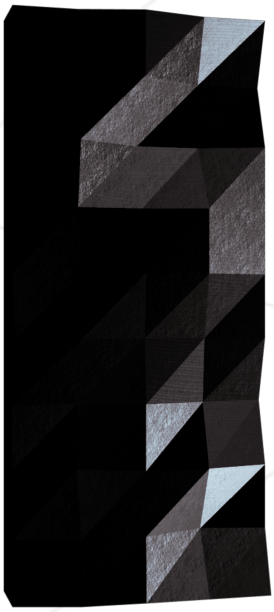

Web

Projet Hypertube

42 Staff pedago@staff.42.fr

Résumé: Une application web pour le 21^{ième} siècle.

Table des matières

Chapitre I

Préambule

Le mot “cafetière” désignait originellement la personne qui tenait un établissement où l'on pouvait boire du café.Une anecdote de Chamfort [1] ne permet pas de douter que le mot existait en ce sens au dix-huitième siècle puisqu'il nous montre un vieil évêque rigoriste consentant à donner vingt-cinq louis à son neveu qui souhaitait acquérir une “jolie cafetière”, et qui entre en fureur quand il constate que la cafetière porte jupon.

Lorsque la percolation du café est inventée, le mot “cafetière” désigne alors l'ustensile de cuisine utilisé pour préparer du café, tel qu'il est connu de nos jours.

Les cafetières, dans les années 1990, ont grandement contribué au développement des technologies modernes.

Par exemple, les chercheurs de l'université de Cambridge en Angleterre ont créé, en 1991, le premier prototype de webcam, dans le seul et unique but de surveiller le niveau de café restant dans la cafetière de la “trojan room” [2].

FIGURE I.1 – La cafetière de la Trojan Room

En 1998, The Internet Society publie un document de référence [3] concernant la gestion d'objets connectés de type “distributeurs de boisson chaude électroniques” [4], suivi directement de la référence du “Hyper Text Coffee Pot Control Protocol” (HTCPCP), définissant un protocole de communication client-serveur, extension du protocole HTTP, permettant le contrôle, la surveillance et le diagnostic de cafetière.

1. Caractères et Anecdotes no 1173
2. http://www.cl.cam.ac.uk/coffee/coffee.html
3. https://tools.ietf.org/html/rfc2325
4. Plus précisement : “Definitions of Managed Objects for Drip-Type Heated Beverage Hardware Devices using SMIv2”

Les requêtes sur une cafetière sont alors identifiées par le schéma d'URI `coffee://` (ou le nom du café dans l'une des 29 langues listées dans la RFC, incluant le français), et supportent les méthodes suivantes :

- `GET` récupère le café depuis la cafetière [5].
- `POST` (ou `BREW`) déclenche l'infusion du café par la cafetière.
- `PROPFIND` affiche des métadonnées sur le café.
- `WHEN` notifie la cafetière de stopper l'écoulement du lait dans le café (s'il y a lieu).

Des headers (champs d'en-tête) `Accept-Additions` peuvent être ajoutés pour préciser le type de lait, de sirop, d'édulcorant, d'épice ou encore d'alcool, pensons aux irlandais. [6]

Afin d'éviter tout accident bête, une théière se doit de renvoyer un code d'erreur 418 "I'm a teapot". L'absence de protocole bien défini pour ces dernières provoqua d'ailleurs une vive polémique parmi les amateurs de thé.

En 2014, Imran Nazar soumet à l'Internet Engineering Task Force la référence du "Hyper Text Coffee Pot Control Protocol for Tea Efflux Appliances" (HTCPCP-TEA) [7], une variante du HTCPCP pour les théières, mettant ainsi un terme aux hostilités.

Bien que ces RFC soient des poissons d'avril, elles sont suffisamment bien décrites pour être implémentées.

`À ce jour, aucun étudiant de 42 n'a réussi à créer une cafetière "HTCPCP compliant".`

5. Ce point est sujet à interprétation, voir la RFC
6. https://tools.ietf.org/html/rfc2324#section-2.2.2.1
7. https://tools.ietf.org/html/rfc7168

Chapitre II

Introduction

Ce projet vous propose de créer une application web permettant à un utilisateur de rechercher et visionner des vidéos.

