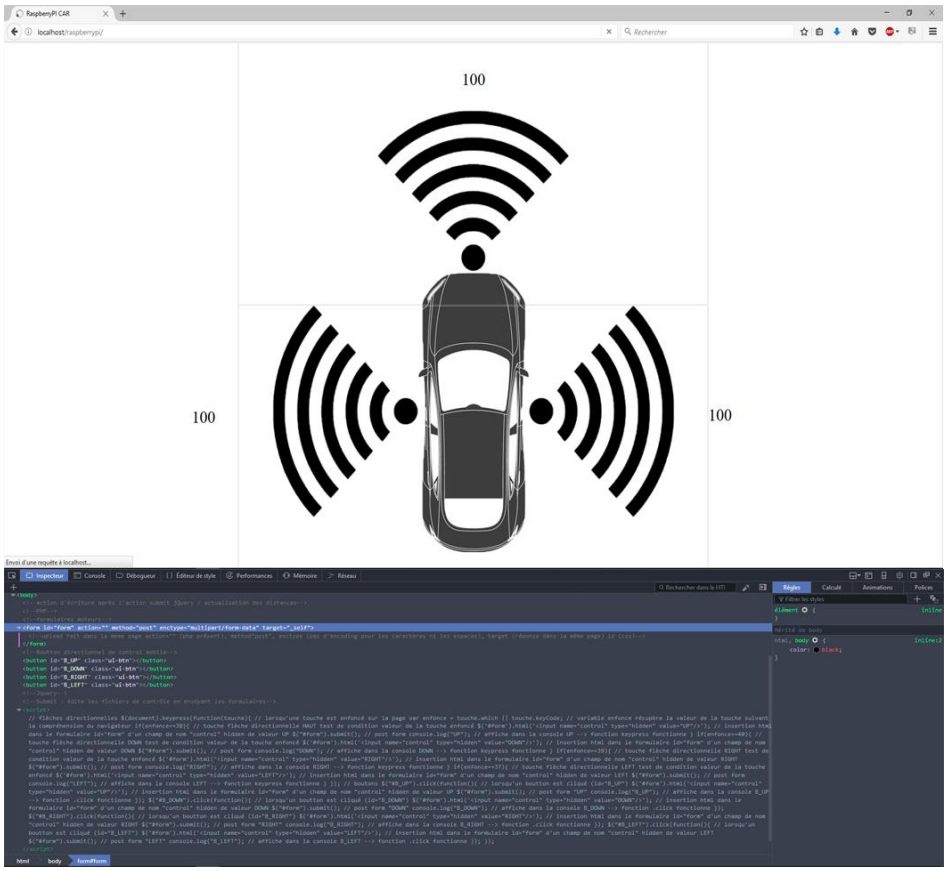
Raspberry-Pi Car



BOUFFETIER Thibault **CLAVIJO Matthis**

CLAVIJO Raphaël

1) Présentation :

Le **Raspberry Pi Car** est un projet de voiture à partir d'un Raspberry Pi. Un Raspberry Pi est un micro-ordinateur de la taille d'une paume de main. Ce dernier propose différentes entrées et sorties. Il est alimenté de la même manière qu'un smartphone par un port femelle micro-USB type B. Ce qui permet par exemple de le raccorder à une batterie portable. Pour afficher la sortie vidéo il existe 2 moyens. On peut passer par un câble HDMI. Autrement on peut passer par le sans fil en ajoutant une clé usb wifi et en utilisant VNC viewer sur n'importe quelle plateforme (PC, tablette, téléphone). Le Raspberry Pi que nous utilisons a pour système d'exploitation Raspian. Ce dernier fournit des outils informatiques comme une boîte de commande linux, un éditeur python, et des éditeurs d'autres langages de programmation. Le Raspberry Pi nous permet aussi d'héberger un serveur et les différents codes permettant le bon fonctionnement de la voiture. Ce projet contient deux modules. Le module d'une voiture autonome, la voiture est capable de rouler seule et de faire des choix pour éviter des obstacles fixes. Le Raspberry Pi contient également le module de la voiture manuelle. Un utilisateur à distance peut se connecter au serveur hébergé par le Raspberry Pi et contrôler la voiture à distance. L'utilisateur connaît en temps réel les distances des obstacles autour de la voiture.

