



IOT SURVEILLANCE VEHICLE

Projet IoT – GLSI2

par: Amine Saidani & Dhia Zorai

1. Objective et Concept



Idée de départ

Recyclage d'une voiture RC classique pour la transformer en un véhicule intelligent.



Fonctionnalités innovantes

- Contrôle Wi-Fi
- Vision embarquée
- Détection faciale et émotionnelle



Enjeux IoT

- Connectivité et communication

2. Architecture Globale du Système



Contrôle Moteur

Le contrôle du véhicule est assuré par une chaîne de commande suivante :

- Smartphone : Interface utilisateur pour envoyer les commandes.
- Carte ESP8266 : Reçoit les ordres du smartphone via Wi-Fi.
- Driver L298N : contrôler la vitesse et le sens de rotation des moteurs DC.
- 2 Moteurs DC : Exécutent les mouvements du véhicule.



Vision et Traitement

Le système intègre des capacités de vision avancées :

- Carte ESP32-CAM : Acquiert les flux vidéo en temps réel.
- Connexion WiFi : Transmet les données vidéo vers l'ordinateur.
- Script Python (PC) : Traite les images pour la détection faciale et émotionnelle.



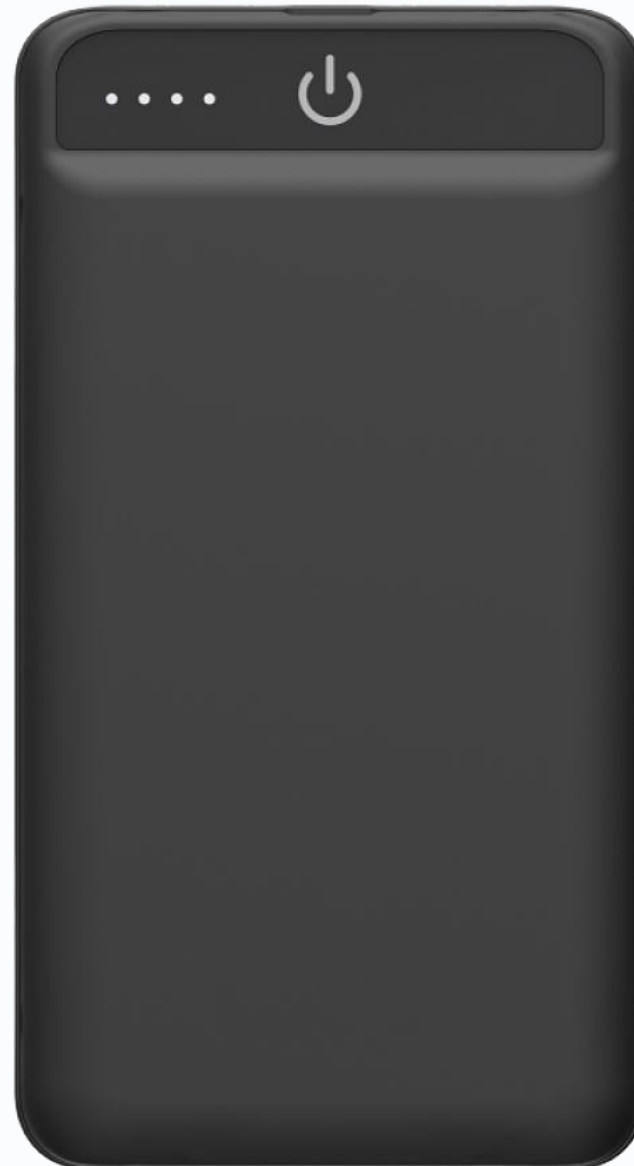
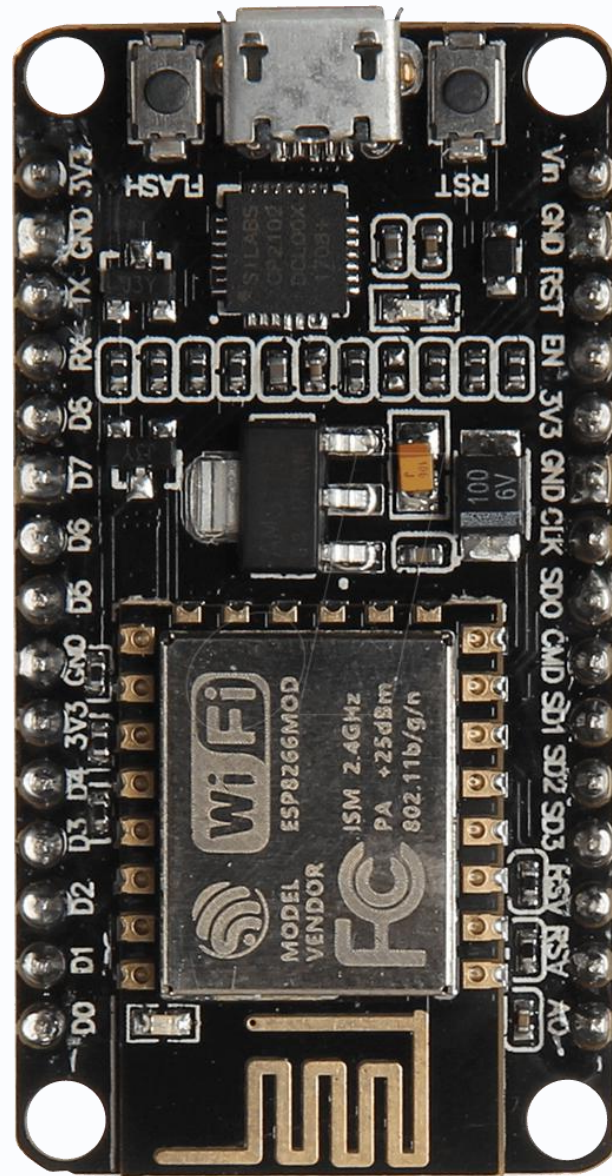
Lien entre les Modules

