Design and Implementation of Blog System Based on B/S Architecture

Yue Wang

School of Computer and Software, Jincheng College, Sichuan University, Chengdu 611731, China

Abstract: ***This article briefly describes a knowledge sharing blog app designed and developed based on the Vue framework. This app is based on the concept of "easy to share, beautiful experience" and draws on the advantages of high-quality blog apps. It is a knowledge sharing blog app that is easy to operate and has simple and beautiful pages. The front-end part of this project uses Vant components, Element components, and Vue framework, while the back-end framework adopts Express framework and MySQL database management system to store data. The running environment is Node.js, making it an efficient and convenient lightweight blog app. This app has functions such as viewing articles, publishing articles, logging in and registering, collecting, commenting, and liking.***

Keywords: Blog App; Vue; MySQL; Express; Vant.

1. INTRODUCTION

With the development of the Internet, the means for people to obtain information has changed from books to electronic products. While obtaining information, people can also express their views on the Internet. But these contents are disorganized and scattered across various forums and websites. People have to rely on search engines to retrieve several pieces of information, and then find the parts they need from this information. The emergence of blogs has solved this problem very well. People can not only search for the information they need by keywords and categories on blogs, but also express their opinions on blogs, which is convenient for communication and future viewing. People can share their professional knowledge and also share the beautiful moments in their lives. This article will summarize and introduce the implementation process of the app, including the main selected technologies, user needs analysis, and functional modules. Pang et al. (2024) unveiled diabetes knowledge and conducted risk prognosis by leveraging electronic health record (EHR)-based data using a data-driven approach [1]. Yin et al. (2024) employed deep learning techniques for classifying crystal systems in lithium-ion batteries, enhancing the understanding and performance of these energy storage devices [2]. Luo et al. (2024) improved e-commerce chatbots using Falcon-7B and 16-bit full quantization, resulting in more efficient and effective customer interactions [3]. Liu et al. (2024) explored the impact of supply chain and digitization on environmental technology development, highlighting the influence of inflation and consumption in G7 nations [4]. Li (2025) optimized clinical trial strategies for anti-HER2 drugs through Bayesian optimization and deep learning, potentially improving treatment efficacy and patient outcomes [5].

Jin et al. (2021) introduced FAST, an FPGA-based subgraph matching system for massive graphs, demonstrating its efficiency and scalability [6]. Yang et al. (2021) proposed HUGE, another subgraph enumeration system that is both efficient and scalable, addressing challenges in graph data processing [7]. Chen et al. (2023) presented a channel-aware 5G RAN slicing framework with customizable schedulers, enhancing network flexibility and performance [8]. Peng et al. (2024) developed a dual-augmentor framework for domain generalization in 3D human pose estimation, improving the robustness and adaptability of pose estimation models [9].

Bi et al. (2024) discussed the role of AI in financial forecasting, particularly the potential and challenges of using ChatGPT for this purpose [10]. Zhou et al. (2024) optimized an automated garbage recognition model based on ResNet-50 and weakly supervised CNN, contributing to sustainable urban development [11]. Fan et al. (2025) researched an online update method for retrieval-augmented generation (RAG) models with incremental learning, aiming to improve model performance over time [12]. Xu et al. (2025) introduced AI-enhanced tools for cross-cultural game design, supporting online character conceptualization and collaborative sketching, which may facilitate the creation of more diverse and engaging games [13].

2. INTRODUCTION TO MAIN TECHNICAL SELECTION

2.1 Vue.js

Vue.js is a lightweight, component-based MVVM library that is easy to use and can better organize and simplify web development.

2.2 Element Desktop Component Library

Element UI is a set of UI libraries used in conjunction with Vue. Element simplifies the encapsulation of commonly used components and improves the usability of backend systems.

2.3 Node.js

Node.js is a runtime environment for the Javascript language.

2.4 Requirement Analysis

This design is a knowledge sharing blog app aimed at providing a platform for netizens to share knowledge. This design refers to the functional design of popular apps of the same type, targeting all age groups, with a focus on the young group of 15 to 35 who use electronic products the most.

Based on the analysis of current popular dating and sharing software, this project needs to have the following functions:

User functions: Login, register, exit account, modify personal information, set different permissions for different types of visitors.

