System Requirements Specification Index

For

Implementing Per-View Caching for Optimizing Blog Post Detail Page Performance

(Topic:- Django Caching)

Version 1.0

Problem Statement

Scenario:

You are a Django developer working at a content platform where users can write and read blog posts. The platform is gaining traction, and users are starting to experience slow loading times when viewing the blog posts. Specifically, the blog post detail pages are taking a long time to load because the database is queried for each request, even if the content hasn't changed.

The team has decided to implement per-view caching to reduce load times and improve performance. The goal is to cache the blog post detail page for each post and store it in the cache for a set period of time. This caching should only apply to the blog post detail view and must be invalidated if the post is updated.

Your task:

- **Implement per-view caching for the blog post detail page in Django using Django's caching framework.**
- **Cache the page based on the post's ID and invalidate the cache when the blog post is updated.**
- **Ensure that the cache is set with a proper timeout and that the page can still be updated correctly when the post changes.**

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.