UNIVERSITÉ DE LORRAINE | IUT nancy Charlemagne

AS informatique
Technologies du Web

Projet — Réalisation d'un site de réduction d'URL
2 avril 2014

Ce projet PHP, à réaliser de préférence par binôme, devra être rendu pour le 25 avril à 22h00. Le soin apporté et l'architecture déployée seront des éléments d'appréciation.

1 Énoncé général

On vous demande de réaliser un site de réduction d'URL en PHP disposant de fonctionnalités minimales. Des sites comme `http://www.goo.gl` ou `http://www.bit.ly` devraient vous permettre de comprendre assez rapidement ce qui est attendu[1]. Vous êtes incités à les utiliser un peu afin de bien comprendre la logique de ce qui vous est demandé dans le cadre de ce projet.

Sur le fond, l'objectif est de concevoir un site Web permettant de réduire les URL, ainsi que d'assurer les redirections des URL réduites produites vers les URL originales. Voici les grandes lignes des fonctionnalités attendues :

1. La page d'accueil du site doit permettre la saisie d'une URL et, après validation, générer une URL réduite permettant d'accéder à la même ressource que l'URL originale.
2. Le site doit permettre de répondre aux requêtes correspondant à ces URL réduites et ainsi assurer la redirection vers les URL originales.
3. Les URL réduites produites doivent perdurer dans le temps. A priori, une URL réduite produite n'a pas à être supprimée par la suite.
4. Le site à construire doit pouvoir fonctionner pour des visiteurs anonymes.
5. Il doit également être possible de se créer un compte sur le site. Dans ce cas, lorsque le site est utilisé par un membre connecté :
   - les URL réduites générées doivent être associées au compte du membre connecté,
   - un membre doit pouvoir accéder facilement à toutes ses URL réduites créées,
   - les URL réduites créées doivent être associées à un certain nombre d'informations et d'opérations : l'URL originale, la date de création de l'URL réduite, le nombre d'utilisations de l'URL réduite, des statistiques sur ces utilisations, un état sur la validité de l'URL originale, la possibilité de supprimer l'URL réduite.

2 Analyse sommaire

Les besoins de représentation de données sont relativement simples, il s'agit d'être capable de stocker :
- Des utilisateurs, définis par un nom, un prénom, un pseudo, un mot de passe et une adresse mail.
- Des ressources, définies par une URL originale, une URL réduite, une date de création, les dates d'utilisation.

[1] Attention toutefois, ce qui est demandé dans le cadre de ce projet diffère du fonctionnement de ces sites sur certains points. Lisez donc bien l'énoncé pour comprendre **ce qui vous est demandé exactement**. Les sites donnés en exemple n'ont pour but que de vous faire comprendre le fonctionnement de ce genre de sites.

3 Contraintes d'utilisation

1 Généralités

Le projet en lui-même doit rester relativement simple et ne comportera que les opérations élémentaires liées à ce type de réalisation. Les opérations qu'il faut implanter dans le cadre de ce projet consistent simplement à pouvoir :

- s'enregistrer, se connecter et se déconnecter sur le site,
- créer une nouvelle ressource, supprimer une ressource existante,
- consulter les ressources existantes et pouvoir avoir des informations statistiques sur leurs utilisations.

