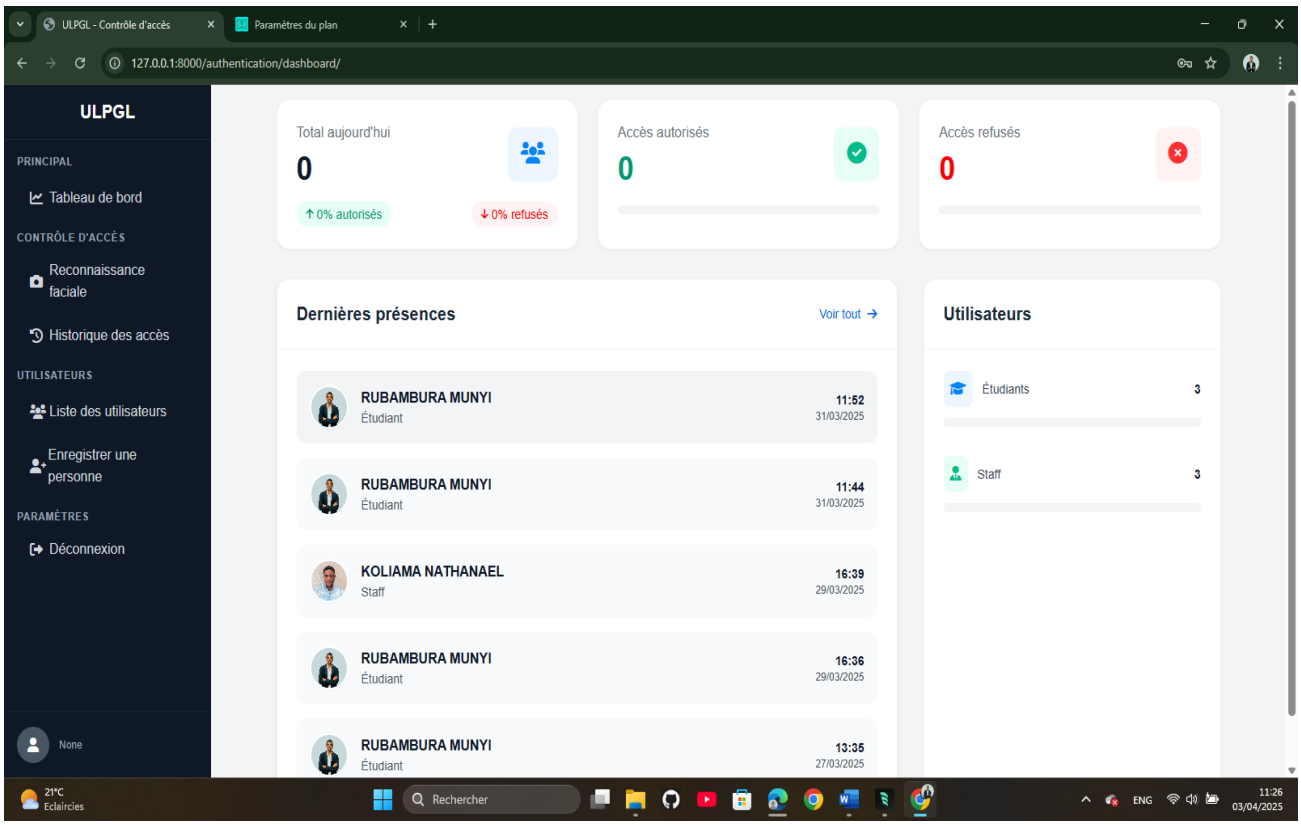


Figure 20:La page d'accueil (Dashboard)

Il est important de signaler que toutes les pages qui suivront auront 2 grandes parties, à savoir :

- **Le sidebar :** c'est la partie se trouvant à gauche, il contient les menus essentiels que l'on a dans le système.

- **La partie centrale** : elle change de contenu selon une fonctionnalité à l'autre.

La Figure 21 ci-dessous montre la partie centrale du Dashboard :

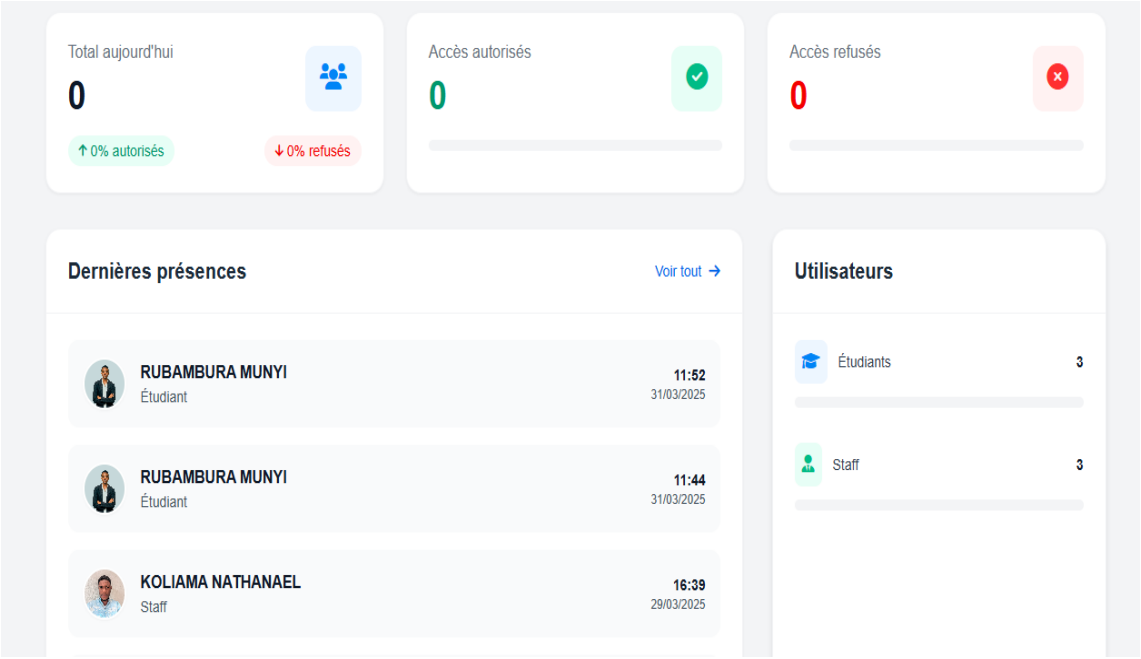


Figure 21: Page d'accueil avec la partie centrale

Chaque fois qu'il y a reconnaissance faciale sur une image directement le Dashboard change ses statistiques comme illustré à la Figure 22 après 8 tentatives d'accès dont 4 accordées. À chaque reconnaissance faciale sur une image, le tableau de bord met automatiquement à jour ses statistiques, comme illustré dans la Figure 22 après huit tentatives d'accès, dont quatre ont été accordées.

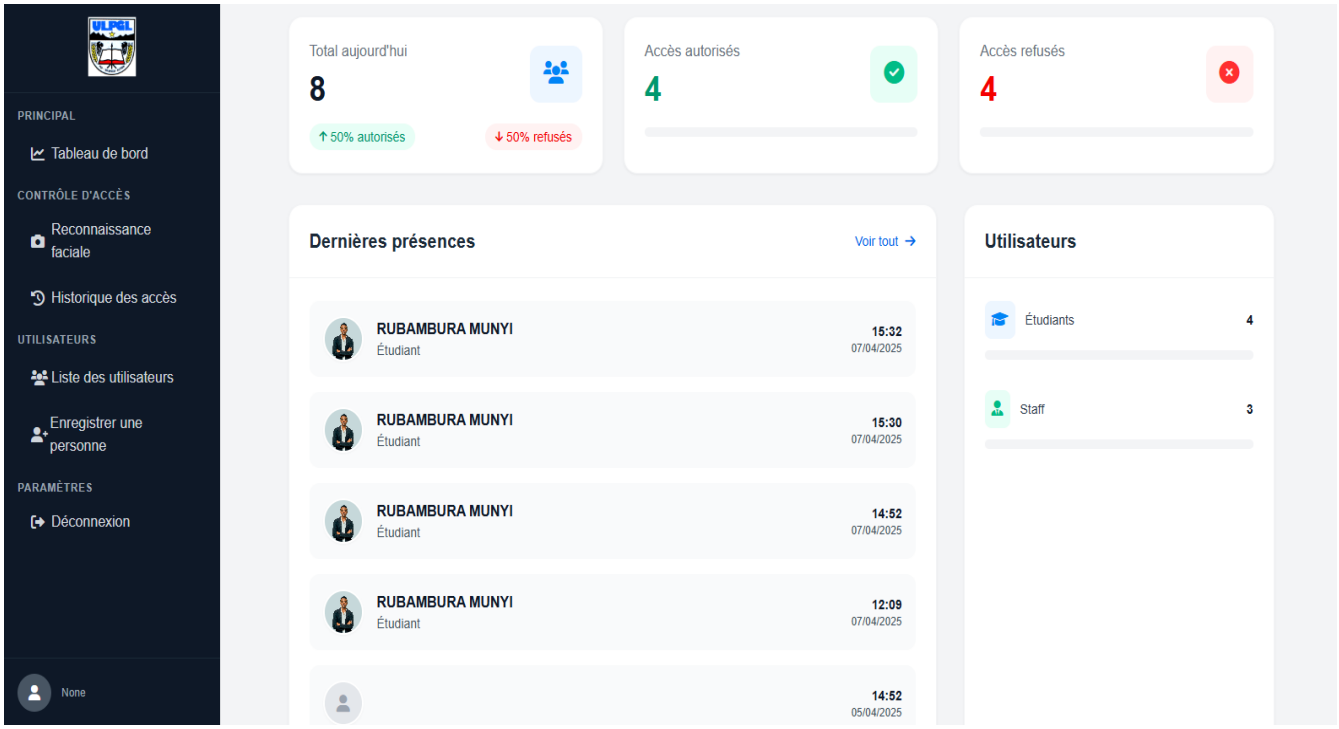


Figure 22: : Page d'accueil avec changement des statistiques

3.2.3 La page des utilisateurs

Cette page, montrée à la Figure 23, permet l'enregistrement des utilisateurs, qu'il s'agisse d'étudiants ou de membres du personnel.

L'ajout d'un utilisateur commence par le choix du type d'utilisateur à enregistrer : personnel ou étudiant.

Figure 23: La page pour choisir un utilisateur a enreregistré

Une fois le type d'utilisateur choisi, il est possible de remplir et soumettre le formulaire contenant les informations de la personne.

