

Systeme de gestion d'éclairage avec gestes avec ESP32-cam et Edge impulse

Encadré par:

• Pr. Anass Bouayad

Réalisé par:

- Ouayazza Abdelaziz
- Dairi Omar
- Azroul Mohamed Khalil

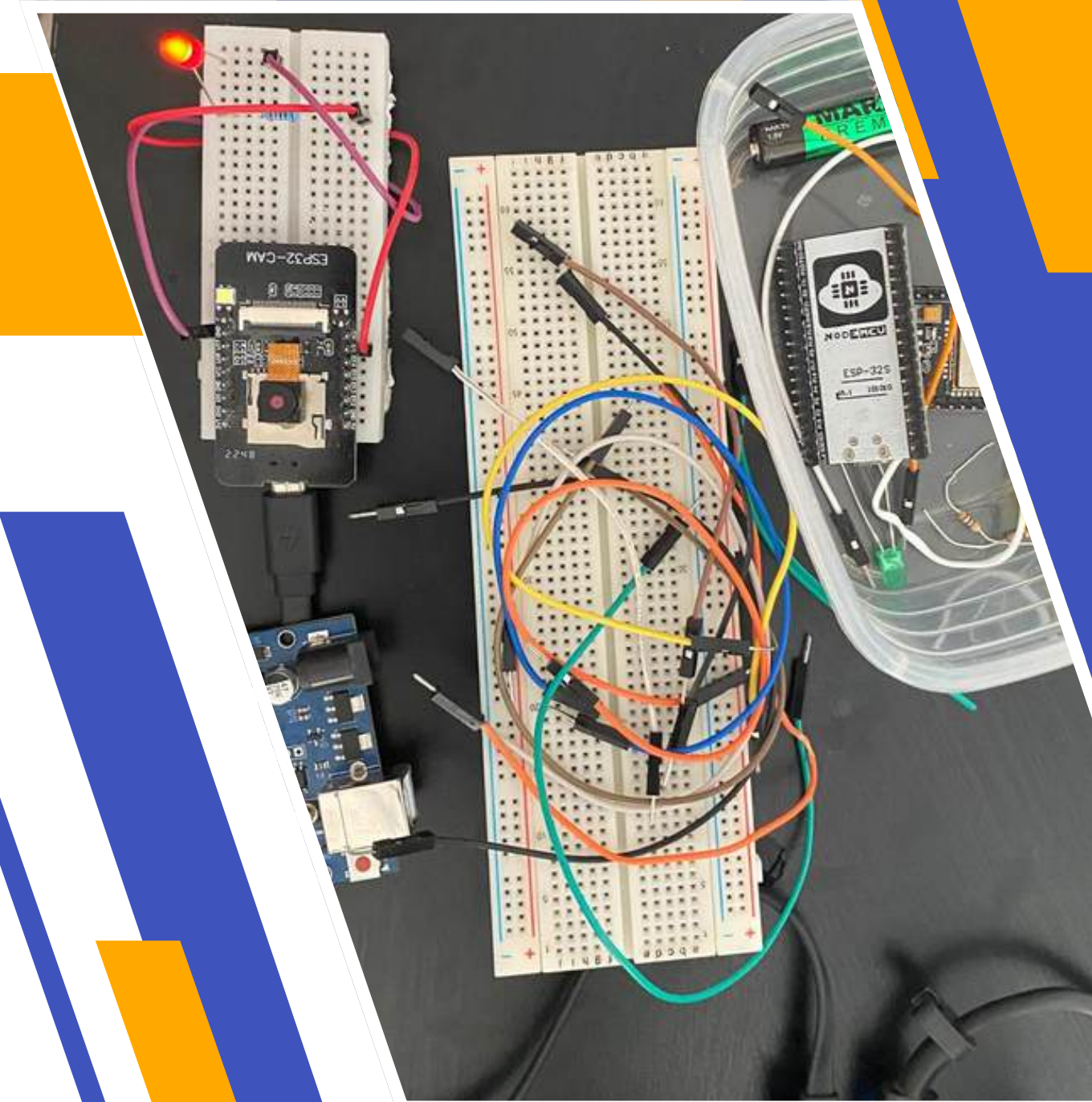
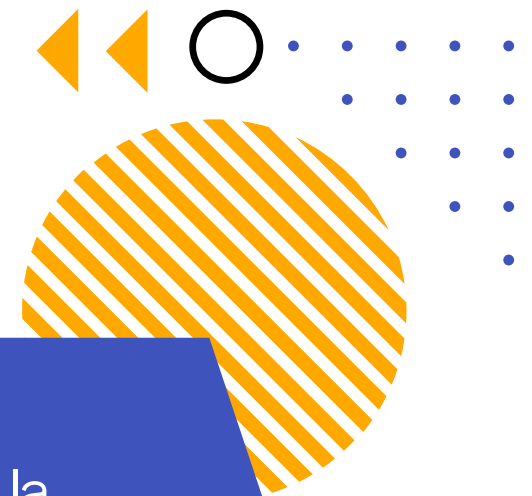




Table des matières



1

Introduction

6

Description du fonctionnement de la solution

2

Problématique

7

Démonstration et test

3

Objectif de solution

8

conclusion

4

Matériels utilisés

9

perspective

5

Architecture de la solution

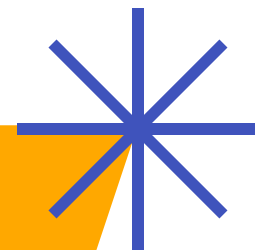
Introduction

Ce projet a pour objectif de développer un système d'éclairage intelligent contrôlé par des gestes de la main. Grâce à la caméra ESP32-CAM et à un modèle d'intelligence artificielle embarquée conçu avec Edge Impulse, l'utilisateur peut allumer ou éteindre la lumière simplement en effectuant un geste devant la caméra, sans contact physique.

Cette solution vise à offrir une alternative moderne aux interrupteurs classiques, en améliorant le confort, l'hygiène et l'accessibilité, notamment dans les lieux publics ou pour les personnes à mobilité réduite. Le traitement se fait localement, ce qui permet un fonctionnement rapide, autonome et sans connexion Internet.



Problématique



Capteur (ESP32-CAM)

Comment capter efficacement les gestes de l'utilisateur en temps réel ?

Traitement (Edge Impulse)

Comment entraîner un modèle d'IA précis et léger pour fonctionner sur un microcontrôleur ?

Autonomie

Comment assurer un fonctionnement rapide, autonome et sans dépendance à Internet ?

Reconnaissance

Comment garantir une reconnaissance fiable des gestes dans des conditions variées ?

Commande

Comment convertir la détection du geste en une action concrète sur l'éclairage (allumer/éteindre) ?

Objectif de Solution

**gestion
d'éclairage**

Automatisation

Gérer l'éclairage automatiquement grâce à la détection de gestes, sans intervention manuelle.

Caméra ESP32-CAM

Utiliser une caméra pour capter les gestes de l'utilisateur

IA embarquée

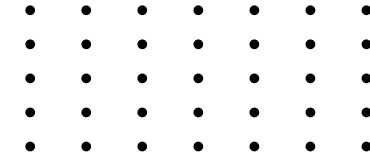
Intégrer un modèle d'intelligence artificielle pour la reconnaissance des gestes.

