Design and Implementation of Personal Blog System

Yang Hao

Ulster College at Shaanxi University of Science & Technology

May 27, 2025

1 Background

2 Project Architecture

3 Project Implementation

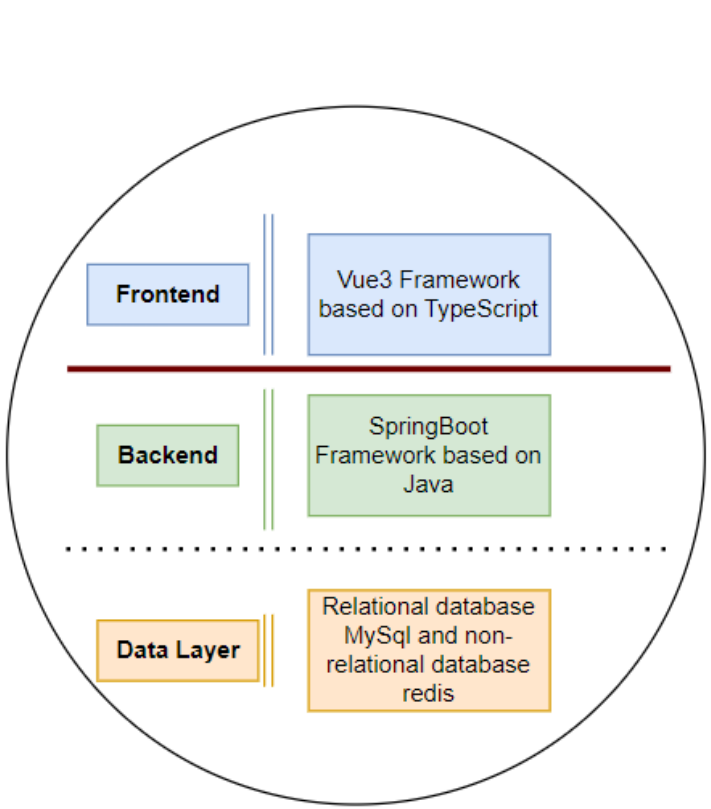
4 Functionality Verification

5 References

1 Background

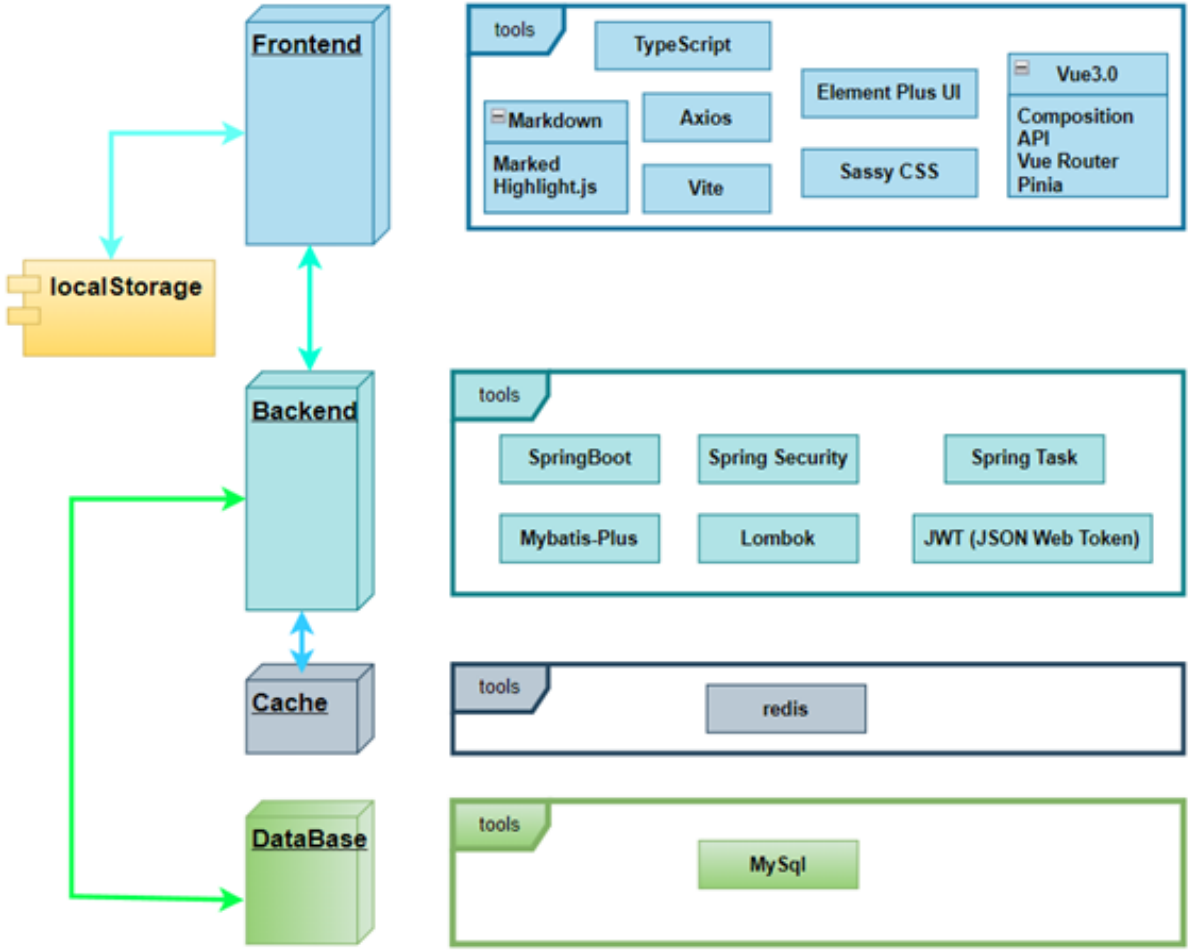
With the development of the internet, blogs have become an important platform for information sharing and personal expression. However, traditional systems often fall short in terms of interface design, user interaction, and maintenance efficiency. The rise of large language models has further intensified competition among blog platforms. In this context, **the combination of Vue3 and Spring Boot offers a new approach to building high-performance**, easily maintainable blog systems, providing both practical application value and support for research in web architecture.

