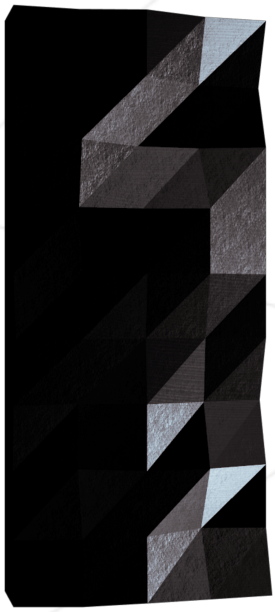

Rush01

Dernière ligne droite !

Vincent Montécot vmonteco@student.42.fr

Résumé: Il est temps de vous frotter à un vrai projet !

Table des matières

Chapitre I

Consignes

- Seule cette page servira de référence : ne vous fiez pas aux bruits de couloir.
- Le sujet peut changer jusqu'à une heure avant le rendu.
- Si aucune information contraire n'est explicitement présente, vous devez partir du principe que les versions des langages et framework utilisés sont les suivantes (ou ultérieures) :
  - `Python 3`
  - `HTML5`
  - `CSS 3`
  - `Javascript EM6`
  - `Django 1.9`
  - `psycopg2 2.6`
- Sauf indication contraire dans le sujet, les fichiers en python de chaque exercice sur `Python` seul (d01, d02 et d03) doivent comporter à leur fin un bloc `if __name__ == '__main__'`: afin d'y insérer le point d'entrée dans le cas d'un programme, ou des tests dans le cas d'un module.
- Sauf indication contraire dans le sujet, chaque exercice des journées portant sur `Django` aura sa propre application dans le projet à rendre pour des raisons pédagogiques.
- Les exercices sont très précisément ordonnés du plus simple au plus complexe. En aucun cas nous ne porterons attention ni ne prendrons en compte un exercice complexe si un exercice plus simple n'est pas parfaitement réussi.
- Attention aux droits de vos fichiers et de vos répertoires.
- Vous devez suivre la procédure de rendu pour tous vos exercices : seul le travail présent sur votre dépot GIT sera évalué en soutenance.
- Vos exercices seront évalués par vos camarades de piscine.
- Vous ne devez laisser dans votre répertoire aucun autre fichier que ceux explicitement specifiés par les énoncés des exercices.
- Sauf indication contraire dans le sujet vous ne devez pas inclure dans votre rendu :

  - Les dossiers `__pycache__`.
  - Les éventuelles migrations.
    Attention, il vous est tout de même conseillé de rendre le fichier `migrations/__init__.py`, il n'est pas nécessaire mais simplifie la construction des migrations.
    Ne pas ajouter ce fichier n'invalidera pas votre rendu mais vous *devez* être capables de gérer vos migrations en correction dans ce cas.
  - Le dossier créé par la commande `collectstatic` de `manage.py` (avec pour chemin la valeur de la variable `STATIC_ROOT`).
  - Les fichier en bytecode Python (Les fichiers avec une extension en `.pyc`).
  - Les fichiers de base de donnée (notamment avec `sqlite`).
  - Tout fichier ou dossier devant ou pouvant être créé par le comportement normal du travail rendu.

  Il vous est recommandé de modifier votre `.gitignore` afin d'éviter les accidents.

- Lorsque vous devez obtenir une sortie précise dans vos programmes, il est bien entendu interdit d'afficher une sortie précalculée au lieu de réaliser l'exercice correctement.
- Vous avez une question ? Demandez à votre voisin de droite. Sinon, essayez avec votre voisin de gauche.
- Votre manuel de référence s'appelle `Google / man / Internet / ...`.
- Pensez à discuter sur le forum Piscine de votre Intra !
- Lisez attentivement les exemples. Ils pourraient bien requérir des choses qui ne sont pas autrement précisées dans le sujet...
- Par pitié, par Thor et par Odin ! Réfléchissez nom d'une pipe !