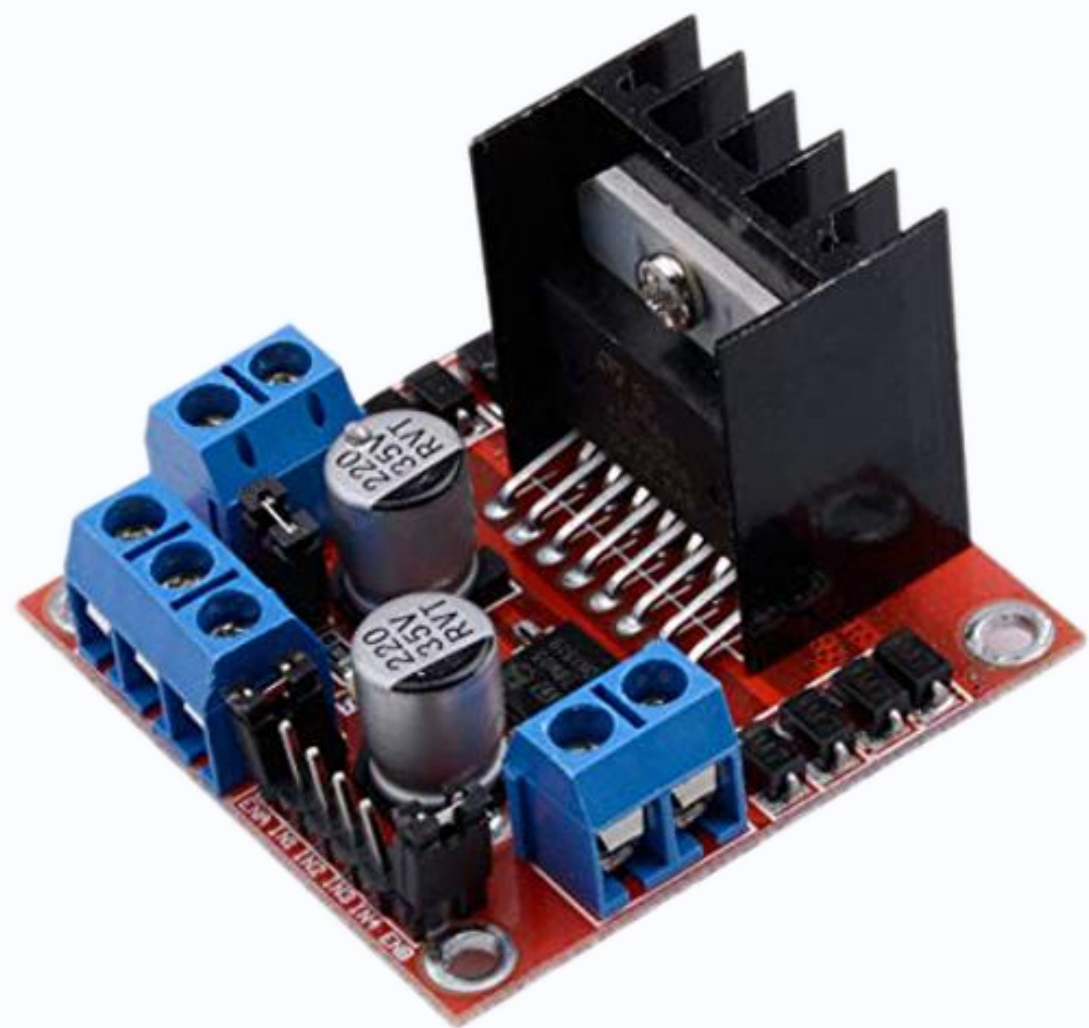
La communication est le pilier de cette architecture distribuée :

- Protocoles HTTP : Utilisé pour toutes les communications entre le smartphone, les cartes ESP et le PC.