Advantages of This Project Implementation



- Flexible
- Efficient
- Community richness
- Easy to use
- Stable

2 Project Architecture



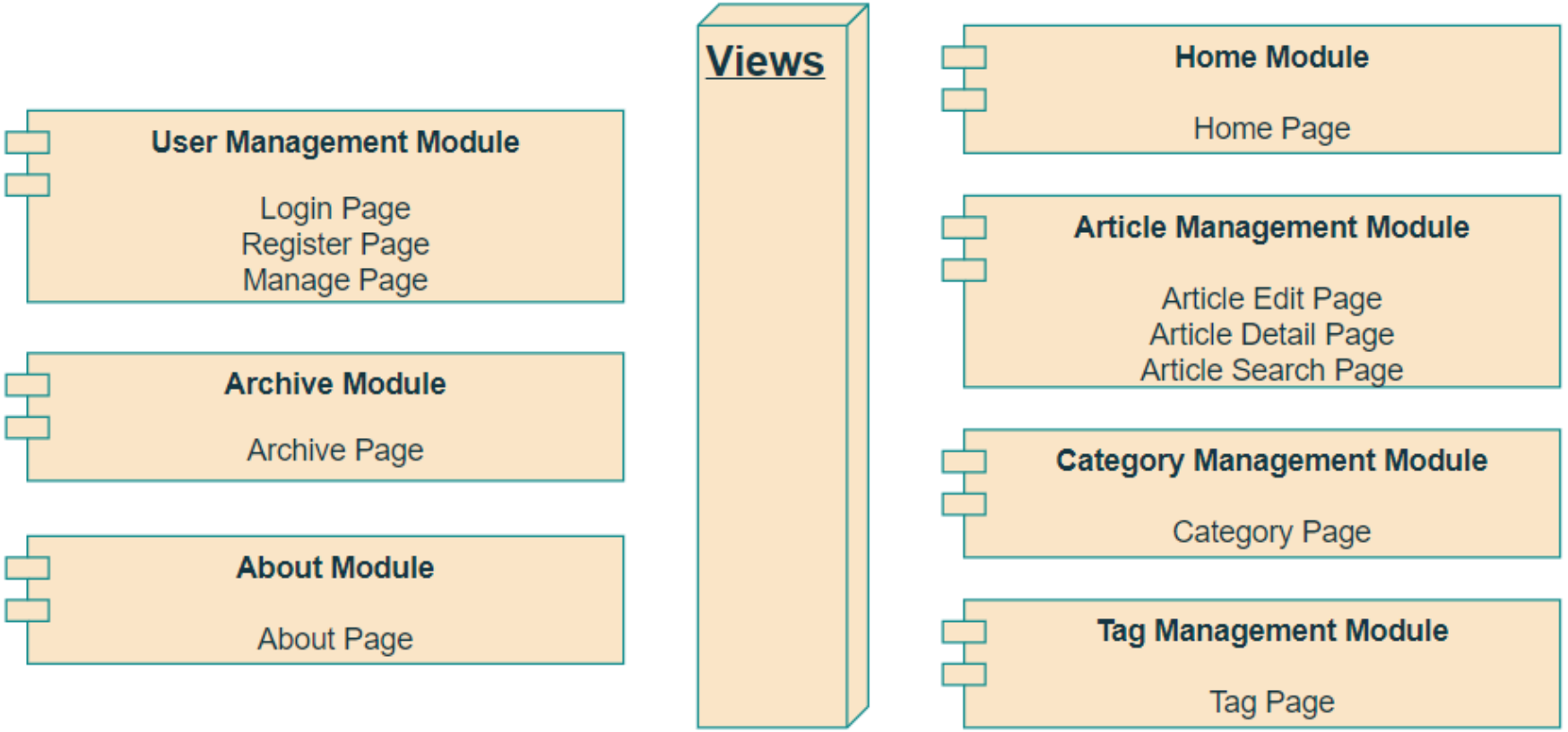
The front end of this blog system is built with Vue3 and TypeScript, utilizing Element Plus, Pinia, and Axios for UI construction and data interaction. The back end is developed with Spring Boot, integrating Spring Security and JWT for access control, MyBatis-Plus for simplified data operations, Redis for performance enhancement, and MySQL for data storage.