Chapitre II

Règles spécifiques de la journée

- Vous pourrez-utiliser pour cette journée les paquets suivants (plus vos propres paquets à construire et installer lors de la correction) :

```
asgi-redis==0.14.1
asgiref==0.14.0
autobahn==0.16.0
channels==0.17.3
daphne==0.15.0
Django==1.9.5
django-bootstrap3==7.1.0
gunicorn==19.6.0
msgpack-python==0.4.8
Pillow==3.4.1
redis==2.10.5
six==1.10.0
Twisted==16.4.1
txaio==2.5.1
uWSGI==2.0.14
zope.interface==4.3.2
```

- Vous pouvez également utiliser JQuery et Nginx comme bon vous semble.
- Vous êtes libres d'organiser votre projet comme bon vous semble.
- N'oubliez pas votre fichier de dépendances `pip`.
- Lors de la correction le correcteur pourra tout à fait modifier le code côté navigateur pour tenter de mettre en échec votre rendu. Vous devez prendre cela en considération pour rendre votre projet suffisamment robuste.
- Votre rendu sera cloné avec la commande `git clone <repo> ~/rush01`. Vous devez donc faire ensemble que votre projet fonctionne à cet emplacement.

Chapitre III

Préambule

Voici une sélection de prix Ig Nobel remis ces dernières années :

Prix de probabilités 2013 : à Bert Tolkamp, Marie Haskell, Fritha Langford, David Roberts et Colin Morgan, pour avoir découvert que la probabilité qu'une vache se lève augmente avec le temps durant lequel elle reste couchée, et qu'il est au contraire quasiment impossible de prédire à quel moment elle se recouchera.

Prix de la paix 2013 : à Alexander Lukashenko, président de Biélorussie, pour avoir rendu illégal l'applaudissement en public et à la police Biélorusse, pour avoir arrêté un unimanchiste, accusé de ce délit.

Prix de Psychologie 2013 : à Laurent Bègue, Brad Bushman, Oulmann Zerhouni, Baptiste Subra et Medhi Ourabah pour avoir expérimentalement confirmé que les personnes pensant être saoules pensent également être attirantes.

Prix de Physique 2014 : à à Kiyoshi Mabuchi, Kensei Tanaka, Daichi Uchijima et Rina Sakai pour avoir étudié les coefficients de frottement entre le sol, une peau de banane et une semelle de chaussure.

Prix de Neurosciences 2014 : à Jiangang Liu, Jun Li, Lu Feng, Ling Li, Jie Tian et Kang Lee, pour avoir tenté de comprendre comment certaines personnes voient le visage de Jésus dans un morceau de pain grillé.

Prix de Physique 2015 : à Patricia Yang et David Hu pour avoir démontré que les mammifères de plus de 3kg vidaient leur vessie en 21s (+/- 13s).

Prix de Biologie 2015 : à Bruno Grossi, Omar Larach, Mauricio Canals, Rodrigo A. Vásquez et José Iriarte-Díaz, pour avoir observé qu'un poulet avec un bâton fixé au croupion adoptait une démarche similaire à celle qu'ont sans doute eue les dinosaures non-aviens.

Prix de Littérature 2015 : à Mark Dingemanse, Francisco Torreira et Nick J. Enfield pour avoir découvert que le mot "huh" existait partout sans qu'on ne sache vraiment pourquoi.

Chapitre IV

Partie obligatoire

Vous voici arrivés à ce deuxième rush qui clôturera ces deux semaines de piscine.

C'est l'occasion pour vous de démontrer vos compétences avec Django à travers la réalisation d'un intranet.

IV.1 Projet de base

Vous devez commencer par réaliser un projet de base avec Django comportant plusieurs parties, parmis lesquelles :

IV.1.1 Une configuration HTTP

Une configuration de serveur pour servir votre projet avec Nginx (Vous ne devez *PAS* utiliser le serveur de développement, `DEBUG` *DOIT* être à `False`).

L'ensemble des fonctionnalités doit fonctionner avec cette configuration. (Si une fonctionnalité fonctionne avec `./manage.py runserver` mais pas avec `Nginx`, cette fonctionnalité n'est pas valide).

