渭南师范学院

本科毕业设计

题　　目：基于 Vue+Midway 的个人主页设计与开发

学　　院：计算机学院

专业班级：计专升本 2022 级 2 班

毕业年份：2024 年

姓　　名：郭佳龙

学　　号：221321060

指导教师：哈渭涛

职　　称：教授

第二导师：纪宁

工作单位：渭南市高新区第一小学

渭南师范学院教务处 制

学位论文原创性声明

本人声明所呈交的学位论文是我在导师的指导下进行研究工作所取得的研究成果。尽我所知，除文中已经注明引用的内容和致谢的地方外，本论文不包含其他个人或集体已经发表或撰写过的研究成果，也不包含本人或他人已申请学位或其他用途使用过的成果。对本文的研究做出重要贡献的个人和集体，均已在文中作了明确说明并表示谢意。

本学位论文若有不实或者侵犯他人权利的，本人愿意承担一切相关的法律责任。

作者签名： **日期：** 年 月 日

学位论文知识产权及使用授权声明书

本人在导师指导下所完成的学位论文及相关成果，知识产权归属渭南师范学院。本人完全了解渭南师范学院有关保存、使用学位论文的规定，允许本论文被查阅和借阅，学校有权保留学位论文并向国家有关部门或机构送交论文的纸质版和电子版，有权将本论文的全部或部分内容编入有关数据库进行检索，可以采用任何复制手段保存和汇编本论文。本人保证毕业离校后，发表本论文或使用本论文成果时署名单位仍为渭南师范学院。

保密论文解密后适用本声明。

作者签名： **日期：** 年 月 日

摘　要

在互联网时代，个人主页扮演着重要的角色，是展示个人信息、技能和成就的关键平台。通过本文的研究和探索，旨在实现一个功能完善、易用且高性能的个人主页系统。

首先，在论文中全面分析了个人主页的需求性、适用人群、意义及目的。接着，着重介绍了 Vue 框架、Midway 框架以及 MongoDB 数据库。在系统设计与架构部分，对系统的需求进行了分析，并将其划分为不同的功能模块，详细描述了技术选型和系统架构的设计过程，以确保系统的稳定性和可扩展性。同时，详细介绍了前端和后端的开发过程。此外，通过对数据和数据库技术的研究，设计出一套更适合个人主页系统的数据模型，用以承载系统的各种数据。

最后，对系统进行了部署及性能评测，并分析了该系统的优缺点。该个人主页系统设计简洁易用，可作为技术记录站点或日常生活记录站点。在设计过程中，充分考虑了系统的安全性和稳定性，同时兼顾了简约优雅的界面风格，并提供了一定的自定义性。编辑功能基于 Markdown 实现，轻松实现各种功能。前后端都采用 TypeScript 开发，数据库也是基于 BSON 的数据存储格式，实现了一种语言全栈开发，降低了开发难度。

关键词：Vue3；Midway；MongoDB；网站开发；博客系统；TypeScript 全栈

Design and Development of Personal Homepage Based on Vue+Midway

Abstract: In the internet era, personal homepages play a crucial role as key platforms for showcasing personal information, skills, and achievements. Through the research and exploration presented in this paper, the aim is to achieve a feature-rich, user-friendly, and high-performance personal homepage system.

Initially, the paper comprehensively analyzes the requirements, target users, significance, and objectives of personal homepages. Subsequently, it focuses on 0-introducing the Vue framework, Midway framework, and MongoDB database. In the system design and architecture section, the paper analyzes the system's requirements, divides it into different functional modules, and elaborates on the process of technical selection and system architecture design to ensure stability and scalability. Additionally, the development processes of both frontend and backend are detailed. Furthermore, through research on data and database technologies, a data model more suitable for the personal homepage system is designed to accommodate various types of data.

Finally, the system is deployed and its performance evaluated, followed by an analysis of its strengths and weaknesses. The personal homepage system designed is concise and user-friendly, serving as a technical documentation site for geeks or a daily life recording site for various user groups. In the design process, emphasis is placed on the system's security and stability, while also considering a minimalist and elegant interface style with a certain degree of customization. The editing function is implemented based on Markdown, enabling easy implementation of various functions when writing articles. Both frontend and backend are developed using TypeScript, and the database adopts the BSON data storage format, achieving full-stack development in a single language and reducing development difficulty.

KeyWords: Vue3; Midway; MongoDB; website development; blog system; TypeScript full-stack

目　录

1 引言

1.1 开发背景

在互联网时代，个人主页成为展示个人信息、技能和成就的重要平台。虽然现在众多博客网站功能非常完善，易用性也非常高，拥有着精美的界面和流畅的操作，但是它也存在着自己部署的个人网站所没有的特点。

在自己部署的个人网站中，用户可以拥有数据的所有权。在工作和思考中产生的内容，一般具有比较高的私有性和归属性，这类数据如果存放在服务商的数据库中，虽然很难出现数据丢失的情况，把数据存在服务商反而能够降低数据的丢失风险，但是数据的控制权和一部分使用权却在服务商手中，用户很难决定数据如何存储，存储在哪，以及如何展示和分析自己的数据；如果使用私有部署的个人网站，用户则可以随时彻底的删除敏感数据，这是在服务商基本不存在的权力，因为在比较规范的数据库设计中，是尽量避免数据删除的，删除操作实质上是给数据添加一个删除标识，数据只是业务上删除了，物理上实际并没有删除，这就导致了你想彻底删除一条数据，但是服务商不会把你数据彻底销毁。

同时，用户还拥有高度的自定义功能。在服务商提供的产品上，基本上都是服务商自己的特色，也许很多功能和交互让用户使用起来非常的舒服，但毕竟众口难调，总有一些用户有一些自己的想法，所以部署自己的个人网站可以完全地增加或删除相应的功能、调整交互逻辑、控制业务。

随着互联网飞速地发展，各种计算机技术也在加速迭代，尤其是前端技术，越来越新的技术层出不穷，后端技术和数据库也有比较大的提升。新的技术用上了更优秀的设计理念，更现代的设计逻辑和审美标准，本个人主页系统使用的技术也将是比较前沿、流行并且可靠的技术，这将使这个项目拥有更好的架构，更健壮的系统，以及更好的维护性。

1.2 开发内容

在本项目中，主要是为了记录、分享，然后再加一些实用的工具和额外的功能。记录分享的内容主要是以博客为主，其他媒体为辅。博客主要是通过富文本来展示的。

网站需要有用户管理系统、博客管理系统、文件管理系统、扩展等功能。用户管理需要有用户信息的持久化，以及各种信息的增删改查；博客管理系统需要有博客数据及信息的持久化，以及各种信息的增删改查；文件管理系统对用户的各种文件做持久化处理，需要有基本的增删查功能，主要是用于存储用户的头像、博客插图等内容，管理员和站长对该模块操作权限较大，用户持有较小的操作权限；扩展功能需要具备良好的可扩展性，对于后期随时加入功能提供良好的基础。

