

Find your ride

CFPT en informatique
Technicien ES en informatique
Travail de diplôme

Classe : T.IS-E2A

Session : 2016-2017

Elève :
Lucien Camuglia

Enseignant :
M. Zeltner

23 mai 2017

Table des matières

1	Introduction	3
1.1	Résumé	3
1.2	Abstract	3
2	Cahier des charges	4
3	Analyse de l'existant	5
3.1	Garmin BaseCamp	5
3.2	http://www.calculitineraires.fr/	5
3.3	http://www.bestbikingroads.com	5
4	Analyse fonctionnelle	6
4.1	Généralités	6
4.2	Description des fonctionnalités globales	7
4.2.1	Connexion	7
4.2.2	Inscription	7
4.2.3	Création de trajet	7
4.2.4	Exportation	7
4.2.5	Suppression d'un trajet	7
4.2.6	Visualiser le trajet	7
4.3	Description de l'interface	8
4.3.1	Inscription	8
4.3.2	Map	8
4.3.3	Detail trajet	9
4.3.4	Connexion	9
4.3.5	Vos trajets	10
4.4	Description des éléments de sécurité	10
4.4.1	Fichier .htaccess	10
4.4.2	Utilisateur de la base de données	10
4.4.3	Requêtes	10
5	Analyse organique	11
5.1	Résumé	11
5.2	Google API	11
5.2.1	Google Maps Javascript	11

5.2.2	Google Direction	11
5.2.3	Google Elevation	11
5.2.4	Google Roads	11
5.3	Fonction PHP pour la base de données	12
5.3.1	connexion à la base de données	12
5.3.2	Requête préparée	12
5.4	Inscription sur le site	14
5.5	Connexion au site	15
6	Problèmes rencontré	17
6.1	Google Api	17
6.2	Async	17
7	Conclusion	17

1 Introduction

1.1 Résumé

Site WEB permettant le partage d'itinéraire entre passionné de la moto. Ce site est réalisé en HTML/PHP/CSS/AJAX et incluant les API¹ Google.

Il permet à un utilisateur de créer un trajet, de le partager et de le modifier.

Un utilisateur peut rechercher un itinéraire avec différents critères :

- Durée
- Type de route
- Changement d'altitude

Le site fournit à l'utilisateur différentes données comme le temps du trajet ou la consommation théorique de la moto pour la balade.

Le motard peut aussi importer et exporter des fichiers directement depuis son GPS.

1.2 Abstract

The website allows sharing itineraries between motorcycle enthusiasts. The site is set-up with HTML/PHP/CSS/AJAX and Google's API. It allows users to create trips, to share and modify them.

Different criteria can be defined by the user to create the itinerary :

- Travel Time
- Type of road
- Change in altitude, etc.

This website gives the user diverse information. One example could be the theoretical consumption of their motorcycle for the ride or travel time. The rider can also upload or download files directly from his GPS.

1. *Application Programming Interface* (Interface de Programation Applicative) https://fr.wikipedia.org/wiki/Interface_de_programmation

2 Cahier des charges

TODO inclure le CDC

3 Analyse de l'existant

3.1 Garmin BaseCamp

Garmin BaseCamp est un logiciel fournit par Garmin.

Il permet à un utilisateur de créer un trajet et de l'importer sur son GPS.

Il donne aussi la possibilités a l'utilisateur de visionnée ses différent déplacement.

Positif	Négatif
<ul style="list-style-type: none">• Connexion directe avec le GPS	<ul style="list-style-type: none">• Pas de partage• Obligation de posseder un GPS Garmin

3.2 <http://www.calculitineraires.fr/>

www.calculitineraires.fr est un site de partage d'itinéraire pour la course à pied, le vélo et la randonnée

Il permet l'import/export de fichier GPX et TCX, la rechercher et le partage d'itinéraire.

