TC ADVISOR

PROJET WEB – DOCUMENT TECHNIQUE

Hugo Dupré
Théo Moutier
Jean Cassou-Mounat
Alicia Lebreton

Département Télécommunications, Usages et Services

Mai 2016

Description du projet

TCAdvisor est un site fait pour que chaque étudiant du département Télecommunication de l'INSA Lyon, qui a pour souhait de partir en échange ou en stage à l'étranger, puisse accéder facilement aux informations liées aux stages et aux échanges du département.
Cette base de données est remplie par des étudiants déjà partis.
Chaque TC voulant faire un **échange** peut ainsi accéder aux :

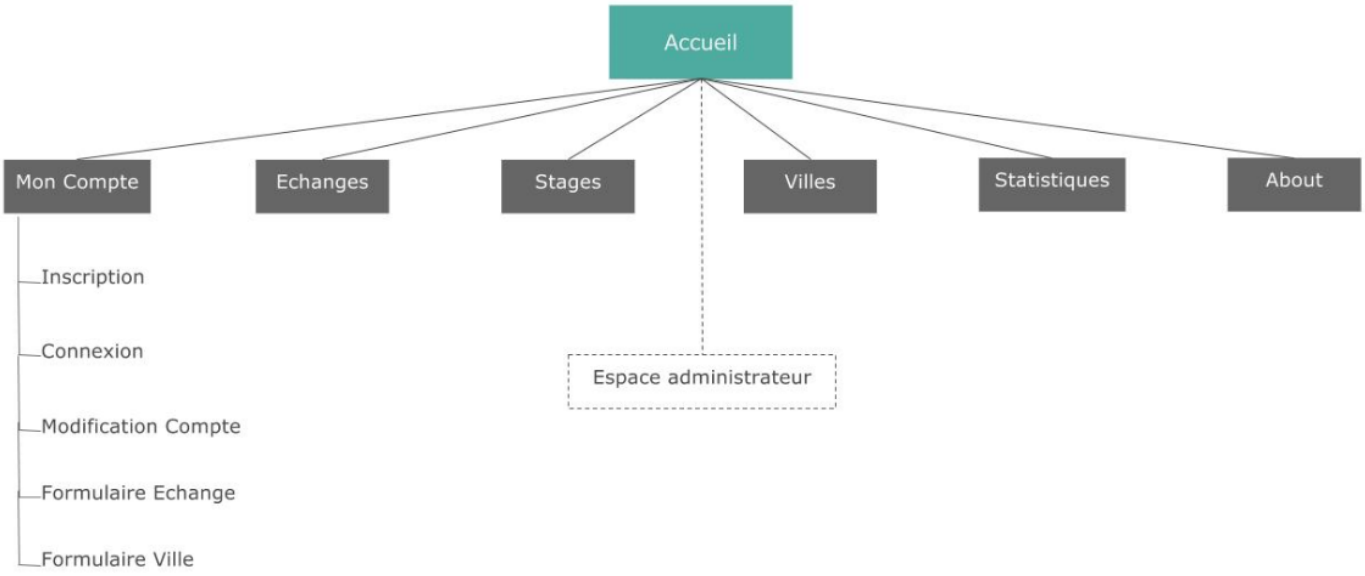
- contrats d'études liés à l'université qui l'intéresse
- cours intéressants et langues de ceux-ci
- contacts (étudiants) qui sont partis là-bas (nom, adresse-mail et téléphone si celui-ci est renseigné)
- s'informer sur la ville liée à l'université

Mais il peut aussi chercher des contacts d'étudiants partis en **stages** à l'étrangers dans une entreprise spécifique.