1.3 意义和目的

本项目基于个人主页对于部分人群的必要性，从而开发一款本地化、易用、简约美观的个人主页系统。通过分析了数据的所有权、高度的自定义化，从而确定了个人主页系统为部分人群提供了哪些特有的功能和目的。

同时也为想独立开发个人主页的人群提供了一个开发模板，包括技术栈的分析、系统架构设计、数据库的设计、前后端的设计等，尤其是本项目中使用到的一些特殊的业务处理方式，为广大开发者提供了一些解决方案。

2 技术背景

2.1 技术栈分析

对于本个人网站项目来说，是一个典型的 WEB 项目，技术栈选型是非常容易的，互联网上拥有大量的技术方案，可以直接套用。比如非常典型的例子：前端采用 React，后端采用 Spring，数据库采用 MySQL，部署平台使用 Linux 操作系统，运行时可以使用 Apache 部署前端页面，使用 Tomcat 部署 Java 后端。这是一套成熟稳定的技术栈，久经众多开发者考验的，同时也是拥有非常丰富生态的技术栈，众多大型软件公司迄今依然采用这套技术。

但是互联网发展非常迅速，尤其是前端这种对用户来说非常直观的技术方向，更是日新月异，发展出了很多更易用的、更具现代化的、设计思想更先进的技术框架，比如 Vue 前端框架，他对于开发者来说使用起来很容易，也拥有着良好的设计框架，使项目架构更清晰、更直观、更结构化。对于小型到中型项目完全可以由 Vue 胜任，它的学习难度远小于 React、Angular 等框架，最新的 Vue3 则引入了更底层的组件式编程，提倡使用 TypeScript 类型严格型[1]编程语言进行编程，进一步提高了项目开发的灵活性和健壮性，所以本项目的前端部分则使用 TypeScript 配合 Vue3 进行开发。

借着前端的发展，后端的技术也是不断进步，也发展了许多优异的技术和框架，比如 Go 语言下的 Gin、BeeGo 等框架；Python 下的 Django、Flask 等框架，甚至把前端语言 JavaScript 也搬到了后端，开发出了 Node.js 运行时[2]，使前端开发者也能轻松踏入后端开发领域。但是相对于比较新的 Go 语言来说，JavaScript 的性能表现远远不如 Go 这种和 C 语言媲美的语言。但是他们各有优势，Go 开发 WEB 后端非常容易，且天生支持多线程，能够非常容易开发出高性能的后端程序，但是毕竟是一款新兴语言，没有其他语言沉淀的那么深厚；Python 开发简单，代码量非常少，也拥有很多库的支持，生态是极其完善，但是性能却远远比不上其他平台；JavaScript 语言因为前端的发展，多年占据开发语言流行度榜一，所以就有开发者尝试把 JavaScript 搬到后端，打通前后端开发的壁垒，于是就诞生了 Nodejs 这款强大的运行时，它采用互联网大厂 Google 开发的 V8 引擎，用于解释执行 JavaScript，而 V8 则采用 C++开发，虽然在此引擎上运行 JavaScript

并没有 C++的性能那么好，但是 V8 的优化是非常优秀的，所以 JavaScript 的性能也是非常可观的，至少相对于 Python 来说性能是非常好的，同时 JavaScript 是单线程的，异步[3]做的非常好，它是天生处理高 IO 任务的。除了以上平台，还有像 PHP、JSP 等后端语言，但是它们技术发展缓慢，已逐渐被时代抛弃。考虑到 JavaScript 也能在后端开发，可以一种语言全栈开发，所以本项目采用 JavaScript 的类型严格版 TypeScript 进行后端开发，在 Node.js 运行时上也存在着许多优异的开发框架，比较原生的 express、koa、egg 等框架，更企业级的有 Next.js、Midway 等，Midway 是由国内阿里团队开发的框架，拥有比较完善的中文文档，所以本项目将基于 Midway 框架进行后端开发。

数据库也有很多优秀的产品，比较流行的有 MySQL、SQL Server、SQLite 等，还有 NoSQL 型的 MongoDB、Redis 等，其中 MongoDB 它是文档型数据库，可以存取任意类型的数据，没有格式限制，同时也支持非常丰富的基本数据类型，它的索引数据是缓存在内存中的，存取性能远超 SQL 型数据库，数据是用 BSON 作为数据存储和网络传输格式的，BSON 和 JSON 在结构上非常相似，只是前者以二进制存取，效率更高，空间更小，它也完美兼容 JSON，JSON 则是 JavaScript 原生类型，所以使用 MongoDB 作为数据库是全平台使用 JavaScript 开发非常合适的选择。

整个部署系统则采用 Linux 下的 CentOS 发行版作为操作系统平台，它非常成熟稳定，且性能开销比较小；前端则用 Nginx 部署，它是一款轻量级的 Web 服务器；后端则用 Node.js 运行时部署代码；它们在 CentOS 中都有非常好的支持。

2.2 Vue 框架介绍

Vue 是一个渐进式 JavaScript 框架[4]，用于构建用户界面。由 Evan You 开发并于 2014 年首次发布，Vue 因其简单、灵活和高性能[5]而迅速受到开发者的青睐。它通常与其他前端框架（如 React 和 Angular）进行比较，但以其简约和易集成而脱颖而出。在 Vue3 中官方推荐使用 TypeScript 组合式 API 进行更自由、更灵活的开发。

轻量快速是 Vue 的一大特点，它具有最小的开销，即使对于大型应用程序也是快速高效的；组件默认是响应式的，这意味着它们在数据变化时自动重新渲染，这简化了状态管理，并确保了一致的用户界面；Vue 也提供了定义计算属性和监视器的机制，实现了高效的数据操作和响应性行为。

Vue 拥有一个充满活力和支持的开发者社区，包括开发者、贡献者和爱好者。它拥有丰富的库、工具和资源生态系统，包括 Vuex 用于状态管理、Vue Router 用于路由和 Vue CLI 用于项目脚手架。

总而言之，Vue 提供了一个实用和易用的解决方案，用于构建现代 Web 应用程序。其简单、灵活和高性能使其成为全球开发者的首选。无论是初学者还是经验丰富的开发者，Vue 都提供了创建优雅且高效的用户界面所需的工具和资源。

2.3 Midway 框架介绍

Midway 框架是一款基于 TypeScript 的 Node.js 框架，由阿里巴巴的开发团队开发，它为开发者提供了一种高效、稳定的构建应用程序的方式。开发者可以充分利用 TypeScript 的静态类型检查、智能提示等特性，提高代码的可维护性和可读性。

除了 TypeScript 的支持外，Midway 框架还提供了依赖注入机制，通过容器管理依赖关系，使得应用的组件解耦合，提高了代码的灵活性和可维护性。此外，Midway 框架内置了 Egg.js、Koa.js 等中间件机制，开发者可以方便地编写和使用中间件来处理请求和响应，实现功能的解耦和复用。同时，Midway 框架支持各种插件，包括数据库插件、日志插件、缓存插件等，开发者可以根据实际需求选择和配置相应的插件。