IV.1.2 Un système d'authentification

Seules seront accessibles hors connexion :

- Une page d'inscription.

  Le formulaire de cette page doit comprendre un champ pour le nom d'utilisateur et deux champs pour le mot de passe (qui ne doit pas apparaître lorsqu'il est tapé). La validation nécessitera que le nom d'utilisateur ne soit pas déjà utilisé et que les valeurs entrées pour le mot de passe soient identiques.

  Une fois la validation effectuée, un utilisateur est créé avec les données entrées et activé. Le visiteur est alors connecté avec cet utilisateur et est redirigé vers la page d'accueil.

Le mot de passe ne doit évidemment pas être stocké en clair en base de donnée.

- Une page de connexion.

  Le formulaire de cette page doit comprendre un champ pour le nom d'utilisateur et un champ pour le mot de passe (qui ne doit pas apparaître lorsqu'il est tapé). La validation nécessitera que l'utilisateur existe, et que l'authentification avec lui soit possible.

  Une fois le formulaire validé, le visiteur est connecté avec cet utilisateur et est redirigé vers la page d'accueil.
  Une fois connecté, l'utilisateur doit être redirigé vers la page d'accueil.

Une fois connecté, ces pages ne sont plus accessibles. L'utilisateur aura alors accès à un bouton de déconnexion.

IV.1.3 Une page d'accueil

Une page d'accueil permettant d'accéder à l'ensemble des fonctionnalités que vous pourrez implémenter.

IV.1.4 Gérer des profils utilisateurs

Vous devez associer à chaque utilisateur certaines informations parmis lesquelles :

- Son nom.
- Son prénom.
- Son email.
- Sa description.
- Sa photo de profil.

Vous devez proposer sur la page de profil de chaque utilisateur la possibilité (pour cet utilisateur et pour un administrateur) de consulter et d'éditer ces informations.

Les administrateurs doivent également avoir la possibilité d'attribuer ou de retirer les droits d'administration à un utilisateur (Il doit donc pouvoir savoir si un utilisateur donné a les droits d'administration).

Un administrateur peut très bien être un superutilisateur, un utilisateur avec un ensemble de permissions ou faisant partie d'un groupe. À votre convenance.

IV.2 Fonctionnalité au choix

Une fois ce projet fait, vous implémenterez au moins une des deux fonctionnalités suivante (si vous faites les deux vous aurez des points supplémentaires).

IV.2.1 Forum

Ce forum, bien que simple, doit implémenter un système de posts.

Ces posts devront être listés par 10 sur une page (faites apparaître un système de pagination lorsque les posts sont trop nombreux).

Des liens vers le détail des posts listés seront présents.

Un post devra avoir un titre, un auteur, un contenu, une date et une heure de création ainsi qu'un ensemble de commentaires qui lui seront associés.

Les commentaires auront un contenu, une date et une heure de création ainsi qu'un auteur.

Le détail d'un post affichera l'ensemble des informations de ce post ainsi que l'ensemble de ses commentaires (et leurs informations respectives également).

Il devra être possible de commenter un post comme il sera possible de commenter un commentaire (Vos commentaires seront organisés sous forme d'arbre). L'affichage indiquera clairement de quoi un commentaire est le commentaire (du post ou de tel ou tel commentaire).

Tout utilisateur doit avoir la possibilité de créer des posts et de commenter.

IV.2.2 Un système de messagerie

Votre système de messagerie devra permettre à un utilisateur de tenir une conversation avec d'autres utilisateurs en échangeant des messages.

Une conversation se fera toujours entre deux utilisateurs *différents* (Et le contenu de cette conversation n'est accessible qu'à ces utilisateurs seuls).

Votre système de messagerie comprendra deux pages :

- Une page listant les conversations *démarrées* impliquant l'utilisateur courant. Vous devez afficher les informations du dernier message reçu/émis (date/heure, contenu tronqué à 30 caractères, interlocuteur...) pour chaque conversation.

