# Mini Blog API — Project Documentation

## Project Overview

Mini Blog API is a lightweight Django + Django REST Framework project that implements a blog backend with Posts and Comments. The project supports JWT authentication, input validation, pagination, and full CRUD operations for posts and comments. It is designed for evaluation of Python fundamentals and REST API skills.

## Quick Facts

**Stack:** Python, Django, Django REST Framework, SimpleJWT, SQLite  
**Project structure:** Django project "**MiniBlogAPI**" with app "blog"  
**Authentication:** JWT (Bearer tokens) using djangorestframework-simplejwt

## Setup & Run (Development)

1) Create & activate virtual environment (recommended).

2) Install dependencies:  
 pip install django djangorestframework djangorestframework-simplejwt

3) Apply migrations:  
 python manage.py makemigrations  
 python manage.py migrate

4) Create a superuser (optional):  
 python manage.py createsuperuser

5) Run server:  
 python manage.py runserver  
 API root: http://127.0.0.1:8000/api/

## Data Models (Django ORM)

- Post: title (max 200), content (max 2000), author (FK to User), created\_at (auto\_now\_add)  
- Comment: post (FK to Post), text (max 500), author (FK to User), created\_at (auto\_now\_add)

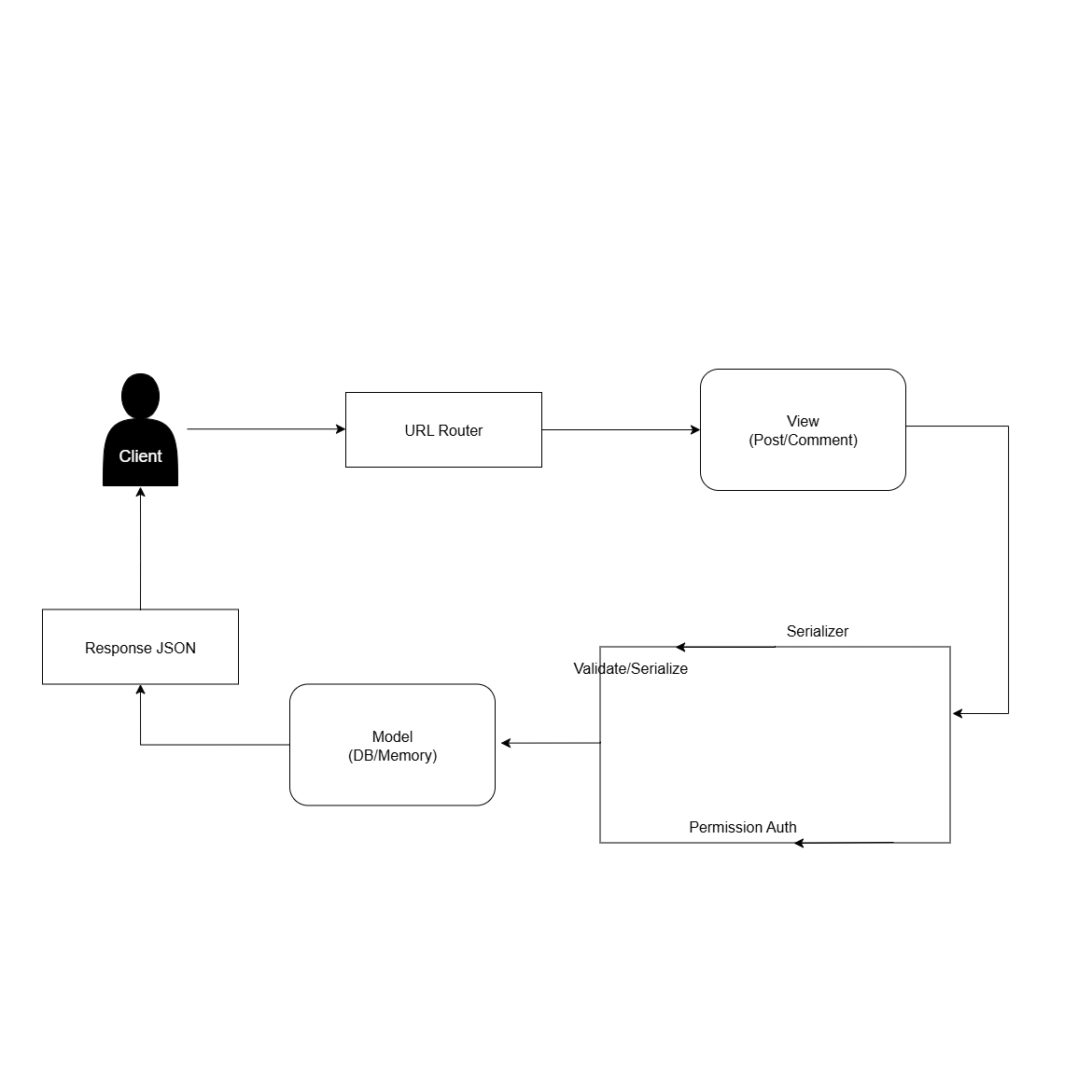
## Serializers

- PostSerializer: id, title, content, author (read-only username), created\_at, nested comments  
- CommentSerializer: id, text, created\_at (author and post set in views)

## Authentication (JWT)

Tokens via /api/token/ (username & password). Use the access token in Authorization header:  
Authorization: Bearer <access\_token>  
Use /api/token/refresh/ to refresh.

**Project Structure:**

****

## API Endpoints & Examples

POST /api/token/ -> get access & refresh tokens

POST /api/token/refresh/ -> refresh access token

POST /api/posts/ -> create post (Auth required). Body: {"title":"..","content":".."}

GET /api/posts/?page=1&page\_size=5 -> list posts latest-first

GET /api/posts/<id>/ -> get single post with nested comments

PUT /api/posts/<id>/ -> update post (author only). Body: {"title":"..","content":".."}

DELETE /api/posts/<id>/ -> delete post (author only)

POST /api/posts/<id>/comments/ -> add comment (Auth). Body: {"text":".."}

PUT /api/comments/<id>/ -> update comment (author only). Body: {"text":".."}

DELETE /api/comments/<id>/ -> delete comment (author only)

## Validation Rules

- title: required, max 200  
- content: required, max 2000  
- comment.text: required, max 500  
- Pagination page/page\_size positive integers  
- Update/Delete restricted to resource owner

## Testing Flow (Postman)

1) POST /api/token/ with username/password -> receive tokens

2) Set Authorization: Bearer <access\_token>

3) POST /api/posts/ to create posts

4) GET /api/posts/ to verify ordering/pagination

5) POST /api/posts/<id>/comments/ to add comments

6) PUT/DELETE on /api/comments/<id>/ to modify/delete comments (author only)

## Database & Migrations

Default: SQLite (db.sqlite3). Use:  
python manage.py makemigrations  
python manage.py migrate  
To inspect: python manage.py dbshell and SQLite commands .tables and .schema <table>

## Troubleshooting

- 401 Unauthorized: check Authorization header (Bearer token) or token expiry  
- 400 Bad Request: check JSON keys vs serializer fields (e.g., 'text' for comments)  
- 500 Internal Server Error: check server trace in terminal for exact cause

This documentation for **Mini Blog API**

Date: 2025-09-25