在部署方面，Midway 框架支持多种部署方式，包括传统的单机部署、容器化部署以及 Serverless 部署，满足了不同场景下的部署需求。此外，Midway 框架拥有丰富的生态系统，包括社区插件、工具和文档等，为开发者提供了全面的支持和帮助。

2.4 MongoDB 数据库介绍

MongoDB 是一种流行的 NoSQL 数据库管理系统[6]，它与传统的关系型数据库有着显著的不同。MongoDB 采用面向文档的存储模型，数据以类似 JSON 的 BSON（Binary JSON）格式存储在文档中。这种文档型数据库的设计使得 MongoDB 非常适合存储半结构化和非结构化数据，例如日志、用户配置信息、产品信息等等。

与传统的关系型数据库相比，MongoDB 具有更灵活的模式设计。文档的结构可以随时修改，字段的类型和数量可以根据需求动态改变，无需进行严格的模式定义。这种灵活性使得 MongoDB 特别适用于需求变化频繁的应用场景，能够快速响应业务需求的变化。

MongoDB 还具有优秀的性能和可扩展性。它能够处理大规模的数据和高并发的请求，并支持水平扩展，通过添加更多的节点来实现性能和容量的增加。同时，MongoDB 还支持数据的复制和自动故障转移，通过复制集（Replica Set）来提供数据的冗余备份，确保数据的可用性和持久性。除此之外，MongoDB 提供了丰富的查询语言和操作符，包括等值查询、范围查询、正则表达式、聚合等功能，能够满足各种复杂的查询需求。它还支持地理空间索引和全文搜索功能，可以对地理位置信息和文本进行高效的搜索和分析。

MongoDB 是一款功能强大、灵活且易于使用的 NoSQL 数据库，适用于各种类型的应用程序。它在应对大规模数据、灵活数据模型以及高性能查询等方面表现非常出色。

3 系统设计与架构

3.1 系统需求分析

在个人网站中，核心是信息展示，也就是博客文章的编辑和展示，所以系统的主要需求则是围绕着博客的功能去做。对于其他的附加功能，可以设计一个良好的系统架构和数据结构去支持未来的功能扩展。

关于博客的一些必要功能和额外功能有以下条目：

1. 博客的发布、阅读、修改、删除；
2. 博客的点赞、收藏、评论；
3. 博客的信息展示，如阅读量、收藏量、点赞量；
4. 用户和博客之间的信息，如是否点赞、是否收藏；
5. 展示博客作者的基本信息。

博客需要用户去操作，所以对于用户的必要功能有以下条目：

1. 用户的注册、登录；
2. 用户的鉴权；
3. 用户的基本信息存储与获取。

在博客中，可能会有图片资源，所以需要有对于文件的需求：

1、文件的上传、删除、获取。

个人网站系统不仅限于简单的博客编辑阅读功能，它还可以扩展很多其他的功能，比如博客合集、好友、消息等，所以在设计博客和用户时，需要将未来扩展的功能考虑进去，这主要在数据库设计部分着重研究。

以上是面向用户对系统的使用进行了一个简单的需求分析，我们还需要对一些功能进行详细的需求分析，以完善系统的功能和增加系统的性能。

1. 博客内容需要支持富文本编辑与展示；
2. 注册用户时，需要避免恶意注册刷号；
3. 用户和博客之间的信息需要在拥有大量用户和博客的环境下以很快的速度和较少的服务器资源去获取；
4. 对于用户的博客、用户的收藏、博客的评论、博客的点赞等数据，要考虑超大量的数据如何存储的情况；
5. 文件存储需要考虑不同用户上传相同文件，避免重复存储；
6. 获取文件时，需要考虑大文件对服务器资源的消耗，尤其是多用户同时下载同一个大文件，需要选择合适的方法以减轻服务器的内存压力；
7. 前后端敏感数据传输的安全性需要保障；

8. 前端 UI 需要简约大方、清晰明了，并且不失美感；
9. 后端接口需要稳定安全、易于扩展、结构合理，并且需要记录服务运行中的出错信息。

在开发中，至少需要考虑并解决以上问题，这将在后续章节中逐一提供解决方案。

3.2 功能模块划分

基于上一节（系统需求分析）给出的需求，制定一份功能模块划分，这可以有效的加快开发速度，并且在开发过程中，能够合理的分出不同的模块功能，使代码结构非常清晰，这有助于代码管理、扩展。