Accessibilité et hygiène

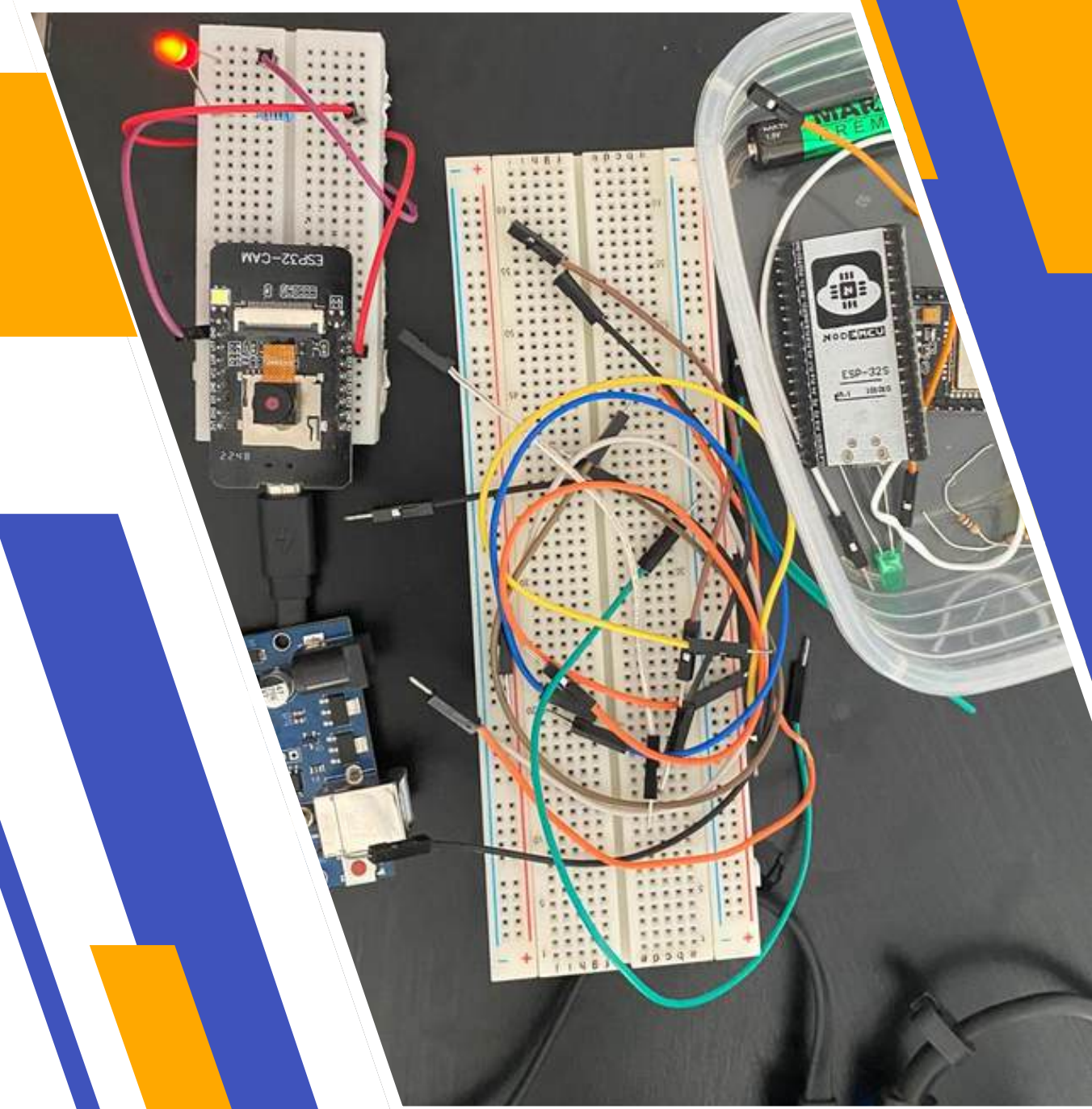
Permettre une gestion sans contact, accessible à tous, y compris aux personnes à mobilité réduite.