2) Analyse du besoin :

L'idée de départ était d'utiliser d'une manière originale le langage python et de ce fait faire un projet créatif. L'idée qui a rapidement émergée été celle d'une voiture semi-autonome contrôlable à partir d'un site web. Ce qui nous a poussé à choisir un projet alliant informatique et électronique est dû à notre attrait pour la technologie en général. Les voitures autonomes se développent de jours en jours avec de grands constructeurs comme Tesla, BMW, Mercedes. Le but intrinsèque de ce projet était donc de comprendre les bases du fonctionnement de tout code autonome.
Pour ce faire, nous avions prévu d'utiliser un raspberry pi pour héberger nos code python et qu'il puisse les lancer. Des projets de voiture autonome ont déjà été réalisés mais nous voulions créer notre propre projet et nous avons donc décidé de faire une voiture autonome et une voiture manuelle. Pour l'autonomie de la voiture, nous utilisons des émetteurs et des récepteurs à ultrasons. Cela nous permet de détecter des obstacles. Pour alimenter les moteurs des roues pour déplacer la voiture, nous utilisons l'énergie de piles AA. Grâce à une batterie portable à forte capacité, nous alimentons en énergie le raspberry pi.
De plus, nous voulions utiliser les connaissances que nous avions déjà acquises en python mais également apprendre de nouveaux langages et les faires interagir entre eux. Ainsi, nous avions l'ambition de mêler plusieurs langages de programmation et les relier pour produire un projet cohérent alliant plusieurs modules divers autant en fonctionnalités qu'en méthodes de fonctionnement.

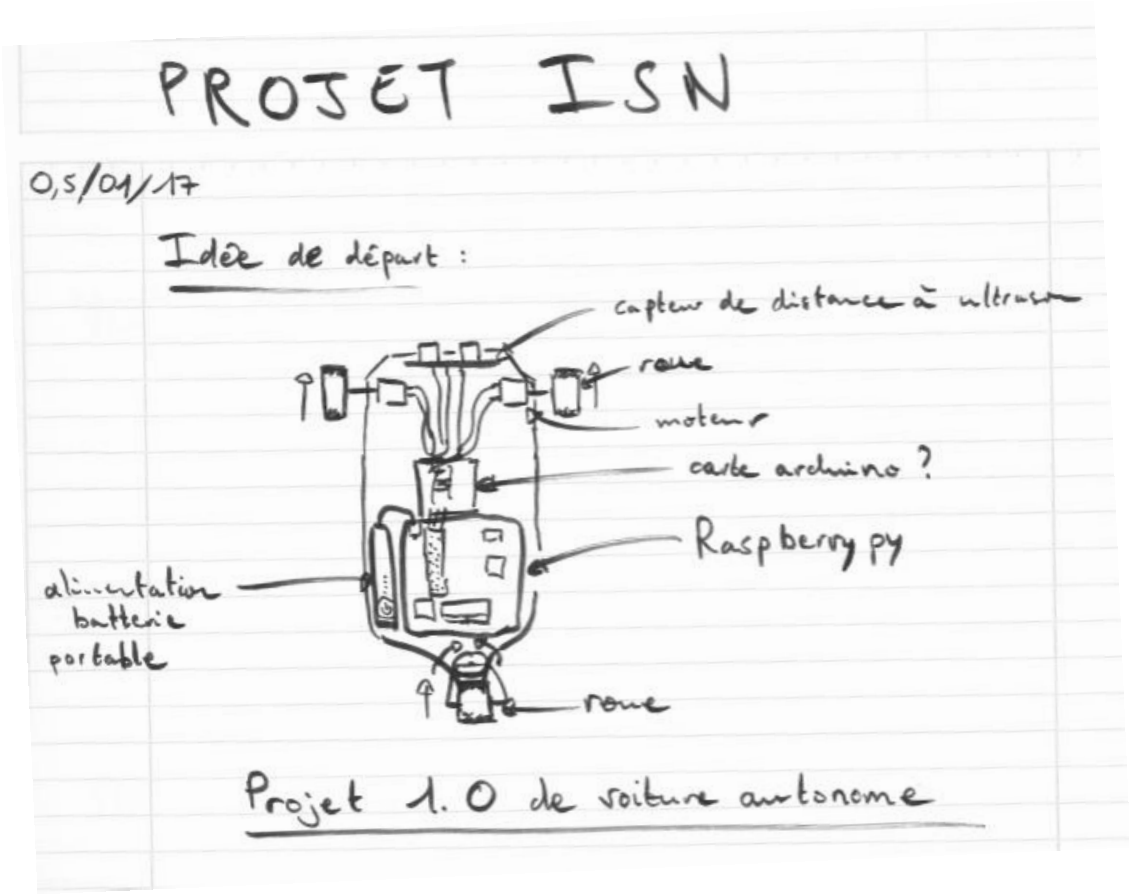
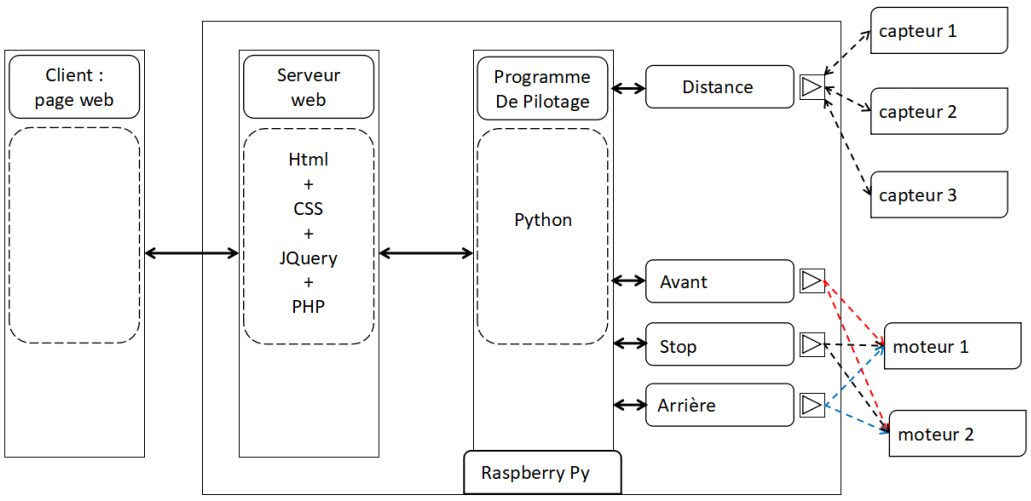