考虑到系统的扩展性，可以对系统分为六大模块，分别为：博客模块、用户模块、配置模块、文件模块、消息模块、日志模块。

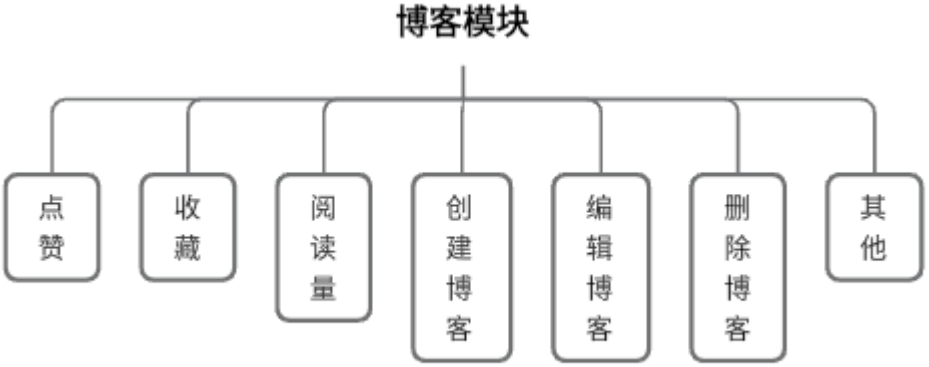


图 3.1 博客模块功能图

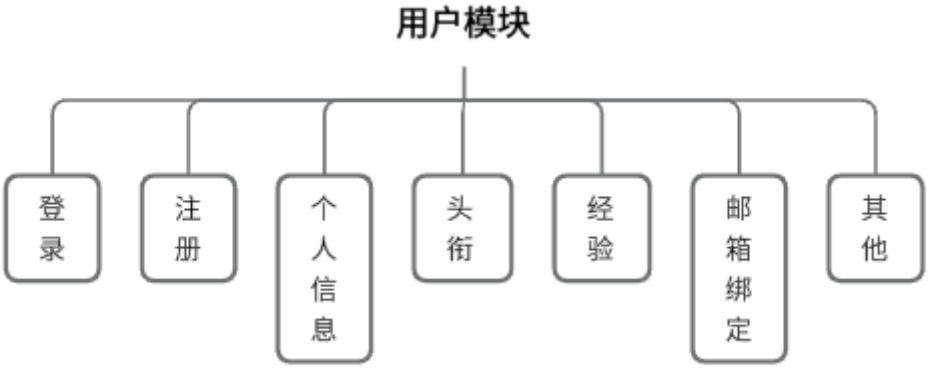


图 3.2 用户模块功能图

博客模块（图 3.1）和用户模块（图 3.2）为系统的核心模块，这也是本系统现阶段需要完成的模块；

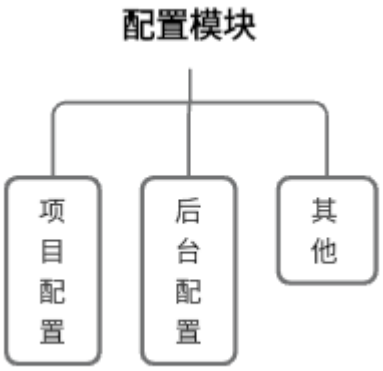


图 3.3 配置模块功能图

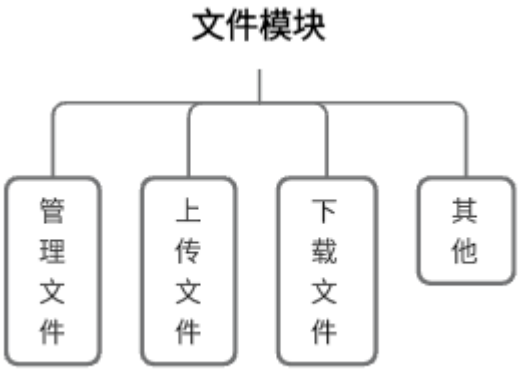


图 3.4 文件模块功能图

配置模块（图 3.3）则可以存储系统和其他功能的参数，基于配置数据，能实现大部分额外功能；文件模块（图 3.4）主要负责管理系统的所有文件；

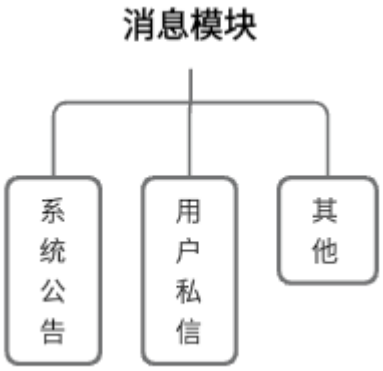


图 3.5 消息模块功能图

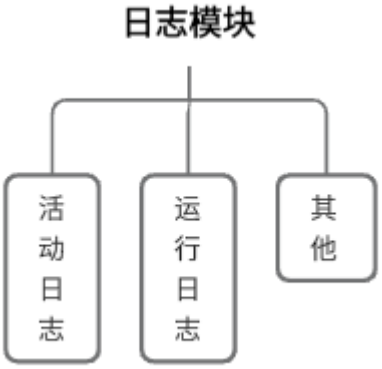


图 3.6 日志模块功能图

消息模块（图 3.5）主要是服务于系统对用户和用户之间的消息传递和存储；日志模块（图 3.6）则是存储后端系统运行时的各种信息，用以分析错误和提升性能。

这六大模块是开发的标准，里边包含了开发中需要实现的基础功能和可以扩展的功能。如果需要扩展一个功能，可以先提出需求并制定功能，然后分析该功能应该划分到哪一个模块，需要涉及到哪些模块。有时候一个复杂的功能可能需要不同模块的多个功能，比如要实现博客富文本功能，则需要博客和文件两个模块共同实现。

3.3 系统架构设计

系统架构是对一个系统的整体结构、组成部分、各部分之间的关系以及它们之间的交互方式的综合描述和设计。现根据 3.1（系统需求分析）小节和 3.2（功能模块划分）小节设计一份系统架构（图 3.7），以清晰地展示系统的各个部分。

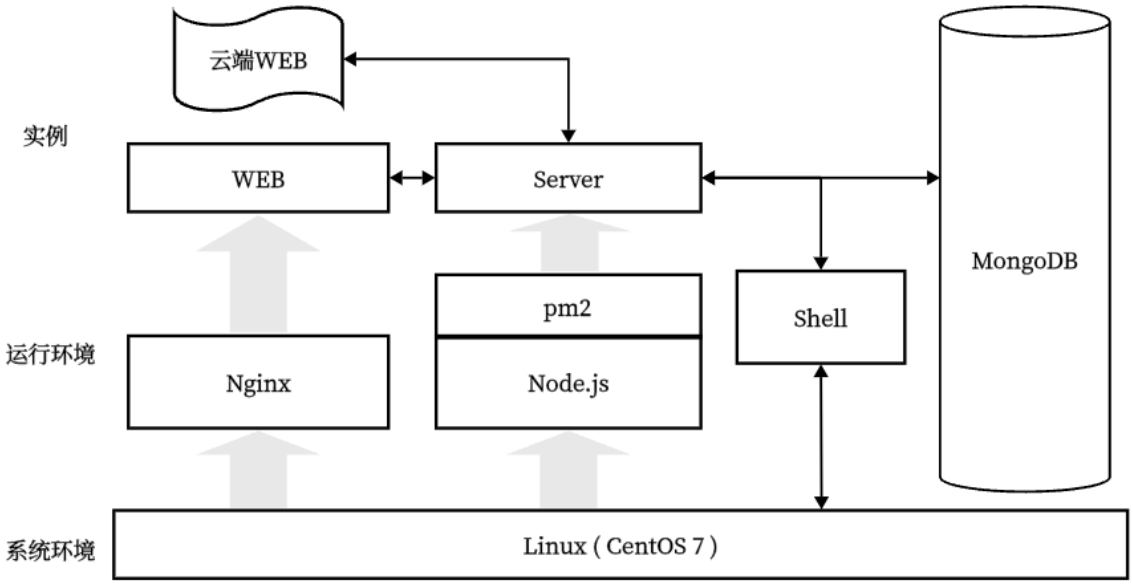


图 3.7 系统架构图

系统环境选用 Linux 发行版 CentOS 7 操作系统，所有程序都是运行在该系统环境下的；前端部署程序为 Nginx，它可以快速轻量地开启 WEB 服务，同时可以运行多个 WEB 实例；后端部署程序为 Node.js，它拥有一个名叫 pm2 的工具，此工具可以方便的管理 Server 实例，也支持自动重启、负载均衡，资源监控等功能；数据库则直接安装在系统环境，为 Server 实例提供数据操作接口；Shell 则为原生程序，在 Server 层可以调用该程序用以间接访问操作系统，提供系统级的额外支持。

在图 3.7 中，实例部分的 WEB 矩形是运行在本地的前端实例，它可以提供给客户端网页服务，同时还有不规则形状的 WEB 部分，它是部署在云端的前端实例，我们可以把域名重定向到云端服务器，让客户端去向云端服务器请求网页，这样可以减轻本地服务器网络压力；不管是本地 WEB 实例还是云端 WEB 实例，都可以去向本地的 Server 实例请求服务，只不过云端 WEB 实例需要通过公网地址去请求 Server，而本地 WEB 实例则可以通过内网地址去请求；本地的 Server 可以通过内网地址直接去访问数据库或者原生程序，以达到业务需求。