- **Le formulaire pour un étudiant :**

  Ce formulaire est composé de :

  - Des champs pour soumettre le nom, le postnom, le matricule, la promotion, la faculté, et le champ pour soumettre une photo comme illustré sur la Figure 24 :

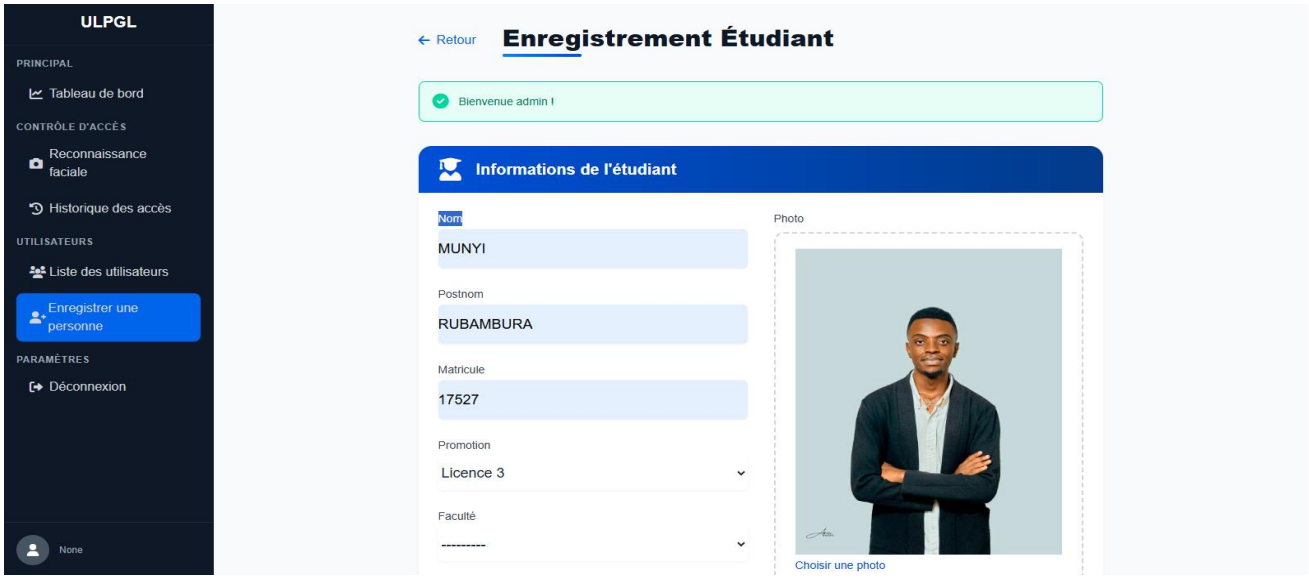


Figure 24: Champs pour ajouter un etudiant

Une fois le formulaire complété, il peut être validé en cliquant sur le bouton « Enregistrer ».

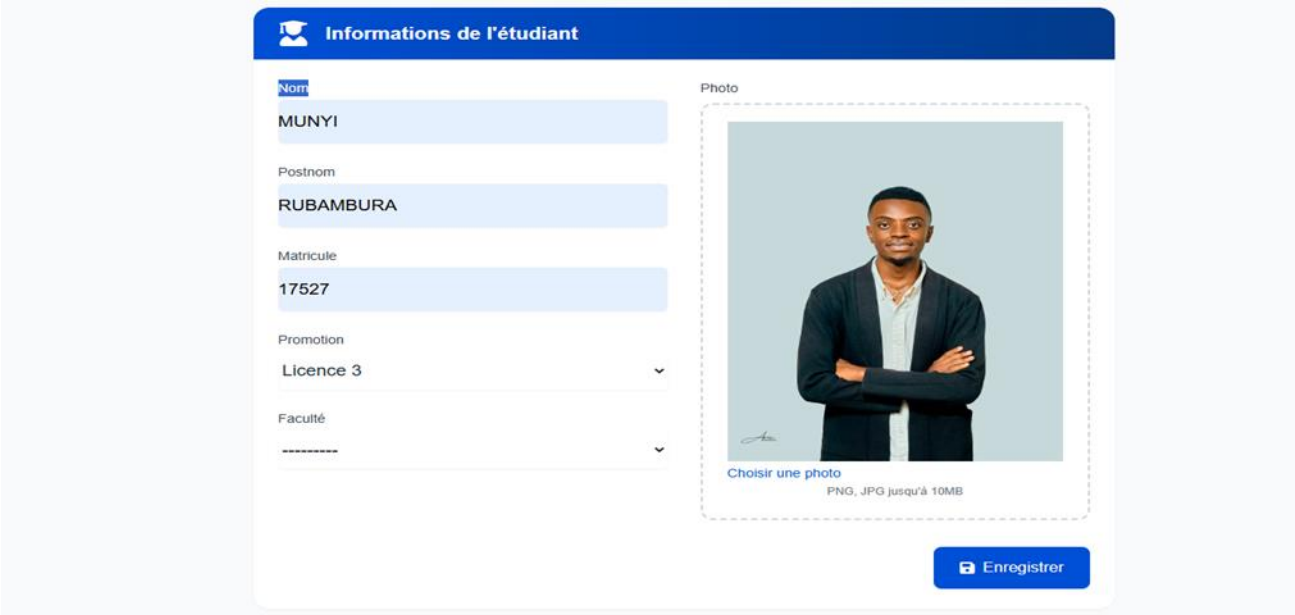


Figure 25: Formulaire ajout etudiant avec le bouton enregistrer

- **Formulaire pour un membre du personnel :**

Pour le personnel, le formulaire montré à la Figure 26 comporte les champs suivants : nom, post-nom, fonction, département, un champ pour sélectionner une photo, ainsi qu'un bouton « Enregistrer ».

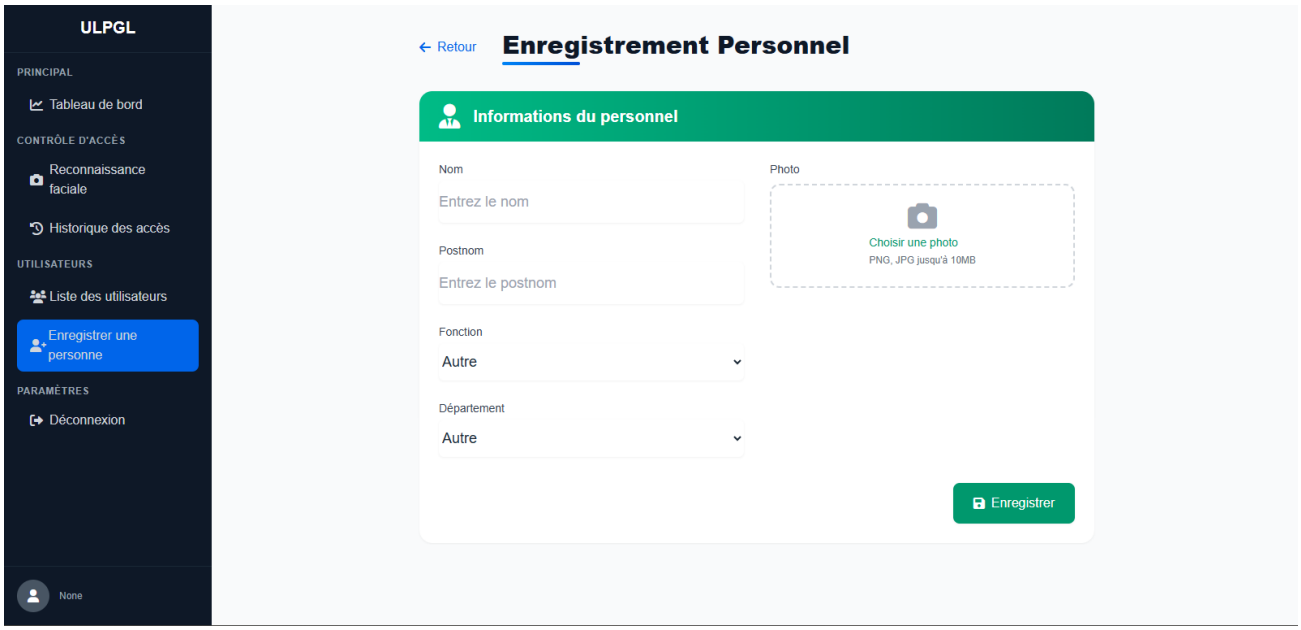


Figure 26: Formulaire d'enregistreerment du Personnel

Une fois on clique sur la zone choisir une photo, on peut déjà voir le répertoire des photos s'ouvrir et en choisir une comme l'illustre la Figure 27 :

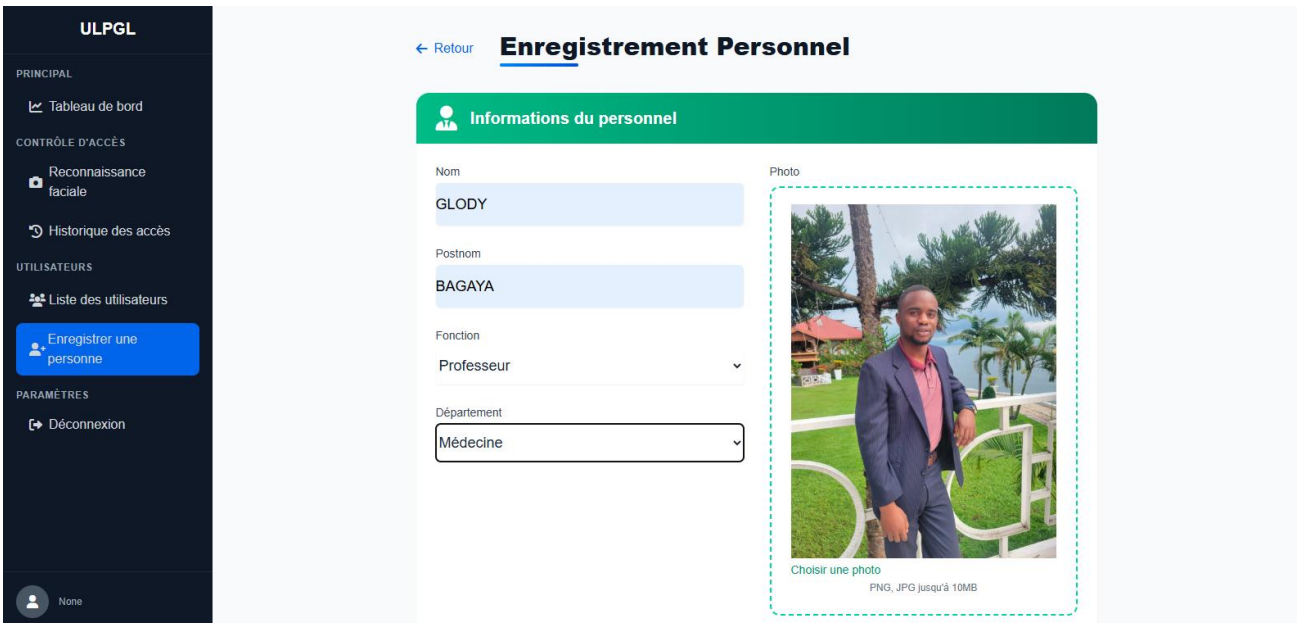


Figure 27: Formulaire personnel avec l'image déjà choisie

Une fois le formulaire validé, un message « Enregistré avec succès » est affiché, comme illustré à la Figure 28.