  Affichez les par 10 et faites apparaître un système de pagination si les messages sont trop nombreux pour être affichés sur une seule page.

implémentez également pour chaque conversation listée un lien vers le détail de celle-ci.

Ajoutez également un lien pour démarrer une conversation avec un utilisateur *différent* sur sa page de profil (Dans le cas où la conversation n'existerait pas encore).

- Le détail de chacune des conversations.

  Vous devrez y lister l'ensemble des messages échangés par ordre chronologiques avec leur contenu ainsi que la date et l'heure de leur émission.

  L'affichage doit également indiquer clairement si un message a été reçu ou envoyé.

  L'utilisateur avec lequel se tient la conversation doit également apparaître.

Chapitre V

Partie facultative

Une fois la partie obligatoire réalisée, vous pouvez réaliser des bonus parmi les suivants :

V.0.1 Bonus Forum : implémentation de l'`AJAX`

Recharger la page ne doit pas être nécessaire pour afficher un formulaire pour commentaire, le remplir, le soumettre et voir les changements s'afficher après leur validation par le serveur, puis recommencer.

Bien entendu, ce bonus n'est faisable que si vous avez réalisé le forum.

V.0.2 Bonus messagerie : utilisation de websockets

L'utilisation de websockets dans votre système de messagerie :

Les messages vous étant envoyés par un autre utilisateur doivent apparaître instantannément dans le fil de conversation sans avoir à recharger la page.

Bien entendu, ce bonus n'est faisable que si vous avez implémenté un système de messagerie.

V.0.3 Push notifications

Vous devez, à l'aide des websockets, implémenter un système de push notifications, deux cas peuvent se présenter :

- Vous devez faire apparaître une notification lorsqu'un nouveau post est créé si vous avez fait le forum. Cette notification doit au moins faire apparaître le titre du post tronqué à 30 caractères.

- Vous devez faire apparaître une notification lorsqu'un nouveau message vous est envoyé si vous avez réalisé le système de messagerie. Vous devez au moins afficher

le nom de l'expéditeur et le contenu du message tronqué à 30 caractères.

Si vous avez validé à la fois le système de messagerie et le forum, traîter un seul des deux cas suffit pour valider ce bonus.

V.0.4 SSL

Servez votre site en HTTPS et utilisez vos éventuels websockets en WSS à l'aide d'une clé et d'un certificat SSL.

Lors d'un accès à votre site par le protocole HTTP, la redirection vers la version en HTTPS est automatiquement effectuée.

V.0.5 Package

Rendez une de vos application sous forme de package qui devra être construit et installé avec `pip` lors de la correction.

V.0.6 Système de votes

Implémentez un système de vote permettant de soumettre des éléments de votre site au vote des utilisateurs.

Ces votes pourront être des votes positifs (upvotes) ou négatifs (downvotes).

Cette fonctionnalité permettra l'affichage de certaines informations :

- Des boutons de votes (pour downvoter et upvoter l'élément).
- Un récapitulatif des votes (positifs comme négatifs) pour cet élément.
- Le vote actuel de l'utilisateur pour cet élément (upvoté, downvoté ou aucun vote)

Un utilisateur ne pourra voter qu'une seule fois sur un élément et ne pourra pas simultanément upvoter et downvoter cet élément.

Si l'utilisateur clique une seconde fois sur un même bouton de vote, le vote précédent est simplement annulé.

Si l'utilisateur clique pour downvoter un élément après l'avoir upvoté, l'élément est downvoté et l'upvote précédent est annulé, l'inverse est aussi vrai.

Une fois que le clic que un bouton est effectué, les informations se mettent à jour sur l'affichage.

Une fois votre système de vote réalisé, testez-le sur votre forum, (Sur les posts et commentaires). Vérifiez que le fait d'avoir plusieurs élément soumis au vote sur une même

page ne perturbe pas les différents boutons de vote et que chacun se comporte correctement (par exemple, cliquer sur un bouton d'upvote n'upvote pas tous les éléments de la page.