4 数据库设计

4.1 数据库技术分析

MongoDB 数据库是 NoSQL 型数据库，它的一些名词和 SQL 型数据库不太一样，表 4-1 是对该数据库的名词解释：

表 4-1 MongoDB 名词解释

MongoDB 名词	SQL 型数据库名词	解释
集合	表	集合是存储文档的容器

文档	行	文档是基本的数据单元
字段	域	字段是文档的一个键值对，键表示字段名，值表示字段值
索引	主键	索引可以快速查找文档，并且具有唯一性

MongoDB 的基本数据类型如表 4-2 所示：

表 4-2 MongoDB 基本数据类型

类型	说明
ObjectId	由时间戳、机器 ID、进程 ID、自增计数器构成，能满足分布式系统需求，并在时间上保持有序；
String	String，字符串，必须是 UTF-8 格式；
Boolean	布尔值，true 或 false；
Integer	整数型，包括 Int32 和 Int64；
Double	浮点数；
Arrays	数组或者 List；
Object	对象类型，相当于一层文档；
Null	空数据类型；
Timestamp	时间戳；
Date	时间的 Unix 格式；
BinaryData	二进制数据；

根据 MongoDB 数据库官方文档[7]说明，数据库名不区分大小写；集合名称限制在 255 字节以内，集合内不能超过 64 个索引，如果不设置最大文档数，则集合内不限制文档数量；单个文档最大为 16M，并且不支持嵌套 100 层深度的文档。

通过官方文档的限制说明，我们可以大概了解该数据库的一些特性，也就能更好地设计一份系统的数据模型。除过以上说明，我们还能通过一些属性来探索出其他的特性：

1. MongoDB 使用 BSON 格式存储文档数据，所以文档是无格式的，也就是说它没有 SQL 型数据库那样严格的域结构，BSON 可以实现任意数据的存储，哪怕在一个集合中，但至少需要一个索引，默认索引是 _id，该索引是 ObjectId 类型的，是一个 24 字节的唯一标识。
2. MongoDB 可以在单个文档中存储二进制数据，且最大可存储 16M 字节，这也为文件存储在数据库中提供了一种思路。

3. MongoDB 可以配合副本集实现数据库事务操作，这能够保证多集合操作的统一性、原子性。
4. MongoDB 使用大量内存去缓存集合的索引，所以查询速度是非常快的。

其中最具有特点的是无格式的文档，这将允许我们更灵活地存储数据资源，但是一个规范的数据模型设计，结构不能够混乱，所以选用 MongoDB 的一个 js 库 mongoose 作为中间件去操作数据库，mongoose 有一个概念为 Schemas，即模式，官方称 mongoose 的一切始于 schema，每个 schema 都会映射到一个集合，schema 相当于限定了集合中文档的规范，我们要在一个集合中插入一条文档，就必须依照 schema 的格式去创建数据，这能保证数据模型的结构性。

4.2 数据汇总与分析

在设计数据模型时，需要先提前将可能出现的数据进行汇总，然后加以分析设计出合适的数据模型。

根据 3.2 小节中划分的功能模块，可以整理出系统的基本数据：

1. Blog 相关数据：博客的内容、博客的信息（包括不限于标题、标签、评论、点赞量、收藏量、阅读量等）；
2. User 相关数据：用户 ID、用户名、用户头像、用户其他基础信息、用户的博客信息、用户的收藏、用户对博客的点赞收藏信息等；
3. File 相关数据：文件二进制数据、文件信息（包括不限于文件名、文件上传者、文件散列值、文件尺寸等）
4. Message 相关数据：消息内容、发送者和接收者、发送时间等；
5. Relation 相关数据：用户的关注和粉丝、关注时间等。

在扩展系统时可能会产生更多的数据，只需要整理好附加到 mongoose 的 Schemas 中即可。现在整理出来的数据，需要考虑一些比较重要的问题：

问题一：用户和博客之间的信息需要在拥有大量用户和博客的环境下以很快的速度和较少的服务器资源去获取。

在用户点开一个博客时，不仅需要向用户展示博客的信息，还需要向用户展示用户和该博客的交互信息，比如用户是否对该博客点赞、收藏，此时我们可以给该博客维护一个用户操作列表，包括哪些用户点赞了，哪些用户收藏了，当用户获取博客信息时在服务端再判断以下用户是否存在于用户操作列表里，就可以判断用户是否点赞和收藏。

但是这样做的代价比较大，每次请求博客信息时需要做额外的判断处理，在非常大的列表里寻找用户非常消耗性能，而且用户操作表能够保存的数据也是有限制的。

我们可以新建一个集合，用户博客交互集合，里面包含着用户和博客一一对应的信息，比如用户 A 和博客 a、b、c 之间就存在三条文档，文档的索引则是用

户 ID+博客 ID，例如：Aa、Ab、Ac，每条文档存储的就是该用户对该博客的点赞和收藏信息。在获取博客信息时，可以通过用户 ID 和博客 ID 在用户博客交互集合中获取它们之间的信息，这样就避免了在服务端去进行非常消耗性能的处理。

问题二：对于用户的博客、用户的收藏、博客的评论、博客的点赞等数据，要考虑超大量的数据如何存储的情况。

虽然在 MongoDB 中存在 Array 类型，但这只适合少量数据存储，而一条博客可能存在上万，甚至千万级的点赞量，显然不适合使用 Array 去存储这么多的数据。

考虑到这种超大量数据一般不会频繁地随机访问，主要是从头开始一点一点去向后访问，所以链表就非常适合这类数据，链表没有长度限制，插入删除效率也是非常优异的，但是 MongoDB 不存在链表格式，不过我们可以通过文档去构造一个链表。一个文档就是一个节点，节点包含前节点和后节点的 ID，相当于访问地址，这是一个典型的双向链表，数据部分为了不浪费空间利用率，采用定长数组去存储数据，这就变成了数组链表复合数据结构，这样的复合结构操作复杂度比较高，但是能省去频繁获取文档的开销。这种复合结构将在后端抽象出操作 API，以供其他服务调用，称为 List。

问题三：文件存储需要考虑不同用户上传相同文件，避免重复存储。

在服务器中，如果把所有文件的归属权都归一个用户所有，这将浪费大量的存储空间，因为存在许多一致的文件被不同用户上传，所以文件应该拥有共享性，但是也不能缺失所属权。所以在文件上传时先利用客户端资源去计算该文件的散列值，以获取该文件的唯一 ID，上传至服务器后，通过文件的唯一 ID 先在数据库寻找是否存在该文件，如果存在，则在文件信息集合中向上传者列表添加该用户，而不去实际地存储该文件；在用户删除文件时，只需要删除上传者列表中的用户，使其失去所属权即可。还可以添加一种逻辑，即文件有超过一定数量的上传者后，使该文件永不删除，哪怕上传者列表被清空后，也不删除文件，因为被一定数量用户上传的文件，在未来也极有可能被上传。

问题四：获取文件时，需要考虑大文件对服务器资源的消耗，尤其是多用户同时下载同一个大文件，需要选择合适的方法以减轻服务器的内存压力。

在实际开发测试中，我发现如果一个用户去请求下载一个 100M 的文件时，服务端程序将立刻增加了 100M 内存使用，如果多个用户去请求，可能会成倍地增加服务端程序的内存使用，这严重地影响了系统性能。

但也正因为 MongoDB 单个文档最大尺寸限制为 16M，使我不得不将文件分片存储，所以我同样可以分片响应给客户端。经过测试，性能有非常巨大的提升，文件分片存储和响应将在 6.4 小节详细叙述。