During the project development, Git was uniformly used for version control. The front end and back end were developed using VSCode and IDEA respectively in a Windows environment, while the database was hosted in a Linux environment.

3 Project Implementation

Front-end Section

The front end adopts a **templated and component-based design**, utilizing Pinia for state management and Vue Router for routing control. It is divided into modules such as article browsing, editing, management, category and tag management, user authentication, and search. Each component follows the single-responsibility principle and communicates through Props and Events, enhancing the system's reusability and maintainability.



The back end is built on Spring Boot using a layered architecture. Requests are preprocessed by the Filter and then forwarded by the Controller layer to the Service layer, which handles business logic with transaction control and caching mechanisms to ensure performance and consistency. The data layer, implemented with MyBatis-Plus, supports efficient database operations, including pagination and complex queries. The system integrates global exception handling, scheduled tasks, and monitoring, resulting in a clear and organized structure.

Password Encryption

The front end applies MD5 encryption to the password before transmission, while the back end performs secondary encryption and verification using Spring Security's BCrypt.

Transactional Login

The user login method uses the @Transactional annotation to ensure atomicity, preventing inconsistent states caused by authentication or cache failures.

Access Control

The system implements fine-grained access control by combining URL patterns with HTTP methods, with all access paths centrally managed in the SecurityConfig.

Global Rate Limiting

The system implements global rate limiting based on IP, using a custom filter to restrict each IP to a maximum of 50 requests per 60 seconds; requests exceeding the limit return a 429 status.