Cahier des Charges

Fonctions		Critères
Principales	Permettre à n'importe quel utilisateur d'avoir accès à la base de données répertoriant les contrats d'études et autres informations	C1 : Pas besoin de créer un compte
	Pouvoir créer un espace perso, rentrer ses informations et les modifier	C1 : Page d'inscription C2 : Modification de la base de données C3 : Connexion et déconnexion utilisateur
Secondaires	Créer un super-utilisateur	C1 : Créer un compte avec des fonctions différentes C2 : Pouvoir supprimer des comptes C3 : Pouvoir supprimer seulement des informations et non seulement un utilisateur (un peu radical)
	Sécuriser les données : protéger les données de la barre url	C1 : Ne pas avoir accès simplement aux mots de passes de chaque utilisateur
Cerise sur le gateau	Un joli design	C1 : Site agréable visuellement C2 : Facile à utiliser
	Donner des statistiques sur les TC partis à l'étrangers	C1 : Afficher des graphiques
	Signaler les champs obligatoires	C1 : Signaler par des astérisques

Arborescence du site



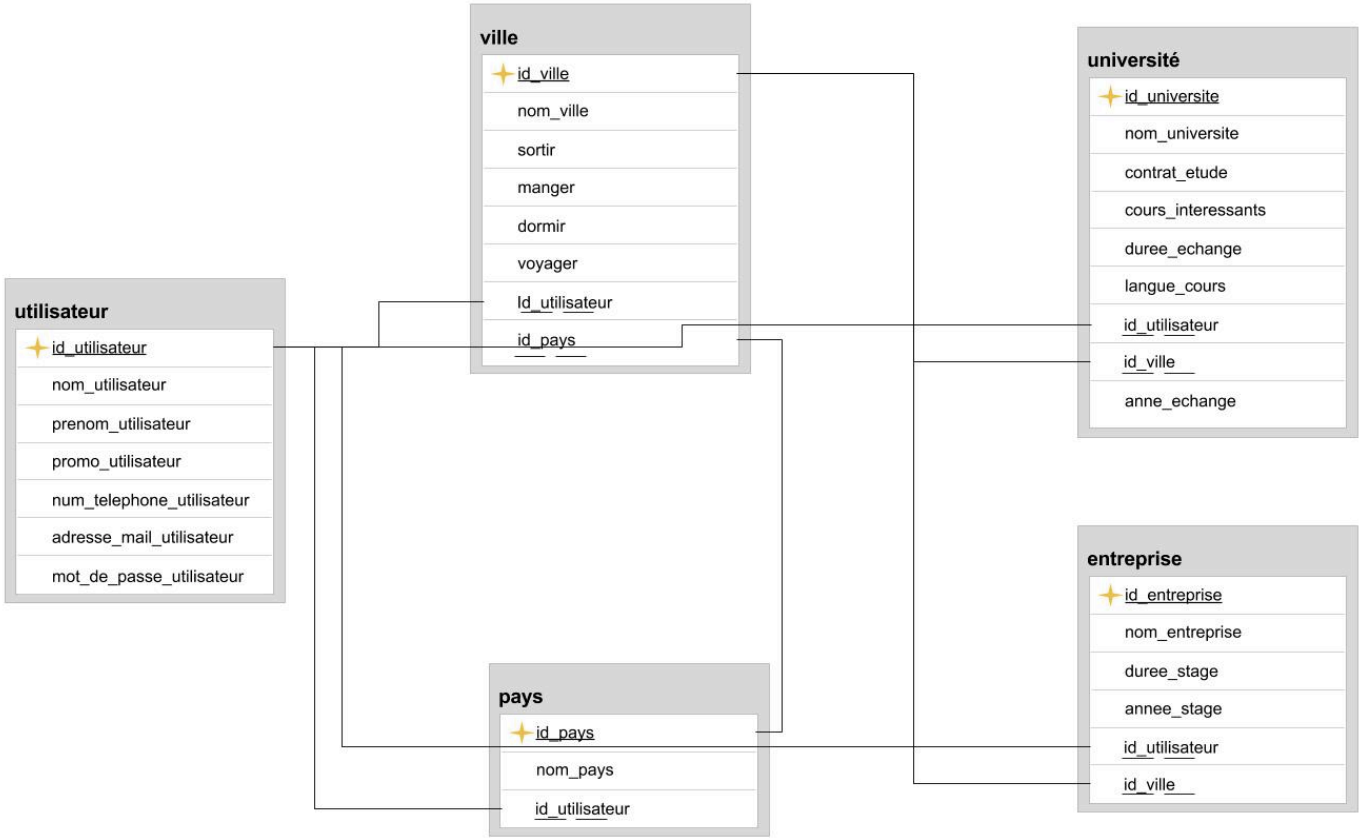
L'arborescence de notre site est assez simple.

