

2016



Projet RandoNear

Banque collaborative de randonnées

Lénaïc Couëllan

Tuteur : Jean-Marc Lecarpentier

Année universitaire : 2015-2016



REMERCIEMENTS

Tout d'abord, je tiens personnellement à remercier :

- Jean-Marc Lecarpentier pour la proposition du sujet. Son aide sur les différents points du projet et comment les appréhender, ainsi que ses aides techniques lors de la réalisation des tâches.
- L'ensemble du corps enseignant de m'avoir donné les outils nécessaires pour mener à bien ce projet.
- L'Université Caen Normandie pour la mise à disposition du matériel et de permettre à tous les étudiants de bénéficier de logiciels nécessaires pour le bon déroulement de nos projets.

SOMMAIRE

Remerciements	2
Table des illustrations	5
Introduction	6
Analyse du projet	7
Contexte	7
Objectifs de l'application	8
Modélisation de la base de données et du site	9
Modélisation de la base de données	9
Modélisation du site	10
Le choix des outils	11
L'API de cartes embarquées	11
Le Framework PHP	12
Le Framework CSS	13
L'IDE	13
Réalisation du projet	14
Utilisation de l'API Google Maps pour la gestion des cartes géo localisées	14
Qu'est ce qu'un fichier GPX ?	14
La création des tracés GPS	15
La création de la carte regroupant toutes les randonnées	17
La création de mes pages avec Symfony	19
Qu'est ce qu'un MVC ?	19
L'architecture MVC de Symfony	20
L'affichage d'une randonnée	21
La création d'une randonnée	23
La recherche de randonnées	24
Le profil des randonneurs	25
L'accès manager de Symfony	25
Problèmes rencontrés et améliorations possibles	26
Problèmes rencontrés	26

Améliorations possibles	26
Conclusion	27
Webographie	28
Annexes	29

TABLE DES ILLUSTRATIONS

Figure 1 : Backlog à jour	8
Figure 2 : Logo Google Développeurs	11
Figure 3 : Logo Symfony	12
Figure 4 : Logo Bootstrap	13
Figure 5 : Logo PHPStorm	13
Figure 6 : Tracé d'une randonnée dans Caen	15
Figure 7 : Carte page d'accueil centrée sur Caen	17
Figure 8 : XML listant les randonnées du site	18
Figure 9 : Schéma MVC provenant de Openclassrooms	19
Figure 10 : Architecture du site	20
Figure 11 : Affichage d'une randonnée	21
Figure 12 : Formulaire de création de randonnée	23
Figure 13 : Recherche de randonnées	24
Figure 14 : Profil utilisateur	25
Figure 15 : MCD	29
Figure 16 : Wireframe page d'accueil	30
Figure 17 : Tracé d'une randonnée au format GPX	31

INTRODUCTION

Le projet RandoNear est un site internet qui a pour objectif de proposer aux utilisateurs du site, des randonnées près de là où ils se trouvent et adaptées à leur besoin.

Le but de ce site comparé aux concurrents est de proposer un tracé de la randonnée via l'enregistrement récupéré par le téléphone des randonneurs. Nous souhaitons aussi un aspect communautaire important. Seuls les utilisateurs du site pourront déposer leurs randonnées. Les autres utilisateurs pourront ainsi évaluer et commenter les randonnées.

Lorsque le projet a été proposé, les choix des différentes technologies étaient libres pour l'étudiant. Une analyse du projet afin de choisir au mieux les outils a donc été une première étape indispensable au projet.

C'est pourquoi, dans une première partie, je vous parlerai de l'analyse préalable du projet avec le contexte et les différents choix qu'il m'a été amené de faire. J'en profiterai pour faire une description de tous les outils principaux qui m'ont servi pendant le projet. Notamment au niveau des cartes de géo localisation ainsi que des Frameworks¹ choisis pour le projet.

Dans un second temps, je ferai une partie plus technique sur le projet en lui même avec une description de chaque fonctionnalité intégrée au site. J'en profiterai pour rappeler les librairies qui m'ont permis de réaliser ces fonctionnalités et comment je les ai intégré. Cette partie sera décomposée en deux sous-parties, une sous-partie concernant la gestion et l'utilisation de mon API² de création de cartes géo localisées ainsi qu'une sous-partie concernant les fonctionnalités de gestion des utilisateurs et des randonnées.

Enfin je parlerai des aspects de déroulement du projet, notamment les problèmes rencontrés. J'envisagerai les possibilités d'amélioration de mon site avant de conclure.

¹ Ensemble cohérent de composants structurels, qui sert à créer les fondations d'un site web.

² Ensemble normalisé de classes, de méthodes ou de fonctions fournies par un site ou logiciel afin de délivrer des services à d'autres sites ou logiciels.

ANALYSE DU PROJET

CONTEXTE

Le projet RandoNear a été proposé par Jean-Marc Lecarpentier. Sa réflexion était la suivante : Aujourd'hui tous les téléphones sont équipés d'un capteur GPS et de nombreuses applications utilisent ce procédé (Twisto, Les applications pour se faire livrer ...), et notamment les applications de "Running".

Cependant, les sites qui proposent des randonnées, n'utilisent pas ce procédé. Par exemple, il faut dessiner sa randonnée sur une carte vide. C'est une étape rébarbative et ce serait beaucoup plus simple d'enregistrer sa randonnée avec son téléphone et de n'avoir qu'à la mettre en ligne sur le site !

L'objectif du site est donc de pouvoir référencer toutes les randonnées qui se trouvent autour de l'utilisateur. Il pourra ensuite filtrer de manière simple, quelles randonnées l'intéressent.

Seuls les utilisateurs pourront créer une randonnée, pour une facilité d'utilisation, l'utilisateur pourra mettre le tracé de sa randonnée enregistrée sur son téléphone directement sur le site. Ainsi le tracé de la randonnée sera directement affichée sur une carte, il n'aura donc pas besoin de gérer lui-même l'affichage.

Ainsi nous pourrons avoir un site géré par la communauté, qui pourra tirer partie des technologies de géo localisation afin d'avoir une interface facile à utiliser et à naviguer pour l'internaute.

OBJECTIFS DE L'APPLICATION

Afin de créer une application qui corresponde aux attentes du tuteur, j'ai créé un Backlog afin que le tuteur informe des fonctionnalités principales de l'application et ainsi que l'application soit en adéquation avec ce qu'il recherche.