Article function: Publish articles, view articles, and categorize articles.

User interaction function: Comment on articles, like articles, bookmark articles, follow other users.

3. PROJECT DESIGN

3.1 Module Design

This app is divided into six modules for module development and design: message management module, comment management module, collection management module, classification management module, user management module, and content management module. The purpose is to provide users with a concise, practical, and convenient online sharing blog app. Users can view article content without logging in, but cannot comment, like, follow, publish articles, and other functions. When an unregistered user clicks on these functions, the app will redirect to the login page to remind the user to log in. The module design is shown in Figure 1.

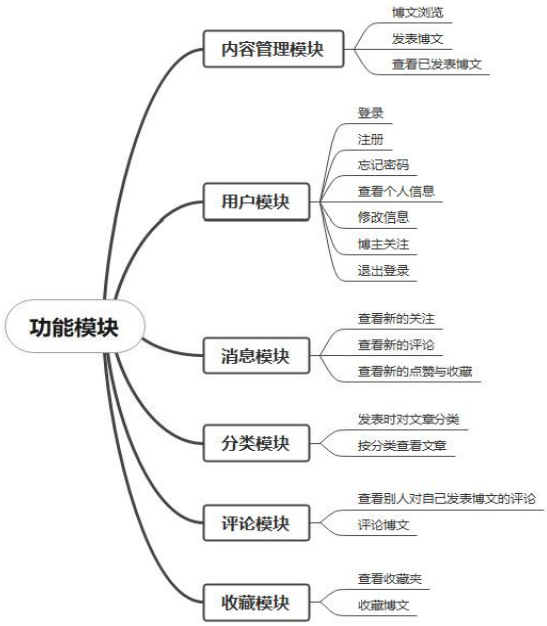


Figure 1: Functional Module Design

3.2 Process Design for Publishing Articles

After logging into the account, users can publish articles. In the article publishing interface, they can enter the title and content of the article, select the category, upload images (images can be omitted), and click the publish button. The front-end will transfer the data to the back-end through Axios and jump to the successful publication interface. The uploaded image size cannot exceed 2MB and must be in JPG format.

3.3 Database Design

Six tables were designed based on the requirements of this project:

(1) User table: User ID, username, user login password, user profile picture, ID for posting blog posts, gender, age.

(2) Article Table: Blog ID, username, blog title, blog image, number of likes, blog content, blog type, user ID, blog publication date.

(3) Like table: like username, like username, liked username, like date, like user avatar, like blog ID, like blog title.
(4) Collection table: Collection blog ID, user ID, username, blog title, collection time, blog content.

(5) Follow Table: Username, User ID, Followed User ID, Followed User ID, User Avatar, Follow Date, Followed User ID, Followed User Avatar.

(6) Comment form: Blog ID, username, user avatar, time of user comment, content of user comment, ID of the commented user, user ID, article content.

3.4 Backend Design

The backend is responsible for storing data in the database and accessing or modifying data in the database when needed. The backend connects to the database through creatConnect() to operate on the database level and return a Connection object. The front-end can query whether a user has already registered by accessing the user table in the database.

The routing jumps used in this project include $router. replace() and $router. push(). The difference between the two is that push() can add a history record locally, while replace() does not add it and jumps directly. The data is obtained through Axios, using a unified POST request. After obtaining data from the page, it is stored in the store

of the state management library. When the page refreshes or triggers certain events, the changed data is placed in the store through pre-set functions to ensure dynamic updates of the page.

4. PROJECT IMPLEMENTATION

4.1 User Management Module

This module implements user login and registration functions. Users who have not registered before can register on the registration page and log in after successful registration. Users with accounts can directly enter their account and password on the user page to log in.

When users click on the message interface, article publishing interface, and personal center interface in the app, the interface will remind them to log in. Non logged in users only have the permission to browse blog posts.

When users log in, they need to enter their account and password. After clicking login, the server will search for the corresponding user ID in the status management database. If it exists, the password will be matched. If the password does not match or the user does not exist, the server will prompt the user to enter the correct username and password.

Unregistered users can click on the registration link to register.