Positif	Négatif
<ul style="list-style-type: none">• Assez complet• Import / Export des fichiers GPS• Création d'itinéraire	<ul style="list-style-type: none">• Interface compliquée d'utilisation• Pas de contribution pour la moto

3.3 <http://www.bestbikingroads.com>

www.bestbikingroads.com est un site de partage d'itinéraire moto.

Il permet l'import/export de fichier GPX, la rechercher, le partage d'itinéraire ainsi que la notation des balades.

Positif	Négatif
<ul style="list-style-type: none">• Assez complet• Import / Export des fichiers GPS• Création d'itinéraire• Notation des balades	<ul style="list-style-type: none">• Tout les tracés s'affiche en meme temps sur la carte• pas de modification possible

4 Analyse fonctionnelle

4.1 Généralités

Ci-dessous se trouve le schéma initial de mon site web. Les ronds représentent des pages et les flèches entre ceux-ci représentent d'éventuelles actions ou états.

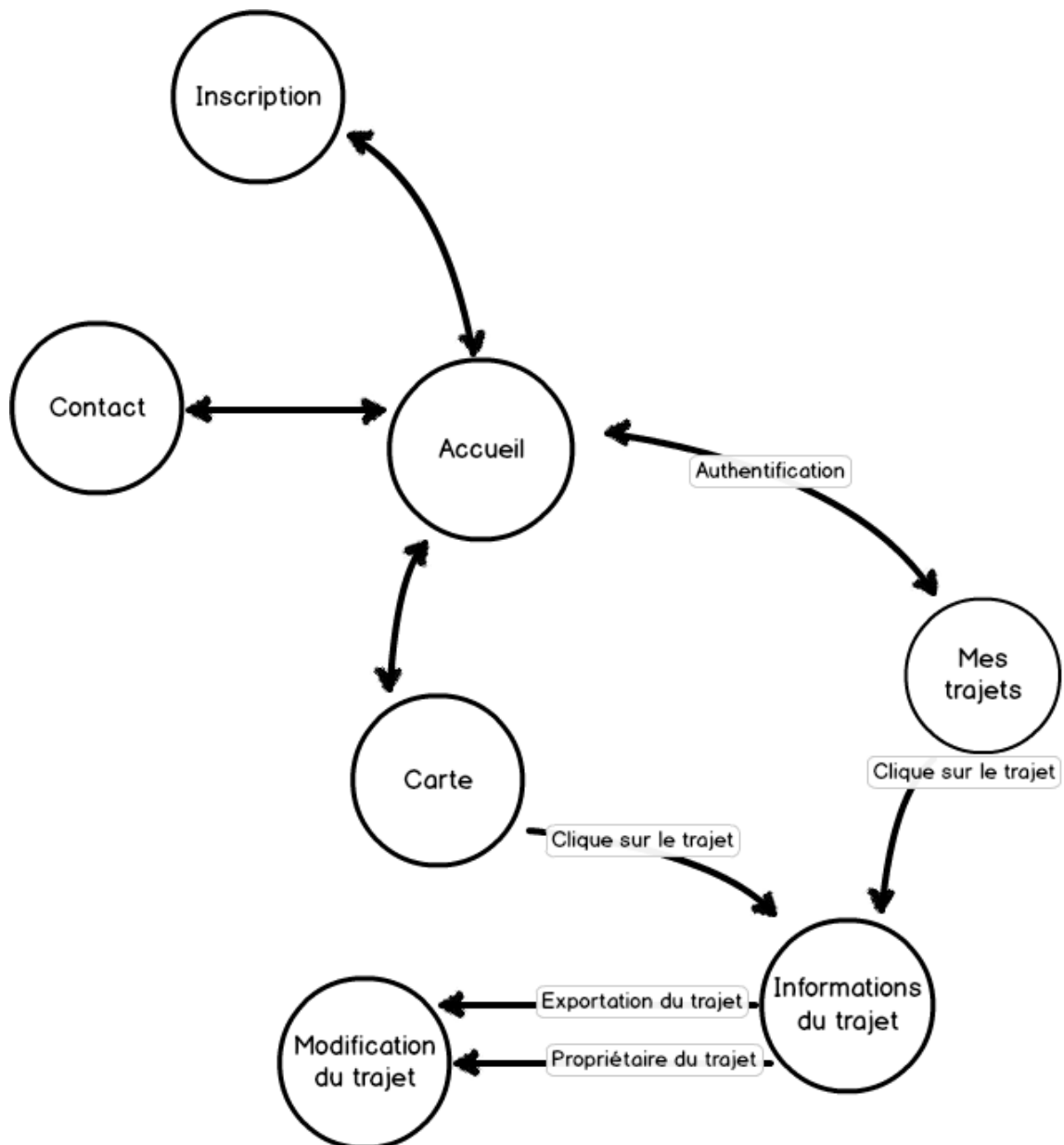


FIGURE 1 – Schéma du site

4.2 Description des fonctionnalités globales

4.2.1 Connexion

Cette fonctionnalité permet à un utilisateur de s'authentifier et d'accéder à ses trajets mit en ligne ou partager de nouveaux trajets.

4.2.2 Inscription

