Web Development Using Application Frameworks
Coding Assignment: Generic Class-Based Views
Instructions

Overview

The Generic Class-Based Views coding assignment is the next 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 refactor 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 and PyCharm.

Tool Versions

In the current semester, I am expecting you to use Python 3.9.7 and Django 3.2.5.

Tutorial Parts

This is a seven-part tutorial.

Part 1 – RedirectView and TemplateView

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

While accomplishing this goal, we will use the starter files provided in:

- `gcbv_starter_files_parts_1_through_3.zip`

Part 2 – 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_parts_1_through_3.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 3 – 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 4 – DetailViews
In this part of the tutorial, we begin by working together to refactor the following views so that they use the DetailView GCBV:

- InstructorDetail
- SectionDetail

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

- CourseDetail
- SemesterDetail
- StudentDetail
- RegistrationDetail

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 the views above, you will complete this part by doing a similar refactoring of the following views on your own:

- CourseCreate
- SemesterCreate
- StudentCreate
- RegistrationCreate

Part 6 – UpdateViews
In this part of the tutorial, we begin 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 the views above, you will complete this part by doing a similar refactoring of the following views on your own:

- CourseUpdate
- SemesterUpdate
- StudentUpdate
- RegistrationUpdate

Part 7 – DeleteViews
In this part of the tutorial, we begin by working together to refactor the following views so that they use the DeleteView GCBV:

- RegistrationDelete
- InstructorDelete

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

- SectionDelete
- CourseDelete
- SemesterDelete
- StudentDelete

Exercises

1. **Exercise 1 (Required)**
   Follow Parts 1 through 7 of the tutorial instructions exactly.

2. **Exercise 2 (Optional Challenge Exercise)**
   The grading rubric for the Final Project requires that you use at least 1 GCBV in your project. Create a plan for implementing GCBVs in your Final Project. Your plan should include the following:

   - How much effort on average do you estimate it will take to refactor a view to use GCBVs?
   - Based upon the estimate above, what do you estimate it will take to refactor all of your views?
   - Do you have any views in your Final Project that do not lend themselves to using GCBVs? Why not?
   - How many lines of code do you estimate would be eliminated from your Final Project by refactoring to GCBVs?

   Please write no more than 1 page of text (single spaced). Convert your document to a PDF file named *final_project_gcbv_plan.pdf*. Place it in the *courseinfo* directory of your PyCharm project with program files like *urls.py* and *views.py.*

Code Deliverables

You are expected to submit one properly organized PyCharm Django project that is ready to be tested using PyCharm. Please refer to my tutorial video for details. Even if you have decided to do the Challenge Exercise, just submit one Django project.

Non-Code Deliverables

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

1. A test user (username = "tester", password = "{iSchoolUI}". PLEASE NOTE: We have changed the password that in the current semester. The old password is mentioned in some of the tutorial videos. Please be sure to use the new password instead.
2. Sufficient test data present in the database to allow for testing all functions

Submission Method

Follow the process that I demonstrated in the tutorial video on submitting your work. This involves:

- Locating the properly named directory associated with your project in the file system.
- Compressing that directory into a single .ZIP file using a utility program.
- Submitting the properly named zip file to the submission activity for this assignment.

File and Directory Naming

Please use the following naming scheme for naming your PyCharm project:

```
surname_givenname_ezu
```

If this were my own project, I would name my PyCharm project as follows:

```
trainor_kevin_ezu
```

Use a zip utility to create one zip file that contain the PyCharm project directory. The zip file should be named according to the following scheme:

```
surname_givenname_ezu.zip
```

If this were my own project, I would name the zip file as follows:

```
trainor_kevin_ezu.zip
```

PLEASE NOTE: All file and directory names must be in lower case. Deductions will be made for submissions that do not conform to this standard.

Due Date

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

Last Revised

2022-04-05