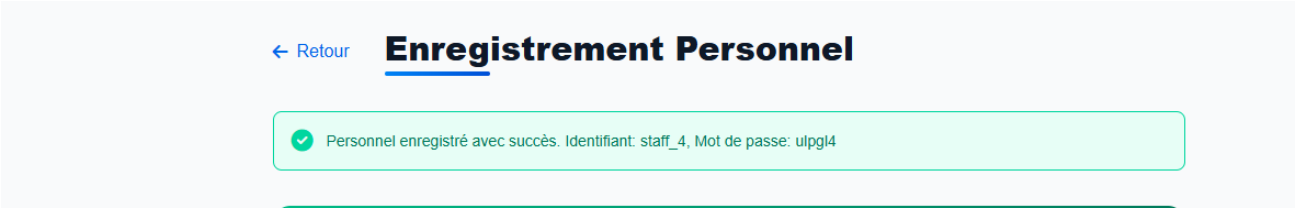


Figure 28: Message de validation du formulaire

Une fois les utilisateurs enregistrés, leurs profils peuvent être consultés, comme montré à la Figure 29.

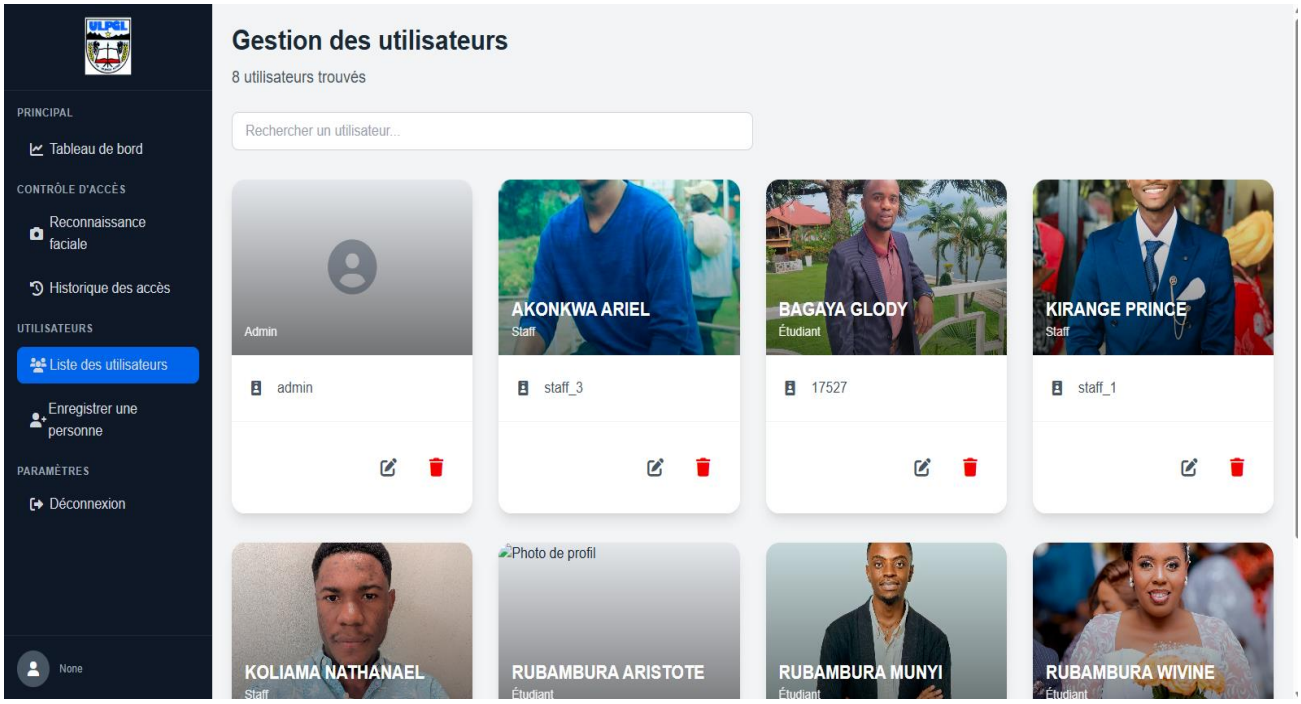


Figure 29: Page Profils utilisateurs

3.2.4 La page Pour la reconnaissance faciale

La page illustrée à la Figure 30 présente le déroulement du processus de reconnaissance faciale avec notre système.

Dans un premier temps, l'utilisateur accède à une page où il doit cliquer sur le bouton « Démarrer la caméra » comme vous le voyez à la Figure 30 :

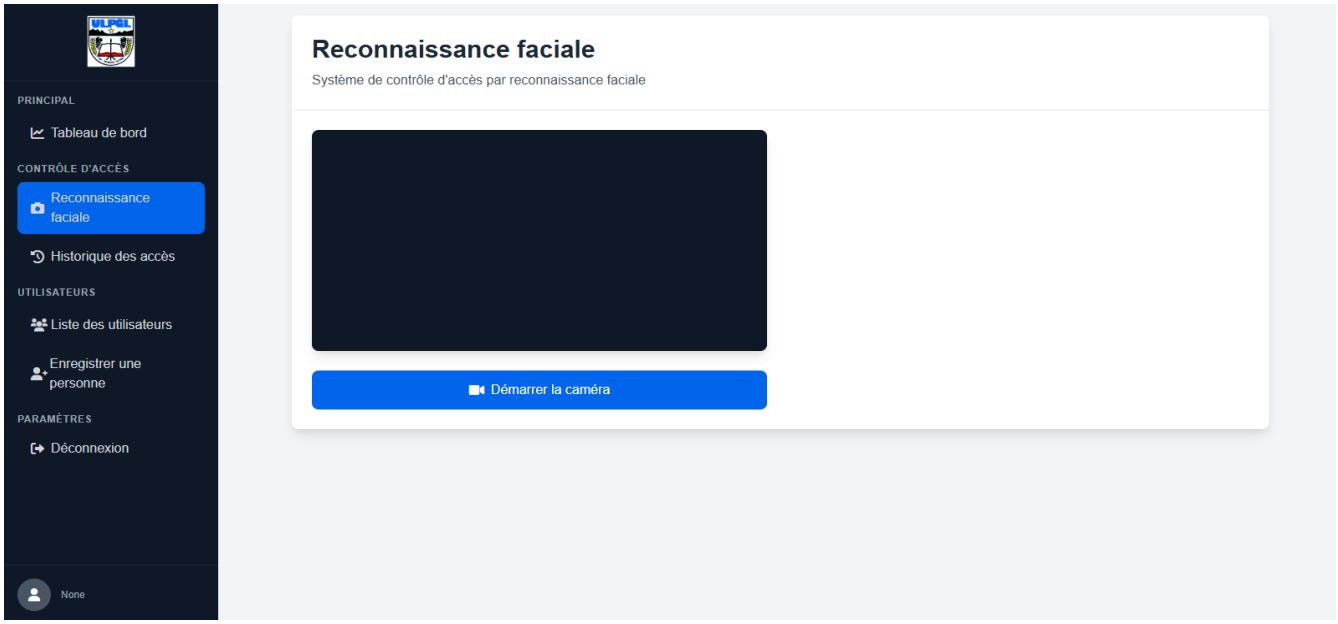


Figure 30: Page pour démarrer la camera

Une fois la caméra ouverte, elle commence à recevoir un flux vidéo pour initier la reconnaissance faciale. À ce stade, la page affiche le message « Personne reconnue » lorsque la reconnaissance est réussie. Les informations de la personne apparaissent à droite, accompagnées du message « Accès autorisé ». Cela est illustré à la Figure 31.

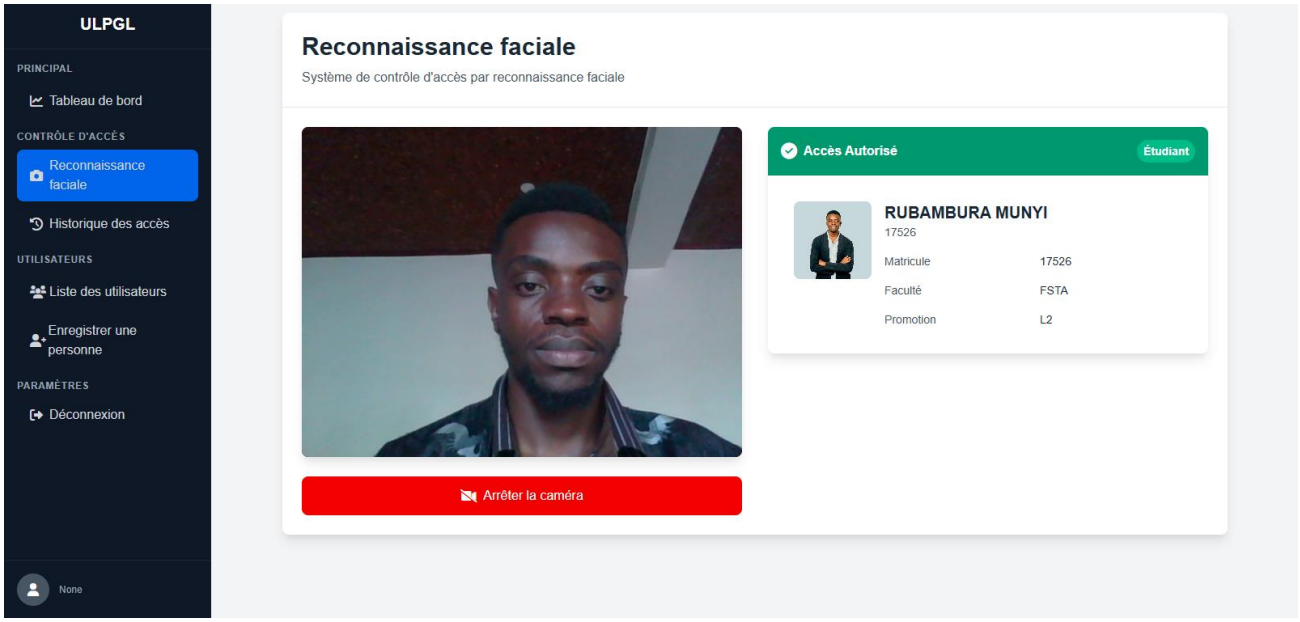


Figure 31: Page pour la reconnaissance faciale avec le résultat « Personne reconnue »

Après une reconnaissance réussie, un message sera affiché pour signaler qu'il faut attendre une minute avant de procéder à une nouvelle reconnaissance pour la même figure ou le même visage détecté. Un message « Veuillez attendre 60 secondes avant votre prochaine présence » sera également affiché, comme illustré à la Figure 32.