3. Composants Matériels

Voici les principaux composants matériels utilisés dans l'architecture du système :





Power Bank

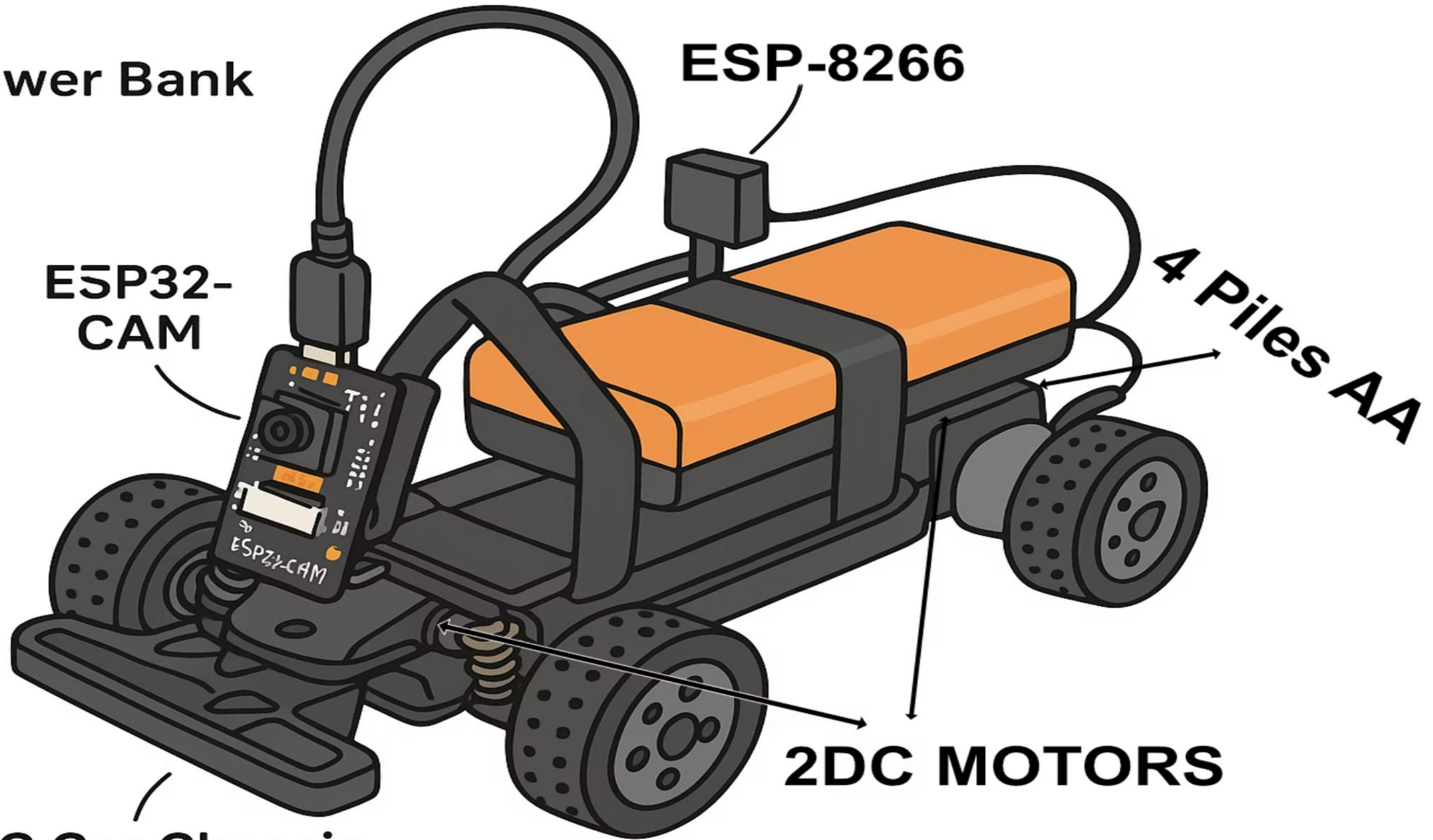
ESP-8266

ESP32-CAM

4 Piles AA

RC Car Chassis

2DC MOTORS



5. Architecture Logicielle

01

ESP8266

Héberge un mini serveur HTTP avec les commandes:

- Forward
- Backward
- Left
- Right
- Stop

02

ESP32-CAM

Diffuse le flux MJPEG vers le script Python.