Schéma des modules et des langages utilisés pour réaliser le projet :

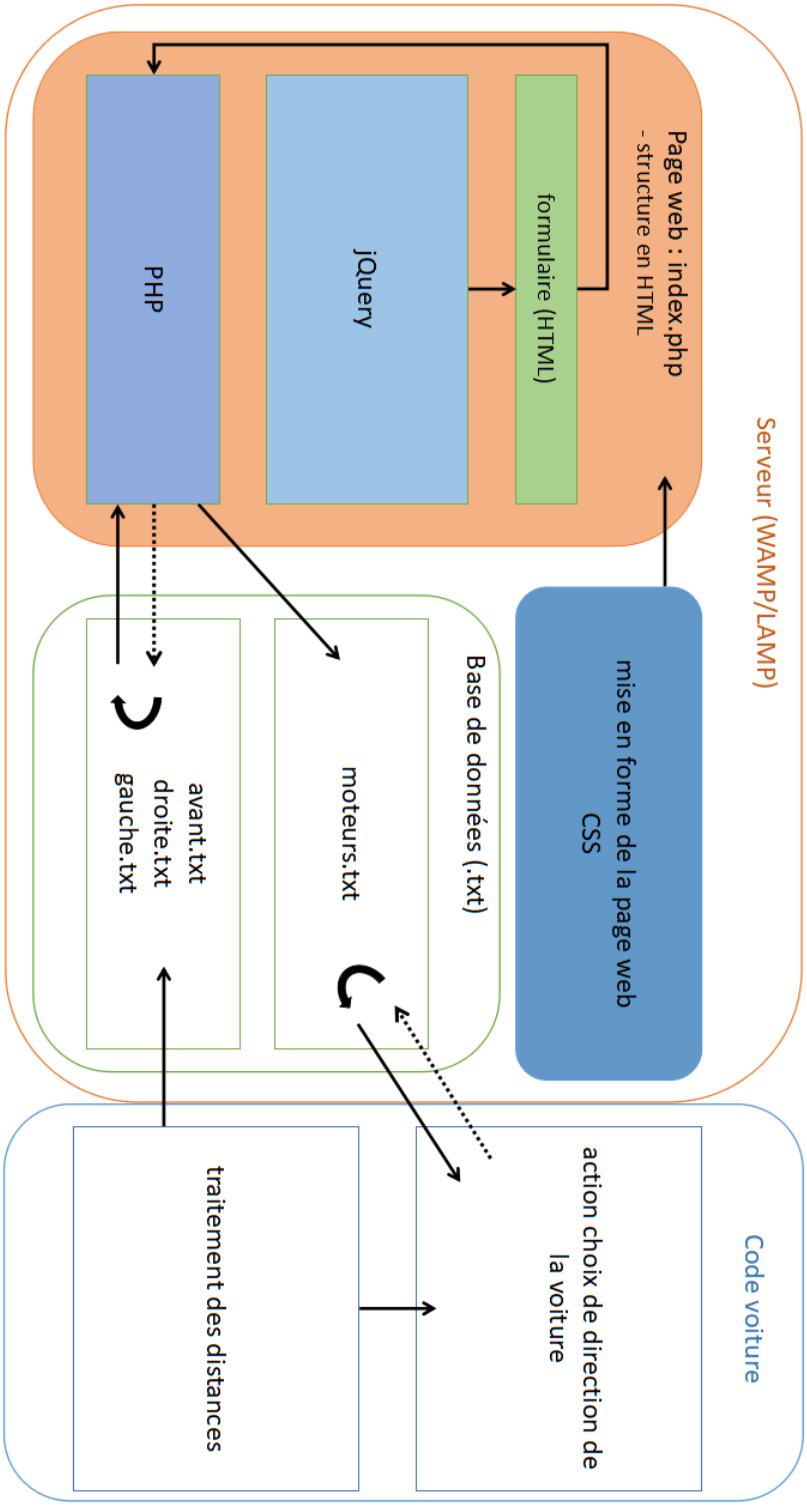


3) Répartition des tâches et démarche collaborative:

Hardware	Hardware-Software	Software
Thibault	Thibault	
		Raphaël
	Matthis	Matthis

Grâce à la plateforme Github, nous avons pu créer notre projet ISN avec tous les membres de notre groupe. Nous pouvions poster et mettre à jour nos codes sur la plateforme Github ce qui permettait à tous les membres de consulter les codes et l'avancement du projet. Nous avons donc pu travailler en équipe pendant nos leçons d'ISN, réfléchir sur l'adaptation du projet, informer les différents membres de l'équipe sur l'avancement des différents modules, en nous organisant et en se réunissant chez un membre de l'équipe. Nous avons également organisé des vidéos conférences pour travailler ensemble sur notre projet et nous concerter sur l'avancement de celui-ci.

4) Réalisation Personnelle



Ma contribution au projet étant l'interface utilisateur et la communication entre l'utilisateur et la machine, je me suis mis à me documenter sur la conception d'une page internet dynamique et la gestion du serveur (apache). Ayant des connaissances en HTML et CSS, j'étais prédisposé dans le groupe pour la réalisation de la page web.

Nous évoquerons trois types de langages différents, le HTML, le PHP, et le jQuery permettant la réalisation d'une page web dynamique :

- Le HTML est un langage structurel
- Le PHP est un langage de requêtes côté serveur
- Le jQuery est un langage d'événements côté client

Le HTML (« HyperText Mark-Up Language ») est un langage qui définit la structure d'une page web et ses différents modules ainsi que les liens que la page a avec les autres pages du site web. La structure est donnée grâce à différentes balises définissant les modules présents dans la page web.
Bien que le HTML permette de réaliser une page statique avec la structure complexe de la page permettant l'incorporation d'autres langages de programmation web, il ne permet pas d'exécuter des actions. En effet, avec une page html, lorsque la requête pour y accéder a été formulée (url), le serveur la délivre au client et la connexion client/serveur est définitivement coupée puisque la page *.html ne permet aucune action côté serveur.
Cependant, il permet la mise en place de formulaires permettant le transfert de données entrées par le client grâce au langage PHP. Ainsi, le HTML met en place la structure du transfert de données sans pour autant pouvoir l'exécuter.

