Submitted by,

Nibin Joseph

24pmc137

Django Blog Project

A modern, feature-rich blog platform built with Django that supports user authentication, post management, comments, and likes.

Features

User Management

- User registration and authentication
- Email-based login system
- User profiles
- Admin dashboard for site management
- Forgot password
- Verify otp
- Change password

Blog Posts

- Create, edit, and delete blog posts (Admin only)
- Categorize posts
- View post details with related posts
- Track post views
- Search functionality

Comments and Interactions

• Comment on blog posts (No admin approval required)

- Like/unlike posts
- Real-time comment updates using AJAX

Admin Features

- Dashboard with statistics(custom admin)
- Post management
- User management
- Category management

Technical Stack

• Backend: Django 4.2

• Frontend: HTML5, CSS3, JavaScript

• **Database**: SQLite (default)

• Authentication: Django's built-in authentication system

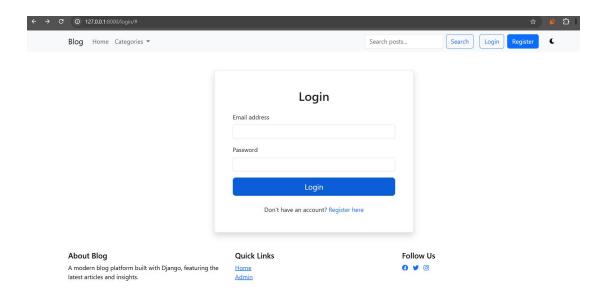
• AJAX: For real-time interactions

Github Repo Link

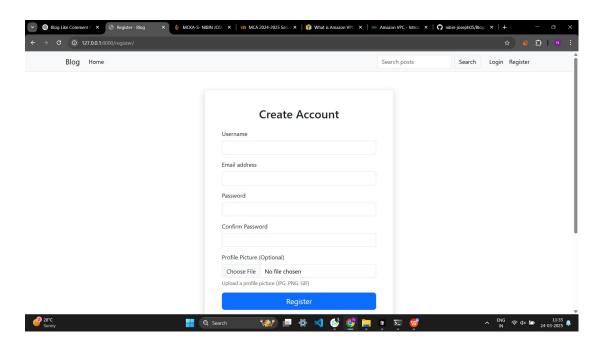
https://github.com/nibin-joseph05/Blog

ScreenShots

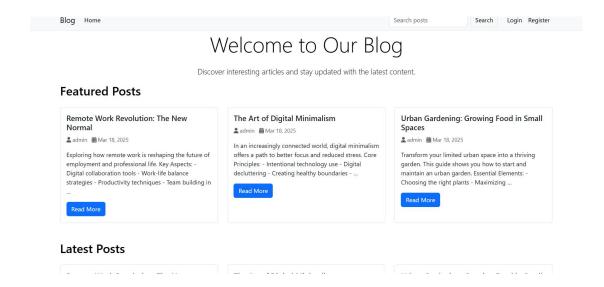
login



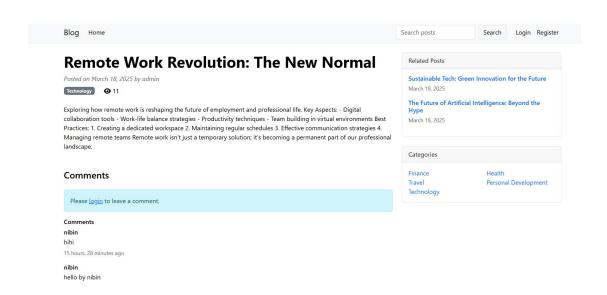
Register



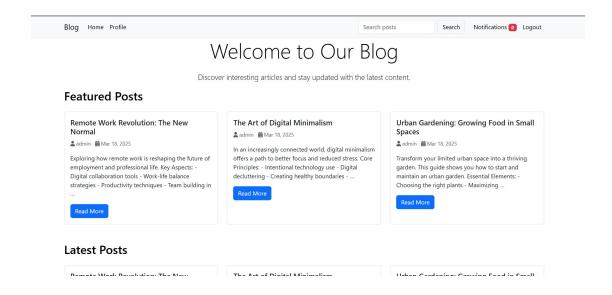
Home



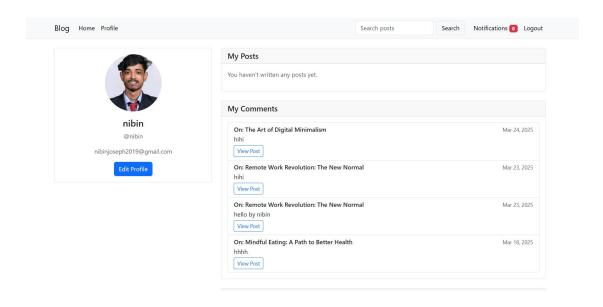