D'un côté, nous avons l'espace utilisateur, qui concerne les étudiants déjà partis en échange ou en stage et qui viennent donner leurs informations. Après s'être inscrits, ils peuvent modifier leurs informations personnelles (dans le cas où par exemple ils changeraient d'adresse mail ou de numéro de téléphone), et poster les informations sur leur échange/stage puis sur la ville dans laquelle ils sont partis.
Toutes ces fonctions demandent une **inscription à l'utilisateur**. Ceci permet de lier un échange ou un stage à un profil et donc à un contact (numéro de téléphone, adresse mail). De plus, une fois connecté, il est possible de modifier ou d'ajouter des données sur un échange que nous avons réalisé.

L'autre partie du site est elle entièrement accessible **sans inscription**. Cette partie va permettre d'afficher, selon la recherche de l'utilisateur, les informations disponibles sur les échanges effectués dans une université, les stages effectués dans une entreprise, ou les bons plans sur une ville.
Dernière partie, l'espace administrateur, ~~qui~~ permet à un compte bien précis de pouvoir voir la liste des personnes inscrites sur le site et de pouvoir les bannir dans le cas ou certains s'amuseraient à poster des informations sans rapport.

Structure de la base de données

Voici le schéma de la base de données que nous avons utilisé:

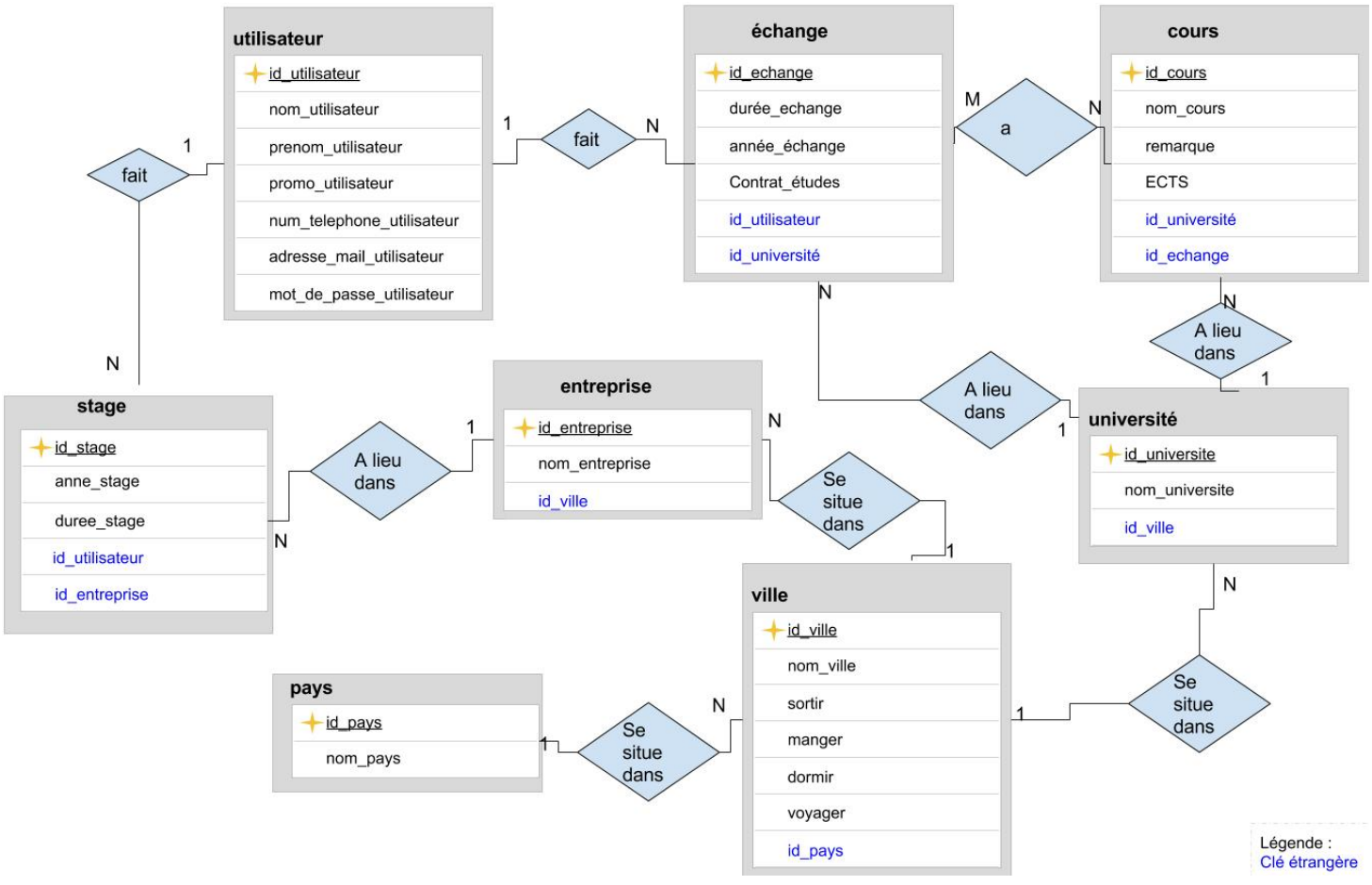


La table principale est la table **utilisateurs**: elle va conserver les informations relatives aux utilisateurs et qui seront affichées sur le site plus tard, telles que leurs noms/prénoms, leur promo, et leur mail/numéro de téléphone s'ils veulent être contactés par d'autres étudiants plus tard. La clé primaire est l'id_utilisateur, qui est unique à chaque utilisateur. Cette id est présente dans toutes les autres tables de notre base de données, nous expliquerons un peu en dessous pourquoi.
Dans la table **pays**, on va simplement stocker l'id des pays (clé primaire unique), et le nom du pays correspondant à cette id.
Cette table pays est directement reliée à la table **ville**: la table ville contient une clé étrangère (id_pays), qui permet de relier chaque ville au pays dans lequel elle appartient. La table va ensuite contenir d'autres informations propres à la ville, comme les endroits ou manger, dormir, sortir…