03

Python (OpenCV+MediaPipe)

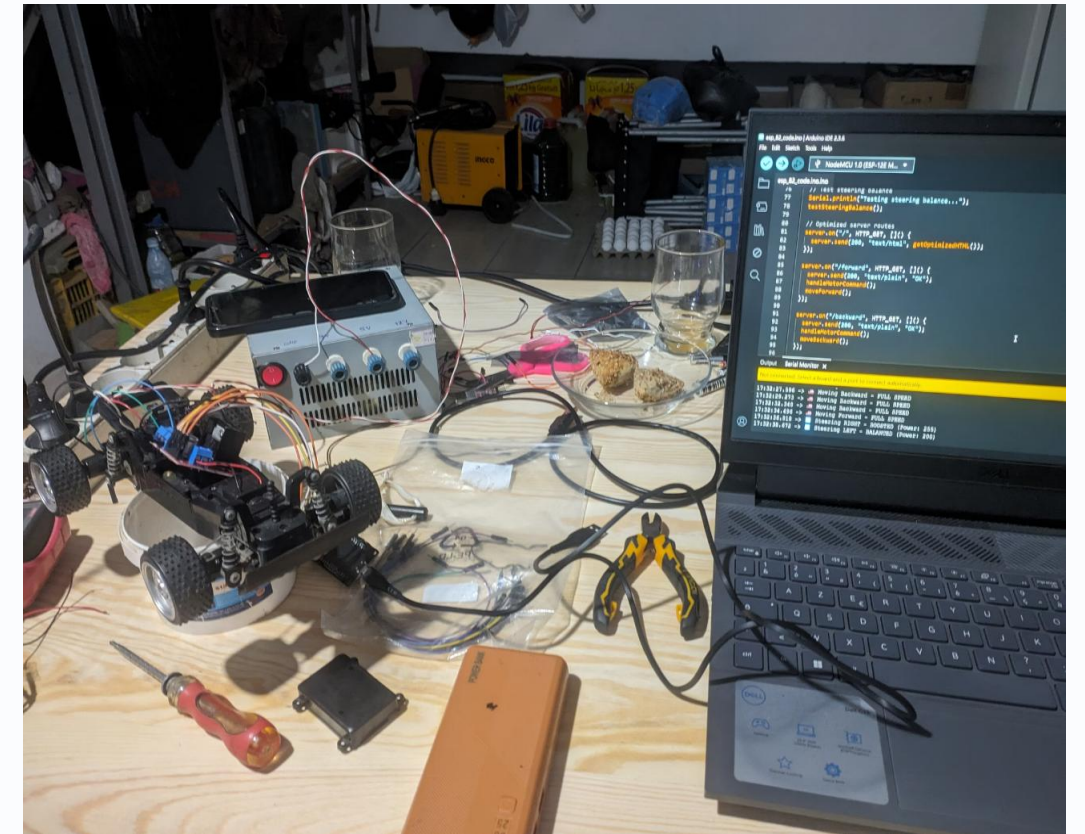
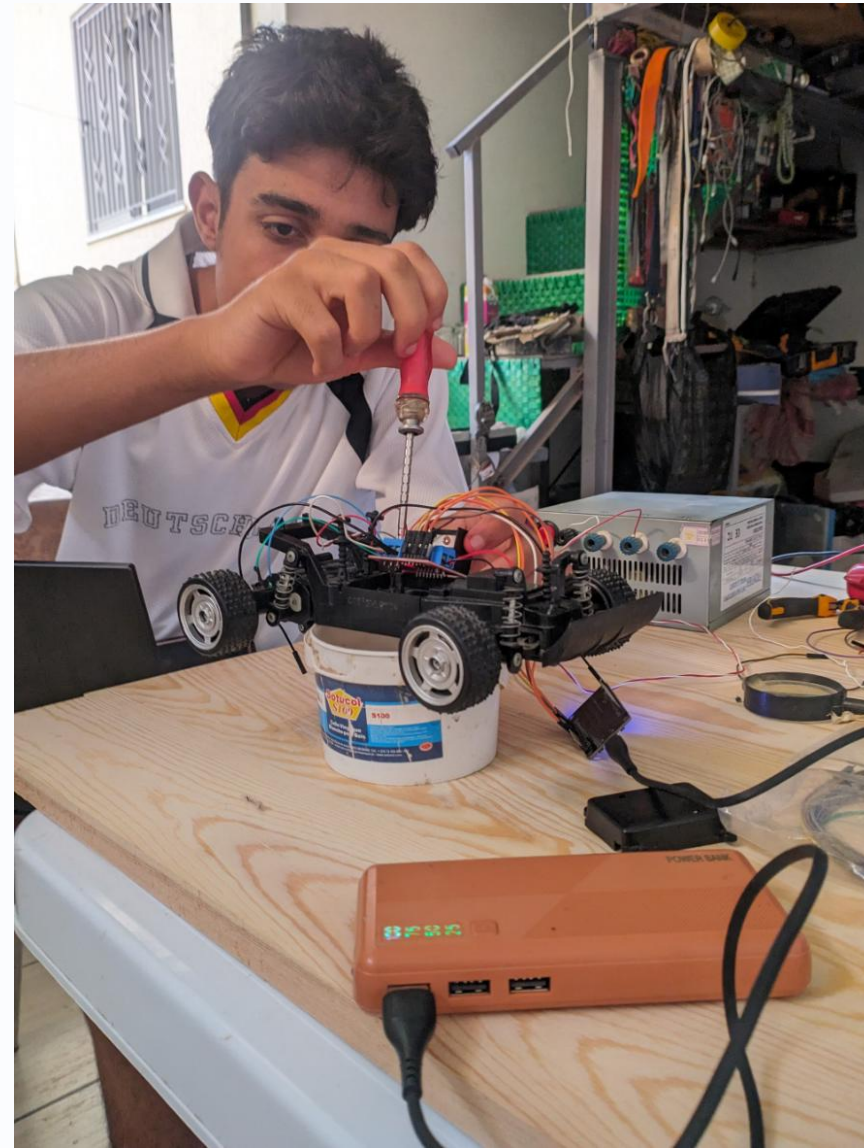
Effectue les opérations suivantes :

