Dans ce petit guide qui est le résultat de mes recherches, je vais essayer détailler ce que j’ai compris du processus de contrôle d’un robot mobile sur un site web via internet. Avec cela, tant que nous avons accès à Internet, nous pouvons accéder à notre robot partout.  
  
Nous aurons besoin d’un kit de voiture robotisée, d’un ESP32-CAM et de quelques lignes de code pour le serveur afin de pouvoir le déployer (Nous pouvons utiliser Heroku).

Le composant principal permettant de fonctionner via Internet est un microcontrôleur doté d'un module WiFi. Pour ce contrôle côté serveur et côté client, nous pouvons nous servir du serveur node.js et de quelques éléments front de base tels que javascript.

Pour le réaliser, nous pouvons nous servir de :

**Composants matériels**

1 – Kit complet voiture robotisée comprenant châssis, roues,

2 – ESP32 – CAM

3 – Pilote moteur L298N

4 - Alimentation 7 - 12 V  
5 - FTDI TTL

6 - Planche d’essai ou breadbord

**Composants logiciels**

1 – L’IDE Arduino

2 – Visual Studio Code

3 – Compte Heroku

**Etape 1 : Présentation du système**  
  
Tout d’abord, nos moteurs seront connectés via ESP32-CAM pour être contrôlés lorsqu'il y aura une commande de déplacement depuis notre site Web en utilisant le protocole de communication WebSocket. L'avantage des WebSockets (à l'instar des sockets) est de créer une connexion permanente qui permet au serveur d’envoyer des données lorsqu'il en a à sa disposition. L'avantage est que le client n'a pas besoin de demander régulièrement des données et qu'il n'y a pas de trace réseau redondante. L’inconvénient des WebSockets réside dans l’augmentation du nombre de connexions.   
  
Avec cela, nous pouvons établir une connexion en temps réel avec celui-ci et n'avons pas besoin de demander régulièrement des données comme HTTP. Ainsi, sur le site Web, il y aura un ensemble de boutons, sur lesquels, lorsque nous cliquons, il enverra une commande à ESP32-CAM et ce dernier la fera correspondre à l'action spécifique correspondant au mouvement.

SITE WEB CÔTE CLIENT ( Page de contrôle )

ESP32-CAM

MOTEUR

HEROKU CÔTE SERVEUR

**Etape 2 : Construction du robot**

Pour la construction de la voiture, nous pouvons nous servirons d’un kit complet prêt à l’emploi sans avoir besoin de nous soucier du châssis ou de tout autre pièce difficile à se procurer.

Pour le montage nous pouvons trouver des vidéos telles que :   
<https://www.youtube.com/watch?v=uW8YVcBjPGU>

<https://www.youtube.com/watch?v=kewza7RyKMQ>

**Etape 3 : Création du contrôleur (côté frontend)**

Le Websocket sera divisé en trois parties, à savoir le côté frontend, le côté backend et le côté ESP32-CAM.

Dans cette partie, nous allons créer notre contrôleur de site Web en utilisant uniquement du HTML, CSS et JavaScript pour le déployer sur le serveur afin que nous puissions y accéder partout. Le côté frontend se compose de :

1. **index.html**

2. **style.css** utilisé pour styliser le web

3. **index.js** pour notre DOM et gérer une connexion WebSocket à un serveur, ainsi que pour l'envoi et la réception des données.

**CODE index.html**

**Résultat**

**Etape 4 : Création du serveur (côté back end)**

C’est ici que nous connecterons notre côté frontend (contrôleur) à l'ESP32-CAM, en gérant le serveur WebSocket pour maintenir une connexion avec tous les clients qui se connectent, lorsqu'il recevra des données, il diffusera la commande au client existant actuellement connecté au serveur.

Nous réaliserons le backend en utilisant Node.js et avec sa bibliothèque qui est **express** et **ws**. Une fois l'installation de node.js terminée, nous pouvons installer la bibliothèque nécessaire via le terminal dans votre dossier du projet

npm install ws

npm install express

**ws** est la bibliothèque dont nous avons besoin pour la partie WebSocket et **express** est utilisé pour créer le serveur

**Etape 5 : Inscription Heroku et déploiement**

Une fois que nous avons fini de construire le frontend et le backend, nous pouvons maintenant les déployer sur le serveur. Le dossier que nous allons déployer sur Heroku est le dossier **website .**