MySQL

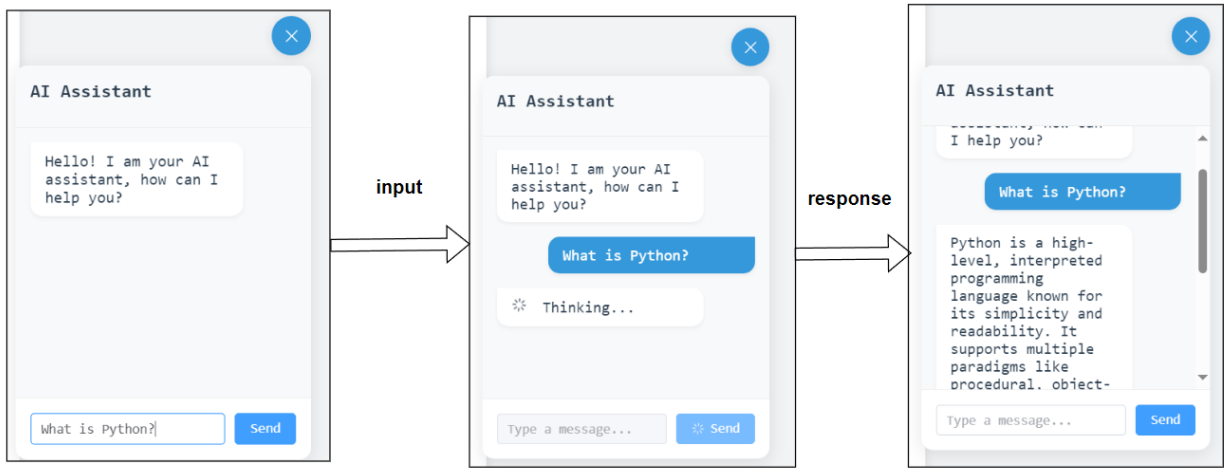
- Store core data such as users, articles, and categories.
- Utilize MyBatis-Plus to achieve efficient querying and pagination.
- Implement an RBAC model to ensure data consistency and secure access control.

Redis

- Cache hotspot data such as user information, verification codes, and view counts to reduce database load and improve response speed.
- Enhance system resilience and fault tolerance by supporting IP rate limiting and persistence mechanisms, thereby improving overall security and stability.

Large Language Model Dialogue

Users can open the chat interface via a sidebar button. The system, combined with backend prompt engineering, supports intelligent Q&A for programming-related questions, enhancing user interaction and practicality.



Front-end

- Global Component: Fixed in the top right corner, collapsible and expandable, unobtrusive when browsing blog content.

Back-end

- Implement the backend functionality of the large model dialogue in the plugin directory.
- Inject the API key and model parameters uniformly from the configuration file using annotations.

LLM Invocation

- Simply modify the configuration file to enable the use of the large model dialogue.
- The interface is highly extensible, supporting API calls as well as local Ollama and Huggingface invocations.

4 Functionality Verification

Unit tests were conducted on the core functions of the article service in the back end, using Mockito to mock dependencies and verify the correctness of method logic and invocations.

Unit Testing

Tested the functionality, performance, and security of the system's APIs, verifying correct responses and data return for article CRUD and tag/category retrieval features.

API Testing

Verify interface responses and data synchronization by simulating user actions to ensure complete and accurate front-end display.

Front-end and Back-end Interaction Testing

5 References

[1] Zhong, Y. & Guo, Y. (2021). Design and implementation of a blog management system based on Springboot. *Modern Information Technology*, 5(7), pp.18–20, 24.

[2] Mo, W., et al. (2024). Research and application of a personal blog platform based on Spring Boot technology. *Science and Technology Wind*, (14), pp.94–96.

[3] Wang, H. & Zhang, L. (2024). Design and implementation of a web-based microblogging system. *Computer Knowledge and Technology*, 20(17), pp.65–68, 77.

[4] Dong, Z. & Pereira, C. (2024). Full-stack website independent development technology development. *Wireless Internet Technology*, 21(24), pp.51–56.

[5] Li, J. (2023). Analysis of computer software testing technology and development application strategies. *Information Recording Materials*, 24(3), pp.50–52.

[6] Wang, Q. & Chen, L. (2025). The application of value engineering in software engineering. *Hebei Enterprises*, (1), pp.69–73.

[7] Cui, P. (2019). Security issues of Java platform and application Java technology. *Information and Computer (Theory Edition)*, (15), pp.160–161.

[8] Spring Team. (2025). *Spring Boot documentation*. Retrieved from https://docs.spring.io/spring-boot/index.html

[9] Wei, C. & Zhang, R. (2024). Design and implementation of a student information management system based on Vue 3 and SpringBoot. *Computer Programming Techniques and Maintenance*, (10), pp.3–6, 20.

Thank you for your time and attention.

May 27, 2025