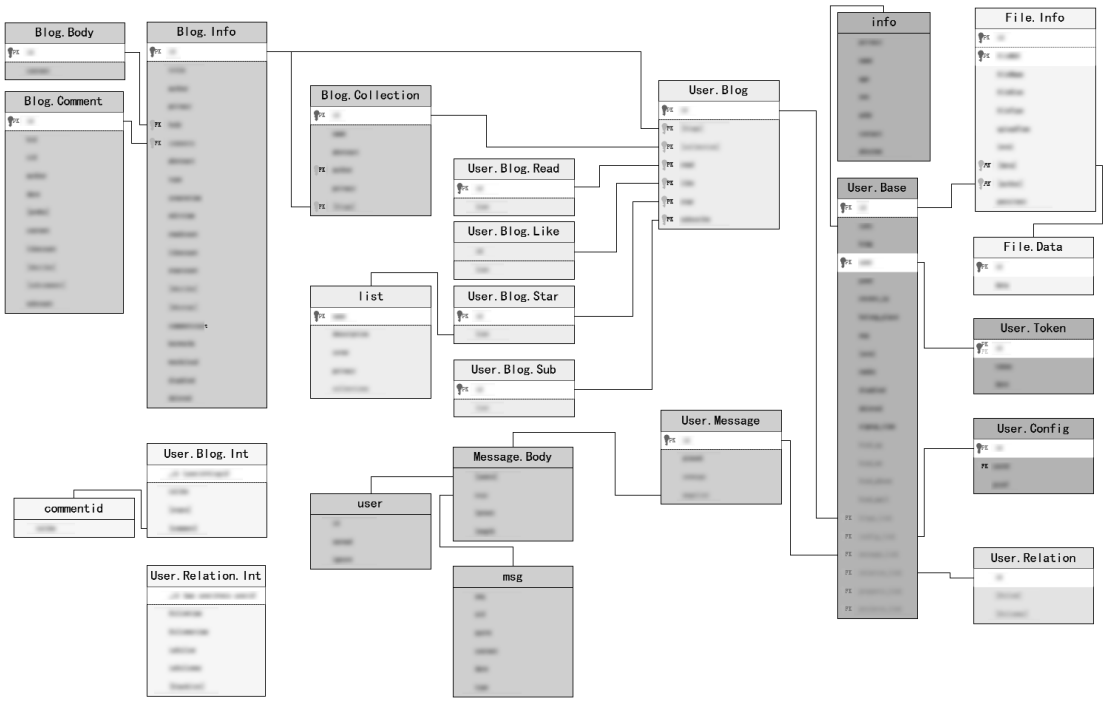


图 4.1 数据模型概览

在图 4.1 中，展示了各种与系统相关数据的概览，其中每个矩形框图则是一个 schema，也就是一个集合中文档的规范，代表着不同属性的数据集，其中包含有 User、Blog、File、Message、List、Relation 等属性，每个 schema 之间的连线则是不同属性中的共同字段，起到映射数据集的作用，相当于外键。主要 schema 的具体设计将在 4.3 小节详细描述。

4.3 数据模型设计

通过对数据库技术的分析和数据汇总及分析，现将该系统的主要数据模型作以展示：

表 4-3 User.Base 集合

字段名	字段类型	说明
himg	String	用户头像文件 ID
user	String	用户账号类型，唯一
pswd	String	用户密码
about_me	String	用户签名
recent_ip	[Object]	最近登录的 IP 列表
belong_place	String	通过最后登录的 IP 获取的归属地
exp	Number	经验值，默认 0
level	Number	用户级别，'visitor', 'user', 'admin', 'aster'，默认为 'user'
ranks	[String]	限制特定功能使用的等级

signup_date	Date	注册日期，默认当前时间
disabled	Boolean	是否禁用账号
deleted	Boolean	是否删除账号
bind_qq	String	绑定 QQ
bind_we	String	绑定微信
bind_phone	String	绑定电话号码
bind_mail	String	绑定邮件
blogs_link	ObjectId	博客外键
config_link	ObjectId	配置外键
message_link	ObjectId	消息外键
relation_link	ObjectId	关系外键
property_link	ObjectId	财产外键
projects_link	ObjectId	项目外键

表 4-4 User.Blog 集合

字段名	字段类型	说明
blogs	List	用户的所有博客
collections	List	用户的所有收藏
read	ObjectId	用户阅读过的所有项目的链接
like	ObjectId	用户点赞过的所有项目的链接
star	ObjectId	用户收藏过的所有项目的链接
readcount	Number	用户阅读次数，默认为 0
likecount	Number	用户点赞次数，默认为 0
starcount	Number	用户收藏次数，默认为 0
blogcount	Number	用户博客数量，默认为 0

表 4-5 Blog.Info 集合

字段名	字段类型	说明
title	String	文章标题
author	ObjectId	作者
privacy	Number	隐私类型，默认为公开
body	ObjectId	文章正文
type	Number	类型
cover	String	封面图片链接
abstract	String	文章摘要
comments	List	文章评论列表

createtime	Date	创建时间，默认为当前时间
edittime	Date	编辑时间
readcount	Number	阅读次数，默认为 0
likecount	Number	点赞次数，默认为 0
starcount	Number	收藏次数，默认为 0
commentcount	Number	评论数量，默认为 0
wholike	List	点赞用户列表
whostar	List	收藏用户列表
keywords	[String]	关键词列表
wordcloud	Object	词云
disabled	Boolean	是否禁用文章
deleted	Boolean	是否删除文章

由于数据模型非常多，在此只展示系统核心功能的数据模型，分别是用户基本信息模型（表 4-3）、用户博客信息模型（表 4-4）、博客信息模型（表 4-5）。

数据模型的设计理念是，一切以用户为中心去做扩展，属于用户唯一数据集的，则索引都是用户的 ID，这样可以通过用户 ID 去寻找用户的一切数据。对于存储大量数据的优先使用 List 结构。尽量使模型只含有一条索引，以减少数据库内存消耗。

5 前端开发

5.1 Vue 环境搭建

在 Vue.js 社区中，更推荐使用 Vite 作为 Vue 项目的打包工具，它拥有更快的启动速度和热更新速度，所以我们使用 Vite 去构建项目。

在此之前，Node.js 是不可或缺的运行时环境，我们使用最新的 LTS 版本 v20.11.1 作为运行环境，它同样可以作为后端程序运行时环境。

为了支持更完善的功能，在搭建完 Vue 项目后，需要额外安装表 5-1 中的第三方库。

表 5-1 Vue 项目依赖

库名	版本	描述
@ant-design/icons-vue	6.1.0	Ant Design Vue 的图标组件库。
ant-design-vue	4.1.1	Ant Design Vue UI 组件库。
axios	1.5.0	用于发起 HTTP 请求的 Promise-based HTTP 客户端。

crypto-js	4.2.0	JavaScript 加密库，提供了各种加密算法。
js-cookie	3.0.5	用于操作 Cookie 的 JavaScript 库。
lottie-web	5.12.2	用于在 Web 上渲染 Lottie 动画的 JavaScript 库。
mavon-editor	3.0.1	基于 Vue.js 的 Markdown 编辑器组件。
ts-md5	1.3.1	用于计算 MD5 哈希值的 TypeScript 库。
uid	2.0.2	生成唯一标识符的 JavaScript 库。
v-lazy-component	3.0.9	Vue 的懒加载组件，用于延迟加载组件。
vue-draggable-plus	0.3.5	Vue.js 的可拖动列表组件。
vue-router	4.0.13	Vue.js 的路由管理器。
vuex	4.1.0	Vue.js 的状态管理库。
less	4.1.3	Less 预处理器，用于编写可维护的 CSS。
less-loader	11.1.3	加载 Less 文件的 loader，用于将 Less 编译成 CSS。