Cette fonctionnalité permet à un nouvel utilisateur de créer un compte et donc de pouvoir bénéficier des fonctionnalités d'un utilisateur connecté

4.2.3 Création de trajet

Cette fonctionnalité se distingue en deux sous fonctionnalités :

- Création : crée un nouveau trajet depuis le site directement.
- Importation : importe un fichier de type GPX et éventuellement modifier le trajet.

4.2.4 Exportation

Cette fonctionnalité permet à un utilisateur d'exporter le trajet de son choix au format GPX pour l'inclure dans son GPS.

4.2.5 Suppression d'un trajet

Cette fonction permet a un utilisateur de supprimer un de ses trajet.

4.2.6 Visualiser le trajet

Cette fonctionnalité permet à n'importe qui de visualiser les trajets mit en ligne, puis les exporter en les modifiant s'il le souhaite.

4.3 Description de l'interface

4.3.1 Inscription

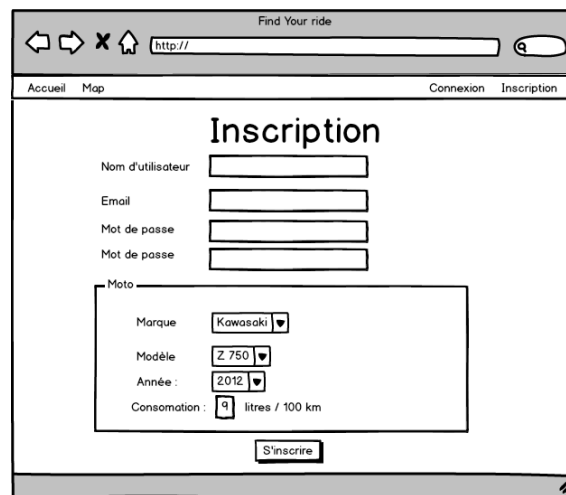


FIGURE 2 – Page inscription

Cette page sert à l'inscription des utilisateurs. On y trouve différents champs :

- Nom d'utilisateur
- E-mail
- Mot de passe, ce champ apparaît deux fois pour avoir la validation de celui-ci
- Moto : diverses informations sur la moto de l'utilisateur comme par exemple sa consommation pour calculer la consommation des trajets. Le champ *Moto* est facultatif mais conseillé.