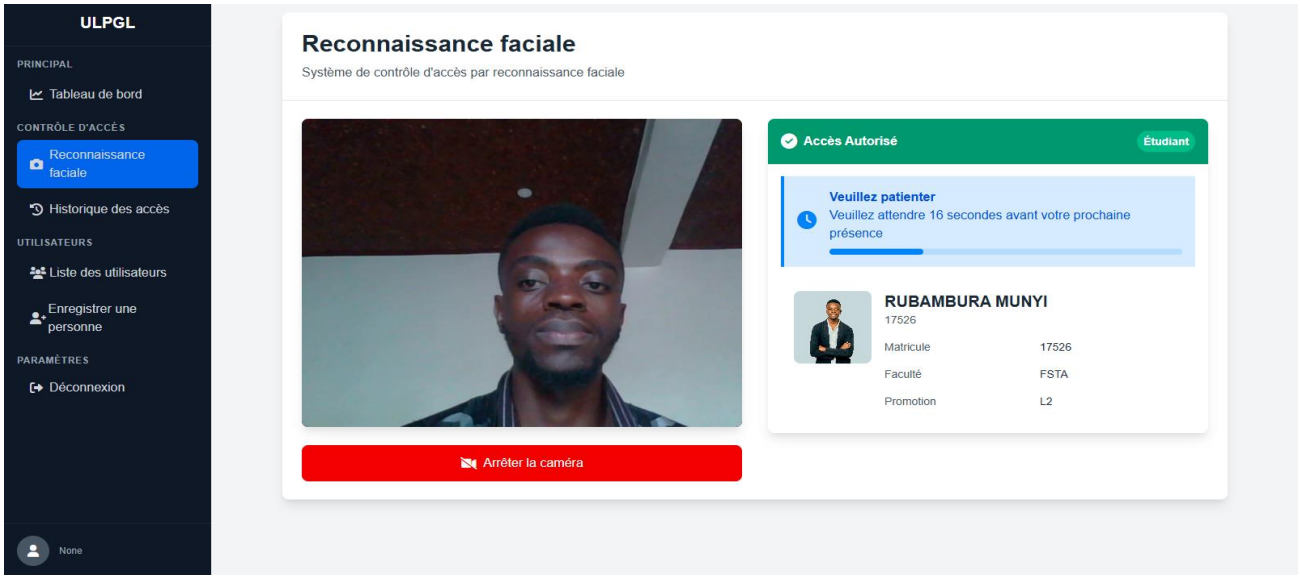


Figure 32: Message :« Veuillez attendre 60 secondes avant votre prochaine présence »

Si le système ne reconnaît pas le visage et identifie la personne comme « inconnu », un message sera affiché, comme illustré à la Figure 33.

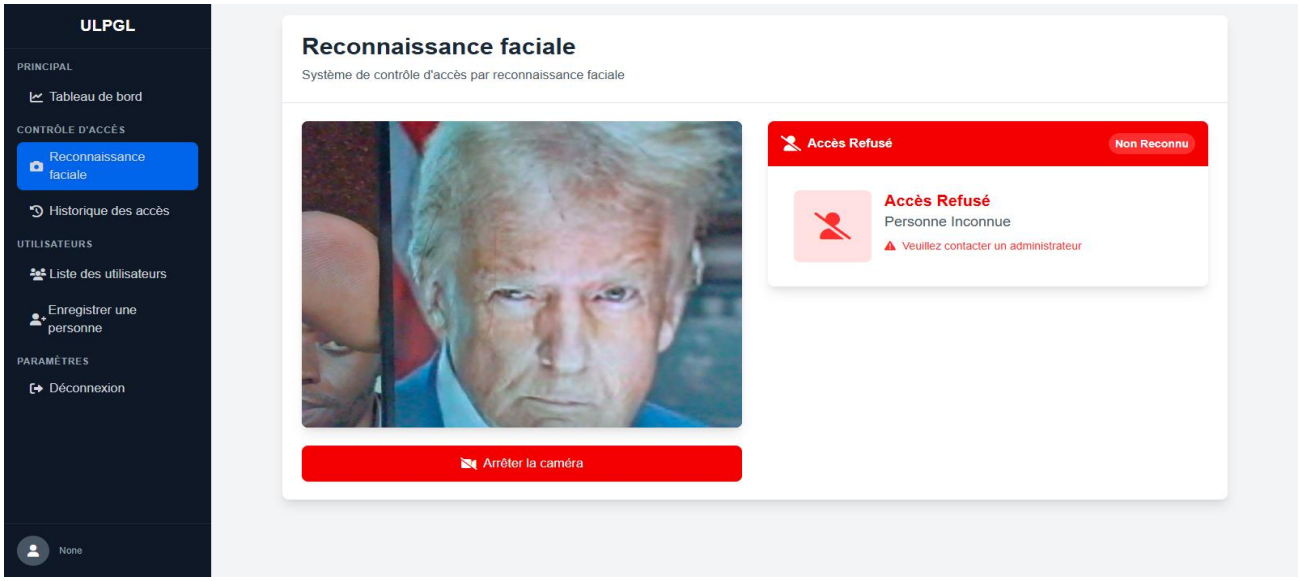


Figure 33 : Message "Acces recusé" pour une personne inconnue

Comme illustré à la Figure 34, lorsque le système détecte plusieurs visages, il affiche le nombre de visages détectés et envoie un message : « Plusieurs visages détectés, veuillez isoler une seule personne devant la caméra ».

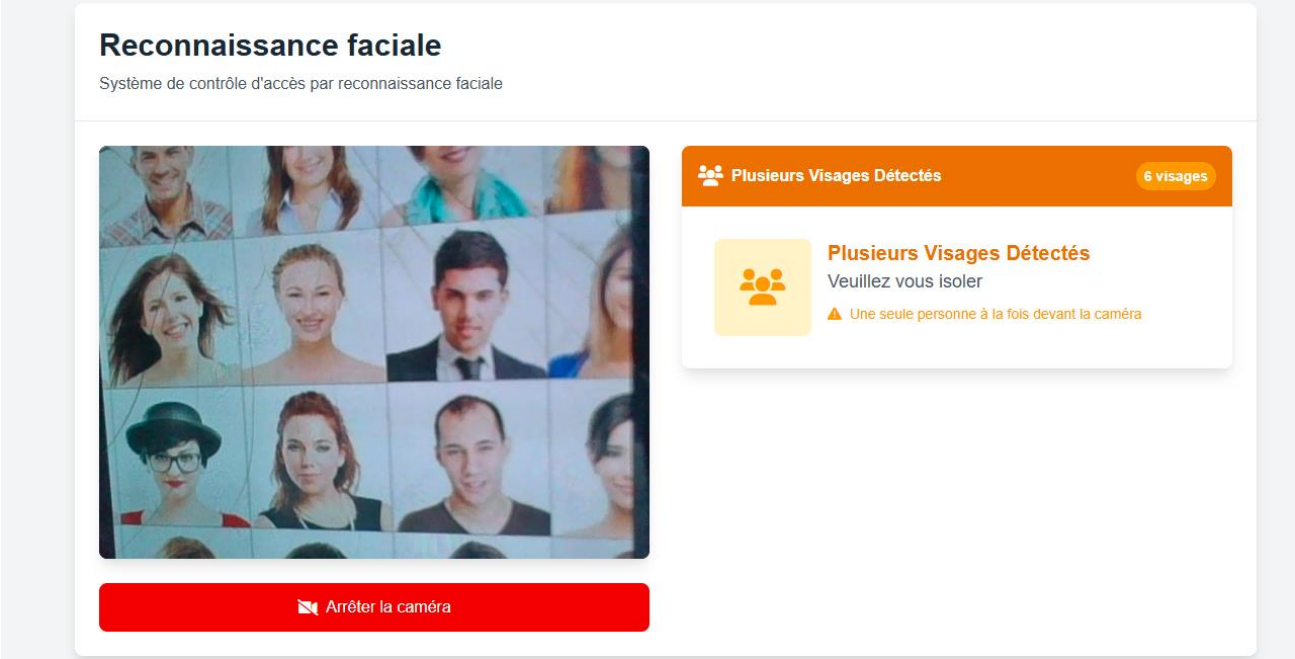


Figure 34: Message « Plusieurs visages détectés , veillez isoler une seule personne devant la caméra»

3.2.5 La page Pour l'historique des Accès

- Cette page, montrée à la Figure 35, présente un tableau récapitulant toutes les entrées, avec les détails correspondants, notamment :**Nom**
- **Identifiant**
- **Type (Étudiant / Personnel)**
- **Date & Heure**
- **Statut (Accès Autorisé / Refusé)**