Internet of Things

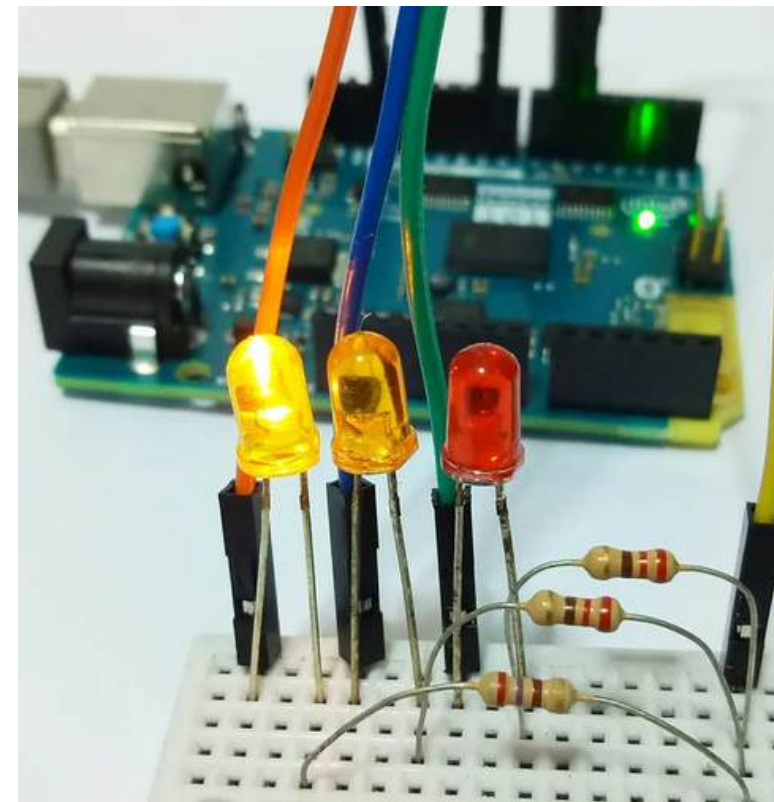


Matériels utilisés

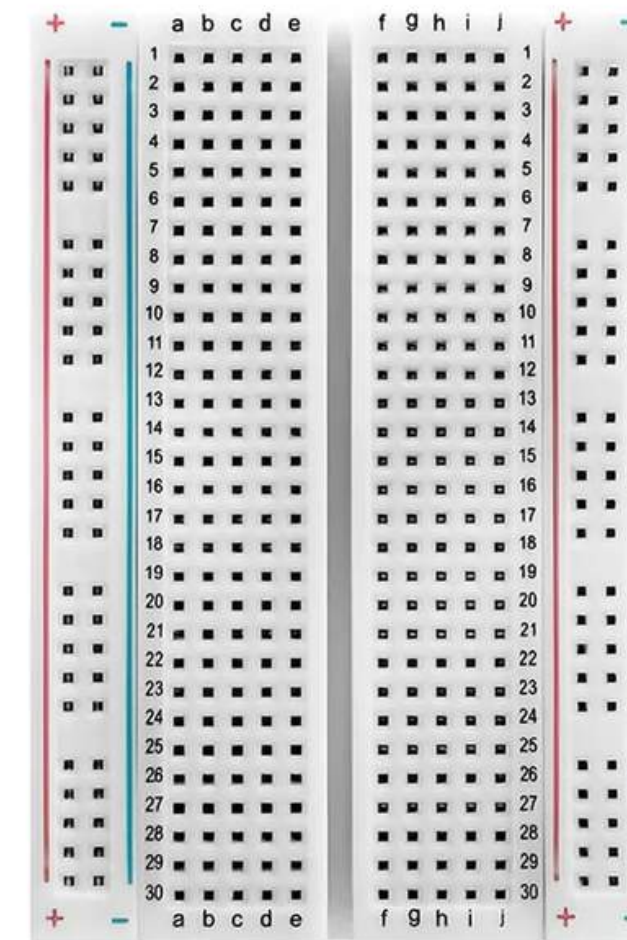




ESP32-CAM



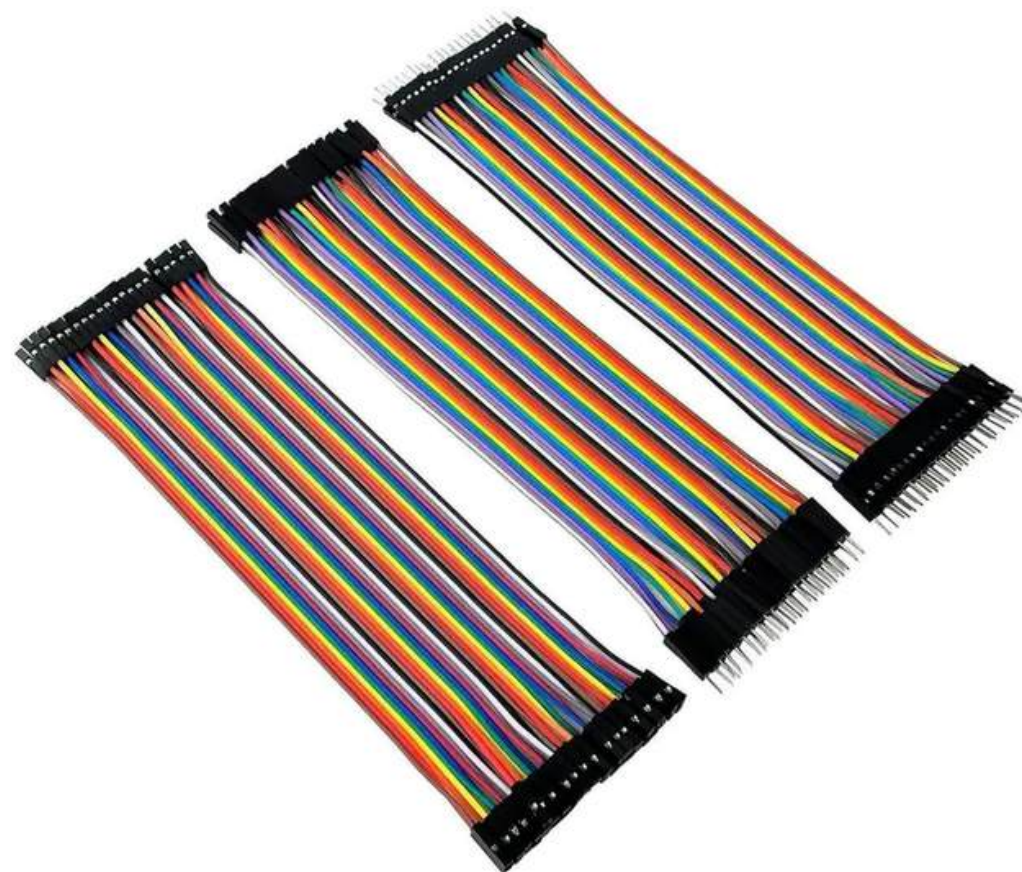
LED



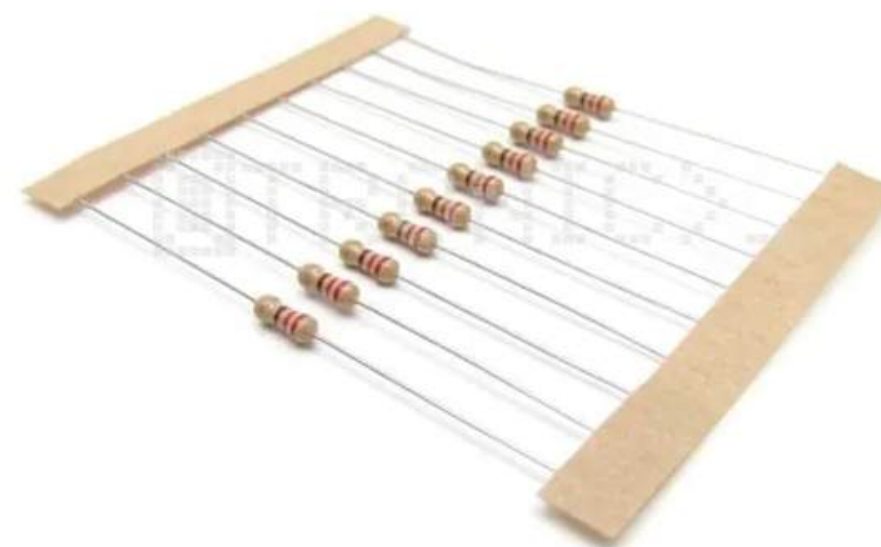
Breadboard



Câbles USB



Câbles Dupont

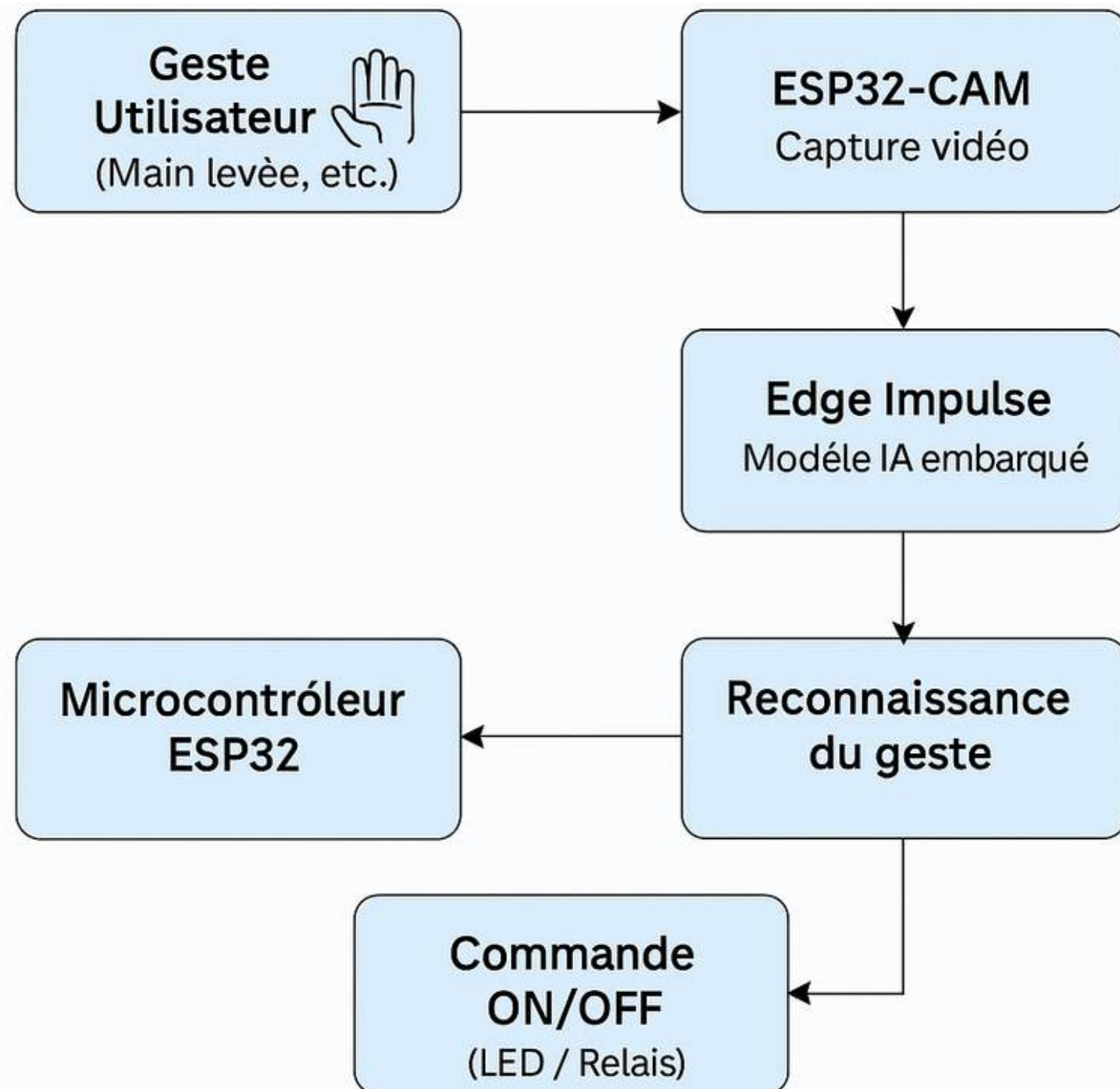


Resistances



téléphone portable

Architecture de la solution



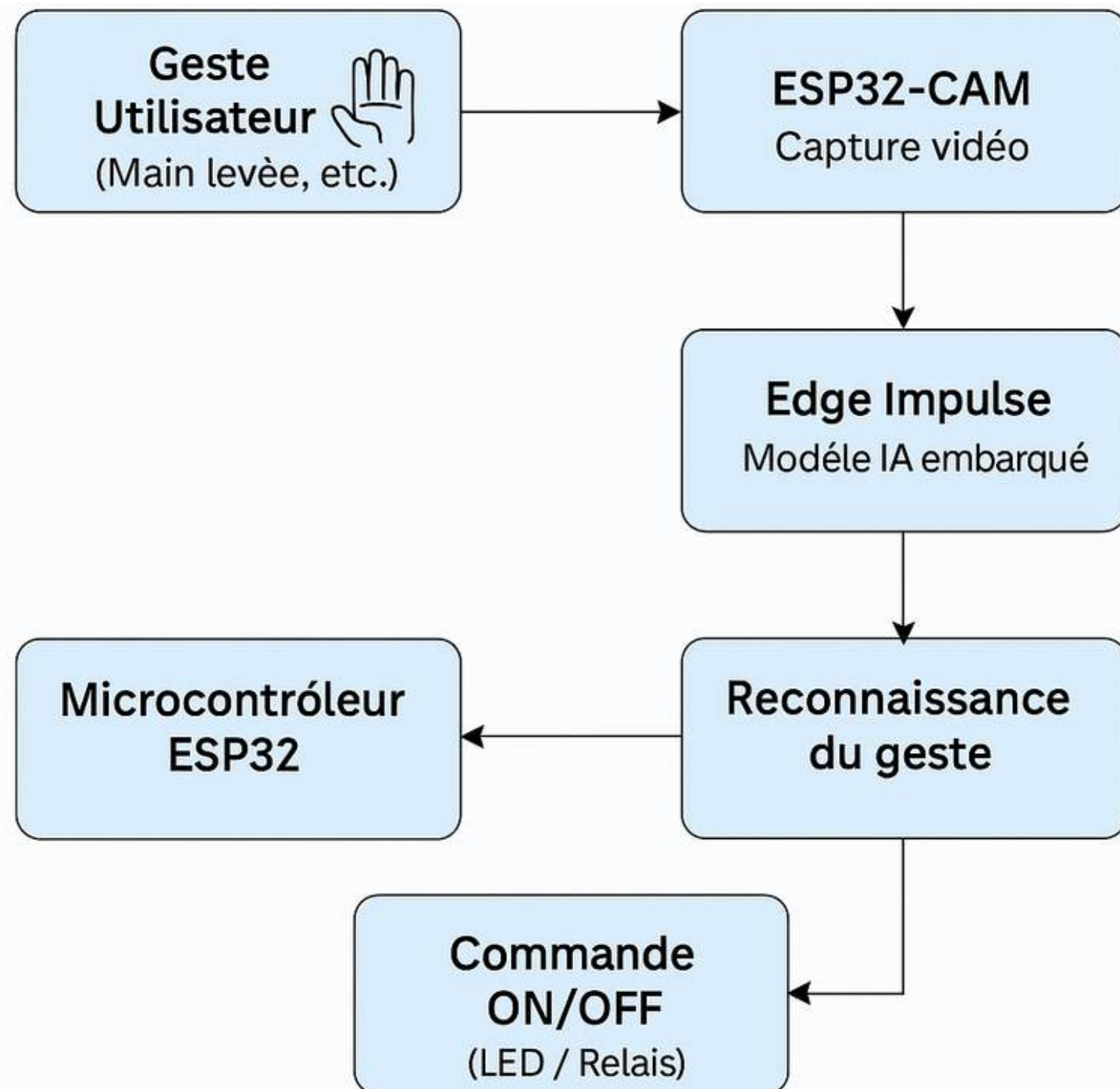
gestion d'éclairage

👋 **1. Geste Utilisateur** : L'utilisateur effectue un geste visible (ex. : lever la main, faire un signe). Ce geste est l'entrée principale du système.

📷 **2. ESP32-CAM (Capture Image)** : un microcontrôleur avec caméra intégrée, capte la vidéo en temps réel du geste de l'utilisateur.

🧠 **3. Edge Impulse – Modèle IA embarqué** : Les images capturées sont traitées par un modèle d'intelligence artificielle embarqué via Edge Impulse. Edge Impulse permet d'entraîner un modèle IA (classification d'image dans ce cas) à reconnaître différents gestes.

Architecture de la solution



gestion d'éclairage

✓ 4. Reconnaissance du geste

Le modèle IA analyse la vidéo et détecte si un geste spécifique correspond à l'un des gestes appris (ex. : main levée = "ON", poing fermé = "OFF").