4.3.2 Map

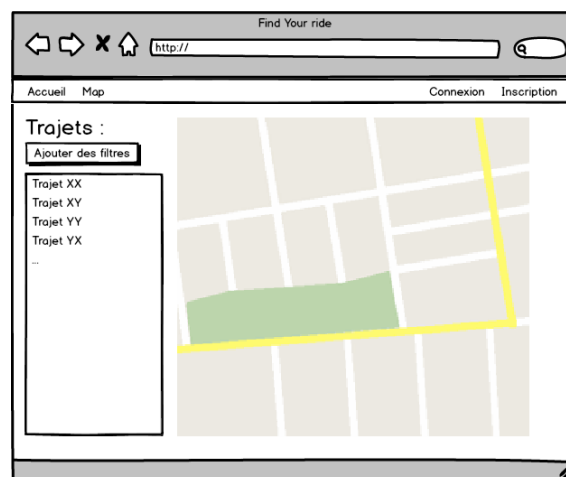


FIGURE 3 – Page inscription

Cette page sert à l'affichage de la carte et des différents trajets.

Sur la gauche apparaissent tous les trajets ainsi qu'un bouton filtre. Ce bouton permet de filtrer les trajets parmi différents critères.

- Avec ou sans autoroute

- Durée
- Dénivelé

Sur la droite une carte Google ou apparaît le tracé sélectionné.

4.3.3 Detail trajet

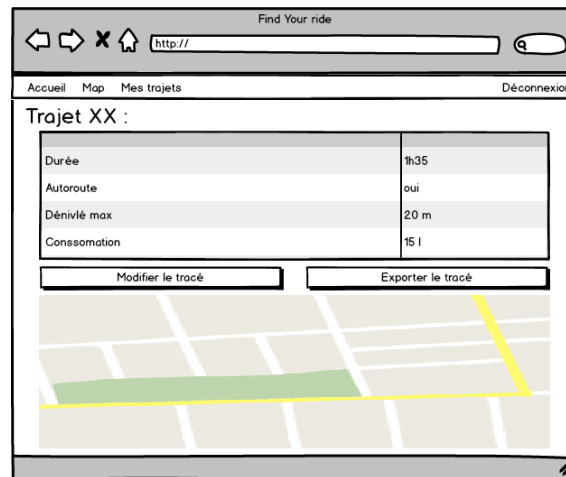


FIGURE 4 – Page Detail Trajet

Cette page sert à afficher le détail d'un trajet.

Sur le haut de la page apparai les détails du trajet.

- Durée du trajet
- S'il contient des autoroutes
- Dénivelé
- La consommation (si l'utilisateur à renseignée les données de la moto)

Deux bouton sont présent pour modifier ou exporter le trajet. Sur le bas de la page, la carte google avec le trajet.

4.3.4 Connexion

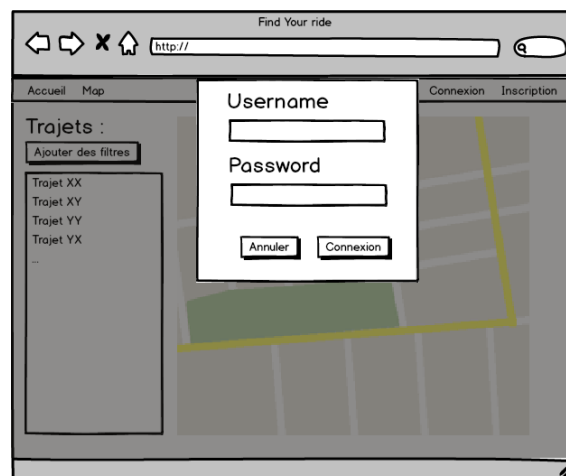


FIGURE 5 – Modal connexion

Le formulaire de connexion est une fenêtre modal qui s'ouvre par dessus les autres pages avec uniquement deux champs.