Cette table ville est elle aussi reliées à deux autres tables: **entreprise** et **université**, qui contiennent toutes les deux une clé étrangère vers id_ville, afin de savoir dans quelle ville se trouve une entreprise ou une université.

Toutes les tables contiennent donc une clé étrangère (id_utilisateur) reliée à la clé primaire de la table utilisateur. Elles n'ont pas pour seul but d'indiquer que cet utilisateur est parti dans cette université/entreprise (sinon il aurait suffi de mettre id_utilisateur uniquement dans les tables entreprise et université), mais elles permettent aussi de retrouver tous les ajouts dans la base de données qu'à fait chaque utilisateur.
En effet, nous avons pris le parti de ne pas imposer une liste fixe de pays / universités / entreprises à nos utilisateurs. Ainsi, en partant d'une base de données vide (lorsque le site

sera mis en ligne par exemple), il n'y aura aucun pays proposé dans la liste et les utilisateurs devront saisir eux même le nom du pays dans lequel ils sont partis. Par la suite, lorsque quelqu'un voudra remplir le même formulaire, le pays renseigné par la personne précédente lui sera proposé. La base de données va donc se construire petit à petit. Pour limiter les abus, le compte de l'ASTUS (superutilisateur) peut accéder à une page d'administration à partir de laquelle il peut voir les informations insérées dans la base par tous les utilisateurs, et supprimer un utilisateur et tous ses ajouts (grâce à un ON DELETE CASCADE).

Cette base de données aurait néamoins pu être améliorée, car elle ne répond pas aux critères de normalisation en vigueur. Cependant, lorsque nous avons réalisé que notre base de données n'était pas optimale, nous avions déjà avancé dans la construction du site. Aussi nous avons décidé de garder cette base de données pour passer du temps sur le reste du site. Voici le la version que nous avons prévu d'utiliser dans un futur proche. Elle respecte la forme normale 3 et chaque attribut d'une entité caractérise explicitement cette dernière.



Choix des langages de programmation

Pour mener à bien ce projet, nous avons principalement utilisé du PHP (en combinaison avec du MySQL), de l'HTML, du Javascript et du CSS .

