Classified Ad Web Site - Milestone 3

Instructor 

In this assignment, you will expand your classified ads web site to add functionality equivalent to:

https://chucklist.dj4e.com/m3

We will add a favoriting capability to your previous milestone by borrowing more parts and pieces from the code that runs:

https://samples.dj4e.com/

Do All of the Challenges

At this point all of the challenges should be working - not all will be tested by the autograder - but we will separately check them.

(1) Make yourself a gravatar at https://en.gravatar.com/ - it is super easy and you will see your avatar when you log in in your application and elsewhere with gravatar enabled apps. The gravatar can be anything you like - it does not have to be a picture of you. The gravatar is associated an email address so make sure to give an email address to the user you create with `createsuperuser`.

(2) Change your `home/static/favicon.ico` to a favicon of your own making. I made my favicon at https://favicon.io/favicon-generator/ - it might not change instantly after you update the favicon because they are cached extensively. Probably the best way to test is to go right to the favicon url after you update the file and press 'Refresh' and/or switch browsers. Sometimes the browswer caching is "too effective" on a favicon so to force a real reload to check if the new favicon is really being served you can add a GET parameter tho the URL to forc it to be re-retrieved:

```
https://chucklist.dj4e.com/favicon.ico?x=42
```

Change the `x` value to something else if you want to test over and over.

(3) Make social login work. Take a look at github_settings-dist.py, copy it into `mysite/mysite/github_settings.py` and go through the process on github to get your client ID and secret. The documentation is in comments of `mysite/mysite/github_settings.py`.

To get your key and secret from github, go to: https://github.com/settings/developers and add a new OAuth2 application. Here are some sample settings:

```
Application name: ChuckList PythonAnywhere
Homepage URL: https://drchuck.pythonanywhere.com
Application Description: Some pithy words...
Authorization callback URL: https://drchuck.pythonanywhere.com/oauth/complete/github/
```


You can register two applications with github - one on localhost and one on PythonAnywhere. If you are using github login on localhost - make sure that you register `http://127.0.0.1:8000/` instead of `http://localhost:8000/` and use that in your browser to test your site. If you use localhost, you probably will get an error message when you login like: `The redirect_uri MUST match the registered callback URL for this application.`

Adding Favorites to the Ads Application

In this section, you will pull bits and pieces of the `favs` sample application into your `ads` application to add support for logged in users to "favorite" and "un-favorite" ads.

(1) Add this to your `ads/model.py` , taking inspiration from `dj4e-samples/favs/models.py`

```
class Ad(models.Model) :

    ...

    # Favorites
    favorites = models.ManyToManyField(settings.AUTH_USER_MODEL,
        through='Fav', related_name='favorite_ads')
    ...

class Fav(models.Model) :
    ad = models.ForeignKey(Ad, on_delete=models.CASCADE)
    user = models.ForeignKey(settings.AUTH_USER_MODEL, on_delete=models.CASCADE)

    # https://docs.djangoproject.com/en/4.2/ref/models/options/#unique-together
    class Meta:
        unique_together = ('ad', 'user')

    def __str__(self) :
        return '%s likes %s'%(self.user.username, self.ad.title[:10])
```

Of course do the migrations once you have modified the model.

(2) Add two routes to your `urls.py` for the favorite features

```
    path('ad/<int:pk>/favorite',
        views.AddFavoriteView.as_view(), name='ad_favorite'),
    path('ad/<int:pk>/unfavorite',
        views.DeleteFavoriteView.as_view(), name='ad_unfavorite'),
    ...
```

(3) Look at how `ThingListView` from `dj4e-samples/favs/views.py` retrieves the list of favorites for the current user and add code to your `AdListView` to retrieve the favorites for the current logged in user.

(4) Alter your `ad_list.html` by looking through `favs/templates/favs/list.html`. Make sure to add the parts that show the stars based on the list of favorites for this user and the `favPost()` JavaScript code at the end.

(5) Pull in and adapt `AddFavoriteView`, and `DeleteFavoriteView` from `dj4e-samples/favs/views.py` into your `views.py`.

Manual Testing

It is always a good idea to manually test your application before submitting it for grading. Here are a set of manual test steps:

- Make two accounts if you have not already done so
- Log in to your application on the first account
- Create an ad, view its details, update, the ad, and delete the ad (test for regression)
- Create more than one ad
- In the list view mark one ad as a favorite and then press 'refresh' and see if the star is the same after refresh as it was when you clicked on the star
- In the list view unfavorite a favorited ad and then press 'refresh' and see if the star is the same after refresh as it was when you clicked on the star
- Log in on the second account - make sure the favorites are not the same as the first account
- Do several favorite and unfavorite operations pressing 'refresh' after each change and make sure the star "sticks" (i.e. has the same value as when you clicked it)

The most common problem is that when you click onthe star if looks good on the screen but the fact that this is not a favorites (or not) did not get recorded in the server. Often you will need to check the developer network console in your browser to find errors in the AJAX code.

Finding and Fixing Errors in the Developer Console

This is the first time you are using AJAX so some of the errors will only be seen in the developer console. If your favoriting code breaks - you won't see the errors on the main screen. Go into the developer console, under the network tab and watch for the AJAX (also known as XHR) calls. Some will fail with errors like 404 or 500 and - if you select the request that is in error and look at the `Response` tab you will usually see what is going wrong in the server.

Sometimes the AJAX errors are a little difficult to see when using the Chrome browser. The developer console in FireFox renders the actual HTML to make it easier to read.

Things that might go wrong

Make sure to turn off your ad blocker. Take a look at your web developer console if the AJAX part of favorites seem to fail. You might see a message like:

```
POST .. net::ERR_BLOCKED_BY_CLIENT
```

Or a similar message - this means your JavaScript tried to do an AJAX request and was stopped by the browser.