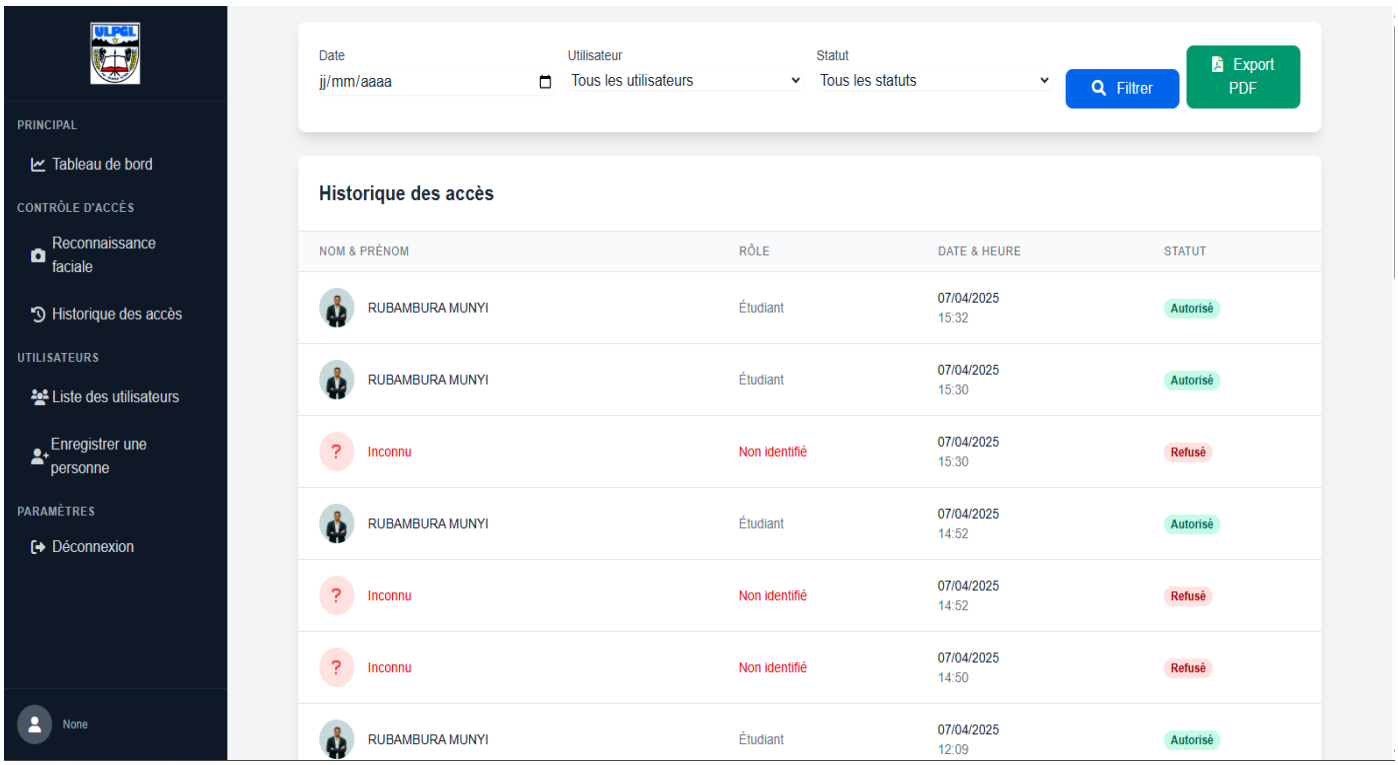


Figure 35: Historique des Accès

. Dans l'historique des accès, il est possible de filtrer les présences selon plusieurs critères : la date, l'utilisateur, et le statut, comme illustré à la Figure 36.

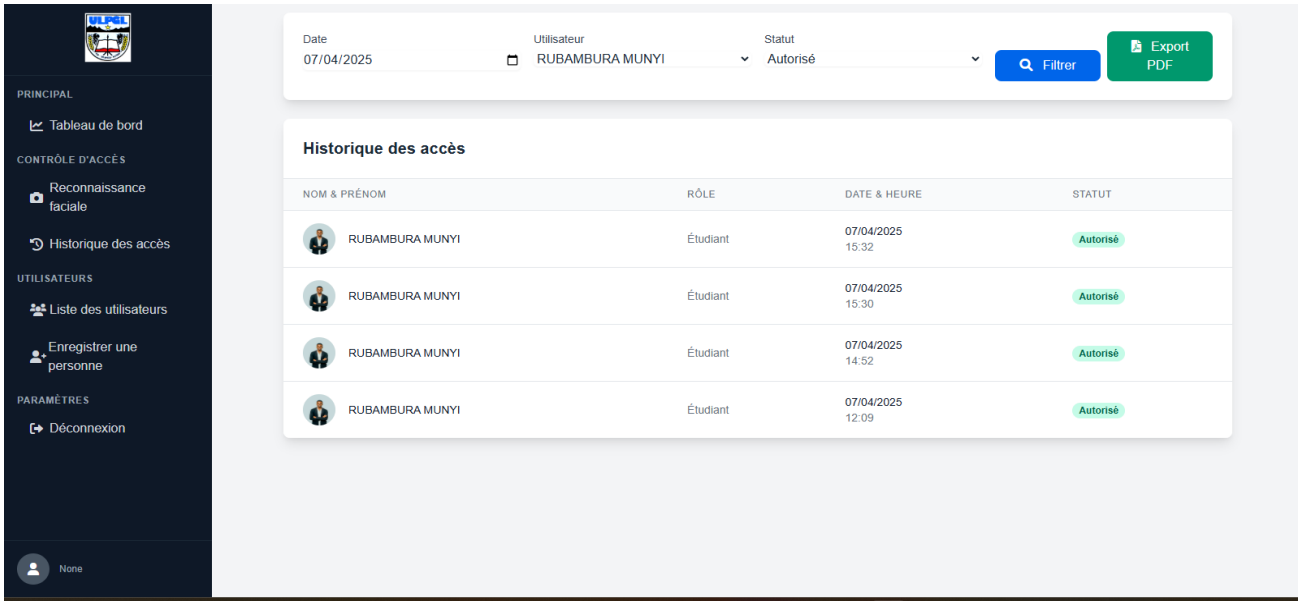


Figure 36: les presence filtrées selon : « La date , utilisateurs, le statuts »

Montrons aussi le cas pour un inconnu a la Figure 37 :

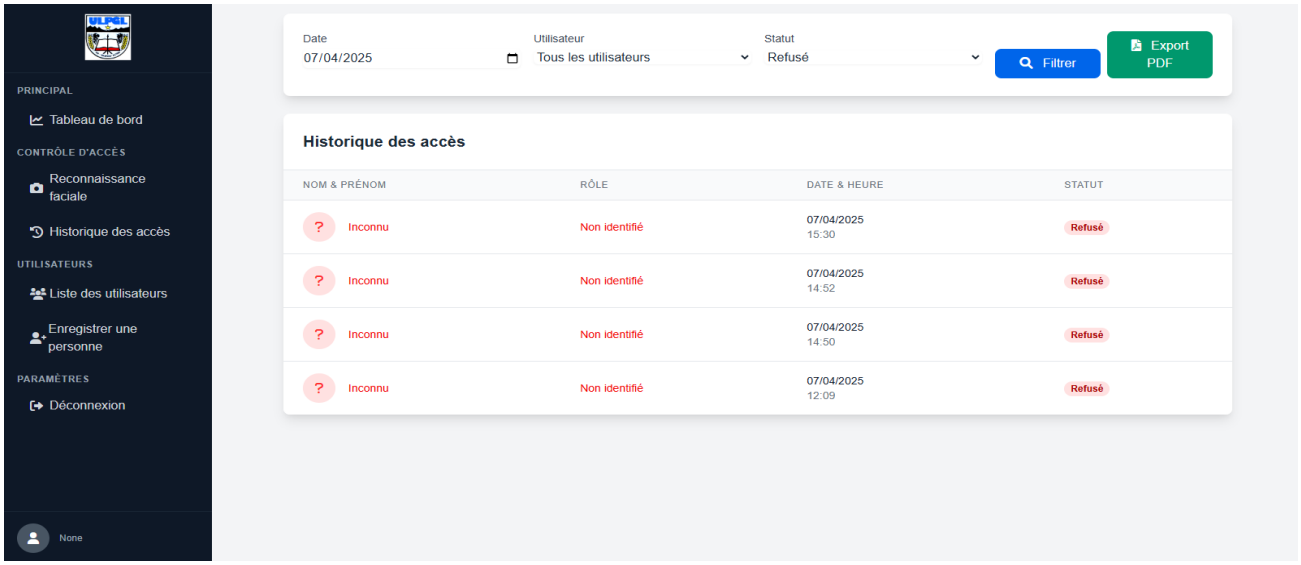
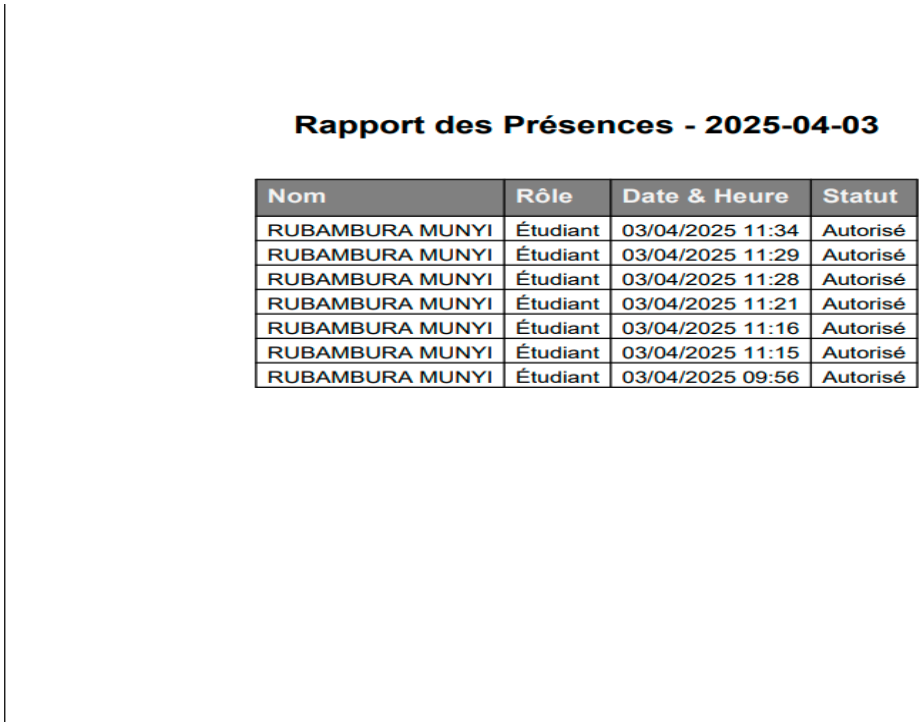


Figure 37: le cas pour filtrer un inconnu

Une fois toutes ces tâches accomplies, le système peut générer une fiche de présence selon les critères sélectionnés, à savoir : « par utilisateur, par date, ou par statut » comme montré à la Figure 38 :



Rapport des Présences - 2025-04-03

Nom	Rôle	Date & Heure	Statut
RUBAMBURA MUNYI	Étudiant	03/04/2025 11:34	Autorisé
RUBAMBURA MUNYI	Étudiant	03/04/2025 11:29	Autorisé
RUBAMBURA MUNYI	Étudiant	03/04/2025 11:28	Autorisé
RUBAMBURA MUNYI	Étudiant	03/04/2025 11:21	Autorisé
RUBAMBURA MUNYI	Étudiant	03/04/2025 11:16	Autorisé
RUBAMBURA MUNYI	Étudiant	03/04/2025 11:15	Autorisé
RUBAMBURA MUNYI	Étudiant	03/04/2025 09:56	Autorisé

Figure 38: Fiche de présences selon l'attente , soit « par utilisateur, date, statut »

3.3 Interprétation des résultats

Le système de contrôle d'accès facial que nous avons conçu présente une efficacité très remarquable, à la fois sur le plan technique et fonctionnel. Le Tableau IV donne en détail les résultats réalisés lors des tests.

