Django Rest Framework Assignment with Signals

Project Overview

You are tasked with creating a Django Rest Framework (DRF) application that allows users to create, read, update, and delete (CRUD) posts. The application should have the following functionality:

1. The user should be able to create a post by providing a title, body, and author.
2. The user should be able to retrieve a list of all posts, or a specific post by ID. 3.
The user should be able to update or delete their own posts.
4. Whenever a new post is created, an email should be sent to the author with a notification of the creation.

Requirements

1. The application should have an authentication system using Django's built-in User model.
2. The Post model should have the following fields:
    - `title` (CharField): The title of the post.
    - `body` (TextField): The body of the post.
    - `author` (ForeignKey to User model): The user who wrote the post.
3. The API endpoints should follow RESTful conventions:
    - `GET /api/posts/`: Returns a list of all posts.
    - `POST /api/posts/`: Creates a new post owned by the authenticated user.
    - `GET /api/posts/<int:pk>/`: Returns a single post by ID.
    - `PUT /api/posts/<int:pk>/`: Updates a single post owned by the authenticated user.
    - `DELETE /api/posts/<int:pk>/`: Deletes a single post owned by the authenticated user.
4. The user should only be able to CRUD their own posts.
5. All API endpoints should return JSON responses.
6. You should use DRF's **Generic APIView** and serializers whenever possible.

7. You should use DRF's authentication and permission classes to enforce authentication and ownership of posts.
8. Whenever a new post is created, an email should be sent to the author notifying them of the creation.
9. You should use Django signals to trigger the email notification whenever a new post is created.

Advanced Permssions

1. Implement custom permissions to allow superusers to have full access to all posts and update or delete any post regardless of ownership.
2. Non-superuser regular users should only be able to CRUD their own posts.

Advanced Serializers:

1.Implement a nested serializer to include the author's username and email when retrieving a single post. The author's full details should not be displayed on the list of all posts.

Requirements:

1. Allow users to filter posts by `title`, `body`, and `author` using query parameters on the `GET /api/posts/` endpoint.
2. Implement pagination for the `GET /api/posts/` endpoint.
3. Implement rate limiting to prevent abuse of the API.
4. Create get api for filtering post by author. Get author id and filter that author's post

Evaluation Criteria

You will be evaluated on the following:

1. How well you follow the above requirements.
2. The correctness and clarity of your code.
3. The effectiveness of your testing.
4. Your use of Django signals to trigger the email notification.