(Je vous conseille de vous référer au schéma ci-dessus pour la suite de l'explication)

Le PHP est un langage côté serveur, c'est-à-dire qu'il réalise la liaison entre le client et le serveur. Il effectue des requêtes permettant d'afficher des données stockées sur le serveur et qui évoluent dans le temps. Il permet également d'envoyer des données pour qu'elles soient stockées sur le serveur. Il permet de rétablir la liaison entre la page web et le serveur qui était perdue avec les pages *.html et qui sont possibles avec les pages *.php.

Le jQuery est un langage côté client, à l'inverse du PHP. Il permet de réaliser des actions sur la page web suivant des événements réalisés par le client. Ainsi, il n'a pas de communication directe avec le serveur et fonctionne indépendamment de celui-ci. Cependant, le jQuery peut être l'événement déclencheur d'un lien entre le client et le serveur en déclenchant une requête PHP. Ainsi, indirectement il permet la liaison entre le client et le serveur à la suite d'une action effectuée par le client.

Pour le projet, ces trois langages étaient nécessaires. En effet, nous avions besoin de transférer des données au serveur en temps réel pour dicter les actions à réaliser à la voiture ainsi que de récupérer en temps réel les distances à l'obstacle captées par les capteurs ultrasonores (PHP). Ce transfert devant s'effectuer suivant la volonté de l'utilisateur de diriger ou non la voiture (événement jQuery).
Bien évidemment, je ne savais pas encore au début du projet que j'aurais recours à ces langages en particulier mais j'avais déjà fractionné les différents modules de mon code ainsi qu'écrit le pseudo-code (python) des événements utilisateurs.

Lorsque j'ai commencé mon projet personnel je me suis documenté sur les différents langages de développement web et leurs caractéristiques pour choisir les plus faciles à utiliser et assimiler rapidement. J'ai utilisé Sublime Text 3 pour le développement car il propose un environnement de développement web complet que ce soit pour le HTML, le PHP, ou le jQuery/Javascript.

J'ai tout d'abord créé un formulaire HTML qui sera édité par le langage client avant d'être envoyé. Il sert donc de support d'édition de données dans le code et d'intermédiaire pour envoyer les données sur le serveur.

```
<!-- formulaires moteurs -->
    <form action="" method="post" enctype="multipart/form-data" target="_self" id="form" >
    </form>
```

Je me suis ensuite intéressé à l'événement utilisateur qui permettrait de transmettre l'action à la voiture en passant par le formulaire. Pour cela, je savais que je devais m'orienter vers un langage comme le JavaScript, langage phare du développement web côté client. J'ai découvert le jQuery, langage côté client plus performant et plus simple que le JavaScript car raccourcissant la longueur du programme.
L'objectif fixé était de contrôler la voiture à partir des flèches directionnelles du clavier. Pour cela j'ai utilisé la fonction .keypress() qui déclenche les fonctions à l'intérieur de celle-ci, suite à l'appui d'une touche du clavier.

```
 $(document).keypress(function(touche){ // lorsqu'une touche est enfoncé sur la page
```

Ainsi, à la suite d'une touche enfoncée je récupère son identifiant avec .which ou .keyCode et le compare, dans des tests conditionnels, avec l'aide du tableau suivant, pour exécuter la commande voulue:

KEY_DOWN = 40;
KEY_UP = 38;
KEY_LEFT = 37;
KEY_RIGHT = 39;

Pour finir, j'édite le formulaire avec .html() en rajoutant une zone cachée dans le formulaire puis en envoyant le formulaire avec la fonction .submit().

```
$('#form').html('<input name="control" type="hidden" value="UP"/>');
$("#form").submit(); // post form
```