- Nom d'utilisateur
- Mot de passe

4.3.5 Vos trajets

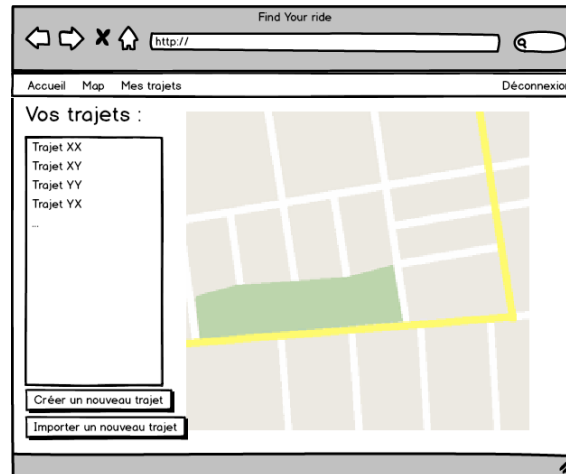


FIGURE 6 – Page vos trajet

Cette page est identique que la page *Map* mais au lieu d'avoir tout les trajet disponible du site, il y a uniquement les trajet de l'utilisateur.

Deux boutons sont disponible sur cette page.

- Créer un nouveau trajet, permet de créer un trajet a partir de la carte
- Importer un nouveau trajet, permet d'importer un fichier GPX avec le trajet.

4.4 Description des éléments de sécurité

4.4.1 Fichier .htaccess

Permet d'empêcher la navigation sur certain dossier/fichiers du site. Ils permettent aussi de définir des pages d'erreurs personnalisées

4.4.2 Utilisateur de la base de données

Utiliser un utilisateur différent que *root* pour accéder à la base de donnée afin de donner des droits qu'au éléments nécessaire.

4.4.3 Requêtes

Utilisation de requêtes préparée pour éviter les injections SQL.

5 Analyse organique

5.1 Résumé

Dans cette section sera expliquer le fonctionnement de l'application.

5.2 Google API

Les API² Google sont une suite d'outils développé et fourni par Google pour permettre à des utilisateurs d'intégrer les services Google dans un site web ou une application.

Il en existe environ 66 dont 16 pour *Google Maps*. Pour la réalisation de ce site j'utilise 4 API *Google Maps* :

- Google Maps Javascript
- Google Direction
- Google Elevation
- Google Roads

5.2.1 Google Maps Javascript

Cette API permet d'intégrer une *Google Map* sur le site web, c'est la seule qui sera visible pour l'utilisateur. C'est sur celle-ci qu'apparaissent les itinéraires et les positions GPS.

5.2.2 Google Direction

Le fonctionnement de cette API est assez simple. Il suffit d'envoyer un point d'origine et une destination à Google Direction et il nous retourne un tableau JSON avec différent paramètres.

- La durée
- La distance
- Les étapes du parcours (positions GPS)
- Les étapes du parcours (Francais, par exemple : Tournez a droite,...)

5.2.3 Google Elevation

TODO

5.2.4 Google Roads

Cette API permet de faire 3 choses différentes :

- *Snap to road*
- *Nearest road*
- *Speed limits*

J'utilise deux fonctionnalités de cette API, *Snap to road* et *Nearest road*.

Snap to road permet de "déplacer" un trajet pour qu'il suive la route. Il faut lui envoyer l'ensemble des point GPS de notre route et il nous retourne un ensemble de point qui suivent la route.

Nearest road permet de donner la position d'un point GPS sur la route la plus proche.

². *Application Programming Interface* (Interface de Programation Applicative) https://fr.wikipedia.org/wiki/Interface_de_programmation

5.3 Fonction PHP pour la base de données

5.3.1 connexion à la base de données

La connexion à la base de données se fait à l'aide de PDO³. PDO a besoin de l'utilisateur de la base de données et du mot de passe ainsi que le nom de la base. Je lui précise aussi le mode d'erreur qui est `PDO::ERRMODE_EXCEPTION` ce mode permet d'afficher le code d'erreur et de déclencher une exception⁴.