Un Backlog est un tableau concernant toutes les fonctionnalités importantes demandées par le client (ici le tuteur). Les fonctionnalités apparaissent dans un ordre chronologique, ainsi nous pouvons voir en temps et en heure l'avancée du projet.

Il contient aussi un ensemble de lots, c'est à dire, une période pour développer les fonctionnalités de l'application, à la fin de chaque lot, un livrable est montré au client afin de lui présenter les avancées du projet et si cela correspond bien à ce qu'il attend.

Story	Priorité	Enoncé	Lot	Avancement	Module	Remarques
1	1	Un utilisateur peut s'orienter sur une carte embarquée	1	Validé	Cartographie	
2	2	Un utilisateur peut voir un tracé de randonnée sur une carte	1	Validé	Cartographie	
3	3	Un utilisateur peut afficher ses tracés GPS	1	Validé	Cartographie	
4	4	Un utilisateur peut créer sa randonnée	2	Validé	Gestion Randonnée	
5	5	Un utilisateur peut consulter une randonnée	2	Validé	Gestion Randonnée	
6	6	Un utilisateur reçoit des suggestions de randonnées	2	Validé	Gestion Utilisateur	Priorité augmentée
7	7	Un utilisateur peut filtrer les randonnées	3	Validé	Gestion Utilisateur	
8	8	Un utilisateur peut évaluer et commenter les autres randonnées	3	Validé	Gestion Utilisateur	Optionnel
9	9	Un utilisateur peut entrer ses préférences	3	À faire	Gestion Utilisateur	
10	10	Une application pour téléphone permet de mettre sur le site sa randonnée	3	À faire	Application	Optionnel

FIGURE 1 : BACKLOG A JOUR

Comme nous pouvons le voir dans ce Backlog, les lots sont équitablement répartis.

Le lot 1 concerne l'utilisation de la carte embarquée. C'est-à-dire la géo localisation des randonnées sur une carte embarquée ainsi que l'affichage d'un tracé de randonnée. Ces parties étant les fonctionnalités primordiales du site, c'est donc pour cela qu'elles doivent être développées en premier.

Le lot 2 concerne la gestion des randonnées. Notamment la création d'une randonnée sur le site et l'affichage de cette randonnée. Cela concerne donc le formulaire de création d'une randonnée, l'insertion dans une base de données³, la liaison entre une randonnée et son parcours et le moteur de recherches de randonnées.

Le lot 3 gèrera quant à lui, les utilisateurs de notre site. La gestion de comptes, les préférences utilisateurs (s'il a l'habitude de faire des randonnées simples, de moins de 2 heures ...) et aussi la gestion communautaire, via notamment le système de commentaires de randonnées.

³ Conteneur informatique permettant de stocker dans un même lieu l'intégralité des informations en rapport avec une activité.

MODELISATION DE LA BASE DE DONNEES ET DU SITE

Afin de savoir comment structurer son site et d'accomplir au mieux les objectifs que je me suis fixé, je me suis d'abord lancé dans plusieurs étapes de modélisation.

Tout d'abord, j'ai créé plusieurs Wireframes⁴ afin de définir mes pages, comment les relier entre elles et ce qu'elles doivent contenir.

J'ai aussi créé un schéma de ma base de données afin de savoir quelles données j'allais devoir stocker ainsi que les relations entre mes différentes tables.

MODELISATION DE LA BASE DE DONNEES

Afin de modéliser ma base de données, j'ai créé un MCD (voir annexe 1), qui m'a permis ensuite de créer ma base de données.

J'ai d'abord créer une table Randonnée, avec un identifiant, les caractéristiques d'une randonnée (nom, longueur, temps). Mais aussi le nom du fichier du parcours (fichier que l'on récupère du téléphone des randonneurs), le lien vers une image et l'identifiant de l'auteur. L'identifiant est récupéré par la table des utilisateurs. Une randonnée est donc reliée à un utilisateur.

Une table Utilisateur qui comprend les informations de connexion. Cependant, pendant le développement, j'ai remarqué que le FOSUserBundle de Symfony permettait la création de cette table, dans le projet, elle est donc auto-générée.

De même, la table Commentaire est remplacée par deux tables. Une table "Thread" qui correspond à un thème de commentaires. Dans notre site, il y'a donc un sujet par randonnée. Ainsi qu'une table "Comments", qui liste tous les commentaires de notre site. Ces tables ont été créées par une autre librairie de Symfony, le FOSCommentBundle (On remarquera que les librairies de Symfony commencent par "FOS", pour "Friends Of Symfony").

A partir de ce MCD, j'ai donc pu créer une première version de ma base.

⁴ Maquette fonctionnelle permettant de définir les zones et composants que doit contenir un site internet.

MODELISATION DU SITE

Pour m'aider à faire mes différents croquis de mon site, j'ai utilisé le logiciel Balsamiq. Vous pourrez trouver un exemple en annexe 2. Il utilise une fenêtre basique et on peut par un système de "cliquer-glisser", poser des éléments HTML⁵. Il fut donc assez simple de voir quels éléments utiliser et comment les mettre en concordance.

Ces maquettes m'ont permis de savoir quels éléments Bootstrap utiliser et comment gérer les fonctionnalités de mon site.

Par exemple, j'ai décidé de mettre la carte de géo localisation des randonnées sur la page d'accueil, comme cela, les utilisateurs n'auront pas à perdre de temps pour trouver une randonnées.

De même, j'ai pu créer ma page de filtrage de randonnées et en accord avec monsieur Lecarpentier, voir quels sont les critères important. Ainsi, j'ai pu en même temps ajouter des champs dans ma table de randonnées.

Ces maquettes et ce MCD m'ont permis de me poser des questions sur le projet et mieux comprendre les attentes de mon tuteur, j'ai pu ainsi mieux appréhender le sujet que si j'avais tout de suite décidé de créer les fonctionnalités.

⁵ Format de données conçu pour représenter des pages web

LE CHOIX DES OUTILS

Afin de m'aider pour le développement complet de mon site internet, j'ai du réfléchir à quels outils utiliser. Il faut savoir choisir avec parcimonie car certains peuvent faire ralentir le projet où le faire tomber dans une impasse. De même utiliser trop d'API ou de Framework peut alourdir le projet et engendrer des conflits. C'est pour cela que j'ai décidé d'utiliser un Framework CSS⁶, un Framework PHP⁷, une API de cartes embarquées ainsi qu'un IDE⁸

L'API DE CARTES EMBARQUÉES



FIGURE 2 : LOGO GOOGLE DEVELOPPEURS

Pour le choix de mon API de cartes embarquées, j'ai regardé en fonction de ce que l'on me proposait comme utilitaires et du résultat final. Mes choix se sont portés sur l'API de Google Maps et l'API de OpenStreetMap. Les deux API proposent des fonctions similaires. Possibilité d'ajouter des points de repère sur une carte. Possibilité d'afficher un parcours et de le générer à partir des parcours enregistrés sur les téléphones.