Tableau IV:Tableau de Résultats

Catégorie	Indicateur	Résultat	Remarques
Reconnaissance Faciale	Taux de reconnaissance	~95%	95 visages sur 100 sont correctement reconnus dans des conditions normales. Excellent taux de fiabilité.
Reconnaissance Faciale	Temps de traitement	1–2 secondes	Temps moyen pour capturer, détecter et identifier un visage. Adapté à une utilisation en temps réel.
Reconnaissance Faciale	Distance optimale de détection	30–100 cm	Distance idéale entre le visage et la caméra. Trop proche ou trop loin peut nuire à la précision.
Reconnaissance Faciale	Angle de détection efficace	±30°	Le visage peut être incliné jusqu'à 30° sans affecter la reconnaissance.

Contrôle d'accès	Temps de réponse total	2–3 secondes	Temps total entre détection faciale et ouverture de la porte.
Contrôle d'accès	Fiabilité du servo-moteur	98%	Ouverture/fermeture fiable avec très peu d'erreurs mécaniques.
Contrôle d'accès	Temps d'ouverture/fermeture	~1 seconde	Temps nécessaire pour l'action mécanique du servo (verrouillage/déverrouillage).
Conditions Optimales	Éclairage requis	300–1000 lux	300 lux ≈ lumière de bureau standard. 1000 lux ≈ lumière naturelle. <300 : détection difficile. >1000 lux : surcharge lumineuse.
Conditions Optimales	Consommation électrique	1.5A moyenne / 2A pic (5V DC)	1.5A : consommation normale. 2A pic : pendant traitement ou activation du servo. Tension standard pour ESP32 et périphériques.

3.4 Conclusion Partielle

Dans ce chapitre, nous avons présenté les différentes interfaces du système ainsi que l'interprétation des résultats, en mettant l'accent sur les interactions entre les pages et en clarifiant les résultats obtenus. Parmi ces interfaces, on trouve notamment :

- Connexion (Login)
- Le Dashboard
- Utilisateurs
- Création d'un utilisateur (Etudiant ou personnel)
- Choisir un utilisateur à enregistrer « soit personnel ou étudiant »
- Profile des utilisateurs
- Validation des accès des utilisateurs

L'ensemble des résultats interprétés montre que le système est fonctionnel, fiable et répond efficacement à nos attentes, sa base de données structurée, et l'efficacité des outils intégrés comme face_recognition permettent une utilisation durable dans un cadre institutionnel ou académique.

Conclusion générale

Ce mémoire a présenté la conception et la mise en œuvre d'un système innovant de contrôle d'accès et de prise de présence par reconnaissance faciale pour l'Université Libre des Pays des Grands Lacs (ULPGL).

Ce projet visait à répondre aux limitations des méthodes traditionnelles telles que les registres papier et les cartes d'étudiants, en offrant une solution automatisée, sécurisée et efficace.

Les résultats obtenus ont confirmé la robustesse du système, ainsi que son intégration matérielle optimale avec l'ESP32, l'ESP32-CAM et le servo-moteur SG90, garantissant une solution fiable pour le contrôle d'accès.

Les principales contributions de ce travail incluent un renforcement de la sécurité grâce à un accès restreint aux seules personnes autorisées via une identification biométrique fiable. Cela permet de réduire les risques de fraude, notamment l'usurpation d'identité et les faux enregistrements. En termes d'efficacité administrative, le système automatise la prise de présence avec un enregistrement en temps réel dans une base de données, éliminant ainsi les erreurs humaines liées aux méthodes manuelles. Sur le plan technologique, le système fait appel à des outils avancés comme Python, les bibliothèques OpenCV et face_recognition pour le traitement des images, tout en intégrant un matériel performant avec l'ESP32-CAM pour la capture d'images et le servo-moteur SG90 pour le contrôle physique de l'accès. Une interface web intuitive a également été développée avec Django et Tailwind CSS.

Cependant, bien que le système présente des avantages indéniables, certaines limites doivent être prises en compte. La dépendance aux conditions d'éclairage est l'une de ces limites, car la reconnaissance faciale peut être moins précise dans des environnements mal éclairés. De plus, le coût initial de l'installation, en raison des investissements matériels et logiciels nécessaires, peut représenter un obstacle pour certaines institutions. Enfin, l'utilisation de données biométriques soulève des questions éthiques et juridiques qu'il convient de traiter avec soin afin d'assurer la conformité avec la législation en vigueur.

Pour l'avenir, plusieurs pistes d'amélioration sont envisagées. D'une part, la reconnaissance faciale pourrait être optimisée grâce à l'intégration de modèles de deep learning, comme ResNet ou FaceNet, pour une meilleure précision et une réduction des faux positifs. Des algorithmes de prétraitement d'images pourraient également être développés pour adapter la reconnaissance faciale aux variations d'éclairage et d'angle. D'autre part, l'ajout d'une seconde caméra permettrait d'enregistrer les départs et de calculer la durée de présence, avec un algorithme de suivi en temps réel pour éviter les doublons, comme dans le cas d'un même individu entrant et sortant dans des délais très courts. Sur le plan matériel, des capteurs infrarouges pourraient être intégrés pour permettre une détection efficace en basse lumière, tandis qu'un système d'alarme pourrait être mis en place en cas de tentative d'intrusion. Enfin, des améliorations logicielles incluraient le développement d'un tableau de bord analytique permettant de visualiser les statistiques de présence sous forme de graphiques et d'exporter les données en format Excel, ainsi que l'implémentation d'alertes automatisées pour signaler les absences répétées ou les tentatives d'accès non autorisées.

Bibliographie

[1] M. Willich, ««2nd End-User Group Meeting on 3D Face Recognition,»,» Berlin, 2008.

[2] M. Benantar, « Access Control Systems: Security, Identity Management and Trust Models,» Springer Science & Business Media, New York, 2006.

[3] M. ,. KAYIHEMBAKO, «Étude et conception d'un dispositif automatique d'ouverture d'une porte par reconnaissance faciale,» 2021.

[4] https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcRKftct9eL10tTWAbtYSbEvpE4iR_SmHj4O1A&s.

[5] https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcQAHzPTlGft9o1HSnmFCi0Hrt1oSS8aNFWOIw&s.

[6] K. &. S. M. Lee, Embedded System Design for Real-Time Face Recognition., IEEE Embedded Systems Letters., (2020).

[7] A. K. F. P. &. R. Jain, . Handbook of Biometrics, New York: . Handbook of Biometrics, 2007.

[8] K. Bouchra, «Mise au point d'une application de reconnaissance faciale,» Université Abou Bakr Belkaid, Tlmencen, 2013.

[9] A. K. J. S. Z. Li, , Handbook of face recognition, michigan: springer, 2004.

[10] Y. M. Y. e. U. S. Adini, «Face recognition:The problem of compensating for changes in illumination direction,» chez Transactions on pattern analysis and machine intelligence, Piscataway, New Jersey, États-Unis, 1997.

[11]]. L. G. N. R. B. e. F. N. Chaby, ««Age-related changes in brain responses to personally known faces : an eventrelated potential (erp) study in humans,»,» *Neuroscience letters,* p. p. 125_129, 2003.

[12] L. J. B. G. N. R. B. e. F. Chaby, « «An erp study of famous face incongruity detection in middle age,»,» *Brain and Cognition,* p. p. 357_377, 2001.

[13] T. H. e. L. G. J. Crook, ««Changes in facial recognition memory across the adult life span,»,» *Journal of Gerontology,* pp. pp. 138-141, 1992.

[14] J. Sullivan, Arduino Robotics, Springer., 2011.

[15] G. Arduino Robotics (Springer) ou McComb, (2020).

[16] Jonathon Phillips and Hyeonjoon Moon and Syed A. Rizvi and Patrick J. Rauss, «"The FERET Evaluation Methodology for Face-Recognition Algorithms",» *IEEE Transactions on Pattern Analysis and Machine Intelligence,,* vol. volume 22, n° %1number 10,, pp. pages 1090-1104,, 2000.

[17] A. P. M. Turk, Eigenfaces for recognition., J. of Cognitive Neuroscience, 1991.

[18] athon Phillips and Hyeonjoon Moon and Syed A. Rizvi and Patrick J. Rauss, «"The FERET Evaluation Methodology for Face-Recognition Algorithms",,» *IEEE Transactions on Pattern Analysis and Machine Intelligence,,* vol. 22, n° %1number 10, pp. pages 1090-1104,, 2000.

[19] P. J. R. S. Der Phillips J., Feret recognition algorithm development and test, Army Research Laboratory technical report,ARL-TR-995, 1996..

[20] T. H. A. &. P. Ahonen, Face recognition with local binary patterns: Application to face recognition, New York: IEEE Transactions on Pattern Analysis and Machine Intelligence, 28(12), 2037-2041, 2004.

[21] M. McRoberts, Programming the ESP32: Getting Started with the ESP32 Development Board, New York, 2009.

[22] W. S. Vincent, "Django for Professionals", Philadelphia: Self-published, 2022.

[23] M. Fowler, UML Distilled: A Brief Guide to the Standard Object Modeling Language, Boston: Addison-Wesley, 2004.

[24] «Django Project Documentation,» [En ligne]. Available: https://docs.djangoproject.com.

[25] L. C. e. a. Z. Liu, Face Recognition Systems: Algorithms, Applications, and Challenges, Springer: springer, 2015.

[26] https://i.sstatic.net/Ye692.png.