2 Détails des contraintes

- La réduction d'une URL doit pouvoir se faire simplement et rapidement : cette opération doit être disponible depuis la page d'accueil du site et nécessiter un minimum d'interactions avec l'utilisateur (le strict minimum donc).
- Les URL courtes générées doivent **obligatoirement** être présentées sur le site sous forme de liens hypertextes.
- Les création et les utilisations de ressources peuvent se faire sans se connecter au site. En revanche, il est nécessaire d'être connecté pour pouvoir administrer et accéder aux statistiques d'une ressource donnée.
- Les membres connectés doivent pouvoir accéder à des opérations d'administration et de consultation des ressources qu'ils ont créées.
- Le site doit par ailleurs être doté d'opérations de maintenance (au niveau des membres, des ressources) disponibles uniquement pour le ou les administrateurs du site.
- Le premier utilisateur créé sera **obligatoirement** un administrateur. Par la suite, il sera possible de créer de nouveaux administrateurs par l'intermédiaire d'options spécifiques proposés aux administrateurs, qui devront ainsi pouvoir promouvoir n'importe quel membre à ce statut.
- Si une ressource est créée par un membre *connecté*, elle doit apparaître dans la liste des ressources de ce membre.
- Un membre doit pouvoir rester connecté, s'il le souhaite, même si son navigateur a été arrêté depuis sa dernière visite.
- Il doit pouvoir être possible de créer, pour un ou plusieurs membres, plusieurs URL réduites correspondant à la même URL originale.
- Les URL réduites créées doivent naturellement être uniques (une URL réduite donnée ne doit pas pouvoir correspondre à plusieurs URL originales).
- Le site doit interdire la création d'URL réduites correspondant à d'autres URL réduites du même site.
- Le site devra proposer sur sa page d'accueil la possibilité de s'inscrire ou de se connecter.
- Une fois connecté, un membre devra pouvoir gérer ses ressources par l'intermédiaire d'un menu.
- Le site doit être personnalisé autant que possible. En particulier, les liens permettant d'effectuer des actions ne devront être présentés que si l'action correspondante est disponible et possible pour l'utilisateur courant (membre, administrateur, anonyme…)
- Des graphiques statistiques devront être générés pour les membres connectés, sur une ou plusieurs caractéristiques des URL raccourcies. Vous devrez utiliser l'API proposée par Google, dont la documentation est disponible sur `https://google-developers.appspot.com/chart/`.

4 Contraintes techniques

- Toutes les informations permettant la connexion à la base de données doivent être personnalisables : nom de la base, nom de l'utilisateur, mot de passe. Ces informations doivent pouvoir être modifiées en un **unique** endroit à l'intérieur de votre projet (typiquement un fichier de configuration) que vous indiquerez. Vous positionnerez **obligatoirement** les valeurs suivantes par défaut lors de la soumission de votre projet :
  - Hôte : `localhost`

  - Nom de base : `racurl`
  - Utilisateur : `racurluser`
  - Mot de passe : `racurlpwd`
- Votre projet sera installé pour être testé. Cette installation consistera en une simple décompression dans un répertoire donné. Attention donc :
  - votre projet ne doit pas requérir d'autre manipulation : aucun fichier n'appartenant pas à votre projet ne doit être configuré pour le faire marcher…
  - votre projet doit fonctionner quel que soit le répertoire utilisé pour l'installation : prenez garde à n'utiliser que des URL relatives dans votre projet…
- Vous êtes vivement encouragés à développer ce projet en utilisant les classes et objets PHP correspondant à la syntaxe PHP 5. À vous de voir quelles classes sont pertinentes dans ce le cadre de ce projet.
- Vous devrez utiliser PDO ou (optionnellement) un framework PHP (symfony, cakephp…) Vous ne devrez donc pas utiliser d'interface native de connexion à un SGBD particulier, quel qu'il soit. Attention à l'installation : votre projet doit être auto-suffisant lors de cette étape. Pour rappel, le correcteur n'installera rien de plus que votre projet sur sa machine pour la correction…
- La correction aura *a priori* lieu sur une machine installée avec Apache 2.2.22 et PHP 5.4.6. Aucune autre configuration ne sera déployée, les fichiers de configuration de ces logiciels ne seront pas modifiés. Vous avez toutefois la possibilité d'inclure un fichier `.htaccess` à votre archive au besoin.

5 Réalisation

1. Créer une base de données SQL appelée `racurl` permettant d'implanter les besoins de ce projet. Trois tables sont suffisantes (la clé primaire s'appelle systématiquement `id`, les clés étrangères sont présentées en *italiques*) :
   - `membres` : `id, nom, prenom, pseudo, mail, mdp`[2] , `activation`[3] , profil[4]
   - `urls` : `id, source`[5] , `courte`[6] , `creation`[7] , *`auteur`*[8]
   - `utilisations` : `id, `*`url`*[9] , `date`[10]