J'ai choisi pour ma part l'API de Google Maps car il s'agit des cartes embarquées qu'on a le plus l'habitude de côtoyer. Les cartes OpenStreetMap sont aussi répandues que celles de Google Maps sur les sites de randonnées mais elles font appel à plus de détails. Il faut rappeler que même les randonneurs les moins expérimentés pourront utiliser notre site, il ne faut donc pas qu'ils soient perdus. Avec cette API, les randonneurs aussi bien novices qu'expérimentés pourront comprendre les tracés proposés sur le site.

⁶ Feuille de style permettant la mise en forme de documents et pages informatiques.

⁷ Langage de programmation libre permettant le développement de pages web dynamiques.

⁸ Ensemble d'outils permettant d'augmenter la productivité des programmeurs.

LE FRAMEWORK PHP



FIGURE 3 : LOGO SYMFONY

Quant au choix du Framework PHP, plusieurs options s'offraient à moi. Déjà pourquoi un Framework PHP plutôt qu'un Framework Javascript⁹. Le choix fut vite opté d'utiliser un Framework PHP car malgré l'API Google qui est utilisée en Javascript, le reste aurait été plus compliqué à apprendre avec un Framework Javascript plutôt qu'un Framework PHP.

J'ai donc décidé de prendre Symfony car c'est le Framework PHP le plus connu, il est aussi utilisé dans les CMS comme Drupal ou PHPBB. Il propose un large panel d'outils notamment le FOSUserBundle, un outil qui gère la connexion utilisateur sur un site ainsi que Doctrine, un outil de gestion de base de données.

De plus, Symfony est un Framework stable (première version sortie en 2005), les programmeurs sont très réactifs et fournissent beaucoup de documentations sur internet. Ce Framework est aussi très réputé, c'est donc une obligation pour un étudiant de Master de le maîtriser.

⁹ Langage de programmation de scripts principalement utilisé dans les pages web interactives mais qui se développent côté serveur.

LE FRAMEWORK CSS



FIGURE 4 : LOGO BOOTSTRAP

J'ai décidé d'utiliser un Framework CSS, choix n'étant pas obligatoire pour le bon déroulement du projet. Cependant, Bootstrap propose de nombreux outils à disposition du développeur afin de faciliter la mise en page. Notamment un affichage en forme de grilles qui permettent au développeur de gérer très facilement le côté responsive design¹⁰.

L'IDE



FIGURE 5 : LOGO PHPSTORM

Je ne m'attarderai pas sur le choix de l'IDE car il s'agit d'un choix personnel et qui n'a que peu d'influences sur le projet. PHPStorm est l'IDE le plus adéquat pour le développement de sites web avec Symfony. Il propose un plugin Symfony très complet que les autres IDE ne proposent pas. PHPStorm est très vivement conseillé par les créateurs de Symfony et leurs développeurs actifs.

¹⁰ Site web permettant un affichage confortable pour l'utilisateur que ce soit sur téléphones, tablettes ou ordinateurs.

REALISATION DU PROJET

Pour la réalisation du site web RandoNear, j'ai abordé deux parties techniques importantes.

La première sur tout la partie de gestion de L'API Google en JavaScript et la manipulation des fichiers de tracés de randonnées. La seconde concernant la création du site en PHP avec notamment la gestion des librairies Symfony.

Cette partie traitera donc en premier temps de l'utilisation de l'API Google Maps et la gestion en JavaScript et en second temps je parlerai de l'utilisation de Symfony et de la création de mon site internet.

UTILISATION DE L'API GOOGLE MAPS POUR LA GESTION DES CARTES GEO LOCALISEES

Pour la gestion de mes cartes Google Maps sur mon site, j'ai utilisé le Google Maps Embed API. Il m'a fallu demander une clé à Google pour avoir accès à l'API et ensuite intégrer les cartes où je le souhaitais.

Le site propose une carte avec les différents marqueurs indiquant les emplacements des randonnées et une carte pour chaque randonnée affichant leur parcours.

Je parlerai donc d'abord de l'affichage d'un parcours de randonnée notamment avec la gestion des fichiers GPX. Ensuite je parlerai de la création des marqueurs en fonction des randonnées enregistrées dans la base de données.

QU'EST CE QU'UN FICHIER GPX ?

Un fichier GPX est un fichier au format XML¹¹ qui répertorie les informations de mouvements d'une personne. Ce fichier stocke notamment la longitude, la latitude, le dénivelé, le temps etc. (Voir annexe 3)

Il est donc important pour notre site, d'utiliser au maximum ce fichier, afin de récupérer les informations de la randonnée.

¹¹ Meta langage informatique permettant la gestion de données structurées sous forme de balises

Il est très simple pour un randonneur de capturer ses mouvements lors d'une randonnée. Il a juste à télécharger une application sur son téléphone (comme Geo Tracker sur Android). Il lance l'application et active le signal GPS lorsqu'il commence une randonnée. L'application tourne en arrière plan avec un chronomètre et lorsque l'utilisateur a décidé que la randonnée est finie, il appuie sur "Stop". Le fichier est alors généré.

L'API Google utilise ce type de fichier et le type de fichier KML (format propre à Google), afin de créer les tracés. En effet, toutes les secondes, les coordonnées GPS sont enregistrées dans le fichier. Google crée donc un point sur sa Google Maps pour chaque coordonnée dans le fichier et il trace ensuite le tracé à partir de tous les points.

LA CRÉATION DES TRACÉS GPS



FIGURE 6 : TRACE D'UNE RANDONNEE DANS CAEN

Afin d'afficher ce tracé GPS d'un fichier GPX. Tout est géré dans un fichier en JavaScript. Ce fichier JavaScript n'est appelé que lors de la consultation d'une randonnée (vu que les tracés sont tous reliés à une randonnée). En effet grâce à un outil de Template¹² utilisé par Symfony appelé Twig. Je peux donc décider de quels fichiers JavaScript j'appelle selon le page que je souhaite consulter.

La gestion de ses tracés se fait en deux temps.