Afin d'avoir un partage d'informations dans le bon format, on définit l'encodage de caractères en UTF-8

```
1 function connexionDb() {
2
3     try {
4         //variables contenant les informations de connexion ainsi
           que la DB
5         $serveur = '127.0.0.1';
6         $pseudo = 'root';
7         $pwd = '';
8         $db = 'findyourride';
9
10        static $pdo = null;
11
12        if ($pdo === NULL) {
13            // Connexion à la base.
14            $pdo_options[PDO::ATTR_ERRMODE] = PDO::
                ERRMODE_EXCEPTION;
15            $pdo = new PDO("mysql:host=$serveur;dbname=$db", $
                pseudo, $pwd, $pdo_options);
16            $pdo->exec("Set Character set UTF8");
17        }
18        return $pdo;
19    } catch (Exception $exc) {
20        echo $exc->getTraceAsString();
21    }
22 }
```

5.3.2 Requête préparée

Afin de sécuriser le site et éviter les injections SQL j'utilise des requête préparée.

On commence par préparer la requête, PDO va substituer les marqueurs (*:marqueur*) pour les valeurs fournies dans le tableau de paramètres fourni au moment de l'exécution. Ensuite, il faut exécuter la requête avec les paramètres.

```
1 Function PrepareExecute($query, $params = NULL) {
2     global $pdo;
3     // Préparation de la requête SQL.
4     $st = $pdo->prepare($query);
5     // Exécution de la requête SQL.
6     $st->execute($params);
7     return $st;
}
```

3. PHP Data Object, <http://php.net/manual/fr/intro.pdo.php>

4. Plus d'informations <http://php.net/manual/fr/pdo.error-handling.php>

s }

5.4 Inscription sur le site

Afin de pouvoir bénéficier de toutes les fonctionnalités du site, l'utilisateur doit se connecter. Si celui-ci n'a pas de compte, il a la possibilité d'en créer un.

La création d'un utilisateur se fait en plusieurs étapes :

- Contrôle en AJAX lors de la saisie des informations par l'utilisateur
- Contrôle en PHP des informations
- Création de l'utilisateur en PHP et SQL

La vérification Ajax est assez simple, lorsque l'utilisateur appuie sur une touche, on affiche une croix rouge puis on envoie une requête à la page PHP qui va nous retourner un booléen à *true* si l'utilisateur existe. S'il est vrai, on laisse la croix rouge sinon on affiche un vu vers.

```

1      $('#Username').on('input', function(e) {
2          text = $("#Username").val();
3          $("#LogoUsername").removeClass("green");
4          $("#LogoUsername").removeClass("glyphicon-ok");
5          $("#LogoUsername").addClass("red");
6          $("#LogoUsername").addClass("glyphicon-remove");
7          if (text.length >= 5) {
8              exist = UserNameExists(text);
9              console.log(exist);
10             if (!exist) {
11                 $("#LogoUsername").removeClass("red");
12                 $("#LogoUsername").removeClass("glyphicon-remove");
13                 $("#LogoUsername").addClass("green");
14                 $("#LogoUsername").addClass("glyphicon-ok");
15             }
16         }
17         checkForEnabled();
18     });

```

```

1 function UserNameExists(username) {
2     exist = false;
3     $.ajax({
4         url: './Includes/ajax.php',
5         type: 'GET',
6         data: {fonction: "UserExists", Username: username},
7         dataType: "json",
8         async: false,
9         timeout: 30000,
10        success: function(result) {
11            if (result.exist) {
12                exist = true;
13                console.log("existe");
14            }
15        }
16    });
17    return exist;
18 }

```

Une fois la vérification faite en javascript/ajax, les données sont envoyées à une fonction PHP qui vérifie si l'utilisateur est déjà présent dans la base, puis, récupère l'identifiant de la moto d'après la marque, le modèle et l'année. Finalement, les données sont ajoutées à la base de données.

```

1 function signin($values) {
2     $values = json_decode($values);

