BackEnd Workshop-4

Clarusway

Subject: Django project with a pizza ordering website.

Learning Goals

- Practice Django forms,
- Using submitted data,
- Adding models
- Using the ModelForms class
- Working with widgets
- Accepting files and multiple forms on a page
- Customizing formsets
- Using local validation
- Delivering errors responsibly
- Styling with CSS

Introduction

In this workshop, we will create a Django app named "Nandia's Garden" which will let users to order pizza.

Most websites require the use of forms to receive data from users, so it is crucial to know how to safely collect and handle data while maintaining a user-friendly experience on your website. In this course, learn how to use Django to create forms.

See the details of the project:

Nadia's Garden Webpage

Models:

In this project, you need to create a Size model, and a Pizza model with fields:

- Size:
    - title
- Pizza:
    - topping1
    - topping2
    - size (ForeignKey)

Forms :

In this project, you need to create a PizzaForm using Pizza model. And also, you need to create a MultiplePizzaForm to order mored than one pizza at a time.

Views:

In this project, you need to create four views:

- home
- order
- pizzas
- edit_order

Templates:

In this project, you need to create four templates:

- home.html
- order.html
- pizzas.html
- edit_order.html

And one optional template for base.html.

URLs:

In this project, you need to create four urls:

- '' : root path will return home page.
- 'order/' : page to order pizza, showing pizza form.
- 'order/<int:pk>/' : editing order.
- 'pizzas/' : to order more than one pizza.

Project Folder Structure

At the end of the project, the folder sturcture will be like:

Code:

Part 1 - Create project and app

1. Create a working directory, with a meaningfull name, and cd to new directory.
2. Create virtual environment as a best practice.
3. Activate scripts, see the (env) sign before your command prompt.
4. Install django.
5. (Optional) See installed python packages.
6. Create the requirements.txt on your working directory, send your installed packages to this file, requirements file must be up to date.
7. Create project.
8. (Optional) If you created your project wihthout "." at the end, change the name of the project main directory as src to distinguish from subfolder with the same name.
9. Create the application. Go to the same level with manage.py file if you create your project with nested folders at step 7. Start app.

10. Go to settings.py and add another line as below under INSTALLED_APPS.
11. Run your project.

Part 2 - Create Models

1. Go to models.py and create Size and Pizza models.
2. See how to use ForeignKey.
3. Discuss the differences between TextField and CharField.
4. Manage migrations.

Part 3 - Create Forms

Create forms.py under pizza app and create your forms.

Part 4 - Create Views and URLs

1. Go to views.py under "pizza" directory, and create your views.
2. Discuss using formsets.
3. Include URL path of the new app to the project url list, go to urls.py.
4. Add urls.py file under pizza directory.

Part 5 - Create Templates

1. Create templates/pizza folder and under that folder, create base.html file as a best practice. Create this base template including bootstrap.
2. Under templates/pizza folder create home.html as the second template, this will serve home page.
3. Under templates/pizza folder create order.html as the third template, this will enable users to order pizza.
4. Under templates/pizza folder create edit_order.html as the third template, this will enable users to edit their orders.
5. Under templates/pizza folder create pizzas.html as the third template, this will enable users to order more than one pizza at the same time.
6. Run our project again to see our view.
7. Go to http://localhost:8000/ in your browser, and check the functionality of your website. To stop the server use "CTRL + C".
8. If you want to send this project to your Github repo, do not forget to add .gitignore file, and secure your sensitive information like keys storing them locally in .env file and using python-decouple module.
9. Update the requirements.txt file after all your installations!
10. (Optional) Deploy your project to Heroku.

☺ **Thanks for Attending** ✍

Clarusway