Final Exam: Building A Web App

Kalu Kalu

Final Exam Description and Questions

The exam tests the student's ability to make a web-app, and share it open-sourced on Github.

For the final exam the students will be asked to launch their terminal, activate their virtual environment and accomplish the following tasks:

1) Use Django to create a simple proof of concept web-app for our client, a retail store, based on the client specifications
2) Upload the entire django project onto Github

Client Specifications

1. Title says "Kool Karen Taobao"
2. Webpage header should say have a large heading 1 with the retail store name, Kool Karen Retail Store
3. Below the retail store name should be the company tagline, "Your convenience is our duty" in smaller font. (you can use your own tagline but it must be catchy!)
4. There should be pictures of at least 3 store items as well as the name of the item.
5. The website should have a working form that allows users to submit their order and address.
6. After a user submits their dish order and address they should see it printed on the webpage below the order form.
7. **Bonus:** Save all user inputs in a database in a table named Order
8. **Bonus**: Create a user admin where we can see data in our database.

Review Will Be Almost Exactly The Same as The Final Exam

This review will be almost exactly the same as what the final exam will ask you. The only difference will be the name of the app, the name of the project, and some text. Instead of making a convenient store website for "Kool Karen Retail Store", where users can post what item they want and their address, you might instead be asked to make an online ice cream store website for "Evelyn Eats Ice Cream Shop" where users post which ice cream they want and what their address it.

Other than that it will be exactly the same.

Rubric:

The students must take a screenshot of their Terminal and save it in their ~/Documents/your_name folder. Immediately after the exam I will look at the pictures and grade their exams.

The exam will be graded based on the following rubric.

1) Django Project (70)
    1) Have a working website which can be visited using the 127.0.0.0:8000 web address. (10 points)
    2) Include a title (specification 1) (10 points)

3) Include the proper webpage (specification 2) (10 points)
4) Include the store name and tagline (specificaiton 3) (10 points)
5) Include three pictures of store items (specification 4) (10 points)
6) Include a working form in home_page (specification 5) (10 points)
7) When user submits they can see their output (specification 6) (10 points)

2) Upload to Github (30)
   1) Create a local repository
   2) Create a GitHub repository
   3) Save all changes to local repository and push to remote repository.

Points Deduction

I will deduct 5 points off your score for any time you need my help to progress or if you speak to another classmate (which is not allowed).

Bonus

1) 10 points for creating a database and database table correctly
2) 10 points for creating a working admin page.

Review

Let's review the final exam.

Creating Our Project and App

The first thing we need to do is make our django project and app. We can call the django project "retail_store" and the app "taobao".

To start with let's open our terminal, cd into your folders and create our project and create our app.

```
KaludeMacBook-Pro:~ kalukalu$ cd ~/Documents/kalu/
KaludeMacBook-Pro:kalu kalukalu$ source ~/
django_and_machine_learning_venv/bin/activate
(django_and_machine_learning_venv) KaludeMacBook-Pro:kalu kalukalu$
django-admin startproject retail_store    1
(django_and_machine_learning_venv) KaludeMacBook-Pro:kalu kalukalu$ cd
retail_store/
(django_and_machine_learning_venv) KaludeMacBook-Pro:retail_store
kalukalu$ python manage.py startapp taobao    2
```

A Note on Creating Our "retail_store" Project

Here we did two things different from normal. Previously our project was named “my_site”. Now our project is named “retail_store”, therefore we need to do django-admin startproject retail_store 1 .

A Note on Creating Our “taobao” App

Next, our app name is also different here than before. Therefore to create an app named “taobao” (instead of the previous blog) we need to do python manage.py. startup taobao 2 .

Adding Our “taobao” App to settings.py

Open your newly created “retail_store” folder in sublime and open the settings.py file. In your settings.py you should add your newly created app to the INCLUDED_APPS list.

your_name/retail_store/retail_store/settings.py

```
1. INSTALLED_APPS = [
2.     'django.contrib.admin',
3.     'django.contrib.auth',
4.     'django.contrib.contenttypes',
5.     'django.contrib.sessions',
6.     'django.contrib.messages',
7.     'django.contrib.staticfiles',
8.     'taobao', 1
9. ]
```

You can see at 1 we added the name of our app as a string.

Project urls.py

Next, we can set up our project-wide urls.py. In sublime open urls.py located at your_name/retail_store/retail_store/urls.py. Then edit it to look like the following.

your_name/retail_store/retail_store/urls.py

```
1. from django.contrib import admin
2. from django.urls import path, include 1
3.
4. urlpatterns = [
5.     path('', include('taobao.urls')), 2
6.     path('admin/', admin.site.urls),
```

```
7. ]
```

We simply need to add “include” to the imports 1 and then add our url path 2 that points to our app urls that we are about to create.