5.2 前端架构设计

在 Vue 项目中需要使用 Vuex 去管理全局属性，用 Vue-Router 去管理路由，各个组件和各个页面也需要适当的划分。

对于多次利用的组件封装起来，供页面使用。在组件或者页面中可以调用 Vuex 的全局属性，Vuex 的属性也需要分类管理；也可以调用工具库，其中包括了一些实用的算法；Axios 中对 API 进行了封装，可以供组件或页面像使用函数一样去请求 API，同时配置了全局错误状态管理。客户端请求的各种 URL，都可以被 Vue-Router 解析出来，然后响应成对应的页面。

在对前端功能做扩展时，可以将业务拆分成不同的类型，然后添加至该架构的对应模块，图 5.1 是前端基本架构的设计。

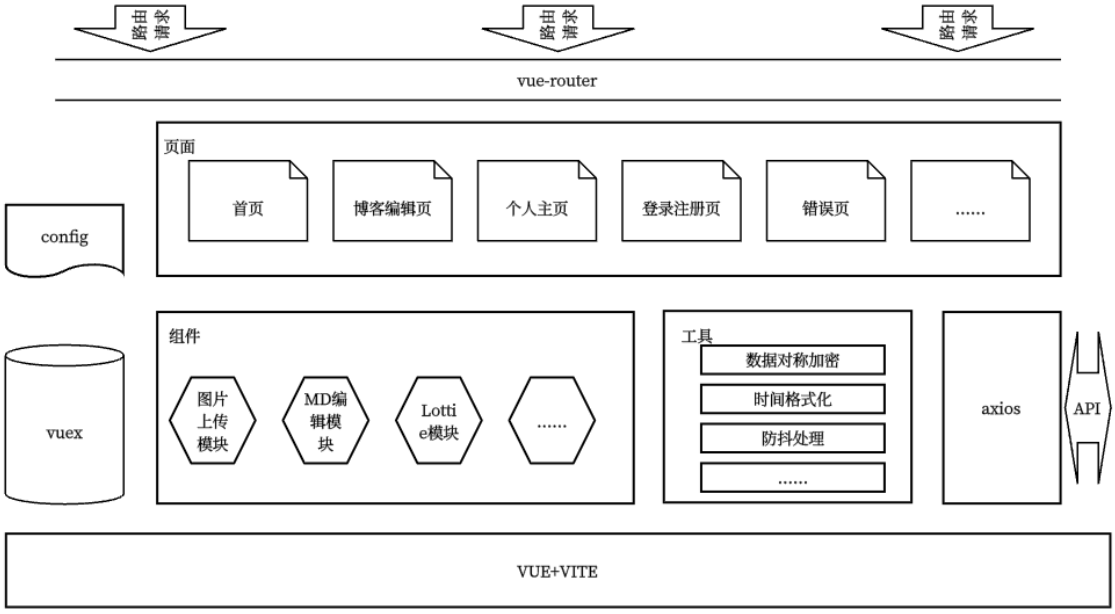


图 5.1 前端基本架构

5.3 核心交互逻辑及实现

在为用户提供界面交互时，需要充分考虑界面 UI 的美观性，以及合理的交互逻辑。在前端程序中，界面采用 Ant Design UI 库进行开发，它是蚂蚁集团开发的一款企业级 UI 库，基于自然、确定性、意义感、生长性四大设计价值观，通过模块化解决方案，降低冗余的生产成本，让设计者专注于更好的用户体验。

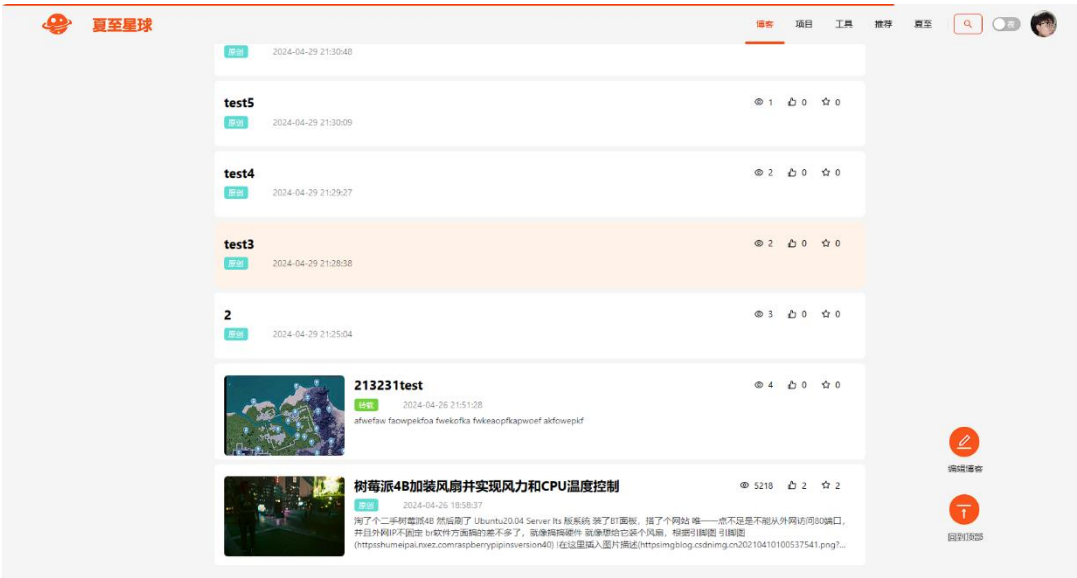


图 5.2 首页 UI

在首页界面图 5.2 中，最上边是网站导航栏，其中包括进程度条，可以根据当前网页浏览进度而展现不同的占有比，方便用户了解当前阅读进度；导航栏的活动滑块是通过算法实现的动态选中，拥有流畅优雅的动画效果。

博客展示组件可以在不包含图片和摘要、包含摘要但不包含图片、包含摘要和图片三种形态自动布局，使信息展示更加丰富；在右下角有功能模块，可以快速跳转到编辑页面或页面顶端。首页整体简约大方，信息直观，同时留有扩展位置提供给额外功能组件。用户登录后会在右上方展示用户头像，点击则会展示用户面板，如图 5.3 展示，在展示用户面板时，会模糊化背景，这有助于专注优先级高的界面。