Nous avons maintenant les événements utilisateurs et le formulaire édité suivant ceux-ci. Cependant, le formulaire ne peut être envoyé que si la page contient un langage côté serveur traitant les différentes zones du formulaire. Ce langage permet d'éditer ou d'afficher sur la page web des fichiers présents sur le serveur dont les données sont amenées à évoluer dans le temps et de manière automatique.
La première question avant d'utiliser le PHP est de savoir ce qu'on veut en faire et surtout où l'on décide de stocker l'information, base de données sql ou simplement dans des fichiers. Avec Matthis, on s'est demandé comment le langage python pouvait récupérer des données externes. On savait que python permettait d'ouvrir des fichiers texte (.txt) pour y écrire ou y lire des informations. Ainsi, on s'est orienté vers une base de données serveur composée de fichiers texte. Par la suite je me suis documenté sur l'écriture et la lecture de fichiers en PHP. Il se trouve que la technique est très similaire à celle utilisée par python.

```php
$fp = fopen("C:\\wamp64\\www\\RaspberryPI\\moteur\\moteurs.txt","w");
$control = $_POST["control"]; /* variable $control correspond au valu
fwrite($fp,$control); /* écriture dans moteurs.txt de la variable $co
fclose($fp); /* fermeture du fichier moteurs.txt */
```

Ouverture, écriture, et fermeture d'un fichier .txt en PHP

```python
fichier=open("C:\\wamp64\\www\\RaspberryPI\\capteurs\\gauche.txt","w")
fichier.write(n[a])
fichier.close()
```

Ouverture, écriture, et fermeture d'un fichier .txt en Python

Le PHP permet ainsi de récupérer les valeurs des champs du formulaire pour ensuite les inscrire dans un fichier texte. Il permet également de récupérer les données des distances mesurées dans différents fichiers pour ensuite les publier sur la page web dans des blocs HTML.

Pour récapituler, il a fallu créer un formulaire HTML vierge prêt à recevoir des champs suivant les actions de l'utilisateur sur les flèches directionnelles du clavier en jQuery pour ensuite traiter le formulaire grâce au PHP.

Les difficultés rencontrées :

L'apprentissage du jQuery est la réelle complexité du langage. Ce n'est pas tant la manière de l'écrire qui est spéciale mais la conception que l'on doit avoir lorsqu'on l'apprend. En effet, le langage est centré sur les événements utilisateur et sur l'action qu'il a sur la page web, tant dans son contenu que dans son apparence, ce qui complexifie l'apprentissage du langage. De plus, les actions se font à travers des fonctions déjà existantes dans la majorité des cas. Le jQuery demande donc

beaucoup de recherche internet pour trouver les fonctions que l'on souhaite utiliser. Ainsi, il faut savoir précisément comment on veut réaliser l'action et réadapter la manière de faire suivant les fonctions que l'on utilise, ce qui n'est pas évident lorsqu'on ne connait pas la méthode de fonctionnement d'un nouveau langage et les fonctions qu'il propose.

La deuxième difficulté est survenue lors de l'insertion du PHP dans la page web. En effet, le PHP étant un langage de traitement et de requêtes serveur, il est courant d'avoir un script PHP dans une page à part et non pas dans la même page sachant que le formulaire HTML y fait référence. Cependant, lorsque le script PHP s'exécute, on perd de vue la page sur laquelle on était pour arriver sur une nouvelle totalement blanche puisque le script PHP est seulement du traitement. Cela devient vite insupportable lorsque les requêtes sont très fréquentes. Une technique consiste à inscrire son script PHP dans sa page internet. Survient assez vite une erreur de variable non définie avant toute action utilisateur car la page exécute tout script PHP dès qu'il est rencontré lors du chargement de la page. Pour remédier à cette erreur, il suffit de vérifier que la variable est définie avant l'exécution du script grâce à la fonction isset() qui renvoie un booléen (True ou False).

Pour finir, on veut que la voiture s'arrête lorsque l'utilisateur n'appuie plus sur les touches. Cependant, l'écriture ne s'effectue que lorsque l'utilisateur appuie sur les flèches directionnelles, ce qui veut dire que le fichier garde en mémoire la valeur précédente avant que l'utilisateur ne lui envoie un contre-ordre.
Ainsi, pour arrêter la voiture, une manière a été d'envoyer à intervalles de temps régulier une nouvelle valeur dans le fichier agissant comme un contre-ordre ordonnant à la voiture de s'arrêter.
Une autre technique a été d'écrire un contre-ordre dans le fichier texte pour chaque ordre exécuté par la voiture au sein de son code mais suffisamment tôt dans le code pour permettre à une nouvelle requête de se superposer indiquant à la voiture de continuer sa route.