Le lecteur sera directement intégré au site, et les vidéos seront téléchargées au travers du protocole BitTorrent.

Le moteur de recherche interrogera plusieurs sources externes de votre choix, comme par exemple http ://www.legittorrents.info, ou encore https ://archive.org.

Une fois un élément sélectionné, il sera téléchargé sur le serveur et diffusé sur le player web en même temps. Autrement dit, le lecteur ne se contentera pas d'afficher la vidéo une fois le téléchargement complété, mais sera capable de streamer directement le flux.

Chapitre III

Consignes générales

Pour ce projet, vous êtes libres d'utiliser le langage de votre choix.

Tous les framework, micro-framework, librairies etc... sont autorisés dans la limite où ils ne servent pas à créer un flux vidéo à partir d'un torrent, limitant ainsi l'interêt pédagogique du projet. Par exemple, des librairies comme `webtorrent`, `pulsar` ou `peerflix` sont interdites.

Vous êtes libres d'utiliser le serveur web de votre choix, que ce soit `Apache`, `Nginx` ou même un `built-in web server`.

L'ensemble de votre application devra être au minimum compatible sur `Firefox` (>= 41) et `Chrome` (>= 46).

Votre site doit avoir une mise en page décente : c'est à dire au moins un header, une section principale et un footer.

Votre site devra être présentable sur mobile, et garder une mise en page acceptable sur de petites résolutions.

Tous vos formulaires doivent avoir des validations correctes, et l'ensemble de votre site devra être sécurisé. Ce point est OBLIGATOIRE et sera vérifié longuement en soutenance. Pour vous faire une petite idée, voici quelques éléments qui ne sont pas considérés comme sécurisés :

- Avoir des mots de passe "en clair" dans une base de données.
- Pouvoir injecter du code HTML ou JavaScript "utilisateur" dans des variables mal protégées.
- Pouvoir uploader du contenu indésirable.
- Pouvoir modifier une requête SQL.

Chapitre IV

Partie obligatoire

Vous devez donc réaliser une application web ayant les fonctionnalités suivantes :

IV.1 Partie utilisateur

- L'application doit permettre à un utilisateur de s'inscrire, en demandant au minimum une adresse email, un nom d'utilisateur, une photo de profil, un nom, un prénom et un mot de passe un tant soit peu sécurisé.

- L'utilisateur doit pouvoir s'inscrire et se connecter via Omniauth. Vous devez donc obligatoirement implémenter au moins deux stratégies : la stratégie `42`, et une stratégie au choix.

- L'utilisateur doit ensuite être capable de se connecter avec son nom d'utilisateur et son mot de passe. Il doit également pouvoir recevoir un mail de réinitialisation de son mot de passe en cas d'oubli.

- L'utilisateur doit pouvoir se déconnecter en un seul clic depuis n'importe quelle page du site.

- L'utilisateur doit pouvoir sélectionner une langue préférée, qui sera par défaut l'anglais.

Un utilisateur devra également pouvoir :

- Modifier son adresse email, sa photo de profil et ses informations.

- Consulter le profil d'un autre utilisateur. C'est à dire afficher sa photo de profil, ses informations. L'email, en revanche, doit rester privé.

IV.2 Partie bibliotèque

`Cette partie ne doit être accessible qu'aux utilisateurs connectés.`

Cette partie doit présenter au minimum :

- Un champ de recherche.
- Une liste de miniatures.

IV.2.1 Recherche

Le moteur de recherche devra interroger au moins deux sources externes de votre choix [1], et retourner l'ensemble des résultats sous la forme de miniatures.

Vous devez limiter les résultats aux vidéos uniquement.

IV.2.2 Miniatures

Si une recherche a été faite, les résultats doivent s'afficher sous la forme d'une liste de miniatures, triées par nom.

Si aucune recherche n'a été faite, vous devrez afficher les médias les plus populaires de vos sources externes, triés selon le critère de votre choix (téléchargements, peers, seeders, etc...).