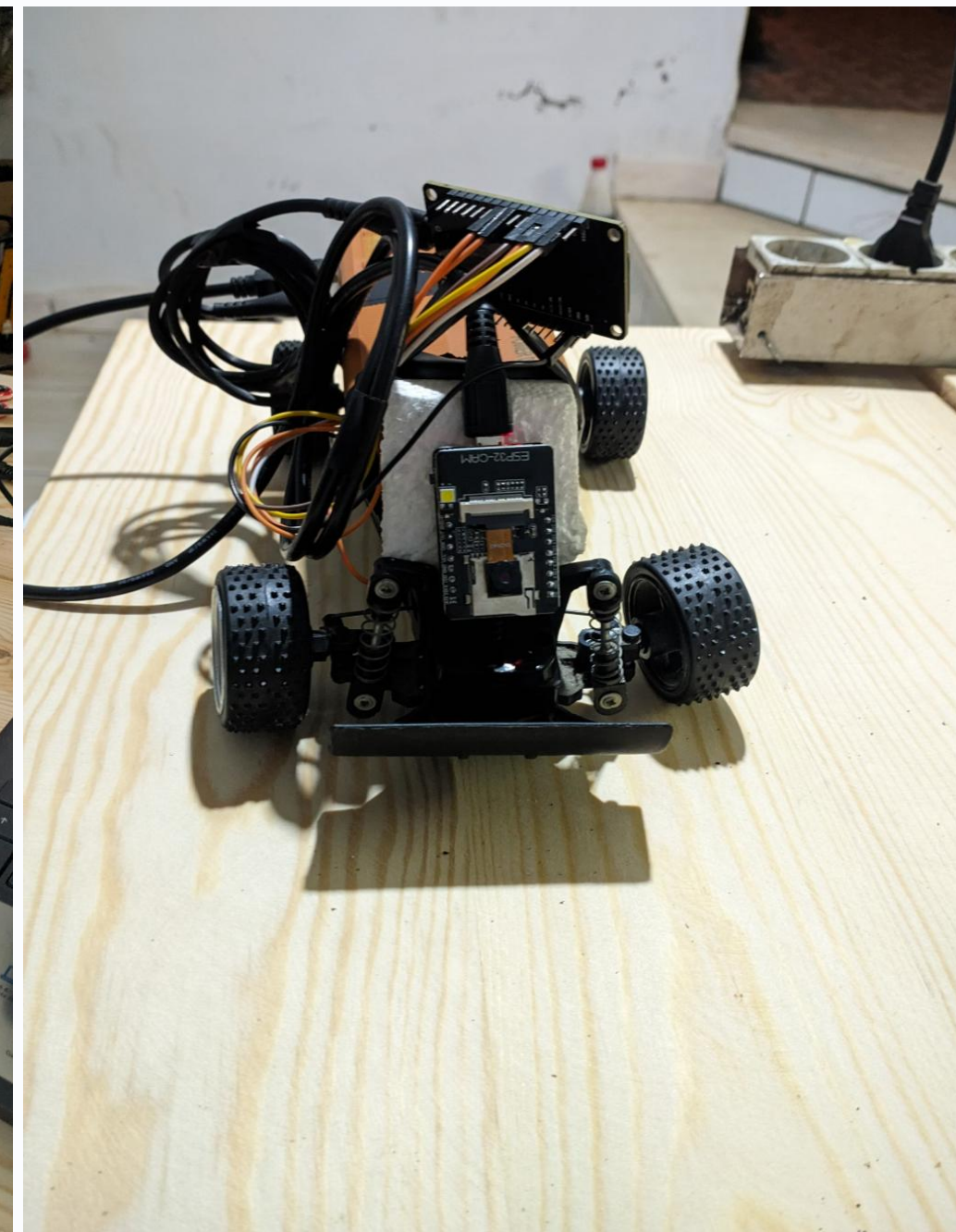
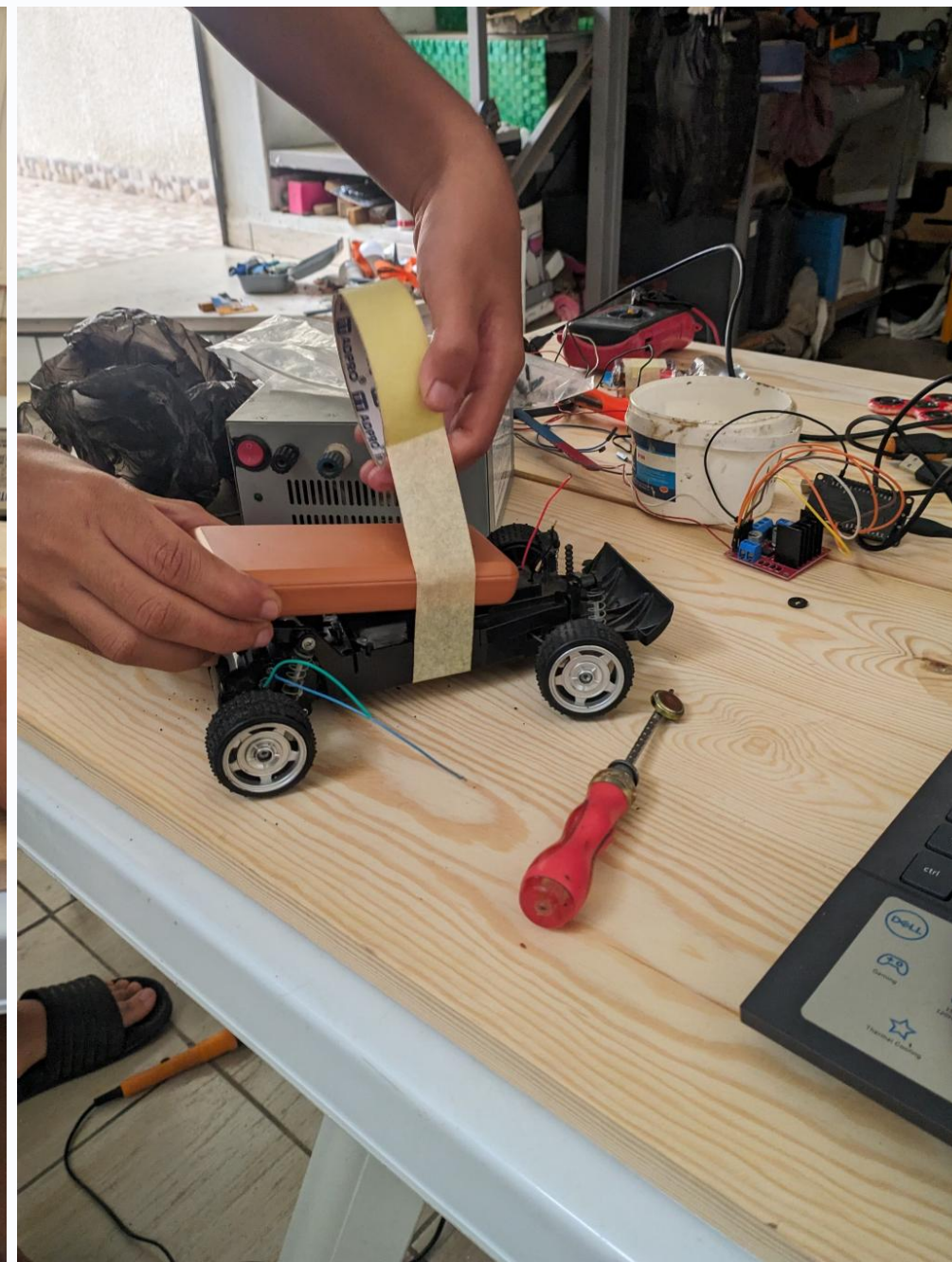
- Capture le flux vidéo
- Détecte les visages
- Génère des emojis dynamiques selon l'expression détectée