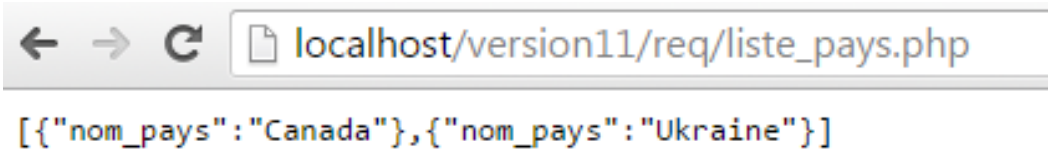
Pour le design et la structure des pages de notre site, nous avons choisi HTML et CSS. Ce choix nous a paru évident, et nous n'avons pas vraiment vu d'alternative viable à ces deux langages.

Le CSS nous a permis de faire un site au design assez simple et agréable à l'utilisation. Nous avons récupéré un template sur http://http://templated.co/, puis nous avons utilisé et modifié les éléments qui nous intéressaient. Nous avons fait un aspect général pour toutes les pages et d'une page à l'autre la structure du code HTML est assez similaire. Pour alléger l'écriture, nous avons également regroupé les sections paramètres à toutes les pages (onglets, sidebar, pied de page) dans des fichiers extérieurs. Ensuite sur chaque page, quelques divisions peuvent varier suivant que l'on se trouve sur une page d'affichage de données ou sur un formulaire.

Le javascript nous permet entre autres de dynamiser nos pages et de réaliser l'interfaçage entre le front-end (notre HTML) et le back-end, en récupérant des objets JSON. Nous avons principalement utilisé la bibliothèque Jquery, mais aussi d'autres fonctions de base du javascript, comme par exemple le placement des eventListener sur certains boutons, ou encore .show() ou .hide() sur certains éléments afin des les afficher/masquer lorsqu'il le fallait.
L'AJAX nous a également permis de récupérer dynamiquement (sans recharger la page) le résultat de requêtes SQL SELECT par exemple. Ceci nous a permis de pouvoir afficher de nouvelles données sans avoir à changer de page, comme nous aurions du le faire si nous avions seulement utilisé du PHP. Voici un exemple de code que nous avons utilisé, qui va récupérer le nom d'un pays selectionné par l'utilisateur et afficher les villes appartenant à ce pays:

```
function checkField(){
  var ville = document.getElementById("liste_villes").value;
  $.ajax({
            type: "POST",
            url: "liste_villes_utilisateur.php",
            data:{ ville: ville}})
    .done(function(data){
```

Pour le backend, nous avons choisi d'utiliser du PHP plutôt que python, principalement dans le but d'apprendre un nouveau langage. Le code PHP est séparé du code HTML, dans le but de séparer le front-end du back-end, et nos pages php renvoient des objets JSON. Voici par exemple ce que nous renvoie la page liste_pays.php, qui renvoie le nom de tous les pays présents dans notre base de données:



Nous n'avons en revanche pas réussi à gérer les utilisateurs (plus précisément leurs connexions/déconnexions) de cette manière, et nous avons utilisé des variables de session ($_SESSION[]) pour transmettre de page en page des informations propres à l'utilisateur connecté, telles que son id par exemple.
Concernant l'environnement de développement PHP, nous avons opté pour XAMPP (https://www.apachefriends.org/fr/index.html). Son gros point fort est qu'il est disponible sur tous les OS, ce qui nous a été très utile vu que nous codions sur Windows, Linux et OSX. Nous n'avons pas utilisé tous les outils proposés par XAMPP, mais certains (comme le serveur Apache) pourraient éventuellement se révéler utiles suivant l'évolution que prendra le site.

Pour simplifier l'écriture de ces pages, nous avons divisé notre dossier de travail en sous-dossiers regroupant un même types de fichiers. Nous avions donc un dossier dédié au javascript, un pour les fichiers exécutant les requêtes SQL, puis un dernier dossier contenant les pages du site à proprement parler. De façon secondaire, nous avont regroupé les blocs (onglets, sidebar, etc..), les images et le css dans trois autres dossiers distincts.
De cette façon, il nous a été plus simple de retrouver les fichiers désirés, et la maintenance du site sera également considérablement facilitée.