App urls.py

your_name/retail_store/taobao/urls.py

```
1. from django.urls import path
2.
3. from . import views
4. app_name = 'taobao' 1
5. urlpatterns = [
6.     path('', views.home_page, name='home_page'),
7. ]
```

What is different here from our blog app is we set our app_name to “taobao” 1 instead of “blog”. Outside of that everything is exactly the same.

views.py

your_name/retail_store/taobao/view.py

```
1. from django.shortcuts import render
2.
3. def home_page(request):
4.
5.     if request.method == 'POST':
6.         item = request.POST.get('item') 1
7.         address = request.POST.get( “address") 2
8.
9.         return render(request, 'taobao/home_page.html', {'item': item, 'address':
   address}) 3
10.
11.     return render(request, ‘taobao/home_page.html’) 4
```

Inside of our if-statement block that checks whether a POST request was made, we first grab the value the user typed in for the input named “dish”, and assign it to a variable named “dish” 1 . Next we grab the value that the user typed in for the texture named “address” , and assign it to a variable named “address”. 2 Both the “item” 1 and “address” 2 arguments that we pass to the respective request.POST.get() method calls, will be discussed

again under the home_page.html section. The arguments we pass in here, “item” and “address”

Then we pass both our dish and address to our home_page.html template inside of a context dictionary 3 .

Lastly, if there is not a POST request, then we will merely just return the home_page.html template without any extra information for our template 4 .

home_page.html

Next we need to create a folder called templates inside of our taobao folder. It should be your_name/retail_store/taobao/templates. Now inside of that templates create another folder called “taobao” your_name/retail_store/taobao/templates . Inside of that folder taobao finally create a file called home_page.html . Please type the following inside of your your_name/retail_store/taobao/templates/home_page..html .

your_name/retail_store/taobao/templates/home_page.html

```
1.  <!DOCTYPE html>
2.  <html>
3.  <head>
4.      <title>Kool Karen Taobao</title>
5.  </head>
6.  <body>
7.      <h1>Kool Karen Retail Store</h1>
8.      <p>Your convenience is our duty</p>
9.
10.     <h2>Here are all our items!</h2>
11.
12.     <!-- Item 1 -->
13.     <p>Water: 12RMB</p>
14.     <img src="https://tse2-mm.cn.bing.net/th?
    id=OIP.QpJtVsC4fgQWo5uM4GVRsQHaHa&pid=Api"> 1
15.     <br>
16.
17.     <!-- Item 2 -->
18.     <p>Toothbrush: 7RMB</p>
19.     <img src="http://pics.drugstore.com/prodimg/146927/450.jpg">
20.
21.     <!-- Item 3 -->
22.     <p>Chips: 10RMB</p>
23.     <img src="http://deals.simplyshellie.com/wp-content/uploads/2012/08/lays-
    chips.jpeg">
24.
```

```
25.     <!-- Order Form -->
26.     <h2>Place your Order</h2>
27.     <form action="{% url 'taobao:home_page' %}" method= “POST">  2
28.             {% csrf_token %}  3
29.             <input type="text" name="item" placeholder="Please Type Which Item You
    Want"><br>  4
30.             <textarea name="address" placeholder="Please Type Your Address” ></
    textarea><br>  5
31.             <input type="submit" value="Place Your Order” >
32.     </form>
33.
34.     <!-- Show Order -->
35.     <h1>Your Order:</h1>
36.     <p>Item Ordered: {{item}}</p>
37.     <p>Delivery Address: {{address}}</p>
38.
39. </body>
40. </html>
```

Adding Pictures

To add a picture is simple. Use bing.com to find a picture. Then right-click on the picture and click “copy image location.” The url address must be from the actual website, not bing. Then inside of your template, wherever you want to insert a picture, simply type img 1 and press the tab key. The img html tag should auto-complete for you, placing your cursor right where it needs to be to paste the URL address of your image inside of src=“” .

Our Form

The Action Attribute

The special part of this is the form. For your form to work correctly you most include an action attribute with a properly set value inside of our form opening tag 2 . In our action we have to set the value to “{% url ‘taobao:home_page’ %}”. This value means that this request should be sent url named “home_page” inside of the “taobao” app. This is all as set up in our ~/your_name/retail_store/taobao/urls.py.

