Web Development Using Application Frameworks
Coding Assignment: Authentication and Authorization
Instructions

Overview
The Authentication and Authorization coding assignment is the next in a series of assignments in which we will be developing the EZU database system, a full C-R-U-D database application for simplified university record keeping. In this assignment, we add login and logout functionality; create users, user groups and permissions; and restrict access within most parts of our application to authorized users.

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

Tool Versions
Use the versions of PyCharm Professional, Anaconda, and Python that we installed during Week 1 of the course when we created the *e4_trainor_django_course* virtual env. These versions are documented in *Instructions for Tool Versions, Installation, and Virtual Environments*.

Important Note Regarding Passwords
PLEASE NOTE: The tutorial videos for this assignment use the old password scheme for logging into our *courseinfo* Django application. Regardless of what might be shown in the tutorial video, please give all users the new password that we are using in the current semester: {iSchoolUI}

Tutorial Parts
This is a 3-part tutorial.

Part 1 – Add Login/Logout Functionality
In this part of the tutorial, we work together to add login/logout functionality to our *courseinfo* app. The following is a list of activities conducted in this part:

1. Add urlpattern and template for login
2. Add LOGIN_REDIRECT_URL to settings.py
3. Add urlpattern for logout
4. Add LOGOUT_REDIRECT_URL to settings.py
5. Test login/logout using admin pages
6. Add markup for login/logout to base.html
7. Test finished version of login/logout

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

- starter_files_for_ezu_authentication_and_authorization_part_1.zip

Part 2 – Create Groups, Users and Permissions
In this part of the tutorial, we work together to create user groups, users, and permissions. The following is a list of activities conducted in this part:

1. Do migration for create groups (see starter file)
2. Create users by hand using Admin app
3. Do migration for create group permissions (see starter file)
4. Explore user powers in Admin app and Courseinfo App.

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

- starter_files_for_ezu_authentication_and_authorization_part_2.zip

Part 3 – Modify Courseinfo App to Enforce Permission Scheme
In this part of the tutorial, we work together to make modifications to the EZU project that enforce previously created login and permission rules. The following is a list of activities conducted in this part:

1. Add LoginRequiredMixin and PermissionRequiredMixin to all class-based views.
2. Add permission_required = 'courseinfo.xxxxxx_yyyyyyyyy' code to each class-based view header.
3. In urls.py, confirm that the pattern that redirects the root url redirects to the about_urlpattern.
4. In settings.py, confirm setting of LOGIN_REDIRECT_URL to about_urlpattern.
5. In settings.py, set LOGIN_URL to reverse_lazy('login_urlpattern').
6. In settings.py, add SESSION_EXPIRE_AT_BROWSER_CLOSE = True.
7. Test the app. Observe that UI options are still shown even when they cause 403 Forbidden.
8. Replace base.html template with new version that hides navigation links (see starter files).
9. Replace all _list.html and _detail.html templates with new version that hides links and buttons in the UI for unauthorized users (see starter files).
10. Test the app. Observe that UI options are shown to users only when they are authorized.

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

- starter_files_for_ezu_authentication_and_authorization_part_3.zip

Exercises

1. **Exercise 1 (Regular)**
   Follow Parts 1 through 3 of the tutorial instructions exactly.

2. **Exercise 2 (Challenge)**
   My EZU tutorial videos do not include unit testing code. In the current semester, we are working in groups during class breakout sessions to create unit testing code for the version of EZU that we created in the previous assignment. To get credit for this Challenge Exercise, you will need to copy the unit testing code developed by your group in the classroom and incorporate it into your own copy of EZU. For this assignment, that means you will incorporate the unit tests for the code that your group developed for the following assignment:

   - Generic Class-Based Views Assignment

   Remember that unit testing code in your copy of EZU should also still include the unit tests created by your group for earlier assignments.

   When incorporated into your copy of EZU, the unit tests should run and show all tests passing. To achieve this, you may need to refactor your code so that names used in the unit test code agree with names used in your EZU project. Also, you will need to correct any errors that are exposed by failing tests.

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. Make sure that all usernames mentioned in the tutorial are created with the expected password (password = "{iSchoolUI}"). PLEASE NOTE: We have changed the password 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
2024-02-20