Comment System Task: Advanced Search and API Documentation Issued on September 26, 2025

1. Overview

This task extends your Django blog application by adding a comment system with advanced search functionality across comment content, post titles, and usernames. You will implement CRUD APIs for comments, create interactive API documentation using `drf-yasg`, and optionally build frontend templates. Only authenticated users can create, update, or delete comments, while all users can view and search comments.

2. Task Requirements

2.1 Database Model

- Create a `Comment` model in SQLite3 with:
  - `content`: TextField (comment text)
  - `author`: ForeignKey to `UserData` model
  - `post`: ForeignKey to `Post` model
  - `created_at`: DateTimeField (auto_now_add=True)
  - `updated_at`: DateTimeField (auto_now=True)

2.2 APIs

Implement the following APIs:

- **Create Comment (/comments/create/)**: Authenticated users add a comment to a post.
- **List Comments (/comments/?post_id=<id>)**: Display comments for a post with pagination (10 per page).
- **Update Comment (/comments/<id>/update/)**: Author edits comment content.
- **Delete Comment (/comments/<id>/delete/)**: Author deletes their comment.
- **Search Comments (/comments/search/?q=<keyword>)**: Search comments by keyword in `content`, `post.title`, or `author.username` (case-insensitive, partial matches).
- Restrict update/delete to the comment's author.

2.3 Advanced Search Functionality

- Implement the search API to:

  - Search `content` (e.g., “great” matches “great post”).
  - Search `post.title` (e.g., “Django” matches “Django Tips”).
  - Search `author.username` (e.g., “john” matches “john_doe”).
  - Use `Q` objects for case-insensitive partial matches (`iContains`).
  - Return paginated results (10 comments per page).
- Example: `GET /comments/search/?q=django` returns comments matching “django” in any of the above fields.

2.4 API Documentation

- Install `drf-yasg` (`pip install drf-yasg`) and configure Swagger UI at `/api/docs/`.
- Document all APIs with descriptions, parameters, and response formats.

2.5 Testing

- Test APIs in Postman, capturing screenshots of:
  - Successful comment creation.
  - Paginated comment list for a post.
  - Updating and deleting a comment.
  - Searching comments across fields (test multiple keywords).
  - Error cases (e.g., unauthorized access, no results).

2.6 Optional Frontend

- Create templates for:
  - A form to add/edit comments under a post.
  - A paginated comment list for a post.
  - A search bar to filter comments by keyword.
- Add CSS (e.g., comment threads, search results layout).

3. Expected Outcomes

- Authenticated users can manage comments (create, update, delete).
- All users can view comments and search across multiple fields.
- Search returns relevant, paginated results.
- APIs return correct responses (e.g., 201 for creation, 403 for unauthorized).
- API documentation is accessible and complete.
- Optional: Frontend is functional and user-friendly.

4. Submission Instructions

- Screenshots of:
  - Database table showing comments with relationships.
  - Postman requests for all APIs (create, list, update, delete, search).
  - Swagger UI for API documentation.
  - Frontend pages with search and pagination (if implemented).
- Zip your Django project folder (including templates if added).
- Submit by end of day, October 7, 2025.

5. Resources

- Django Q Objects
- drf-yasg Documentation
- Postman for API testing
- Install: `pip install drf-yasg`
- Contact your team lead for support

6. Sample API Response Formats

6.1 Create Comment (POST /comments/create/)

Request:

```
{
  "post_id": 1,
  "content": "Great post about Django!"
}
```

Response (201 Created):

```
{
  "status": "success",
  "comment": {
    "id": 1,
    "content": "Great post about Django!",
    "author": "username",
    "post_id": 1,
    "created_at": "2025-09-26T09:30:00Z"
  }
}
```

6.2 Search Comments (GET /comments/search/?q=django)

Response (200 OK):

```
{
  "status": "success",
  "comments": [
    {
      "id": 1,
      "content": "Great post about Django!",
      "author": "username",
      "post_title": "Django Tips",
      "created_at": "2025-09-26T09:30:00Z"
    },
    ...
  ],
  "pagination": {
    "current_page": 1,
    "total_pages": 2,
    "total_comments": 15
  }
}
```

6.3 Error Case (403 Forbidden)

Response (e.g., unauthorized update):

```
{
  "status": "error",
  "message": "You are not authorized to edit this comment."
}
```

7. Support

Reach out via email or team chat for clarification. A Q&A session will be scheduled if needed. Good luck!