   Vous devrez utiliser ce schéma à défaut de tout autre : **il est interdit d'y faire des modifications**. Une implémentation de ce schéma de base, réalisée sous MySQL, est présente en annexe.
2. Écrire des scripts PHP permettant de créer un nouveau membre ainsi que de permettre aux membres existants de se connecter.
3. Écrire des scripts permettant à un administrateur de gérer les membres du site (visualisation, modification, suppression).
4. Écrire des scripts PHP permettant de créer une URL réduite à partir d'une URL originale (tenir compte de l'auteur pour ces scripts s'il correspond à un membre connecté).
5. Écrire des scripts permettant à un administrateur de gérer les URL du site (visualisation, modification, suppression, statistiques d'utilisation).
6. Écrire des scripts PHP permettant de rediriger une requête initiée à partir d'une URL réduite vers l'URL originale correspondante, tout en mettant à jour les structures de données relatives aux statistiques.
7. Enfin, écrire des scripts PHP permettant de gérer les URL créées par des membres connectés (visualisation, suppression, statistiques d'utilisation).
8. Des contrôles sur les données fournies devront être implantés pour garantir une bonne sécurité de l'application (des éléments seront fournis à ce propos lors du dernier cours, mais vous pouvez vous renseigner auparavant à ce sujet et commencer à implanter quelque chose).

[2] Mot de passe crypté
[3] Permettant de gérer l'activation des comptes utilisateur si cette fonctionnalité est implantée (des détails seront fournis en cours)
[4] Permettant de distinguer les utilisateurs « normaux » des administrateurs
[5] URL originale
[6] URL raccourcie
[7] Date et heure de création de cette URL
[8] Identifiant du membre, le cas échéant, ayant créé cette ressource
[9] Identifiant de l'URL concernée
[10] Date et heure d'utilisation

9. Ajouter quelques éléments de styles pour rendre agréable la navigation sur votre site.

6 Comment rendre le projet ?

Votre projet va faire l'objet d'un traitement automatisé. Merci de suivre ***scrupuleusement*** *les consignes suivantes. Des points seront automatiquement ôtés si une ou plusieurs de ces consignes ne sont pas suivies.*

1 Consignes générales

1. Votre projet devra obligatoirement être rendu sur l'ENT. Une section « Remise de projet », présente dans le cours en ligne correspondant à ce cours, sera disponible jusqu'au 25 avril 2014 à 22h00. Si vous avez déjà déposé un projet, il vous sera possible d'en déposer une ou plusieurs versions modifiées jusqu'au délai final. **Attention**, que vous ayez ou pas déposé votre projet, il sera **impossible** de déposer un projet passé le délai final. À vous de prendre vos dispositions.
2. **Aucun fichier ou répertoire** de votre projet ne doit contenir, **dans son nom**, des caractères tels que des espaces, des caractères accentués ou des caractères de ponctuation (à l'exclusion des caractères « `_` `-` `.` »).
3. Votre projet devra être une archive, **obligatoirement** dans l'un des formats suivants (à l'exclusion de tout autre) : `zip`, `rar`, `tar.gz`, `tgz`. Votre choix de format est désigné ci-dessous par l'`extension`.
4. Le nom de votre archive devra **obligatoirement** suivre l'une des syntaxes suivantes :
   — une seule personne : `TP_Nom.extension` (exemple : `TP_Dupont.tar.gz`)
   — deux personnes : `TP_Nom1_Nom2.extension` (exemple : `TP_Durand_Dupont.zip`)
   — trois personnes : `TP_Nom1_Nom2_Nom3.extension` (exemple : `TP_Durand_Dupont_Martin.zip`)
   Si un nom de famille est composé, ne prenez en compte que le premier mot du nom (sauf s'il s'agit d'une particule, que vous pouvez concaténer au nom — exemple : `TP_DeLaFontaine.zip`). Si un nom de famille contient une lettre accentuée, remplacez là par la lettre non accentuée correspondante.
5. Vous avez le choix, dans votre archive, de :
   — placer directement les fichiers et répertoires de votre projet à la racine,
   — placer un unique répertoire à la racine (aucun nom imposé), contenant les fichiers et répertoires de votre projet.