After the user inputs an account, the server will search in the state processor to see if the account already exists. If it exists, a prompt will be displayed. The format for uploading avatar images only allows JPG format and cannot exceed 2MB. After registration is completed, it will redirect to the successful registration page and display the user ID of the current user.

Users can modify their personal information in the personal center, and the gender selection in the personal information modification interface uses the<el radio>(single selection box) in the Element component.

4.2 Classification module

This module divides articles into several categories for easy user search.

When users upload blog posts, they need to select a category. Each blog post can only have one category, which is divided into "sports", "knowledge", "entertainment", and saved as the message_date field in the database.

Users can browse articles of interest through different categories on the homepage.

Classification also increases the connectivity between different articles. After a user reads an article, the system defaults to pushing articles of the same category first.
When users publish articles, they must select a category, and the articles are stored in the data options by category. Using v-for, all categories are rendered to the<el option>option list. The content of the article can include images, which are limited to JPG format and cannot exceed 2MB.

4.3 Content Management Module

This module is used to process the content of articles. The articles published by users will be stored in the database. When users browse blog posts, the frontend will request the backend database to render the article title, content, classification, etc. into the frontend interface.

The content searched by the user on the homepage will be saved to the backend. When clicked again, the search history will be displayed. Each search will determine whether it is the same as the data saved in the state library. If the search content does not exist, it will be stored in the history record. If the content of the search box exists in the history record, the previous old record will be deleted and the new search value will be put back into the array header.

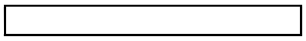
All articles on the homepage of the website will be sorted in chronological order, with the most recent ones placed at the top using blogger - choice(). By default, the homepage will display the first few lines of text for users to

choose whether they are interested in the content they want to read. Clicking on the content they want to read will enter the article page and view all the content.

Articles can be shared with just one click through various channels such as WeChat, Weibo, duplicate links, and QR codes.

4.4 Message module

This module is used to process user messages and display unread messages on the message interface.
In the message interface, logged in users can view the messages they have received, including comments, likes, favorites, and followers.



This page uses v-for to loop and output dynamically changing messages, traversing the comment data in the database and rendering it into the li tag of ul.

For the sake of page aesthetics, three Vant components are introduced, with class names "like" and "follow" as icons for bookmarking, liking, and following, respectively. There is also a small red dot style icon as a marker to distinguish whether a comment has been read or unread. When the user clicks on the comment to enter the corresponding blog post, the index corresponding to the comment is added to the state library index queue. The v-if is used to determine that the index already exists, and the icon corresponding to the comment is hidden to indicate that it has been read.

The front end saves the number of comments requested as comment_info, and the number of followers as concern_info. After obtaining the favorite data, it calls then() to obtain the like data. The two data are merged and saved as info in the state using the array method. The reason for merging is that the tips for favorite likes are put on the same page and need to be merged when displaying the data length. When rendering the page, check if there is any data. If there is no data, it means it is the first time rendering. Simply put the obtained data into the state library. If there is data, it means the page has already been rendered, not the first time rendering. At this time, put the new data into the state library and change the length of the corresponding data, for example, the length of the comment data is comment_info_num. The length of the saved data is for displaying how many messages there will be in the future.

After clicking the All Read button, call read() to save the current number of comments. Call len() to set len, the number of comments, to 0. At the same time, set the number of likes and collections to 0. At this time, there is no comment data in v-for, and there is no data for likes, collections and comments. The number of unread message prompts is the number of collections and likes plus the number of concerns and comments. At this time, it is also 0.

4.5 Comment module

This module stores the relevant data of user comments and articles.

The comment module increases the interactivity between users. Users can post comments in the comment section under the blog content, and the comments received from other users can be viewed in the message interface.

4.6 Favorite and Like Module

This module stores data related to user favorites and likes of articles.

After collecting blog posts, users can view their already collected blog posts in the user center. The backend determines whether a user has collected a blog post by comparing the blog post ID and user ID in the blog post collection table in the database. After the user clicks on the bookmark, the front-end will save the blog ID, user ID, username, user avatar, bookmark date, blog title, and blog content in the bookmark table.

