Homework 4: Extending the Social Network

Due date: ~~February 15, 2016~~ => February 16, 2016, last late day still February 19, 2016

This homework is the second in a series of homeworks in which you will build an increasingly sophisticated nano-blogging site. This site will eventually be a featureful, interactive web application including user registration and authentication, email integration for user verification, photo upload, and quasi-real-time updates.

For this assignment, you will implement extend the application you built in the previous assignment, adding additional features such as the ability to edit and enrich your profile, following, and image uploading. Again, we recommend reading this document in its entirety before starting on your homework.

The learning goals for this assignment are to:

- Demonstrate mastery of many learning goals from the previous assignment, including all of the learning goals noted from earlier in the semester.
- Gain experience with sophisticated data management, including data models with complex relationships and the use of an ORM to execute queries using those relationships.
- Gain familiarity with features common to modern web application frameworks, including:
    - developing modular, reusable views with inheritance,
    - using form classes to encapsulate input validation, and
    - image (or file, in general) upload.

Specification

This section describes the enhancements you will make to your social network application. To begin your assignment, start by copying over your project from the last homework over to the hw4 folder (see Turning in your work for the directory structure).

Effective use of Django Templating

Make the following enhancements to your templates for your social network:

1. Eliminate code duplication in your templates with template inheritance.
2. Eliminate all hard-coded URLs in your Python and templating code by using reverse URL resolution. Particularly, this means usage of the `{% url 'location' %}` template tag as well as the `reverse()` function in Python.

Validation via Django Forms

Refactor your application to validate all user input using Django Forms. This means that you will not be *manually* validating request parameters as you have been currently shown to do, but instead make use of the provided validation functions such as `Form.is_valid()` function.

New features for your social network

In addition to the refactoring in the tasks above, you are to also add the following enhancements to your homework assignment.

- Profiles for logged-in users will include at least the following information if available:
    - first name
    - last name
    - age
    - short bio (430 characters or less)

- Logged-in users are able to edit their profile information.

- Logged-in users are able upload a profile image, which is displayed on the site next to their posts as a small image. The image upload form should be located in the profile editing page.

- Logged-in users may choose to 'follow' and 'unfollow' another user.

- Logged-in users may view a 'follower stream', displaying all posts from the users that the current user is following (in reverse-chronological order).

Implementation hints

You may end up adding something like the following pages:

1. A **edit profile** page; displays a form that allows the current user to edit their first name, last name, age, and short bio, as well as upload a profile image.
2. A **follower stream** page; lists posts from all users that the current user follows.

You may want to re-familiarize yourself with the following guides on the Django reference:

- Relationship fields (i.e. `ForeignKey`, `OneToOneField`, `ManyToManyField`)
- Making queries ("Retrieving specific objects with filters" and "Lookups that span relationships" might be very useful)

Requirements

Your application must also follow these requirements:

- You must meet **all** requirements specified in the previous assignment, including in particular:
  - The empty URL (i.e. http://localhost:8000/) must route to the first page of your application.
  - Your application should not use any hard-coded absolute paths.
  - Your application should run with **Django 1.8.x** or **1.9.x**.
  - Your application should use the default Django database configuration based on a SQLite database file (named `db.sqlite3`) in your project directory.
  - Your application should not crash as a result of any input sent to the server-side or because of any actions the user performs.

- Your application should use template inheritance, reverse URL resolution, as well as complete validation of client-submitted data with Django Forms.

- All tasks and features in the specification must be easily accomplishable using the user interface for your social network.

- Cite all external resources used and any additional notes you would like to convey to your grader in the `README.md` file.

Grading criteria

For substantial credit your solution must clearly demonstrate the learning goals for this assignment, which are described above in the introduction.

Committing your work

As with the previous homeworks, we will be evaluating your version control usage. Keep in mind that good version control usage typically means (1) incremental, modular commits with (2) descriptive and useful commit messages.

Specification fulfillment

Your submission must follow all specifications and requirements introduced in the previous sections.

Validation

As with previous assignments, any client request (achievable or not by your user interface) must not be able to crash the application. Additionally, we are looking to see if you correctly use Django Forms for input validation.

Coverage of technologies

You must demonstrate effective use of the introduced technologies of this assignment.

- template inheritance and reverse URL resolution
- ORM usage (model design, data manipulation, and data retrieval)
- image upload and display

Design

As with previous assignments, we will provide comments and suggestions on your user interface, but will not be grading specifically the aesthetics of your project.

Turning in your work

Your submission should be turned in via Git and should consist of a Django application in the **hw4** directory. Name your project **webapps** and the application **socialnetwork**. The directory structure will look somewhat like the following (some files/directories omitted):

```
[YOUR-ANDREW-ID]/hw4/
   |-- webapps/
       |-- settings.py
       |-- urls.py
       |-- [etc.]
   |-- socialnetwork/
       |-- static/
       |-- templates/
       |-- models.py
       |-- views.py
       |-- [etc.]
   |-- manage.py
   |-- db.sqlite3
   |-- README.md
```