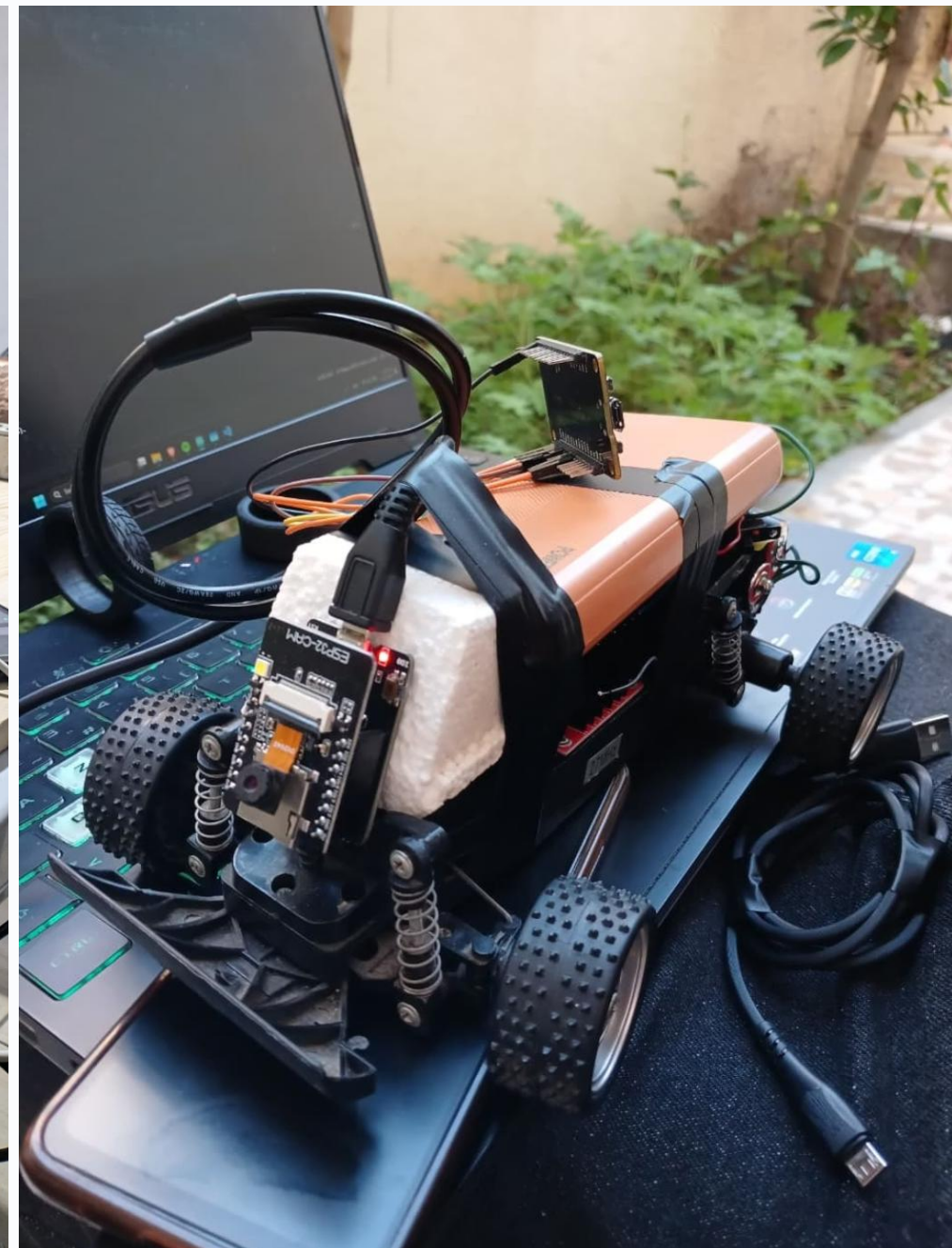
6. Communication et synchronisation

- Connectivité WiFi Partagée: Les deux modules ESP (ESP32-CAM et ESP8266) utilisent le même réseau WiFi fourni par le smartphone, simplifiant la configuration réseau.
- Smartphone comme Interface de Commande: Le smartphone sert d'interface utilisateur principale, envoyant les commandes au robot via le réseau WiFi.
- Traitement du Flux Vidéo par Python: Un script Python sur le smartphone reçoit et traite le flux vidéo en temps réel provenant de l'ESP32-CAM.