Blog detail



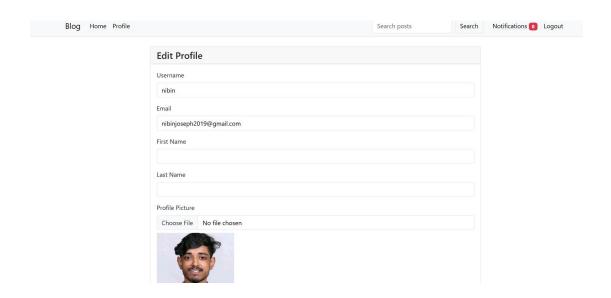
User page



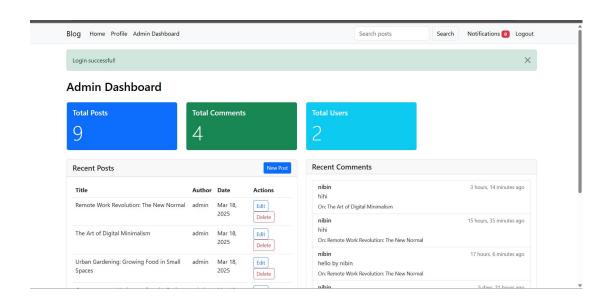
User profile



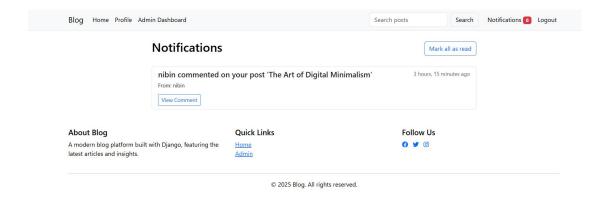
Edit profile



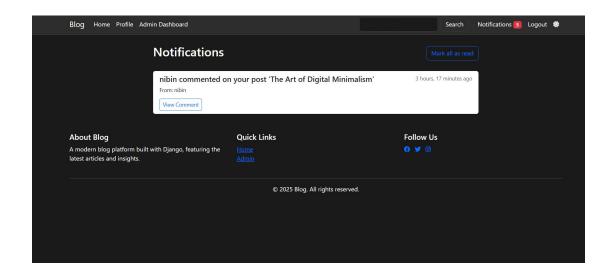
Admin dashboard



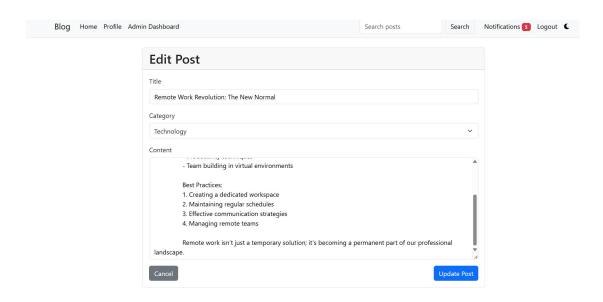
Notification



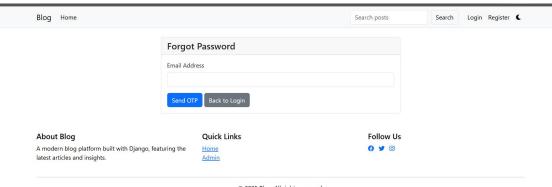
Dark mode



Edit post

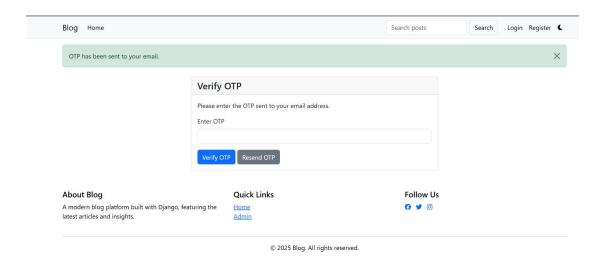


Forgot password



© 2025 Blog. All rights reserved.

Verify otp



Change password



• Challenges & Learnings: Summarize key challenges faced and how you solved them.

> Handling User Interactions (Likes & Comments Without Admin Approval)

Challenge: Initially, the blog required admin approval for comments, which was not user-friendly.

Solution: Modified the comment model to automatically approve comments upon submission, allowing seamless interaction.

• Efficiently Managing Likes & Comments

Challenge: Implementing a dynamic like/unlike system and real-time comment updates.

Solution: Used AJAX for asynchronous requests, enabling instant updates without reloading the page.

· Ensuring Secure and Scalable Authentication

Challenge: Handling user authentication securely while keeping the system scalable

Solution: Utilized Django's built-in authentication and session management, ensuring data security and easy scalability