D'abord, lors de la création de la randonnée, les fichiers GPX sont stockés dans une partie du site (pour l'instant, sur mon ordinateur). Ensuite, dans le "DefaultController" du site, je crée un url pour chaque fichier GPX (/traces/{nomDuFichier}) où le fichier est affiché au format XML.

¹² Patron de mise en pages de textes ou d'images

Puis dans mon fichier "trace.js", je crée une requête Ajax¹³ avec la technologie JQuery¹⁴ pour avoir accès au fichier GPX dont j'ai besoin, et je récupère les informations de géo localisation (Ici la latitude et la longitude). (Voir requête ci-dessous).

```
$.ajax({
  type: "GET",
  url: "../traces/"+file,
  dataType: "xml",
  success: function (xml) {
    var points = [];
    var bounds = new google.maps.LatLngBounds();
    $(xml).find("trkpt").each(function () {
      var lat = $(this).attr("lat");
      var lon = $(this).attr("lon");
      var p = new google.maps.LatLng(lat, lon);
      points.push(p);
      bounds.extend(p);
    });
    var poly = new google.maps.Polyline({
      path: points,
      strokeColor: "#FF00AA",
      strokeOpacity: .7,
      strokeWeight: 4
    });
    poly.setMap(map);
    map.fitBounds(bounds);
  }
});
```

Ensuite grâce à l'API de Google Maps, je fais un lien entre chaque position afin de créer un tracé sur ma carte Google Maps. On remarquera que la carte est centrée sur le tracé car je la positionne avec un zoom élevé sur la première coordonnée de la randonnée.

¹³ Permet de récupérer des informations serveur en JavaScript.

¹⁴ Librairie JavaScript facilitant l'écriture des scripts côté client.

LA CREATION DE LA CARTE REGROUPANT TOUTES LES RANDONNEES

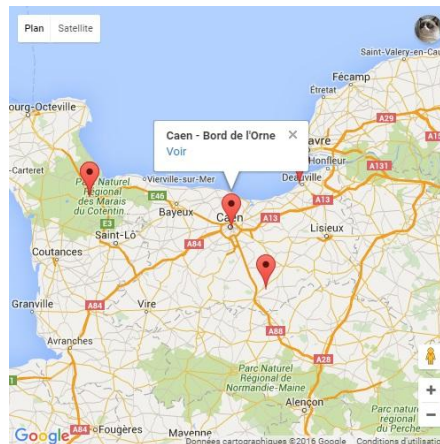


FIGURE 7 : CARTE PAGE D'ACCUEIL CENTREE SUR CAEN

Pour afficher toutes les randonnées sur une carte centrée en fonction de là où on se trouve, j'ai créé un nouveau fichier JavaScript qui est appelé que sur l'accueil.

Ce fichier comprend plusieurs fonctions. Une fonction principale "load" qui prend en paramètres une latitude et une longitude. Deux fonctions "showPosition" et "bindInfoWindow", fonctions génériques qui permettent de récupérer et traiter les informations de localisation du navigateur. Ainsi qu'une fonction "downloadUrl" qui permet de récupérer un fichier XML en JavaScript natif.

COMMENT CELA FONCTIONNE ?

On souhaite récupérer plusieurs caractéristiques d'une randonnée afin de l'afficher correctement. On va donc prendre en base de données les informations suivantes :

- La latitude et la longitude afin qu'une randonnée soit affichée correctement à son lieu de départ.
- Le nom afin d'afficher correctement une randonnée et que les utilisateurs ne soient pas désorientés.
- L'identifiant afin de rediriger le marqueur vers la page de la randonnée, vu que pour aller sur une page de randonnée, on y accède via `"/map/{idRandonnée}"`.

On crée ensuite un fichier XML afin de gérer ces données en JavaScript (Un fichier JSON, autre format de fichier de balisage, aurait fonctionné également).

Dans le "DefaultController" du site, je crée une fonction afin de générer le fichier XML permettant de lister toutes les randonnées avec les informations que l'on souhaite exploiter. Ce fichier est alors affiché sur une page de mon site "/displayMarkers", afin de pouvoir les récupérer en Ajax. (Voir affichage ci-dessous).

```
▼<markers>
  <marker id="1" name="Reserve du coteau Mesnil Soleil" lat="48.927219" lng="-0.148719"/>
  <marker id="2" name="Carentan - Sur les traces des paras" lat="49.304901" lng="-1.240941"/>
  <marker id="3" name="Deauville et ses hauteurs" lat="49.355171" lng="0.057726"/>
  <marker id="4" name="Caen - Bord de l'Orne" lat="49.173363" lng="-0.362593"/>
  <marker id="10" name="Rando ajoutée par formulaire" lat="49.304901" lng="-1.240942"/>
</markers>
```

FIGURE 8 : XML LISTANT LES RANDONNEES DU SITE

Une fois ce fichier construit, je peux utiliser ces données en JavaScript.

Tout d'abord, je dois récupérer les informations du navigateur afin de centrer correctement ma carte embarquée.

Pour cela c'est assez simple, j'utilise la fonction "getCurrentPosition", à partir de ce moment, lorsqu'un internaute va sur mon site, le navigateur lui demande s'il souhaite communiquer ses informations de géo localisation.

S'il refuse, la carte est centrée par défaut sur Caen, sinon je transmets à ma fonction principale "Load", sa latitude et sa longitude, ainsi la carte est centrée sur sa position.

Ensuite afin d'afficher tous mes marqueurs, je récupère mon fichier XML en Ajax. Ainsi, je parcours toutes les randonnées de mon fichier et pour chaque randonnée, je crée un marqueur contenant, le nom et le lien vers la randonnée. (Voir requête ci-dessous).

```
downloadUrl("displayMarkers", function(data) {
    var xml = data.responseXML;
    var markers = xml.documentElement.getElementsByTagName("marker");
    for (var i = 0; i < markers.length; i++) {
        var name = markers[i].getAttribute("name");
        var id = markers[i].getAttribute("id");
        var point = new google.maps.LatLng(
            parseFloat(markers[i].getAttribute("lat")),
            parseFloat(markers[i].getAttribute("lng")));
        var html = "<b>" + name + "</b><br/><b><a href=\"map/" + id + "\">Voir</a></b>";
        var marker = new google.maps.Marker({
            map: map,
            position: point
        });
        bindInfoWindow(marker, map, infoWindow, html);
    }
});
```

LA CREATION DE MES PAGES AVEC SYMFONY

Durant cette partie, j'aborderai tout d'abord l'architecture de mon site avec le modèle MVC (modèle - vue - contrôleur) . Puis je parlerai de chacune des fonctionnalités que j'ai développé avec notamment les outils de Symfony.

