System Requirements

Specification Index

For

Role-Based Permissions for Blog Post Management in a Publishing Platform

(Topic:- Django Authentication and Authorization)

Version 1.0

Scenario

You have joined a content publishing platform as a Django developer. The company aims to build a permission-based blog system, ensuring that only staff members can edit or delete blog posts, while all users (including non-staff) can view them.

This approach is crucial for maintaining content integrity while allowing a broader audience to access published content.

Problem Statement

Your task is to implement a permission system in Django for managing blog posts.

Each blog post must include the following details:

- **title → A unique title for the blog (CharField, max length: 200).**
- **content → The main content of the blog post (TextField).**
- **author → A ForeignKey reference to the Django User model.**
- **created_at → The timestamp when the blog post was created (DateTimeField, auto-generated).**

Your Task

- **Create a Django BlogPost model with the above fields.**
- **Ensure that:**
  - **Only staff users can edit or delete posts.**
  - **All users (even non-authenticated ones) can view posts.**
- **Implement Django permissions using has_permission and has_object_permission.**
- **Write API views to list, create, update, and delete blog posts.**

Execution Steps to Follow:

1. All actions like build, compile, running application, running test cases will be through Command Terminal.
2. To open the command terminal the test takers, need to go to

   Application menu(Three horizontal lines at left top)->Terminal->NewTerminal.

3. The editor Auto Saves the code.
4. If you want to exit (logout) and to continue the coding later anytime(using Save & Exit option on Assessment LandingPage) then you need to use CTRL+Shift+B command compulsorily on code IDE. This will push or save the updated contents in the internal git/repository. Else the code will not be available in the next login.
5. These are time bound assessments the timer would stop if you logout and while

   logging in back using the same credentials the timer would resume from the same time it was stopped from the previous logout.

6. To test any Restful application, the last option on the left panel of IDE, you can find

   ThunderClient, which is the lightweight equivalent of POSTMAN.

7. To test any UI based application the second last option on the left panel of IDE, you can

   find Browser Preview, where you can launch the application.

8. Install ‘djangorestframework’ module before running the code. For this use the following command.

   pip install djangorestframework

9. Use the following command to run the server

   python3 manage.py runserver

10. Mandatory: Before final submission run the following commands to execute testcases

    python3 manage.py test library.test.test_functional

    python3 manage.py test library.test.test_exceptional

    python3 manage.py test library.test.test_boundary

11. To test rest end points

    Click on ‘Thunder Client’ or use Ctrl+Shift+R->Click on ‘New Request’ (at left side of IDE)

12. Once you are done with development and ready with submission, you may navigate to the previous tab and submit the workspace. It is mandatory to click on “Submit Assessment” after you are done with code.

13. You need to use CTRL+Shift+B - command compulsorily on code IDE, before final submission as well. This will push or save the updated contents in the internal git/repository, and will be used to evaluate the code quality.