[27] D. K. J. P. Florian Schroff, «FaceNet: A Unified Embedding for Face Recognition and Clustering,» Google Inc. (via arXiv.org), Mountain View, Californie, 2015.

[28] M. Jiménez, Arduino Sensors: A Beginner's Guide, spinger: springer, 2019.

[29] https://makeradvisor.com/wp-content/uploads/2017/10/esp32-board-bg.jpg.

[30] https://www.hwlibre.com/wp-content/uploads/2022/03/servo-sg90-1024x614.jpg.webp.

[31] JUnit.org. [En ligne]. Available: http://www.junit.org. [Accès le 2 Janvier 2019].

[32] [En ligne]. Available: https://www.bing.com/images/create/proposez-une-image-d27un-jeune-c3a9tudiant-congolais2c-/1-66964ff0c60c442d9f5042d2e1a1f9cc?id=af3kmh3t3IGubEmiMxL8QQ%3d%3d&view=detailv2&idpp=genimg&thId=OIG2.rYuvVum1aujjdpbDWZkJ&FORM=GCRIDP&ajaxhist=0&ajaxserp=0 .

[33] K. M. C. E. Ogata, Modern Control Engineering, Upper Saddle River, NJ.: Prentice Hall, 2010.

[34] R. &. K. ,. R. Karthik, «Smart Access Control Systems.,» IEEE Internet of Things Journal,, New York, NY, 2020.

[35] A. K. F. P. &. R. Jain, Handbook of Biometrics., New York: Springer, 2007.

[36] S. R. e. V. Mirjalili, Python Machine Learning, NY: Packt Publishing, 2019.

[37] A. Rosebrock, Practical Python and OpenCV + Case Studies, San Francisco: PyImageSearch, 2016.

[38] https://rlx.sk/14596-large_default/sg90-micro-servo-motor-ws-15286.jpg.

[39]

[40] https://fr.hwlibre.com/cam%C3%A9ra-esp32/.

[41] https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcSanszgrVo8dGaeHJbKx9A6t0flC41Uh8B_ig&s.

[42] https://makeradvisor.com/wp-content/uploads/2017/10/esp32-board-bg.jpg.

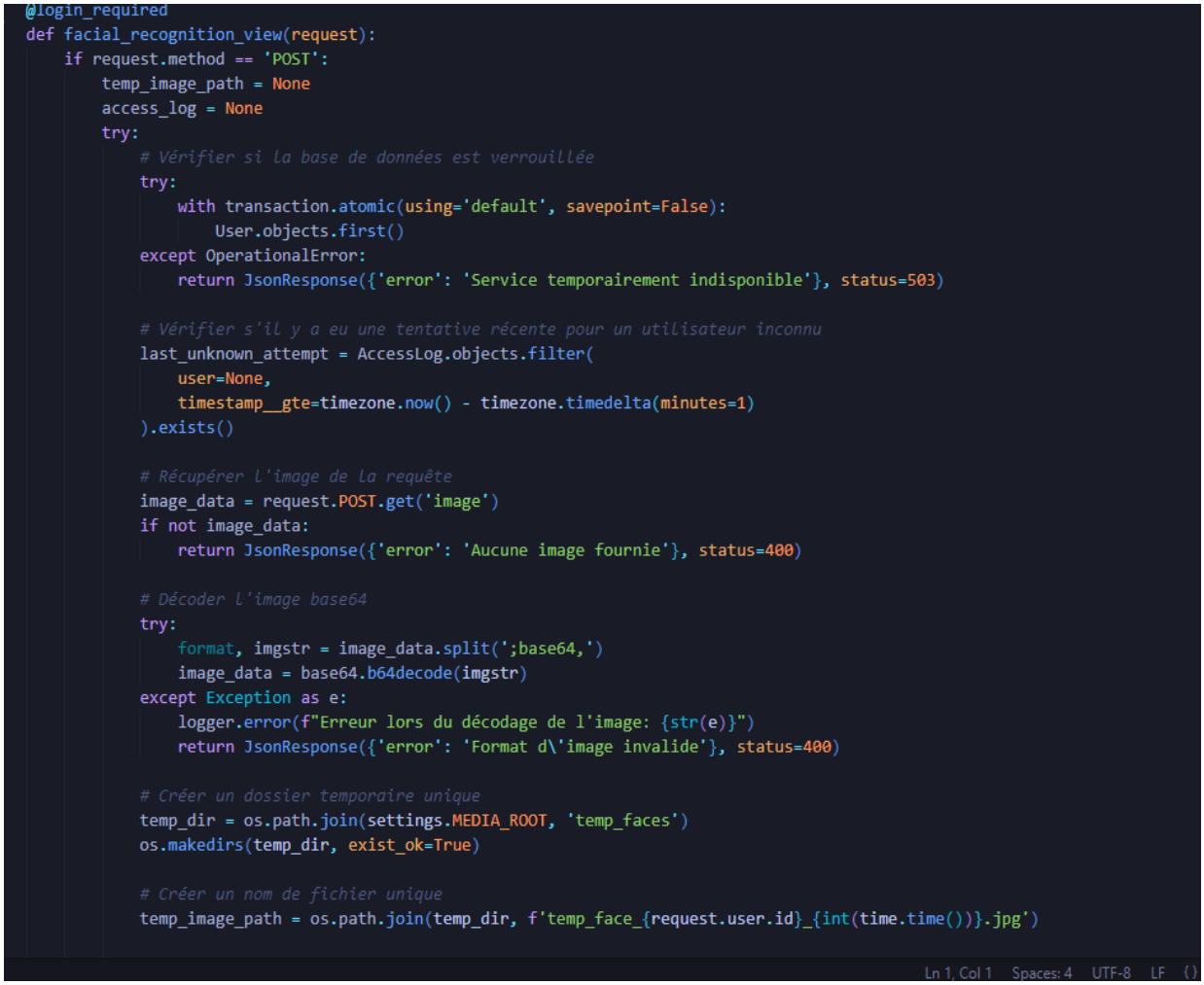
[43] https://makeradvisor.com/wp-content/uploads/2017/10/esp32-board-bg.jpg.

[44] https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.l-acces.com%2Fserrure-poignee-

[45] https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.heracles.fr%2Fheracles-serrure-carenee-sesame-2-x8_SKUM32635.html&psig

Annexe

Extrait du code de la fonction pour la reconnaissance faciale



```python
@login_required
def facial_recognition_view(request):
    if request.method == 'POST':
        temp_image_path = None
        access_log = None
        try:
            # Vérifier si la base de données est verrouillée
            try:
                with transaction.atomic(using='default', savepoint=False):
                    User.objects.first()
            except OperationalError:
                return JsonResponse({'error': 'Service temporairement indisponible'}, status=503)

            # Vérifier s'il y a eu une tentative récente pour un utilisateur inconnu
            last_unknown_attempt = AccessLog.objects.filter(
                user=None,
                timestamp__gte=timezone.now() - timezone.timedelta(minutes=1)
            ).exists()

            # Récupérer l'image de la requête
            image_data = request.POST.get('image')
            if not image_data:
                return JsonResponse({'error': 'Aucune image fournie'}, status=400)

            # Décoder l'image base64
            try:
                format, imgstr = image_data.split(';base64,')
                image_data = base64.b64decode(imgstr)
            except Exception as e:
                logger.error(f"Erreur lors du décodage de l'image: {str(e)}")
                return JsonResponse({'error': 'Format d\'image invalide'}, status=400)

            # Créer un dossier temporaire unique
            temp_dir = os.path.join(settings.MEDIA_ROOT, 'temp_faces')
            os.makedirs(temp_dir, exist_ok=True)

            # Créer un nom de fichier unique
            temp_image_path = os.path.join(temp_dir, f'temp_face_{request.user.id}_{int(time.time())}.jpg')
```

Figure 39: Figure de la fonction pour la raiconnaissance facial

Annexe

Extrait du code pour l'encodage des images

```python
# Encoder le visage détecté
unknown_encoding = face_recognition.face_encodings(unknown_image, [face_locations[0]])[0]

# Comparer avec tous les utilisateurs ayant un encodage facial
best_match = None
best_score = 0
threshold = 0.6  # Seuil de confiance

# Récupérer tous les utilisateurs avec encodage facial en une seule requête
with transaction.atomic():
    users = list(User.objects.exclude(face_encoding=None))

for user in users:
    try:
        known_encoding = user.get_face_encoding_array()
        if known_encoding is not None:
            # Calculer la similarité
            face_distance = face_recognition.face_distance([known_encoding], unknown_encoding)[0]
            similarity_score = 1 - face_distance  # Convertir la distance en score de similarité

            if similarity_score > threshold and similarity_score > best_score:
                best_match = user
                best_score = similarity_score
    except Exception as e:
        logger.error(f"Erreur lors de la comparaison avec l'utilisateur {user.username}: {str(e)}")
        continue

# Si un utilisateur est reconnu
if best_match:
    # Vérifier si l'utilisateur a déjà une présence dans la dernière minute
    last_presence = AccessLog.objects.filter(
        user=best_match,
        timestamp__gte=timezone.now() - timezone.timedelta(minutes=1)
    ).first()
```

Figure 40: Extrait du code pour l'encodage des images

Extrait du code une fois le visage reconnu

```python
# Si un utilisateur est reconnu
if best_match:
    # Vérifier si l'utilisateur a déjà une présence dans la dernière minute
    last_presence = AccessLog.objects.filter(
        user=best_match,
        timestamp__gte=timezone.now() - timezone.timedelta(minutes=1)
    ).first()

    if last_presence:
        # Calculer le temps restant
        time_since_last = timezone.now() - last_presence.timestamp
        seconds_remaining = 60 - time_since_last.seconds

        return JsonResponse({
            'success': False,
            'error_type': 'cooldown',
            'message': f'Veuillez attendre {seconds_remaining} secondes avant votre prochaine présence',
            'seconds_remaining': seconds_remaining
        })

    # Récupérer l (class) User  de profil
    best_match = User.objects.select_related('student_profile', 'staff_profile').get(id=best_match.id)

    # Créer le log d'accès
    with transaction.atomic():
        access_log = AccessLog.objects.create(
            user=best_match,
            access_granted=True,
            recognition_score=best_score,
            photo=image_data
        )
```