QU'EST CE QU'UN MVC ?

L'architecture MVC est apparue lors des premiers développements d'applications en objet. En effet il permet de bien organiser son code source. Le but étant de séparer la logique du code en trois parties :

- Le modèle : Cette partie gère l'interaction avec la base de données, elle récupère les données brutes et les assemble afin que le contrôleur puisse les gérer.
- La vue : Cette partie se concentre sur l'affichage. Elle ne se contente que d'afficher les résultats provenant du contrôleur. Il ne s'agit surtout que de HTML et de quelques boucles simples en PHP.
- Le contrôleur : Cette partie gère la logique du code qui prend les décisions, elle sert à faire le lien entre le modèle et la vue. C'est aussi ici que les algorithmes sont mis en place.

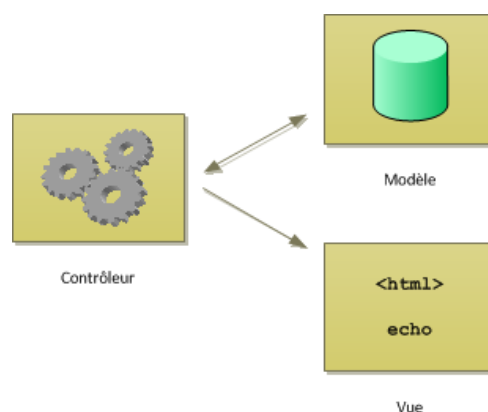


FIGURE 9 : SCHEMA MVC PROVENANT DE OPENCLASSROOMS

L'ARCHITECTURE MVC DE SYMFONY

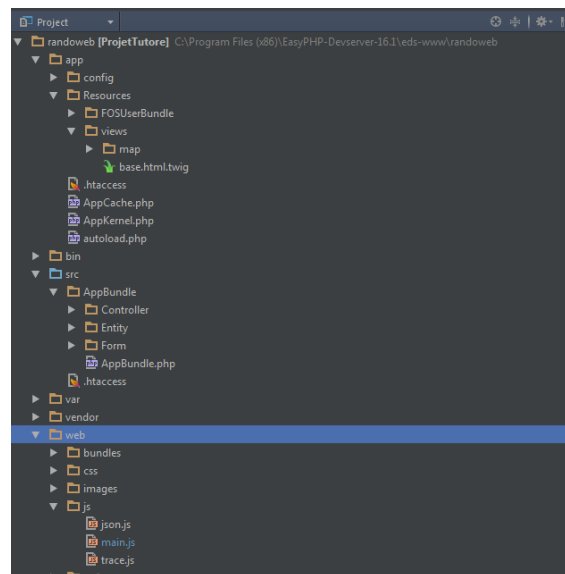


FIGURE 10 : ARCHITECTURE DU SITE

Symfony propose sa propre architecture MVC qui est réutilisée dans plusieurs CMS comme Drupal ou PHPBB. Il s'agit de la suivante :

Un dossier "app" qui regroupe les configurations du site tel que les chemins d'accès à la base de données, les appels aux différentes bibliothèques de Symfony, la traduction du site ... Mais aussi nos pages de Template ("La vue"), le dossier "Ressources" contient toutes les pages Twig qui permettent d'afficher les pages du site.

Un dossier bin qui contient la console de Symfony.

Un dossier "src" qui contient tous les contrôleurs du site, mais aussi, toutes les entités. C'est à dire, les classes principales du site. Comme la classe "Randonnée", "User" ... Il s'agit donc du dossier clé où toutes les fonctions importantes en PHP sont stockées.

Un dossier "var" qui ne contient que les variables de caches, log et sessions.

Un dossier "vendor" qui permet de stocker toutes les librairies utiles à notre site comme Doctrine, FOSUserBundle ...

Et un dossier "web" qui permet de stocker les fichiers enregistrés sur notre site. Il contient par exemple, les tracés des parcours, les images des randonnées. Mais aussi nos fichiers de CSS et de JavaScript.

L'AFFICHAGE D'UNE RANDONNEE

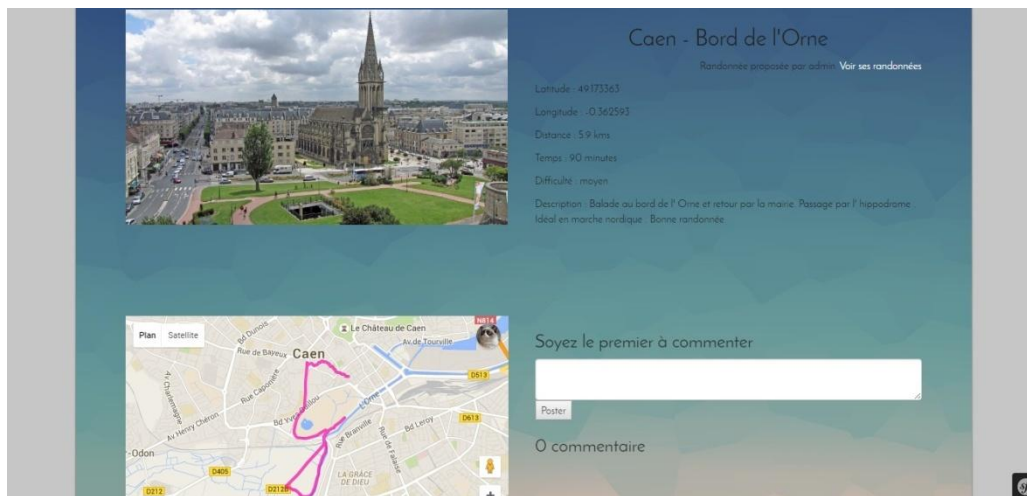


FIGURE 11 : AFFICHAGE D'UNE RANDONNEE

Lorsque l'on arrive sur une randonnée, on voit à l'écran plusieurs informations.

Le titre de la randonnée, l'auteur avec un lien vers son profil. Ainsi que les autres informations comme la distance, le temps, la difficulté ...

On voit aussi le tracé de la randonnée comme expliqué dans la partie du dessus. Mais aussi un espace de commentaires.

Le tracé de la randonnée est un travail à part en JavaScript et l'on a déjà abordé le sujet, cependant le reste de l'affichage se fait en PHP notamment à partir d'un fichier, le "mapController".