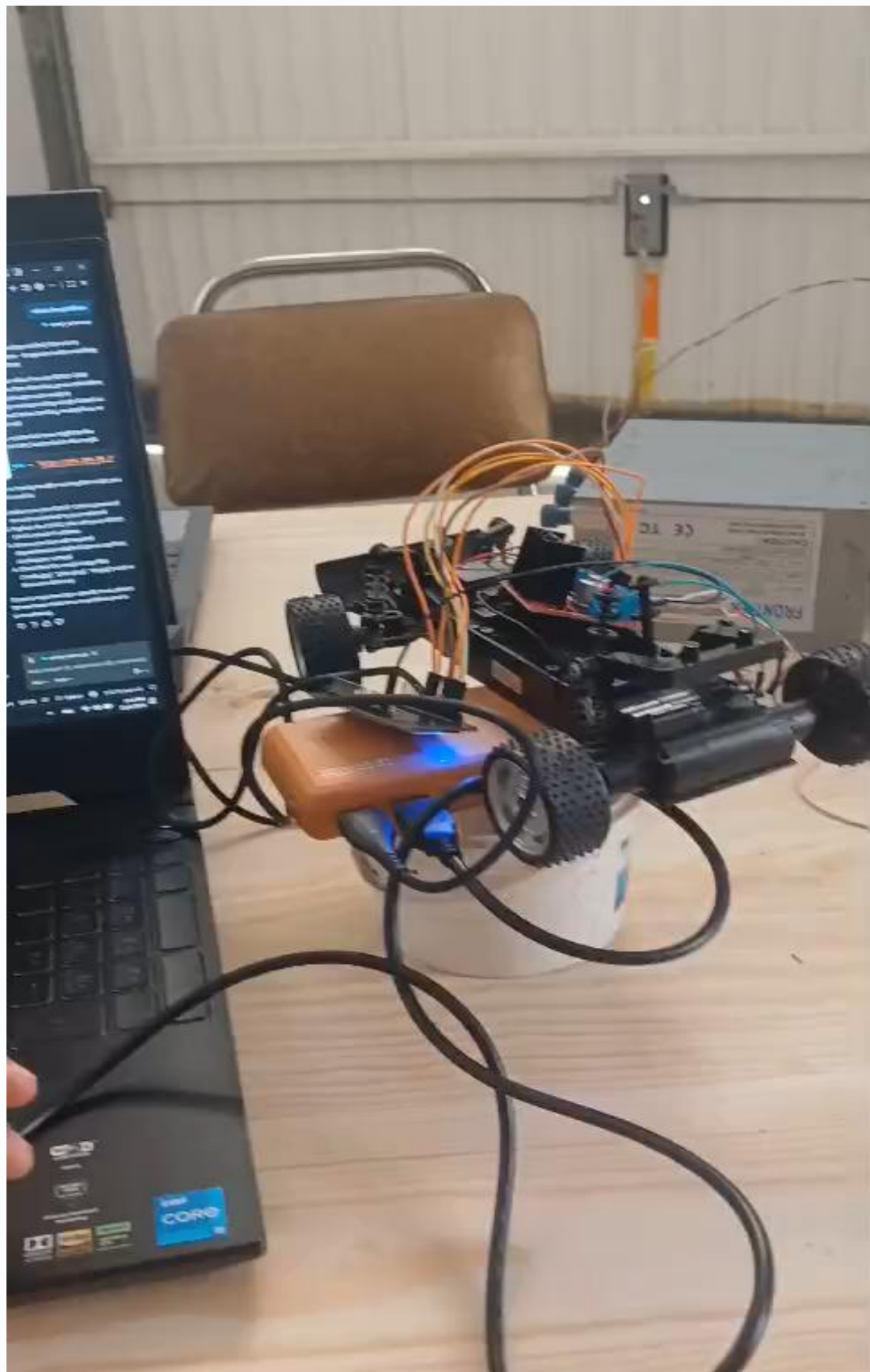
7. Le processus en images.











8. Défis et Solutions

1

PC ne reconnaissant pas l'ESP8266

Problème: Le PC refusait de détecter l'ESP8266 après de nombreux tests.

Solution: Installation d'anciens pilotes (version 3.5 au lieu de 3.9) a permis la détection de la carte.

2

Caméra refusant d'envoyer un flux vidéo

Problème: La caméra ne transmettait pas de flux vidéo, malgré diverses configurations.

Solution: Remplacement du module défectueux ("ribbon cable " endommagé) a résolu le problème.

3

Instabilité de l'ESP32-CAM avec les moteurs

Cause: Chutes de tension dues à la consommation des moteurs, cause une instabilité de l'ESP32-CAM.

Solution: Ajout d'une alimentation séparée via powerbank, améliorant partiellement la stabilité.

4

Vibrations du châssis et flux vidéo flou

Effet: Les vibrations du châssis rendent le flux vidéo tremblant et instable .

État actuel: Non totalement résolu, la caméra n'étant pas conçue pour filmer en mouvement sans stabilisation.

Thanks for
tuning in