🔌 5. Microcontrôleur ESP32

L'ESP32 exécute l'action correspondante en fonction du geste reconnu. Il agit comme centre de décision/action après la détection du geste.

💡 6. Commande ON/OFF (LED / Relais)

En sortie, l'ESP32 commande un périphérique selon le geste détecté.

Exemples :

- Allumer/éteindre une LED
- Activer/désactiver un relais (pour contrôler un appareil plus puissant)

Description du fonctionnement de la solution

01

Acquisition de données :

- Capture des images de gestes (main levée, poing fermé...) avec un téléphone portable ou ESP32-cam.
- Envoi des images vers Edge Impulse pour l'annotation (zones du geste avec bounding boxes).

02

Entraînement du modèle IA

- Utilisation d'un modèle de détection d'objets (ex. FOMO).
- Entraînement et optimisation du modèle pour une exécution sur ESP32-CAM (TinyML).

03

Déploiement :

- Téléchargement du modèle IA compilé pour microcontrôleur.
- Intégration du modèle dans un sketch Arduino ou PlatformIO.

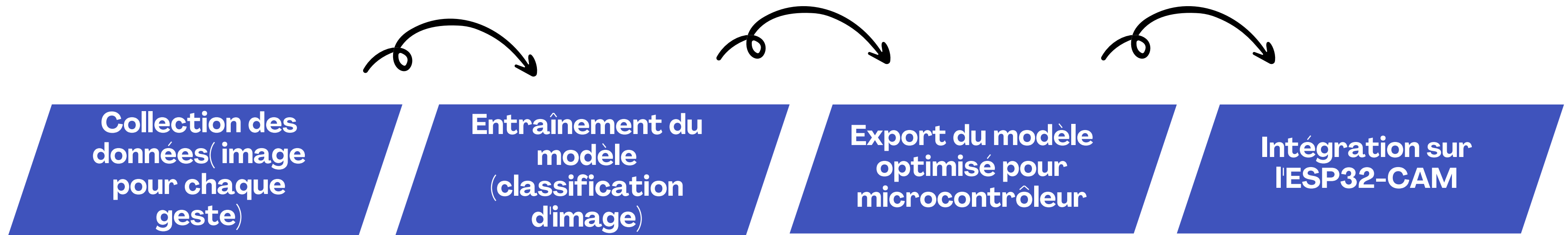
04

Utilisation :

- L'ESP32-CAM capture en direct les gestes de la main.
- Le modèle IA embarqué détecte le geste (ON/OFF).
- Le microcontrôleur exécute l'action (activer ou désactiver une LED ou un relais).