Nous allons prendre la fonction d'affichage d'une randonnée comme exemple afin de montrer comment fonctionne Symfony.

```

/**
 * @Route("map/{randoId}")
 */
public function showRandonnee($randoId, Request $request)
{
    $randonnee = $this->getDoctrine()
        ->getRepository('AppBundle:Randonnee')
        ->find($randoId);
    $em = $this->getDoctrine()->getManager();
    $query = $em->createQuery(
        "SELECT u FROM AppBundle:User u
JOIN AppBundle:Randonnee r WITH r.author = u.id
WHERE r.id= :id"
    )->setParameter('id', $randoId);
    $users = $query->getArrayResult();

    if (!$randonnee) {
        throw $this->createNotFoundException('Désolé, la randonnée n\' a pas pu être trouvée !');
    }

    /* Commentaires */
    $thread = $this->container->get('fos_comment.manager.thread')->findThreadBy(['permalink'=>$request->getUri()]);
    if (null === $thread) {
        $thread = $this->container->get('fos_comment.manager.thread')->createThread();
        $thread->setPermalink($request->getUri());

        // Add the thread
        $this->container->get('fos_comment.manager.thread')->saveThread($thread);
    }
    $comments = $this->container->get('fos_comment.manager.comment')->findCommentTreeByThread($thread);

    $templating = $this->container->get('templating');
    $html = $templating->render('map/rando.html.twig', [
        'randonnee'=>$randonnee,
        'user'=>$users[0],
        'comments' => $comments,
        'thread' => $thread
    ]);
    return new Response($html);
}

```

Pour l'affichage d'une randonnée, on indique sur quelle page on y accède grâce à l'annotation "Route" de Symfony. Par exemple ici, on peut voir que dès que l'on va sur l'url `map/{idRandonnee}`, on utilise la fonction "showRandonnee". On peut utiliser des variables dans notre url comme ici pour "randold". Du coup, si l'on va à l'url `map/4`, le paramètre "4" est donné à la fonction. On peut donc afficher autant de randonnées que l'on souhaite.

Ensuite on voit dans les trois premières lignes que l'on utilise des fonctions non natives de PHP. Ici, nous utilisons la librairie Doctrine. Doctrine gère toute la gestion de la base de données de notre site. On a créé une classe "Randonnée", avec les attributs que l'on souhaitait, et avec des annotations, Doctrine a pu créer la table Randonnée. Cette librairie met à jour la base de données grâce à une commande dans la console de Symfony.

On peut donc voir ici que Doctrine cherche une entité de la classe "Randonnée" à partir de l'identifiant donné en paramètre. On récupère ainsi la randonnée que l'on souhaite afficher.

Ensuite nous avons besoin de récupérer l'auteur de la randonnée. Je passe par une seconde requête car le nom de l'auteur se situe dans une autre table "User" reliée à la table randonnée, je suis donc obligé de faire une jointure entre les deux tables.

Pour les commentaires, j'utilise une autre librairie de Symfony appelée FOSCommentBundle. On peut voir ici que l'on crée un sujet pour chaque randonnée.

Ensuite pour communiquer toutes ces données, nous envoyons une "réponse" à notre vue, ici le fichier "rando.html.twig". On lui donne les informations de randonnées, l'auteur de la randonnée, ainsi que les informations de commentaires.

Le fichier Twig peut alors créer un fichier HTML avec les informations souhaitées récoltées à partir du contrôleur.

LA CRÉATION D'UNE RANDONNÉE

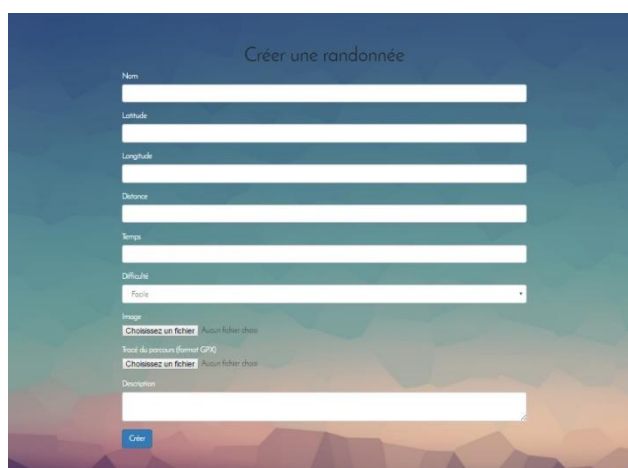


FIGURE 12 : FORMULAIRE DE CREATION DE RANDONNEE

La création de formulaire est assez simple avec Symfony, en effet, il faut créer une classe pour chaque formulaire. Ici j'ai créé la classe "RandonneeForm" qui hérite de "AbstractType", j'ai une fonction "buildForm" qui permet d'indiquer tous les champs du formulaire.

Une fois cette fonction faite, j'indique au formulaire que tous les champs sont reliés à ma classe "Randonnée". Ainsi, lorsque le formulaire remplit correctement les conditions, une nouvelle entité "Randonnée" est créée et elle s'enregistre en base de données.

Pour la gestion des fichiers, je crée un identifiant unique pour chaque fichier (image ou GPX). Je les stocke dans le répertoire "web" du site respectivement "web/images/rando", "web/traces". Seuls les fichiers au format image et GPX peuvent être envoyés, sinon un message d'avertissement apparaît lors de la création de la randonnée.

LA RECHERCHE DE RANDONNÉES



FIGURE 13 : RECHERCHE DE RANDONNEES

Cette fonctionnalité fut assez difficile à mettre en place car elle demandait de savoir comment utiliser les outils de Symfony pour la rendre la plus optimale possible.

J'ai décidé pour ma part de créer deux classes : `RandonneeFilter` et `RandonneeFilterForm`.

En effet `RandonneeFilter` est une classe qui contient tous les paramètres de recherche d'une randonnée comme la distance, le temps etc.

On s'aperçoit que la recherche de randonnées est en fait un formulaire, il a donc fallu créer la classe `RandonneeFilterForm` afin de choisir comment filtrer nos randonnées. Une fois que l'utilisateur clique sur le bouton "Rechercher", une entité `RandonneeFilter` est créée.

Une requête comprenant tous les résultats du formulaire est envoyée au contrôleur. Un contrôle des données est fait antérieurement afin que la requête ne soit pas incorrecte. Une liste de randonnées est retournée. Cette liste est envoyée à la vue et l'utilisateur peut donc se rendre sur les randonnées de son choix.