Figure 41: Extrait du code une fois le visage reconnue

Extrait du code pour l'affichage des informations

```python
# Préparer les informations de profil selon le rôle
profile_info = None
if best_match.role == 'Étudiant' and hasattr(best_match, 'student_profile'):
    profile_info = {
        'student_profile': {
            'matricule': best_match.student_profile.matricule,
            'faculte': best_match.student_profile.faculte,
            'promotion': best_match.student_profile.promotion
        }
    }
elif best_match.role == 'Staff' and hasattr(best_match, 'staff_profile'):
    profile_info = {
        'staff_profile': {
            'departement': best_match.staff_profile.departement
        }
    }

# Construire la réponse
response_data = {
    'success': True,
    'user': {
        'username': best_match.username,
        'full_name': f"{best_match.first_name} {best_match.last_name}",
        'role': best_match.role,
        'photo_url': best_match.profile_image.url if best_match.profile_image else None,
        **(profile_info or {})
    },
    'score': best_score,
    'log_id': access_log.id
}
return JsonResponse(response_data)
```

Figure 42: Extrait du code pour l'affichage des informations après reconnaissance

Extrait du code pour un visage non reconnu

```
            else:
                # Gérer le cas où aucun utilisateur n'est reconnu
                if not last_unknown_attempt:
                    with transaction.atomic():
                        access_log = AccessLog.objects.create(
                            user=None,
                            access_granted=False,
                            recognition_score=0,
                            photo=image_data
                        )
                    Ctrl+I for Command, Ctrl+L for Cascade
                response_data = {
                    'success': False,
                    'message': 'Personne inconnue. Accès refusé.',
                    'error_type': 'unknown_person'
                }
                return JsonResponse(response_data)

        except Exception as e:
            logger.error(f"Erreur lors du traitement de l'image: {str(e)}")
            return JsonResponse({'error': str(e)}, status=500)

    except Exception as e:
        logger.error(f"Erreur lors de la reconnaissance faciale: {str(e)}")
        return JsonResponse({'error': str(e)}, status=500)
```

Figure 43: Extrait du code pour un visage non reconnue

Extrait du code pour exporter la présence

```python
@login_required
def export_presence_pdf(request):
    # Récupérer les mêmes filtres que la vue history
    date_filter = request.GET.get('date', '')
    user_filter = request.GET.get('user', '')
    status_filter = request.GET.get('status', '')

    # Base de la requête
    access_logs = AccessLog.objects.select_related('user').all().order_by('-timestamp')

    # Appliquer les mêmes filtres que la vue history
    if date_filter:
        try:
            filter_date = datetime.strptime(date_filter, '%Y-%m-%d').date()
            access_logs = access_logs.filter(timestamp__date=filter_date)
        except ValueError:
            pass

    if user_filter:
        access_logs = access_logs.filter(user_id=user_filter)

    if status_filter:
        if status_filter == 'granted':
            access_logs = access_logs.filter(access_granted=True)
        elif status_filter == 'denied':
            access_logs = access_logs.filter(access_granted=False)

    # Créer le PDF
    buffer = BytesIO()
    doc = SimpleDocTemplate(buffer, pagesize=letter)
    elements = []

    # Styles
    styles = getSampleStyleSheet()
    title_style = ParagraphStyle(
        'CustomTitle',
        parent=styles['Heading1'],
        alignment=TA_CENTER,
        spaceAfter=30
```

Figure 44: Extrait du code pour exporter la présence

Extrait du code Arduino pour contrôler l'esp32 et le servo moteur

```ino
#include <ESP32Servo.h>

// Configuration du servo
Servo doorServo;
const int servoPin = 13;  // GPIO13 pour le servo
const int CLOSED_ANGLE = 0;
const int OPEN_ANGLE = 90;

// Variables d'état
bool isDoorOpen = false;

// Buffer pour les commandes série
String inputBuffer = "";

// Fonction pour traiter les commandes
void processCommand(String command) {
  command.trim();  // Enlever les espaces
  command.toUpperCase();  // Convertir en majuscules

  if (command == "OPEN") {
    doorServo.write(OPEN_ANGLE);
    isDoorOpen = true;
    Serial.println("OK: Porte ouverte");
  }
  else if (command == "CLOSE") {
    doorServo.write(CLOSED_ANGLE);
    isDoorOpen = false;
    Serial.println("OK: Porte fermée");
  }
  else if (command == "STATUS") {
    String status = isDoorOpen ? "ouverte" : "fermée";
    Serial.println("STATUS: Porte " + status);
  }
  else {
    Serial.println("ERREUR: Commande inconnue");
  }
}
```

Figure 45: Extrait du code arduino pour controler l'ESP32 et le SERVO MOTEUR

Extrait du code du fichier de connexion entre l'application et le coté matériels

```python
import json
import serial
import requests
from typing import Dict, Any

class DeviceCommunication:
    def __init__(self):
        self.wifi_devices: Dict[str, Dict[str, Any]] = {}
        self.serial_connections: Dict[str, serial.Serial] = {}

    def connect_wifi_device(self, device_id: str, ip: str, port: int) -> bool:

        try:
            # Test de connexion avec un ping HTTP
            url = f"http://{ip}:{port}/ping"
            response = requests.get(url, timeout=5)
            if response.status_code == 200:
                self.wifi_devices[device_id] = {
                    "ip": ip,
                    "port": port
                }
                return True
            return False
        except Exception as e:
            print(f"Erreur de connexion WiFi: {e}")
            return False

    def send_command(self, device_id: str, command: Dict[str, Any]) -> bool:
        """
        Envoie une commande à un périphérique spécifique
        """
        try:
            if device_id in self.wifi_devices:
                return self._send_wifi_command(device_id, command)
            elif device_id in self.serial_connections:
                return self._send_serial_command(device_id, command)
            return False
        except Exception as e:
            print(f"Erreur d'envoi de commande: {e}")
```

```python
# Si un utilisateur est reconnu
if best_match:
    # Envoi de la requête HTTP à l'ESP32 pour ouvrir la porte
    try:
        esp32_url = "http://10.215.111.151/on"  # Remplacez par l'IP de votre ESP32
        response = requests.get(esp32_url)
        if response.status_code == 200:
            logger.info(f"Accès accordé à {best_match.username}, porte ouverte.")
        else:
            logger.error(f"Erreur lors de la communication avec l'ESP32.")
    except Exception as e:
        logger.error(f"Erreur de communication avec l'ESP32: {str(e)}")

    # Vérifier si l'utilisateur a déjà une présence dans la dernière minute
    last_presence = AccessLog.objects.filter(
        user=best_match,
        timestamp__gte=timezone.now() - timezone.timedelta(minutes=1)
    ).first()

    if last_presence:
        # Calculer le temps restant
        time_since_last = timezone.now() - last_presence.timestamp
        seconds_remaining = 60 - time_since_last.seconds

        return JsonResponse({
            'success': False,
            'error_type': 'cooldown',
            'message': f'Veuillez attendre {seconds_remaining} secondes avant votre prochaine présence',
            'seconds_remaining': seconds_remaining
        })

    # Récupérer les informations de profil
    best_match = User.objects.select_related('student_profile', 'staff_profile').get(id=best_match.id)
```

Figure 46: Extrait du code du fichier de connexion entre l'application et le coté matériels

Extrait du code du fichier de connexion entre l'application et le coté matériels

```python
# Si un utilisateur est reconnu
if best_match:
    # Envoi de la requête HTTP à l'ESP32 pour ouvrir la porte
    try:
        esp32_url = "http://10.255.248.151/on"  # Remplacez par l'IP de votre ESP32
        response = requests.get(esp32_url)
        if response.status_code == 200:
            logger.info(f"Accès accordé à {best_match.username}, porte ouverte.")
        else:
            logger.error(f"Erreur lors de la communication avec l'ESP32.")
    except Exception as e:
        logger.error(f"Erreur de communication avec l'ESP32: {str(e)}")

    # Vérifier si l'utilisateur a déjà une présence dans la dernière minute
    last_presence = AccessLog.objects.filter(
        user=best_match,
        timestamp__gte=timezone.now() - timezone.timedelta(minutes=1)
    ).first()

    if last_presence:
        # Calculer le temps restant
        time_since_last = timezone.now() - last_presence.timestamp
        seconds_remaining = 60 - time_since_last.seconds

        return JsonResponse({
            'success': False,
            'error_type': 'cooldown',
            'message': f'Veuillez attendre {seconds_remaining} secondes avant votre prochaine présence'
            'seconds_remaining': seconds_remaining
        })

    # Récupérer les informations de profil
    best_match = User.objects.select_related('student_profile', 'staff_profile').get(id=best_match.id)
```

Figure 47: Extrait du code du fichier de connexion entre l'application et le coté matériels

Extrait du code Arduino pour actionner le servomoteur

ES32WEBSERV.ino

```
#include <WiFi.h>
#include <WebServer.h>
#include <ESP32Servo.h>

const char *ssid = "Decarlo";
const char *password = "12345678";
WebServer server(80);
Servo myservo;

const int led = 2;
bool etatLed = 0;
char texteEtatLed[2][10] = {"FERMER!", "OUVERT!"};

unsigned long tempsOuverture = 0;
bool attenteFermeture = false;


void handleOn()
{
    etatLed = 1;
    digitalWrite(led, HIGH);
    for (int pos = 0; pos <= 90; pos++) {
        myservo.write(pos);
        delay(5);
    }

    // Démarrer le compte à rebours de 5 secondes
    tempsOuverture = millis();
    attenteFermeture = true;

    server.sendHeader("Location", "/");
    server.send(303);
}

void handleOff()
{
```

Figure 48:Extrait du code Arduino pour actionner le servomoteur

ES32WEBSERV.ino

```
60  }
61
62  void handleOff()
63  {
64      etatLed = 0;
65      digitalWrite(led, LOW);
66      for (int pos = 90; pos >= 0; pos--) {
67          myservo.write(pos);
68          delay(5);
69      }
70      attenteFermeture = false;
71
72      server.sendHeader("Location", "/");
73      server.send(303);
74  }
75
76  void handleNotFound()
77  {
78      server.send(404, "text/plain", "404: Not found");
79  }
80
81  void setup()
82  {
83      Serial.begin(115200);
84      delay(1000);
85      Serial.println("\n");
86
87      myservo.setPeriodHertz(50);    // Fréquence standard pour les servos
88      myservo.attach(18);            // GPIO18 (choisis une pin PWM)
89      myservo.write(0);
90
91      pinMode(led, OUTPUT);
92      digitalWrite(led, LOW);
93
94      WiFi.persistent(false);
95      WiFi.begin(ssid, password);
```

Figure 49:Extrait du code Arduino pour actionner le servomoteur