```

```

3
4     $query = "Select Username FROM users where Username=:username;"
5     ;
6     $params = array('username' => $values->username);
7     $st = PrepareExecute($query, $params);
8     while ($data = $st->fetch(PDO::FETCH_ASSOC)) {
9         return false;
10    }
11
12    $query = "SELECT idMoto FROM moto WHERE Brand=:brand AND model=
13              :model AND year=:year";
14    $params = array(
15        'brand' => $values->brand,
16        'model' => $values->model,
17        'year' => $values->year . "-01-01"
18    );
19    $st = PrepareExecute($query, $params);
20    $idMoto = $st->fetch(PDO::FETCH_ASSOC)["idMoto"];
21    $query = "INSERT INTO users(Username, Password, email, idMoto,
22              role) VALUES (:username,:password,:email,:idmoto,2)";
23    $params = array(
24        'username' => $values->username,
25        'password' => $values->password,
26        'email' => $values->email,
27        'idmoto' => $idMoto
28    );
29    $st = PrepareExecute($query, $params);
30
31    global $pdo;
32    return $pdo->lastInsertId();
33 }
34
35 function GetRoutes($idUser = NULL) {
36     if ($idUser == NULL) {

```

5.5 Connexion au site

Une fois l'inscription effectuée, l'utilisateur a la possibilité de se connecter au site.

Pour ce faire, l'utilisateur saisi ses informations dans le formulaire et se connecte. Une première vérification en HTML5 vérifie que les champs soient remplis(*required*). Une fois cette vérification effectuée, les informations sont envoyée à la page PHP *connexion.php* qui vérifie encore une fois que les champs soient rempli. Puis, envoie les données à la fonction PHP de connexion qui va récupérer :

- Les ids utilisateur
- Les mots de passe
- Les roles

Une fois ces informations récupérée, on tous les parcourir pour être sûr que le nom d'utilisateur et le mot de passe fourni correspondent bien à un utilisateur existant. Si c'est le cas, on enregistre l'id, le role et le nom d'utilisateur dans des *Sessions* et on retourne *true* pour dire que les informations sont correcte.

```

1 if (!empty($_POST['username'])) {
2     if (!empty($_POST['password'])) {
3         $connexion = ConnexionUser($_POST['username'], sha1($_POST[

```



```
        'password']));
4      if (!$connexion) {
5          $reponse_array['status'] = 'error';
6          $reponse_array['message'] = 'wrong username or password';
7      } else {
8          if ($_SESSION["role"] == 3) {
9              session_destroy();
10             $reponse_array['status'] = 'error';
11             $reponse_array['message'] = 'You are banned';
12         } else {
13             $reponse_array['status'] = 'success';
14         }
15     }
16 } else {
17     $reponse_array['status'] = 'error';
18     $reponse_array['message'] = 'password is required';
19 }
20 } else {
21     $reponse_array['status'] = 'error';
22     $reponse_array['message'] = 'username is required';
23 }
24
25 header('Content-type: application/json');
26 echo json_encode($reponse_array);
```

```
1 function ConnexionUser($User, $Password) {
2     global $pdo;
3
4     $query = 'SELECT idUser, Username, Password, role FROM users ';
5
6     $st = PrepareExecute($query);
7     $connector = false;
8
9     while ($data = $st->fetch(PDO::FETCH_ASSOC)) {
10         if ($data['Username'] == $User && $data['Password'] == $
11             Password) {
12             $id = $data['idUser'];
13             $username = $data["Username"];
14             $role = $data["role"];
15
16             $connector = true;
17             break;
18         }
19     }
20     if ($connector) {
21         $_SESSION["role"] = $role;
22         $_SESSION["id"] = $id;
23         $_SESSION["username"] = $username;
24         return true;
25     } else {
26         return false;
27     }
28 }
```

6 Problèmes rencontré

6.1 Google Api

6.2 Async

7 Conclusion

Table des figures

1	Schéma du site	6
2	Page inscription	8
3	Page inscription	8
4	Page Detail Trajet	9
5	Modal connexion	9
6	Page vos trajet	10