Entraînement du Modèle IA (Edge Impulse)

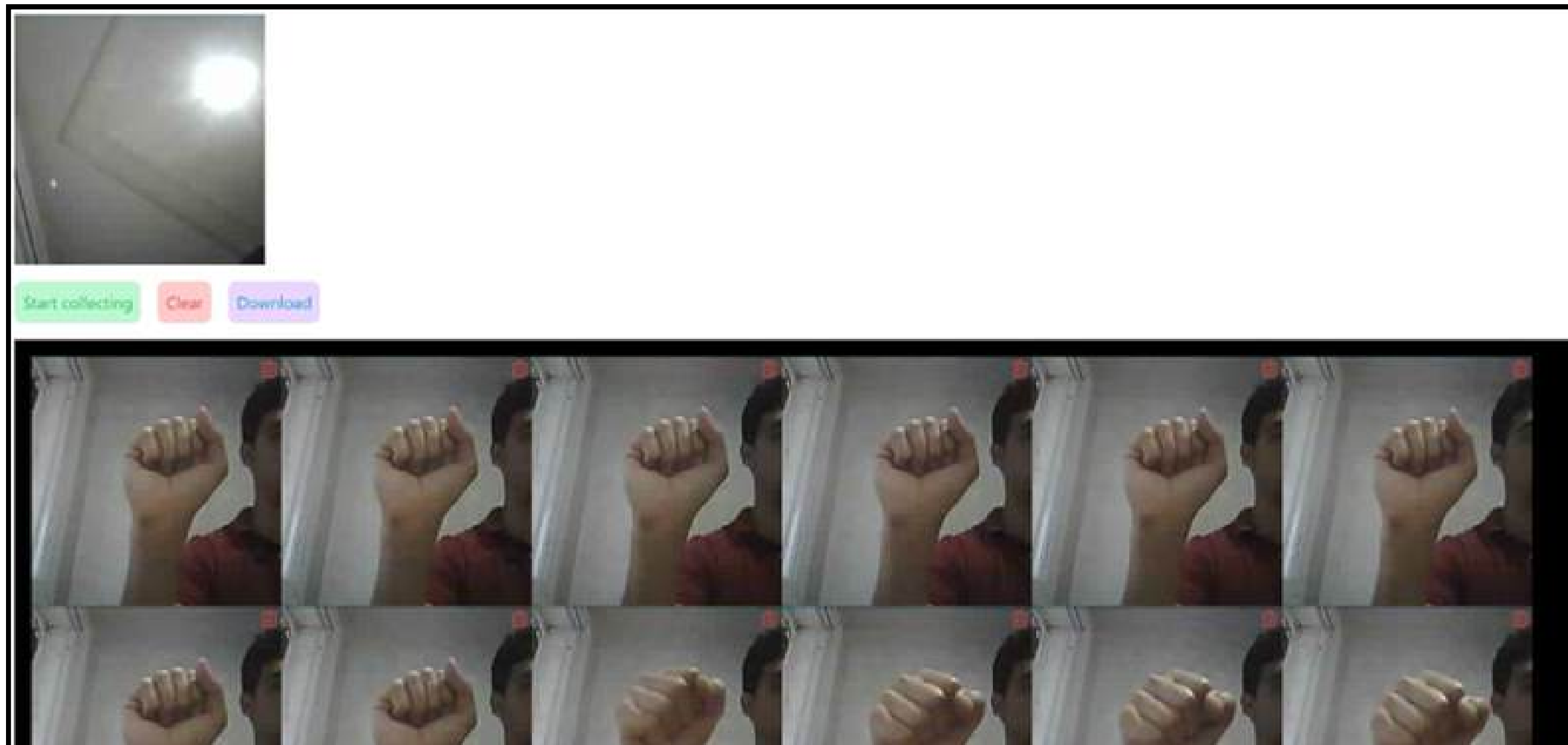
**gestion
d'éclairage**



Collection des données

gestion
d'éclairage

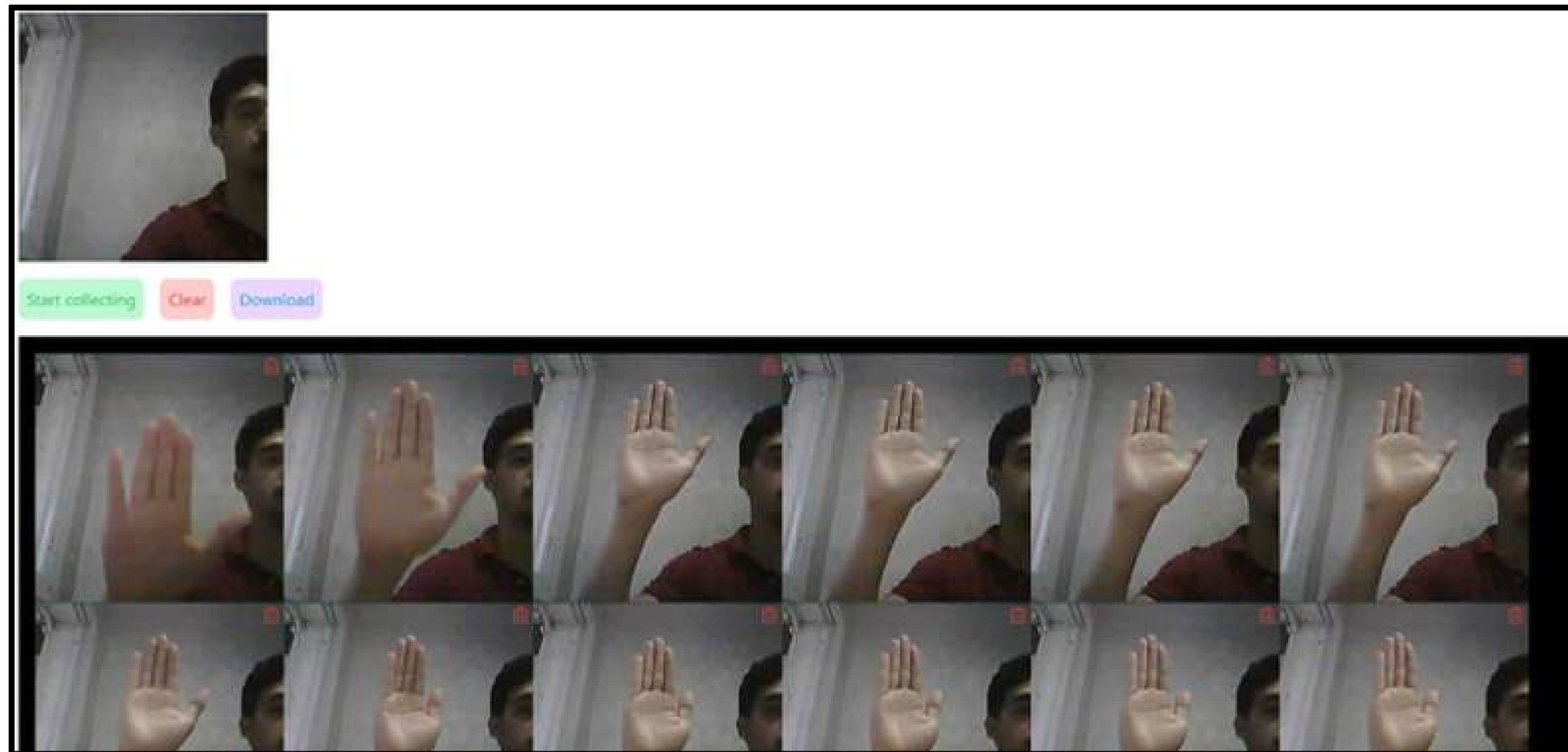
- CLoSe hand gestures



Collection des données

- Open hand gestures

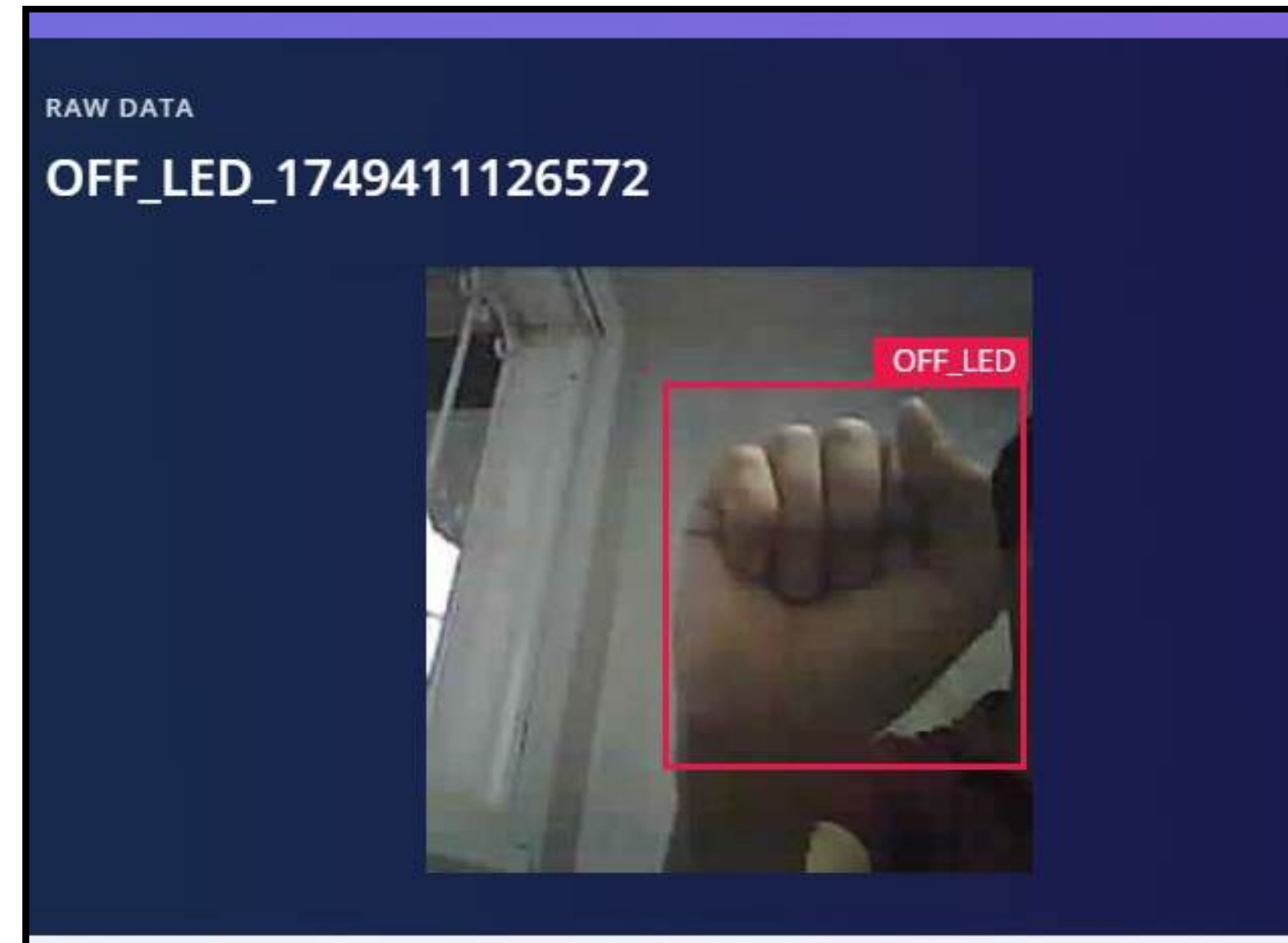
gestion
d'éclairage



Entraînement du modèle (classification d'image)

gestion
d'éclairage

- labelling



Entraînement du modèle (classification d'image)

gestion
d'éclairage

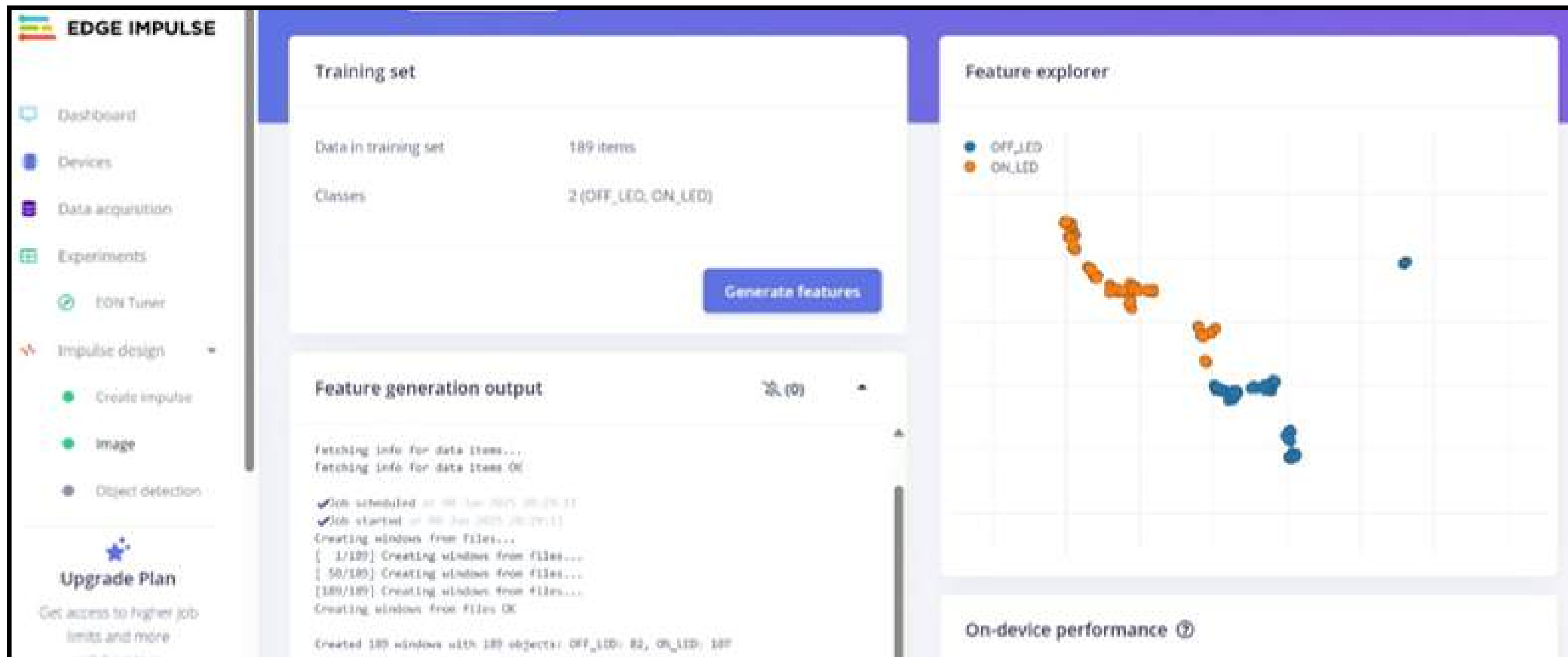
- Create Impulse

The screenshot displays the Edge Impulse web interface. On the left is a sidebar with navigation links: Dashboard, Devices, Data acquisition, Experiments, EON Tuner, Impulse design (selected), Create impulse, Retrain model, and Live classification. Below these is an 'Upgrade Plan' section. The main workspace is titled 'An impulse takes raw data, uses signal processing to extract features, and then uses a learning block to classify new data.' It contains four colored blocks: a red 'Image data' block with input axes (image), width (96), height (96), and a 'Fit shortest' resize mode; a white 'Image' block with a name field (image) and an input axes dropdown (image); a blue 'Object Detection (Images)' block with a name field (Object detection), an input features checkbox (checked for image), and output features (2 (OFF_LED, ON_LED)); and a green 'Output features' block with a checkmark icon and the same output features. A 'Save Impulse' button is located to the right of the output features block. At the bottom, there are dashed boxes for 'Add a processing block' and 'Add a learning block'.