En plus du nom de la vidéo, une miniature doit être composée, si disponible, de son année de production, d'une note et d'une image de couverture.

Vous devrez différencier les vidéos vues des vidéos non-vues, de la manière de votre choix.

La liste devra être paginée, à chaque fin de page, la suivante doit être automatiquement chargée de manière asynchrone. Autrement dit, il ne doit pas y avoir de lien pour chaque page.

La liste devra être triable et filtrable selon des critères tels que le nom, le genre, un intervalle de note, un intervalle d'année de production, etc...

1. comme par exemple http://www.legittorrents.info, ou encore https://archive.org

IV.3 Partie vidéo

`Cette partie ne doit être accessible qu'aux utilisateurs connectés.`

Cette section devra présenter le détail d'une vidéo, c'est à dire afficher le player de la vidéo ainsi que - si disponible - le résumé, le casting (au moins producteur, réalisateur, acteurs principaux, etc...), l'année de production, la durée, la note, une image de couverture et tout ce qui vous semblerait pertinent.

Vous devez également donner aux utilisateurs la possibilité de laisser un commentaire sur la vidéo, et afficher la liste des commentaires précédents.

Lancer la vidéo sur le navigateur doit - si le fichier n'a pas déja été téléchargé précédemment - lancer le téléchargement du torrent associé sur le serveur, et streamer le flux vidéo depuis ce dernier dès que suffisament de données sont téléchargées pour assurer la lecture intégrale de la vidéo sans interruption. Bien évidemment, tout le traitement doit être fait en arrière plan de manière non-bloquante.

Une fois un film téléchargé intégralement, il doit être sauvegardé sur le serveur, de manière à ne pas re-télécharger un film plusieurs fois. Si un film n'est pas visionné pendant un mois, il devra être supprimé.

Si des sous-titres anglais sont disponibles pour cette vidéo, ils devront être téléchargés et sélectionnables sur le player vidéo. De même, si la langue de la vidéo ne correspond pas à la langue préférée de l'utilisateur, et que des sous-titres sont disponibles pour cette langue, ils devront également être téléchargés et sélectionnables.

Si la vidéo n'est pas nativement lisible pour le navigateur [2], vous devrez la convertir à la volée dans un format acceptable. Le support du format `mkv` est un minimum.

2. C'est à dire que ce n'est ni du `mp4`, ni du `webm`.

Chapitre V

Partie bonus

Si la partie obligatoire a été réalisée entièrement et parfaitement, vous pouvez ajouter les bonus que vous souhaitez ; ils seront évalués à la discrétion de vos correcteurs. Vous devez néanmoins toujours respecter les contraintes de base. Par exemple, le téléchargement d'un torrent devra rester coté serveur, en arrière plan.

Si l'inspiration vous manque, voiçi quelques pistes :

- Ajouter des stratégies Omniauth supplémentaires.
- Gérer différentes résolutions de vidéo.
- Développer une API RESTful.
- Streamer la vidéo via l'API MediaStream.

Chapitre VI

Rendu et peer-évaluation

Les consignes suivantes seront présentes dans le barème de soutenance. Soyez très attentifs lors de l'application de ces dernières car elles seront sanctionnées par un 0 sans appel.

VI.1 Consignes éliminatoires

- Votre code ne doit produire aucune erreur, warning ou notice, coté serveur et coté client, dans la console web.
- Tout ce qui n'est pas explicitement autorisé est interdit.
- La moindre faille de sécurité entrainera un 0. Vous devez au minimum gérer ce qui est indiqué dans les consignes générales, c'est à dire ne pas avoir de mot de passe en clair, être protégé contre les injections SQL, et avoir une validation de tous les formulaires de saisie et d'upload.
- Enfin, vous devez rendre, à la racine de votre dépôt de rendu, un fichier `auteur` contenant vos logins, à raison d'un par ligne, de cette façon :

```
$>cat -e auteur
xlogin$
ylogin$
zlogin$
alogin$
$>
```

Vous pouvez poser vos questions sur le forum, Slack, etc...

Bon courage à tous !