2 Consignes propres au projet

1. Votre archive **devra** contenir :
   — un fichier `index.php` à la **racine**, point d'entrée de votre site,
   — un unique fichier contenant toutes les informations nécessaires à la configuration de la base de données (à appeler `config.php` typiquement, sauf si l'usage d'un *framework* vous oblige à le nommer différemment — dans ce cas, consignez le nom dans le fichier `readme.txt`),
   — un fichier `readme.txt`, que vous pourrez utiliser pour me signaler tout point que vous souhaitez porter à ma connaissance,
   — tous les fichiers PHP, HTML, CSS, JPG… strictement nécessaires au bon fonctionnement de votre site.
2. Votre archive **pourra** contenir :
   — un fichier `tests.sql`, contenant des données de test conformes au schéma de base et permettant de peupler la base à des fins d'exemple.
3. Votre archive **ne devra pas** contenir :
   — de fichier `.sql`, contenant le schéma de base utilisé : vous devez utiliser obligatoirement le schéma fourni dans cet énoncé,
   — les fichiers de sauvegarde (les fichiers `~`, les fichiers `.bak`).

Vous avez fini votre projet ? Vous vous apprêtez à le rendre ? Merci de relire l'énoncé et de vérifier que toutes les contraintes qui vous ont été données ont bien été respectées. Chaque contrainte est associé à un nombre de points dans la notation...

7 Annexe : schéma de base sous MySQL

Ce fichier peut-être récupéré sur l'ENT.

```
-- phpMyAdmin SQL Dump
-- version 3.4.11.1deb1
-- http://www.phpmyadmin.net
--
-- Client: localhost
-- Généré le: Mar 02 Avril 2013 à 12:51
-- Version du serveur: 5.5.29
-- Version de PHP: 5.4.6-1ubuntu1.2

SET SQL_MODE="NO_AUTO_VALUE_ON_ZERO";

--
-- Base de données: `racurl`
--

-- --------------------------------------------------------

--
-- Structure de la table `membres`
--

DROP TABLE IF EXISTS `membres`;
CREATE TABLE IF NOT EXISTS `membres` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `nom` varchar(64) NOT NULL,
  `prenom` varchar(64) NOT NULL,
  `pseudo` varchar(64) NOT NULL,
  `mail` varchar(64) NOT NULL,
  `mdp` varchar(40) NOT NULL,
  `activation` varchar(40) NOT NULL,
  `profil` varchar(16) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `pseudo` (`pseudo`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 AUTO_INCREMENT=1 ;

-- --------------------------------------------------------

--
-- Structure de la table `urls`
--

DROP TABLE IF EXISTS `urls`;
CREATE TABLE IF NOT EXISTS `urls` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
```

```
  `source` varchar(1024) NOT NULL,
  `courte` varchar(10) NOT NULL,
  `creation` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `auteur` int(11) DEFAULT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `courte` (`courte`),
  KEY `auteur` (`auteur`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 AUTO_INCREMENT=1 ;

-- --------------------------------------------------------

--
-- Structure de la table `utilisations`
--

DROP TABLE IF EXISTS `utilisations`;
CREATE TABLE IF NOT EXISTS `utilisations` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `url` int(11) NOT NULL,
  `date` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`),
  KEY `url` (`url`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 AUTO_INCREMENT=1 ;

--
-- Contraintes pour les tables exportées
--

--
-- Contraintes pour la table `urls`
--
ALTER TABLE `urls`
  ADD CONSTRAINT `urls_ibfk_1` FOREIGN KEY (`auteur`) REFERENCES `membres` (`id`) ON DELETE
      SET NULL ON UPDATE NO ACTION,
  ADD CONSTRAINT `urls_ibfk_2` FOREIGN KEY (`auteur`) REFERENCES `membres` (`id`) ON DELETE
      SET NULL ON UPDATE NO ACTION;

--
-- Contraintes pour la table `utilisations`
--
ALTER TABLE `utilisations`
  ADD CONSTRAINT `utilisations_ibfk_1` FOREIGN KEY (`url`) REFERENCES `urls` (`id`) ON DELETE
       CASCADE ON UPDATE NO ACTION,
  ADD CONSTRAINT `utilisations_ibfk_2` FOREIGN KEY (`url`) REFERENCES `urls` (`id`) ON DELETE
       CASCADE ON UPDATE NO ACTION;
```