Homework 5: Asynchronous interaction

Due date: October 18, 2018

This homework is the penultimate assignment in a series of homeworks in which you will build an increasingly sophisticated nano-blogging site. For this assignment you will extend your Homework 4 solution to use Ajax to allow user comments on posts in **grumblr**. You will also use Ajax to refresh the global stream to show new posts.

The learning goals for this assignment are:

- Demonstrate mastery of the learning goals from the previous homeworks, including both the technical features of web applications and the development process.
- Gain familiarity with interactive client-side web programming, including the use of jQuery for DOM manipulation and Ajax.

Specification

This section describes the enhancements you will make to **grumblr**. To begin this assignment, start by copying your Homework 4 solution to the `homework/5` directory in your course repository. You should correct any known flaws in your Homework 4 solution and then add two new features:

1. The global stream page should be updated with any new posts every five seconds, without refreshing the HTML page.
2. A logged-in user should be able to add comments to posts anywhere posts are shown to the logged-in user.
    - Comments show the author, the profile image, and the time the comment was made.
    - Comments should be displayed in chronological order on each post, with the bottom comment being the newest comment on that post.
    - It should be clear in your **grumblr** design which comments belong to which posts.
    - Adding a new comment should not require refreshing the entire HTML page.

Requirements

Your application must also meet the following requirements:

- You must meet all requirements specified in the previous assignment, including:
    - 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 in any Django-generated components. You may hard code absolute paths to static components.
    - Your application should run with Django 2.1.x and Python 3.
    - Your application should use the default Django database configuration with 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 or because of any user-performed actions.

    - Your application should use template inheritance, reverse URL resolution, and completely validate all client-submitted data using Django Forms or ModelForms.
    - All specified tasks and features must be easily accomplishable in your solution.
- You must fix any errors mentioned in your previous homework feedback.
- You must use jQuery rather than the standard JavaScript library for DOM manipulation and Ajax in such situations where jQuery is an appropriate tool.
- When implementing the Ajax updates of the global stream, your application must send and load only new posts since the last update. Do not simply copy the original Ajax to-do list example, which generated and reloaded the entire stream.
    - When updating posts or adding a comment to a post, you do not need to check for new comments from other users; you only need to check for new posts that would be visible on the page. (Checking for new comments would be an excellent optional feature, though!)
- Cite all external resources used and any additional notes you would like to convey to your grader in the README.md file. Include the python version you used to develop your solution.

Implementation hints

Recall that you must be able to comment on all posts, including the posts newly added (by Ajax) to the global stream. It's possible to accomplish this by dynamically adding appropriate event handlers to newly-created DOM elements, but you might want to learn more about event delegation: http://learn.jquery.com/events/event-delegation/.

Grading criteria

For substantial credit your solution must clearly demonstrate the learning goals for this assignment, especially a thorough understanding of client-side programming and the asynchronous interaction of client-side and server-side components of a web application.

Committing your work (10 pts)

Please remember that we evaluate your commit history and development process in addition to your final product.

Fulfilling our specification (20 pts)

Input validation (20 pts)

As with previous assignments, any client request (achievable or not by your user interface) must not be able to crash the application. Also, we will still evaluate your use of Django Forms/ModelForms for input validation. Be sure to validate all Ajax requests even if you expect such requests to be generated only programmatically by your application.

Appropriate use of web application technologies (50 pts)

We will evaluate your use of Django technologies in addition to JavaScript and jQuery for DOM manipulation, event handling, and Ajax.

Turning in your work

Your submission should be turned in via Git and should consist of a Django application in the `homework/5` directory. Name your project **webapps** and the application **grumblr**. Your directory structure might look something like:

```
[YOUR-ANDREW-ID]/homework/5/
   webapps/
       settings.py
       urls.py
       [etc.]
   grumblr/
       static/
             js/
               jquery-3.3.1.min.js
               grumblr.js
       templates/
           grumblr/
       models.py
       views.py
       [etc.]
   manage.py
   db.sqlite3
   README.md
```