Python/Django Practical task

Create API for a CMS application

1. The database should contain three tables named User, Post/Blog, and Like.
2. The User table should store user information such as user ID, name, email, password, and other relevant details.
3. The Post/Blog table should store post/blog information such as post ID, title, description, content, creation date, and other relevant details.
4. The Like table should store information about the likes of each post/blog, such as like ID, post ID, user ID, and other relevant details.
5. The following CRUD APIs should be implemented for all three tables:
    - Create API: To add new user/post/like to the corresponding table.
    - Read API: To retrieve a specific user/post/like from the corresponding table.
    - Update API: To update the details of a specific user/post/like in the corresponding table.
    - Delete API: To delete a specific user/post/like from the corresponding table.
6. The GET all post/blog API should also return the number of likes for each post/blog.
7. All APIs must adhere to the following rules:
    - Access to the PUT/DELETE APIs for the Post/Blog table should be restricted to the owner of the post/blog.
    - Any user can access the GET API for a post/blog if it is public. For private posts/blogs, only the owner should be able to access them.
    - There should be only one API endpoint for any given query. Retrieval of both the post/blog and its likes should be completed within a single query.

➔ You must share the complete codebase in a private/unlisted repository on your preferred git platform, please don't submit a zip or drive link.

➔ Plus point for providing a postman collection.