Entraînement du modèle (classification d'image)

gestion
d'éclairage

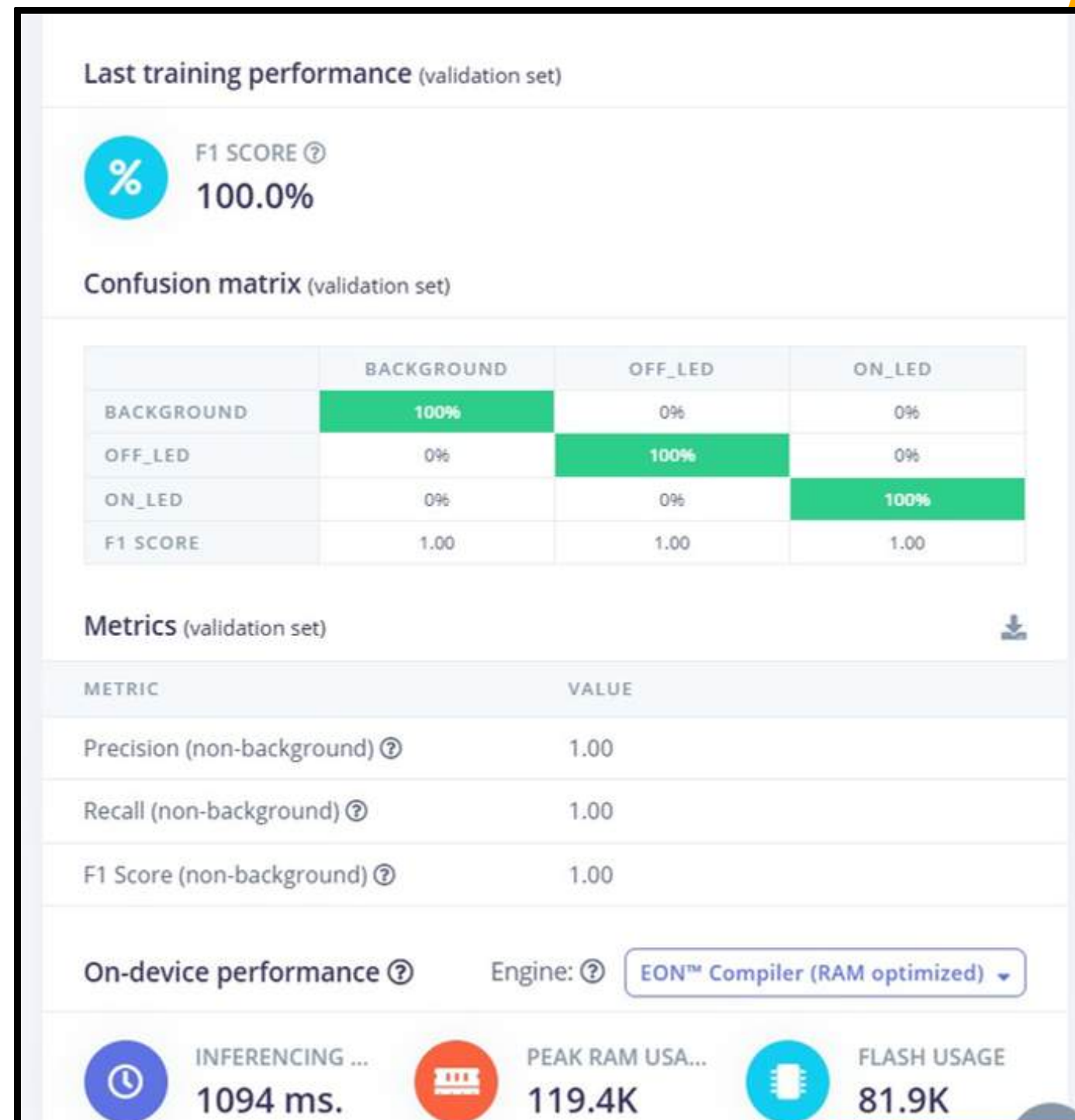
- Training Sets :



Entraînement du modèle (classification d'image)

gestion
d'éclairage

- Object Detection :



Export du modèle optimisé pour microcontrôleur

gestion
d'éclairage

- Deployment :

The screenshot displays the Edge Impulse web interface for a project named 'dair1 / IOT_project'. The left sidebar contains navigation links: Dashboard, Devices, Data acquisition, Experiments, EON Tuner, and Impulse design. Under 'Impulse design', there are options for 'Create impulse', 'Image', and 'Object detection'. An 'Upgrade Plan' section is also visible. The main content area is titled 'Configure your deployment' and includes a search bar with 'Arduino library' entered. Below this, the 'SELECTED DEPLOYMENT' section shows the 'Arduino library' with a description: 'An Arduino library with examples that runs on most Arm-based Arduino development boards.' The 'MODEL OPTIMIZATIONS' section explains that optimizations can increase on-device performance but may reduce accuracy. It features a toggle for 'EON™ Compiler' with a note: 'Same accuracy, 17% less RAM, 36% less ROM.' At the bottom, a table shows quantization results for 'int8' across 'IMAGE', 'OBJECT DETECTION', and 'TOTAL' categories. On the right, the 'Run this model' section provides a QR code and a 'Launch in browser' button. A 'Resume tutorial' button is located in the bottom right corner.

EDGE IMPULSE

dair1 / IOT_project PERSONAL Target: Espressif ESP-EYE...

Configure your deployment

You can deploy your impulse to any device. This makes the model run without an internet connection, minimizes latency, and runs with minimal power consumption. [Read more.](#)

Arduino library X

SELECTED DEPLOYMENT

Arduino library
An Arduino library with examples that runs on most Arm-based Arduino development boards.

MODEL OPTIMIZATIONS
Model optimizations can increase on-device performance but may reduce accuracy.

EON™ Compiler
Same accuracy, 17% less RAM, 36% less ROM.

Quantized (int8)

	IMAGE	OBJECT DETECTION	TOTAL

Run this model
Scan QR code or launch in browser to test your prototype

Launch in browser

Resume tutorial

Intégration sur l'ESP32-CAM

gestion
d'éclairage

- Installer la bibliothèque dans l'IDE Arduino :



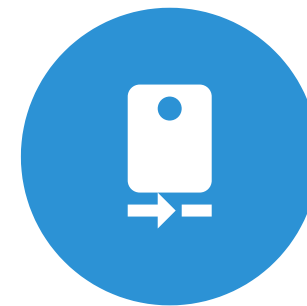
Ouvrir l'IDE Arduino



2. Menu :
Sketch > Include Library >
Add .ZIP Library



3. Sélectionner le
fichier .zip que tu
viens de télécharger



4 .L'IDE va installer
une bibliothèque
nommée comme
ton projet Edge
Impulse



Intégration sur l'ESP32-CAM

gestion
d'éclairage

- Tester un exemple de code d'inférence :

1

Menu : Fichier > Exemples >
Nom_de_ta_bibliothèque_EI >
ESP32 > Camera

2

Choisis un exemple
selon ton projet (image,
son, etc.)