J'ai aussi ajouté un système d'auto complétion grâce à l'outil de JQuery UI, `Autocomplete`. J'ai créé un fichier JSON avec le nom des randonnées récupérées en Ajax. Ainsi les utilisateurs pourront plus facilement retrouver des randonnées.

LE PROFIL DES RANDONNEURS



FIGURE 14 : PROFIL UTILISATEUR

Pour l'instant, peu de choses sont affichées dans le profil des utilisateurs. Tout le monde peut accéder à un profil. On peut voir les randonnées qu'un utilisateur a posté sur le site.

Cette gestion n'est pas gérée par le FOSUserBundle, il s'agit d'une fonction dans le contrôleur "UserController".

On récupère tous les informations de l'utilisateur grâce au nom d'utilisateur donné en paramètre "GET". Une fonction utilisant Doctrine récupère toutes les randonnées qu'il a créé. Les randonnées et les informations utilisateur sont ensuite envoyées à la vue.

L'ACCES MANAGER DE SYMFONY

Je vais parler très brièvement de l'accès manager de Symfony. Il s'agit d'un fichier "security.yml" qui liste tous les droits des utilisateurs du site.

Exemple :

```
access_control:
- { path: ^/login$, role: IS_AUTHENTICATED_ANONYMOUSLY }
- { path: ^/register, role: IS_AUTHENTICATED_ANONYMOUSLY }
- { path: ^/resetting, role: IS_AUTHENTICATED_ANONYMOUSLY }
- { path: ^/admin/, role: ROLE_ADMIN }
- { path: ^/map/new, role: ROLE_USER }
```

On peut voir par exemple que la création d'une nouvelle randonnée demande d'abord l'authentification de l'utilisateur. Si l'utilisateur n'est pas connecté et qu'il essaie de se rendre sur la page de création de randonnée, il est alors redirigé vers la page de connexion.

PROBLEMES RENCONTRES ET AMELIORATIONS POSSIBLES

Le site aujourd'hui présente presque toutes les fonctionnalités prévues dans le Backlog cependant beaucoup d'améliorations sont encore possibles. Le retard pris au début du projet a empêché le développement de ces dernières fonctionnalités.

PROBLEMES RENCONTRES

Le problème majeur qui m'a suivi tout au long du projet et il était attendu, c'est que je n'avais jamais travaillé sur un Framework. Symfony est le Framework le plus difficile à appréhender car il présente énormément d'outils et il faut donc se familiariser avec chacun.

J'ai pris beaucoup de retard notamment au début du projet ne serait-ce que pour mettre en place le site, je remercie d'ailleurs monsieur Lecarpentier de m'avoir appris à faire fonctionner Composer notamment, outil indispensable de Symfony.

J'ai passé autant de temps à lire la documentation de Symfony et à suivre les tutoriels qu'à développer. En effet comme vous avez pu le constater, ce Framework demande un code totalement différent que ce soit au niveau des annotations, des classes, des entités ... Il faut donc passer du temps à comprendre comment mettre tout en place afin d'arriver à ses fins.

Le deuxième problème que j'évoquerai est ma gestion du temps. Nous avons eu peu de temps pour travailler sur le lot 1 et 2 mais beaucoup plus sur le lot 3. Je me suis donc retrouvé en retard sur mes premiers lots. Du coup je n'ai pas été assez à la rencontre de mon tuteur pour décider ensemble des améliorations du projet pour le prochain lot.

AMELIORATIONS POSSIBLES

Trois points peuvent être améliorés sur le site :

- Au niveau du profil utilisateur, on pourrait ajouter un système de préférences afin que lors de la recherche de randonnées, les critères soient directement entrés. De plus, le profil utilisateur est assez vide pour le moment, on pourrait par exemple ajouter un avatar, que l'utilisateur puisse lui même remplir son profil etc.

- Au niveau des randonnées, il faudrait qu'un utilisateur puisse modifier les randonnées qu'il a créé. On souhaiterait aussi que le formulaire de création de randonnée soit moins long à remplir.

- Au niveau du tracé des randonnées, on souhaiterait pouvoir ajouter plus d'images aux randonnées et de pouvoir les lier au tracé. Ainsi l'utilisateur pourra voir directement où les photos ont été prises.

Enfin une application sœur au site afin de pouvoir récupérer ses tracés de randonnées serait utile, ainsi le site serait totalement indépendant.

CONCLUSION

RandoNear est un projet qui m'a beaucoup apporté et dont le travail fourni me servira pour la suite. Il fut très intéressant de travailler avec des outils que l'on aborde pas en cours comme l'API Google Maps ou l'utilisation de fichiers GPX. Je suis satisfait d'avoir pu créer un site qui n'avait aucune base car tous les éléments de base d'un site internet comme la connexion ou la recherche me seront utile pour la suite.

Malgré le temps que j'ai passé à apprendre Symfony, je suis extrêmement satisfait d'avoir utilisé ce Framework. J'ai pu bien appréhender les bases et ce sera très utile pour la suite de mes études et pour ma recherche future d'emploi.

Enfin, j'ai pu utiliser le développement agile pour la création de mon projet, cet outil m'a permis d'avoir une vision différente de rendu de projet. Cette méthode est aujourd'hui très utilisée et demandée dans le monde du travail, cette première approche est donc très importante pour la suite.

