IS590WF – Web Development Using Application Frameworks
Coding Assignment: Staticfiles and Generic Class-Based Views
Instructions

Overview

The Staticfiles and Generic Class-Based Views coding assignment is the eighth in a series of assignments in which we will be developing the EZ University database system, a full C-R-U-D database application for simplified university record keeping. In this assignment, we add styling to our Web application using the Staticfiles feature. Additionally, we refactor substantial portions of our views.py code to take advantage of generic class-based views (GCBVs).

Tools

I am expecting you to use the tools that are demonstrated in the tutorial videos: Anaconda, PyCharm, Git, and SourceTree.

Tool Versions

In the current semester, I am expecting you to use Python 3.6 and Django 2.1.1.

Tutorial Parts

This is a seven-part tutorial. The parts are named Part 0 through Part 6.

Part 0 – Relocate Templates

In this tutorial part, we work together to relocate the templates directory so that it is contained by the *courseinfo* app directory. This is a refactoring to make our code compliant with Django best practices. As with any refactoring, please be sure to test your code both before and after the changes.

Part 1 – Staticfiles

In this part of the tutorial, we work together to configure the Staticfiles features of Django. Then, we install the static files that are contained in the following starter file:

- staticfiles_starter_files.zip

The result of our effort is a styled Web application that is substantially more visually appealing than the unstyled version that we have worked with up to this point.

Part 2 – RedirectView and Templateview

In this part of the tutorial, we work together to add an *About* page to our Web application and implement it using the RedirectView GCBV. Additionally, we refactor the code that redirects the empty URL string so that it is implemented with the TemplateView GCBV.

Files needed for this part of the tutorial are provided in:

- gcbv_starter_files.zip

Part 3 – Paginated ListViews
In this part of the tutorial, we begin by work together to refactor the InstructorList view so that it uses the ListView GCBV. Some code needed for this part of the tutorial is provided in:

- gcbv_starter_files.zip

After we have worked together to code and test our refactoring of InstructorList, you will complete this part by doing a similar refactoring of the StudentList view on your own.

Part 4 – Non-Paginated ListViews
In this part of the tutorial, we begin by working together to refactor the SectionList view so that it uses the ListView GCBV.

After we have worked together to code and test our refactoring of SectionList, you will complete this part by doing a similar refactoring the following views on your own:

- CourseList
- SemesterList
- RegistrationList

Part 5 – CreateViews
In this part of the tutorial, we begin by working together to refactor the following views so that they use the CreateView GCBV:

- InstructorCreate
- SectionCreate

After we have worked together to code and test our refactoring of InstructorCreate and SectionCreate, you will complete this part by doing a similar refactoring of the following views on your own:

- CourseCreate
- SemesterCreate
- StudentCreate
- RegistrationCreate

Part 6 – DeleteView and UpdateViews
In this part of the tutorial, we begin by working together to refactor the following view so that it uses the DeleteView GCBV:

- RegistrationDelete

Note that we will not be refactoring the other Delete views.

We continue this part of the tutorial by working together to refactor the following views so that they use the UpdateView GCBV:

- InstructorUpdate
- SectionUpdate

After we have worked together to code and test our refactoring of InstructorUpdate and SectionUpdate, you will complete this part by doing a similar refactoring of the following views on your own:

- CourseUpdate
- SemesterUpdate
- StudentUpdate
- RegistrationUpdate

Deliverables
Please be sure that the project you submit includes the following:

1. A test user (username = “tester”, password = “(secret)”
2. Sufficient test data present in the database to allow for testing all functions

Submission
Submit 1 properly named and properly organized zip file containing your PyCharm project. Remember to use the following naming conventions:

- Project Name: yourLastName_yourFirstName_ez_university
- File Name: yourLastName_yourFirstName_ez_university.zip

Due Date
Please see the Weekly Schedule for the date and time when this assignment is due.