3

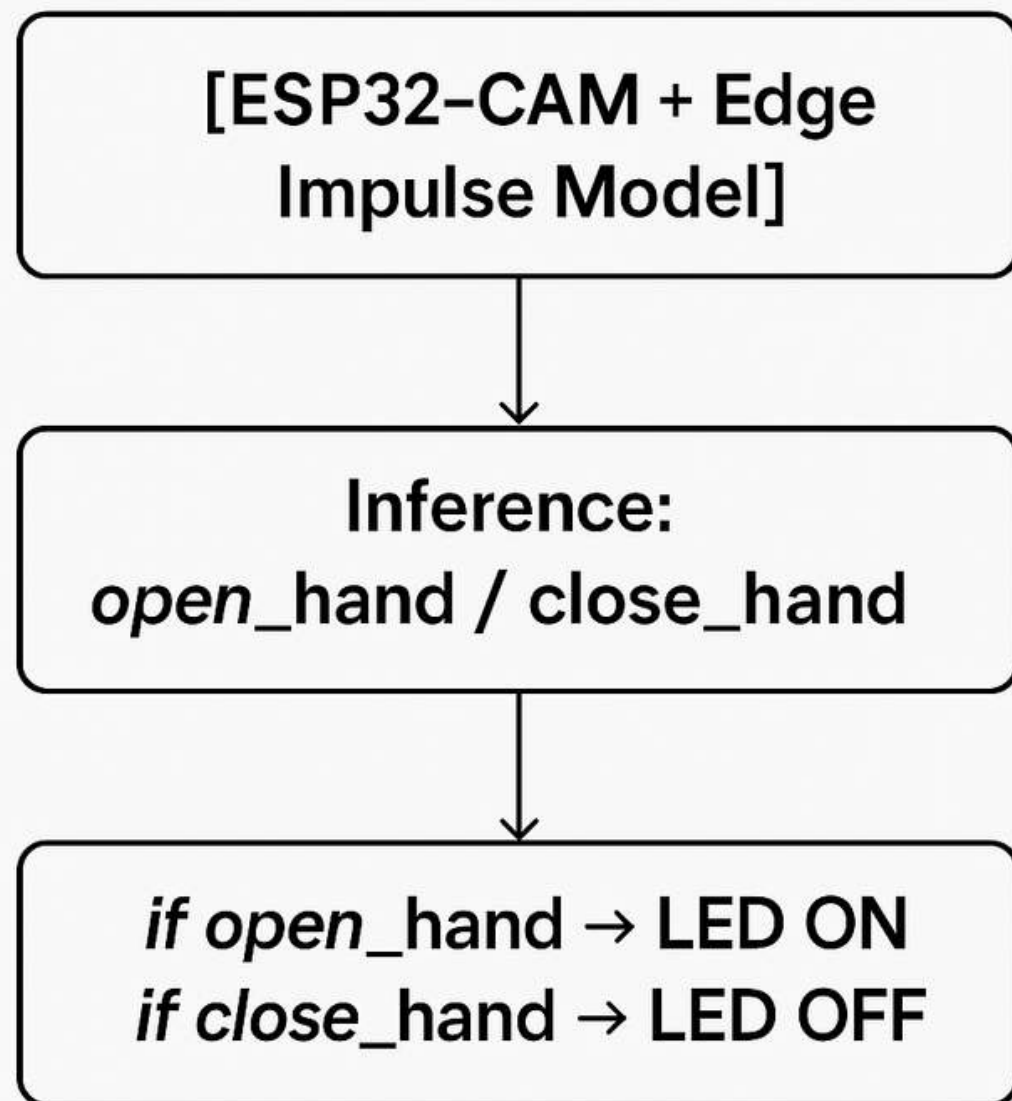
Modifie le code selon
les broches spécifiques
de l'ESP32-CAM
(exemple : caméra,
capteur, GPIO, etc.)

DÉMONSTRATION ET TEST



Contrôle des LED basé sur les résultats d'inférence

gestion
d'éclairage



Dans cette application, l'ESP32-CAM exécute un modèle entraîné par Edge Impulse pour classer les gestes de la main. En fonction de la prédiction la plus élevée (ouvrir ou fermer la main), il contrôle une LED connectée au GPIO choisi.

Contrôle des LED basé sur les résultats d'inférence

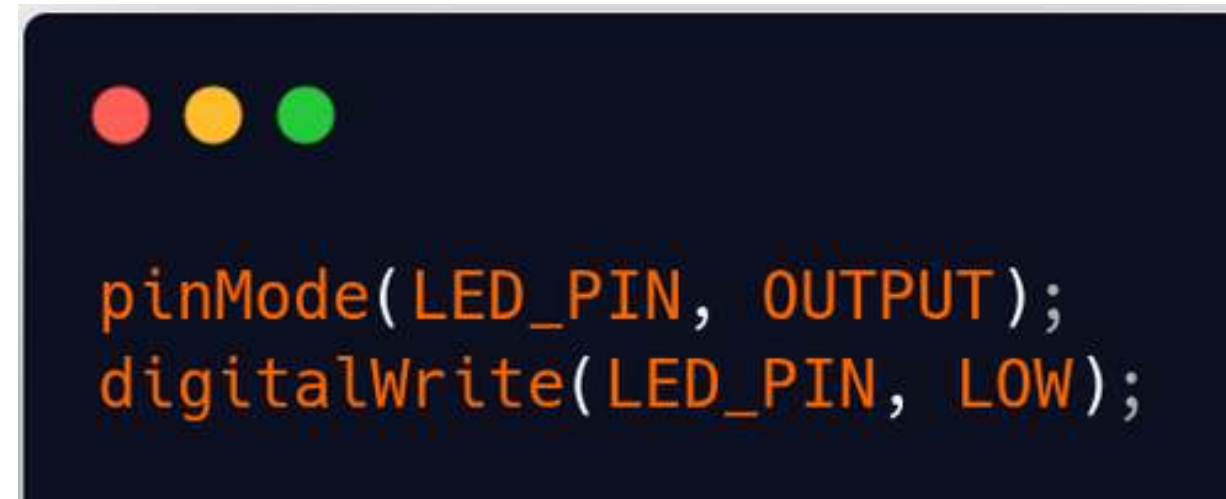
- Code LED:



```
#define led_pin 2
```

- c'est la broche 2 où LED est connectée.

- In Setup():



```
pinMode(LED_PIN, OUTPUT);  
digitalWrite(LED_PIN, LOW);
```

- `pinMode(LED_PIN, OUTPUT);`
Configure la broche LED_PIN (ici la broche 2) en mode sortie.
Ça veut dire que cette broche pourra envoyer un signal (par exemple allumer une LED).
- `digitalWrite(LED_PIN, LOW);`
Met la broche LED_PIN à LOW (0 Volt), donc éteint la LED connectée à cette broche au démarrage.

Contrôle des LED basé sur les résultats d'inférence

- *après la boucle de classification:*

```
// Loop through the classifications
for (size_t i = 0; i < result.classification[0].size; i++) {
    ei_classifier_label_t c =
    result.classification[0].classification[i];

    // If model predicts "OFF_LED" with high confidence, turn LED
    off
    if (strcmp(c.label, "OFF_LED") == 0 && c.value > 0.8f) {
        digitalWrite(LED_PIN, LOW);
    }
    // If model predicts "ON_LED" with high confidence, turn LED on
    else if (strcmp(c.label, "ON_LED") == 0 && c.value > 0.8f) {
        digitalWrite(LED_PIN, HIGH);
    }

    // Print the label and its confidence value
    ei_printf("%s: %.5f\n", c.label, c.value);
}
```

Ce code parcourt tous les labels détectés par un classificateur, affiche leur score, et garde en mémoire le label avec le score le plus élevé. Puis :

- Si le label détecté est "open_hand", il allume la LED (digitalWrite(LED_PIN, HIGH);).
- Si c'est "close_hand", il éteint la LED (digitalWrite(LED_PIN, LOW);).

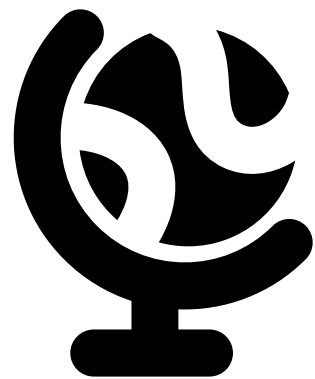


gestion
d'éclairage

Conclusion

- Le système remplit son objectif : reconnaissance fiable de gestes simples avec matériel peu coûteux et traitement embarqué sans cloud.
- Il répond aux besoins de domotique intelligente et d'interfaces naturelles.

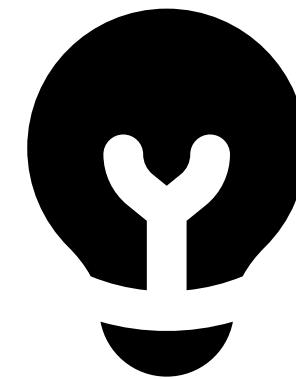
Perspectives



**Ajouter un mode
apprentissage
personnalisé pour les
utilisateurs.**



- **intégrer à un système domotique complet (Home Assistant).**
- **Élargir l'usage à d'autres appareils (TV, volets, alarmes...).**



- **Ajouter des gestes plus complexes (ex : variation d'intensité).**



INTERNET OF THINGS

MERCI POUR VOTRE ATTENTION

Si vous avez d'autres questions, n'hésitez pas à les poser, ou si vous souhaitez explorer notre projet, n'hésitez pas à nous contacter.



Fsdm, Usmba, Fès, Morocco

