MIT AITI
Mobile, Python, Software Development

Lab 06: Forms

You will be adding forms to your blog app. These forms will allow you to add and edit comments. If you get stuck, take a look at these resources:

1. Lecture slides
2. Previous labs
3. Other group members
4. Django documentation
   a. Forms: https://docs.djangoproject.com/en/1.4/topics/forms/
   b. Forms from Models: https://docs.djangoproject.com/en/1.4/topics/forms/modelforms/
5. Google
6. Instructors

Part 1: Adding Comments

1. Browse to your `myblog` project.
```
$ cd ~/Desktop/myblog
```

2. Browse to your `templates/blog` directory and open `detail.html`. Add this code after the `{% endfor %}`
```
<br />
<div>New Comment</div>
    <form method='post' action="">
        {{ form }}
        <input type='submit' name='Submit'>
    </form>
```

3. At the top of your `views.py`, add the following lines:
```
from django.forms import ModelForm
from django.http import HttpResponseRedirect
from django.views.decorators.csrf import csrf_exempt
```

4. In `views.py`, right before your `blog_detail` method, create a class called `CommentForm`. Link the `CommentForm` class to your `Comment` model.

5. In `views.py`, update your `blog_detail` definition to look like this:

   ```
   @csrf_exempt
   def blog_detail(request,id,showComments=False):
   ```

   If we don't put in the `@csrf_exempt`, django will give us a security error. We may teach some information on csrf later on, but for now we will just exempt our views.

6. In your `blog_detail` method, insert the following code after retrieving the list of comments. Make sure you understand what is happening, and ask us for help if you find any of it confusing.

   ```
   #Start of form code
   if request.method == 'POST':
       form = CommentForm(request.POST)
       if form.is_valid():
           form.save()
           return HttpResponseRedirect(request.path)
   else:
       form = CommentForm()
   #end of form code
   ```

7. There is one more change to make before your comment form will be displayed. In `blog_detail`, update the context to include the form:

   ```
   c = Context({'blog':blog, 'comments':comments,
   'form':form.as_p()})
   ```

8. Start your server. Go to the website (http://localhost:8000/blog/list) and choose a post.

9. Add comments to your post and make sure they show up. Make sure you select a blog post from the list.

10. Right now, this works, but users shouldn't have to manually choose a blog post. We can fix that by excluding the post field from the form and automatically choosing a post for a new comment:
    - In the Meta class of your CommentForm class, add this line:

```
exclude=[‘post’]
```

- Insert the bold lines into your blog_detail

```
if request.method == 'POST':
    comment = Comment(post=blog)
    form = CommentForm(request.POST,
instance=comment)
    if form.is_valid():
```

11. You should now have a working submission form that doesn’t force the user to choose a post.

Part 2: Editing Comments

12. Edit your `detail.html` template file and add the bold line. This will add an “Edit Comment” link to your details page :

```
<div>Last updated on {{ comment.updated }}</div>
<div><a href="/blog/editcomment/{{ comment.id
}}">Edit Comment</a></div>
```

13. Now it’s your turn to write your own code. Here’s what you need to do:
    - Set up a template called `editcomment.html` (Hint: copy and modify your other templates)
    - Write a view that will display the form. Make sure that you populate the form fields with the information that is stored in the database. (Hint: You should be able to copy and paste most of the code from another view)
    - When the user has properly submitted the form, it should redirect them to the blog page for that comment. It’s a bit tricky, so here’s the line you should use:

      ```
      return HttpResponseRedirect('/blog/detail/'+
      str(comment.post.id)+'/True')
      ```
    - If you get a CSRF verification error, you forgot to put `@csrf_exempt`
    - Note: You should **not** have to create a new form class. Keep in mind that one of the main features of python and django is code reuse.
    - Edit your `urls.py` to include the path to editcomment

14. Test and make sure you can edit a comment.

15. (Challenge) Notice that in the forms to both create and edit comments, the user must manually insert the date. However, the date created should be set automatically when the user submits the comment, and the date submitted updated should be updated automatically when the comment is edited. Edit your blog website so that these fields do not appear in the forms, but are automatically set appropriately.