The Method Attribute

Also inside our form opening tag is should be a method attribute set to the value “POST” 2 . This is in order to make the request, a post request.

The CSRF Token

The second important thing inside of our form element is the {% crsf_token %} 3 .

According to django, a csrf_token:

> provides easy-to-use protection against Cross Site Request Forgeries. This type of attack occurs when a malicious website contains a link, a form button or some JavaScript that is intended to perform some action on your website, using the credentials of a logged-in user who visits the malicious site in their browser. A related type of attack, ‘login CSRF’, where an attacking site tricks a user’s browser into logging into a site with someone else’s credentials, is also covered.

For more see; https://docs.djangoproject.com/en/2.0/ref/csrf/.

The Name Attribute

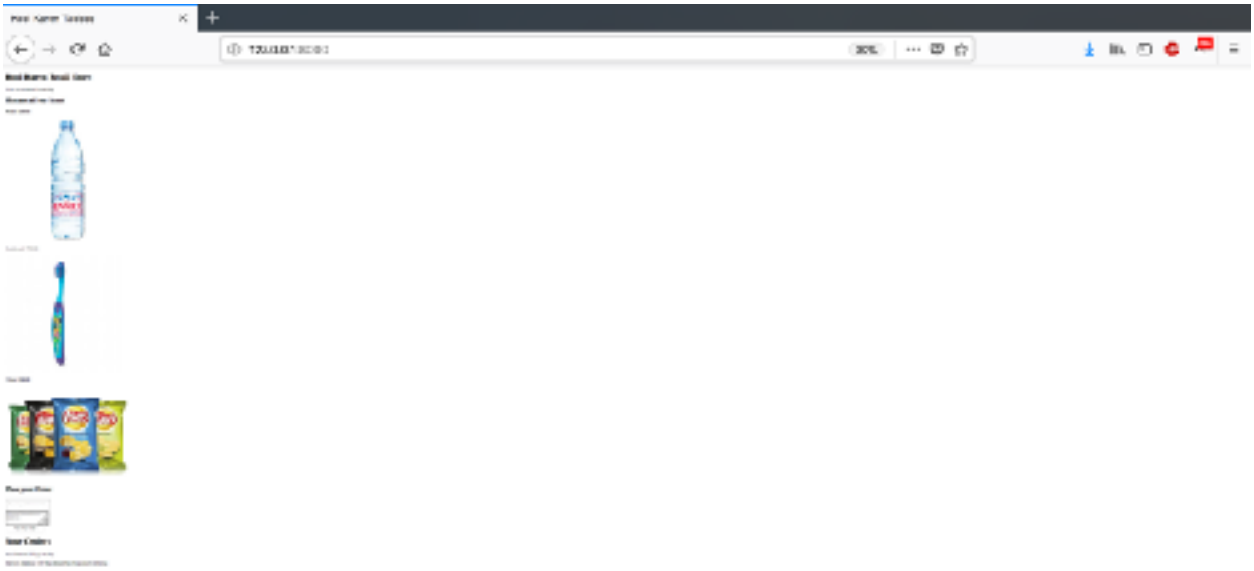
Inside of our input 4 and textarea 5 elements we set a name attribute. We set each name attribute to “item” and “address” respectively. The name attribute is how we refer to the value inputted by the user inside of our views.py. If you remember in our views.py we have the lines of code request.POST.get(‘item’) and request.POST.get(‘address’). This allows us to get the values that the user put in each respective input or text area box inside of our views.py. In others the values you have in your home_page.html input or textarea name attributes must correspond with the argument you pass to the request.POST.get() method inside your views.py. If we had the name attributes set to name=“user” and name=“comment” instead, then in our views.py we must use request.POST.get(“user”) and request.POST.get(“comment”) to get the text inputed by the user.

Running Your Server

Once You have typed up your home_page.html you are done! Congrats! We now simply need to run our server and check out our website. Go back to your Terminal. Make sure that your virtual environment is activated and that you are in the same folder as manage.py (your_name/ retail_store/). Then type the following command to run your server; python manage.py runserver . (<—- do not include the period)

```
(django_and_machine_learning_venv) KaludeMacBook-Pro:retail_store
kalukalu$ python manage.py runserver
```

Then right click on the 127.0.0.0:8000 url address and click copy url address. Paste the address in your web-browser address bar and admire your website!



Place your Order

Please Type Which Item You

Please Type Your Address

Place Your Order

Your Order:

Item Ordered: 4 bags of chips

Delivery Address: 180 Main Street, San Francisco California