Une dernière difficulté fut la mise en page CSS de la page web suivant la compatibilité des différents navigateurs web et comment ils interprétaient les indications CSS.

BONUS :
La page web était exclusivement destinée aux utilisateurs ayant un ordinateur. En effet, les actions sont transmises suivant les actions sur les flèches directionnelles. Cela rend le contrôle de la voiture impossible sans un ordinateur. Ainsi, j'ai rajouté des boutons (zones de contrôles presque transparentes), permettant à tout utilisateur possédant un smartphone ou une tablette de communiquer avec la voiture en cliquant sur les zones. Le fonctionnement est le même que les flèches directionnelles, à la seule différence que la fonction utilisée n'est pas .keypress() mais .click().

```
$("#B_UP").click(function(){ // lorsqu'un boutton est cliqué (id="B_UP")
```

Page web finale

5) Intégration et Validation

Mon projet s'insère parfaitement dans le projet global car il fait la liaison entre l'utilisateur et la voiture et permet de proposer une interface graphique à l'utilisateur pour le contrôle de sa voiture. Ma partie s'est faite avec une forte communication et en accord avec l'équipe pour être sûr que les deux modules (voiture et serveur) communiquent parfaitement et proposent une solution pour répondre au problème de la communication entre l'utilisateur et la voiture.
En ce qui concerne le serveur, nous avons utilisé un serveur apache interprétant le PHP.

Les tests furent d'abord fait sur un serveur local (WAMP pour Windows) avant d'installer un serveur apache2 pour Linux sur le Raspberry Pi.

6) Bilan et Perspective

Notre projet, dans le cas où il serait commercialisé, permettrait à l'entreprise de proposer des voitures complètement autonomes ou des voitures complètement manuelles pilotables à partir d'une interface web et d'une connexion internet.
L'améliorer serait de retravailler la réactivité de la voiture à travers des protocoles et des transferts de données directs ainsi qu'un langage plus performant comme le C par exemple.

Le travail en groupe m'a permis de mettre mes compétences techniques au service du groupe ainsi que la possibilité de produire un projet complexe plus efficacement que seul. Le travail de groupe a permis de réfléchir ensemble sur le projet que l'on désirait mener tous ensemble et sur les techniques pour le mettre en place suivant les compétences et les intérêts de chacun. Ce travail a permis de renforcer l'organisation

et la confiance que l'on avait dans les autres membres de l'équipe car sans le travail et la coopération de tous, le projet n'aurait pu se faire. En effet, les différents modules étaient interdépendants. Sans l'électronique, on n'aurait pas eu de voiture; sans le code principal, elle n'aurait pas été fonctionnelle; sans la page web, il n'y aurait pas eu d'interactions avec l'utilisateur.
De pus, le travail fourni pourrait évoluer dans le temps et suivant les différents modules produits. Le travail n'est pas réalisé pour soi mais pour le groupe et pour les différents autres modules. Ainsi, une confiance et une bonne entente entre les membres du groupe est nécessaire pour mener à bien le projet, d'autant plus que les modules ont des conséquences les uns sur les autres et qu'une modification de l'un entraîne la modification des autres d'où la très importante communication entre les membres du groupe.

Pour finir, tout en étant un travail de groupe, le projet m'a conduit à effectuer de nombreuses recherches personnelles pour mener à bien le rôle que j'ai joué dans ce projet.

Annexes

Code autonome:
https://github.com/ecolelasource92/Raspberry-Pi_Car/blob/master/MODULES/RADAR_MOTOR%20main.py

Code manuel:
https://github.com/ecolelasource92/Raspberry-Pi_Car/blob/master/MODULES/MANUal.py
Et
https://github.com/ecolelasource92/Raspberry-Pi_Car/blob/master/MODULES/MANUal2.py

Code serveur:
https://github.com/ecolelasource92/Raspberry-Pi_Car/blob/master/SERVEUR/index.php
Et
https://github.com/ecolelasource92/Raspberry-Pi_Car/blob/master/SERVEUR/Principal/index.php

Bases de données:
https://github.com/ecolelasource92/Raspberry-Pi_Car/tree/master/SERVEUR/capteurs
Et
https://github.com/ecolelasource92/Raspberry-Pi_Car/tree/master/SERVEUR/moteur