WEBOGRAPHIE

Wikipedia.org

openclassrooms.com

developpez.net

symfony.com

knpuniversity.com

alsacrations.com

ANNEXES

Annexe 1 : MCD

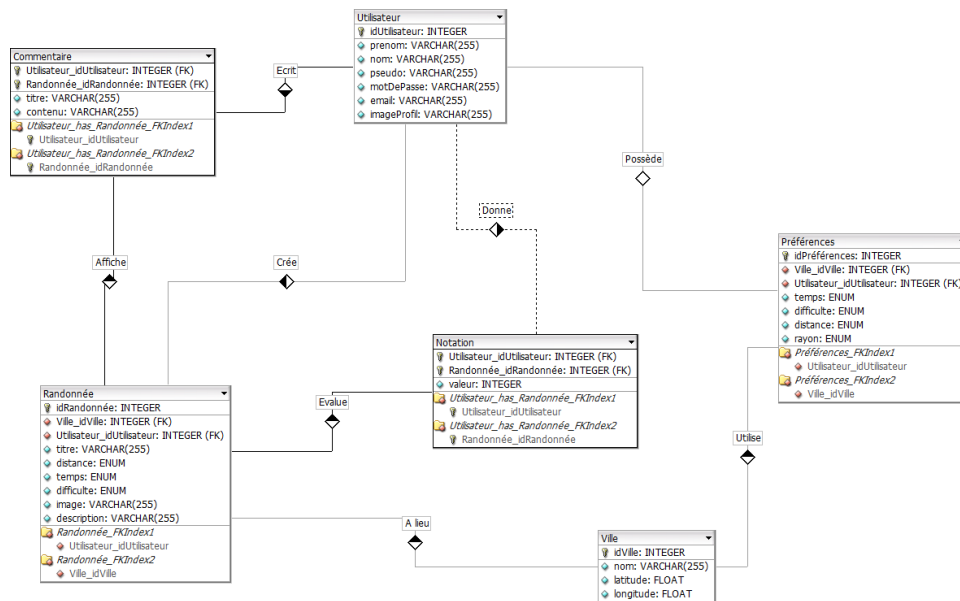


FIGURE 15 : MCD

Annexe 2 : Exemple de Wireframe avec Balsamiq

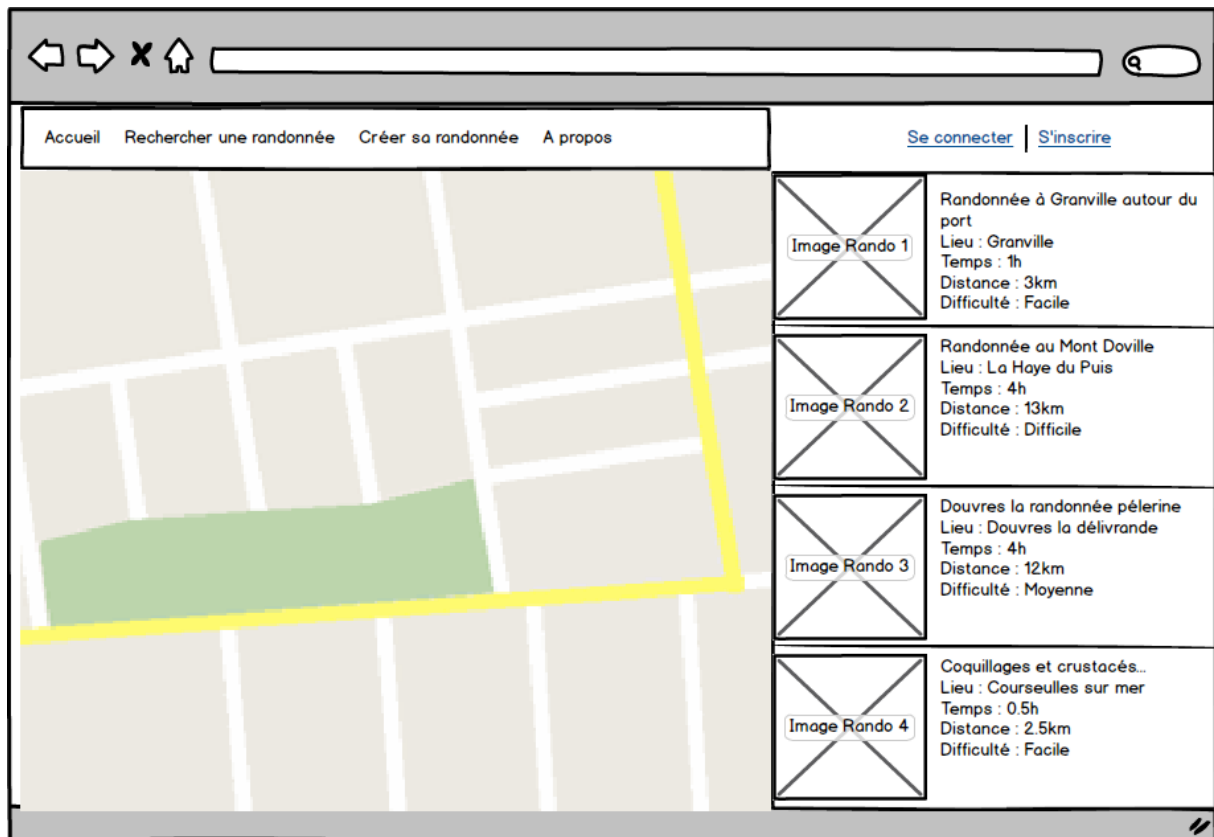


FIGURE 16 : WIREFRAME PAGE D'ACCUEIL

Annexe 3 : Exemple de fichier GPX

```

1 <?xml version='1.0' encoding='UTF-8' standalone='yes' ?>
2 <gpx version="1.1" xmlns="http://www.topografix.com/GPX/1/1" xmlns:geotrack="http://ilyabogdanovich.
  com/gpx/extensions/geotrack" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.topografix.
  com/GPX/1/1 http://www.topografix.com/GPX/1/1/gpx.xsd" creator="Geo Tracker 3.2.2 for Android by Ilya Bogdanovich">
3   <metadata>
4     <name>7 mai 2016 16:58:00</name>
5     <link href="https://play.google.com/store/apps/details?id=com.ilyabogdanovich.geotrack" />
6     <time>2016-05-07T14:58:00.091Z</time>
7     <author>
8       <name>Enregistré par Geo Tracker pour Android de Ilya Bogdanovich</name>
9       <link href="https://play.google.com/store/apps/details?id=com.ilyabogdanovich.geotrack" />
10    </author>
11  </metadata>
12  <trk>
13    <name>7 mai 2016 16:58:00</name>
14    <src>Enregistré par Geo Tracker pour Android de Ilya Bogdanovich</src>
15    <link href="https://play.google.com/store/apps/details?id=com.ilyabogdanovich.geotrack" />
16    <extensions>
17      <geotrack:meta>
18        <length>7642.75</length>
19        <duration>4842454</duration>
20        <creationtime>2016-05-07T14:58:00.091Z</creationtime>
21        <activity>0</activity>
22      </geotrack:meta>
23    </extensions>
24    <trkseg>
25      <trkpt lat="49.2981" lon="-1.251393">
26        <ele>59.88</ele>
27        <time>2016-05-07T14:58:03.462Z</time>
28        <extensions>
29          <geotrack:meta s="0.48" />
30        </extensions>

```

FIGURE 17 : TRACE D'UNE RANDONNEE AU FORMAT GPX