The difference between favorites and likes is that users cannot see the number of favorites, but they can see the number of likes before liking. After bookmarking an article, users can view it again in their own favorites, but they cannot view the articles they have liked. Thumbs can serve as an important reference value for article popularity, and when the system pushes to users, it will also prioritize pushing articles with high likes.

5. CONCLUSION

This system adopts a B/S architecture and uses a modular division method to complete functions such as user login and registration, article browsing and publishing, liking and following. After testing and verification, this system has the characteristics of simple interface, clear functions, and easy operation, which can provide users with a good online reading and dating experience. The current problem with this system is that blog recommendations and hot lists are calculated based on popularity values, which only add likes and favorites, without considering page views and the number of blogger fans, which is not accurate enough. In the future, the likes, favorites, views, and number of followers of the blogger will be calculated to obtain more accurate popularity values, providing users with a better reading experience.

REFERENCES

[1] Pang, H., Zhou, L., Dong, Y., Chen, P., Gu, D., Lyu, T., & Zhang, H. (2024). Electronic Health Records-Based Data-Driven Diabetes Knowledge Unveiling and Risk Prognosis. arXiv preprint arXiv:2412.03961.

[2] Yin, Y., Xu, G., Xie, Y., Luo, Y., Wei, Z., & Li, Z. (2024). Utilizing Deep Learning for Crystal System Classification in Lithium - Ion Batteries. Journal of Theory and Practice of Engineering Science, 4(03), 199–206. https://doi.org/10.53469/jtpes.2024.04(03).19.

[3] Luo, Y., Wei, Z., Xu, G., Li, Z., Xie, Y., & Yin, Y. (2024). Enhancing E-commerce Chatbots with Falcon-7B and 16-bit Full Quantization. Journal of Theory and Practice of Engineering Science, 4(02), 52–57. https://doi.org/10.53469/jtpes.2024.04(02).08.

[4] Liu, H., Li, N., Zhao, S., Xue, P., Zhu, C., & He, Y. (2024). The impact of supply chain and digitization on the development of environmental technologies: Unveiling the role of inflation and consumption in G7 nations. Energy Economics, 108165.

[5] Li, T. (2025). Optimization of Clinical Trial Strategies for Anti-HER2 Drugs Based on Bayesian Optimization and Deep Learning.

[6] Jin, X., Yang, Z., Lin, X., Yang, S., Qin, L., & Peng, Y. (2021, April). Fast: Fpga-based subgraph matching on massive graphs. In 2021 IEEE 37th international conference on data engineering (ICDE) (pp. 1452-1463). IEEE.

[7] Yang, Z., Lai, L., Lin, X., Hao, K., & Zhang, W. (2021, June). Huge: An efficient and scalable subgraph enumeration system. In Proceedings of the 2021 international conference on management of data (pp. 2049-2062).

[8] Chen, Y., Yao, R., Hassanieh, H., & Mittal, R. (2023). {Channel-Aware} 5g {RAN} slicing with customizable schedulers. In 20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23) (pp. 1767-1782).

[9] Peng, Q., Zheng, C., & Chen, C. (2024). A Dual-Augmentor Framework for Domain Generalization in 3D Human Pose Estimation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 2240-2249).

[10] Bi, S., Deng, T., & Xiao, J. (2024). The Role of AI in Financial Forecasting: ChatGPT's Potential and Challenges. arXiv preprint arXiv:2411.13562.

[11] Zhou, Y., Wang, Z., Zheng, S., Zhou, L., Dai, L., Luo, H., ... & Sui, M. (2024). Optimization of automated garbage recognition model based on resnet-50 and weakly supervised cnn for sustainable urban development. Alexandria Engineering Journal, 108, 415-427.

[12] Fan, Y., Wang, Y., Liu, L., Tang, X., Sun, N., & Yu, Z. (2025). Research on the Online Update Method for Retrieval-Augmented Generation (RAG) Model with Incremental Learning. arXiv preprint arXiv:2501.07063.

[13] Xu, Y., Shan, X., Lin, Y. S., & Wang, J. (2025). AI-Enhanced Tools for Cross-Cultural Game Design: Supporting Online Character Conceptualization and Collaborative Sketching. In International Conference on Human-Computer Interaction (pp. 429-446). Springer, Cham.