图 5.3 个人面板

UI 库只是统一界面交互的一套设计语言，本系统的核心是博客模块，所以在博客编辑、阅读上需要更多的功能和视觉体验。

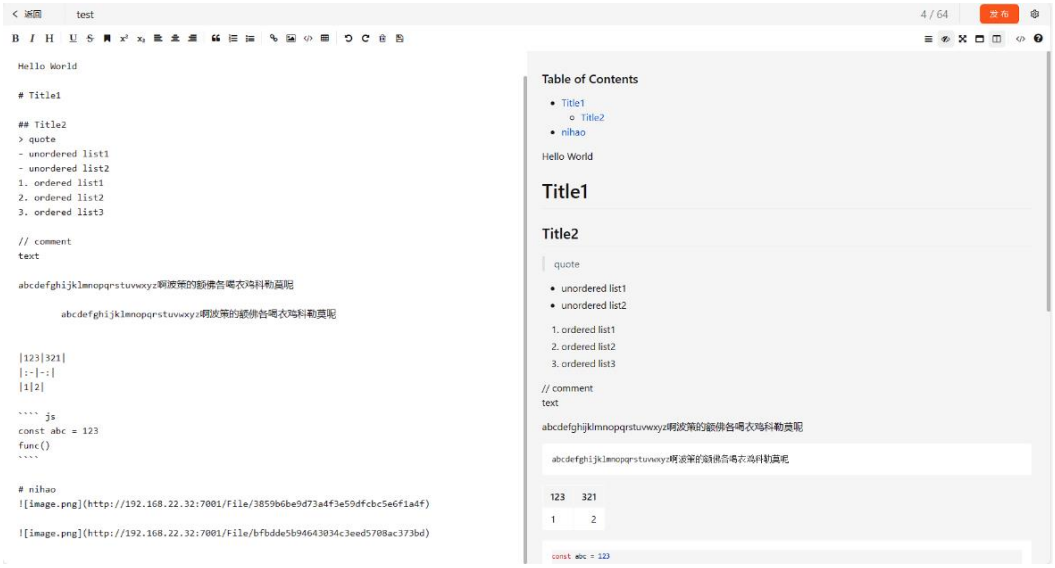


图 5.4 博客编辑页

富文本编辑则可以实现这一需求，常见的网页富文本 Markdown 是一款轻量级标记语言，设计简洁、简单易学、跨平台兼容，同时还支持丰富的内容，包括标题、段落、列表、链接、图片、引用、代码块等。有一款出色的 Markdown 库名为 Mavon-Editor，它 UI 简约而不简单，符合本系统的设计语言。图 5.4 是 Mavon-Editor 组件编辑时的界面，我修改了一点样式表使其颜色和整体 UI 更相配，以及适配了它的黑夜模式。

图 5.5 博客阅览页面

图 5.5 正文则是使用了 Mavon-Editor 的预览模式，除此之外该页面还包含了题目、博客信息、作者信息和交互按钮。

前端的核心交互逻辑以及设计语言向该节展示图片靠拢，后续页面或组件扩展可参考这类设计模式和语言。

5.4 核心组件设计及实现

在前端程序中，有一些组件和模块需要贯穿整个程序发挥着比较的重要的作用。

Header 组件：Header 组件几乎在九成以上的页面都有显示，它和 Footer 组件构成了页面的两大必要组件，但 Footer 只是简单地展示一下网站信息，没有 Header 组件功能复杂。在该组件中，需要实现进度条、目录活动项、页面标题、用户面板。进程条则是通过监听页面滚动事件，计算出当前位置相对于整个页面的占比，然后渲染进度条到合适的位置；目录活动项则监听了鼠标点击事件以及路由，以实现动态切换选中项；页面标题则是注册一个自定义更改标题的事件，以达到其他组件修改标题后，能够自动更新 Header 组件的标题；用户面板使用弹出侧边框来展示用户信息，并提供个人功能的主要控制面板。

Upload 组件：上传文件在很多场景都能够遇到，多数是上传图片，但是用户选择了图片后，实际上传的概率非常不确定，受业务未完成、页面突然关闭、更

换图片等影响。如果选择了图片立即上传会使数据库存储大量不需要的、无价值的文件，所以在该组件，提供了一个控制函数，如果想手动控制上传操作，则在该组件设置 ctl 属性即可，当业务完成图片确定后，再调用组件内提供的 upload 控制函数，便可在图片具有实际意义时上传。

OnDark 模块：由于使用了 Ant Design 的 UI 库，所以更新色彩时，需要在页面顶层去计算并设置主题色和其他配套颜色。但是像开启暗黑模式、设置主题等功能不可能都在顶层页面实现，所以需要在其他子组件或页面去控制色彩。好在 Vue3 官方支持透传功能，该功能可以在父组件定义函数或变量，并把该函数或变量透传到子孙甚至更深的组件中，使之可以使用。该模块则是在顶层实现了计算和设置主题色等功能，并提供接口，具体在哪个组件需要什么样的主题色，只需要在那个组件去调用透传的接口便可全局设置。

Debounce 模块：有时候我们的监听太过于频繁，比如鼠标滚动事件，一秒可达几十上百次，但实际中监听事件的处理函数稍微复杂，就会导致性能的大量损失。Debounce 模块则是自定义的防抖功能，它可以有效处理频繁的监听事件。该模块通过全局的管理回调函数，在每次事件循环时，去判断回调函数的满足条件，从而决定是否执行回调，使非常频繁且大量消耗性能的监听事件可以自定义执行频率。

useHeaderMode 模块：在 Header 组件中，有四种展示形态，分别是恒定展示 Header、当下滑时隐藏 Header、当下滑时展示 Title、自动隐藏 Header，这些形态为不同页面提供了选择，比如在浏览首页时，就只需要展示 Header 就好，但浏览博客页面时，就需要在下滑时展示 Title，上滑时展示 Header。页面需要在退出时还原 Header 形态，如果加载时设置 HeaderMode，卸载时再设置一遍就显得些许繁琐，所以这个模块类似于中间件，只需要一个函数，便可以在组件的整个生命周期做提前预设好的事情，这也是 Vue3 官方支持的技术，称为“组合式函数”(Composables)。

useTitle 模块：该模块也使用了组合式函数来实现不同组件对标题的修改。

6 后端开发

6.1 Midway 环境搭建

在搭建前端环境时已经安装了 Node.js 运行时，所以直接用 Node.js 的包管理器 npm 安装 Midway 框架就好。同时安装第三方库 mongoose 作为数据库操作接口；nodemailer 库作为邮件发送工具。

后端偏业务性，所以不需要太多额外的库便能完成大部分任务。装好上述三个库就可以开始编写后端程序，Midway 提供了 OOP + Class + IoC 编程范式[8]和设计模式，这有助于提高代码的组织性、可维护性、灵活性和可测试性，是现代软件开发中广泛应用的重要概念和技术。

6.2 后端架构设计

总体而言，后端程序只负责提供安全稳定的 API，需要支持用户鉴权、数据响应、数据的简单处理等功能。

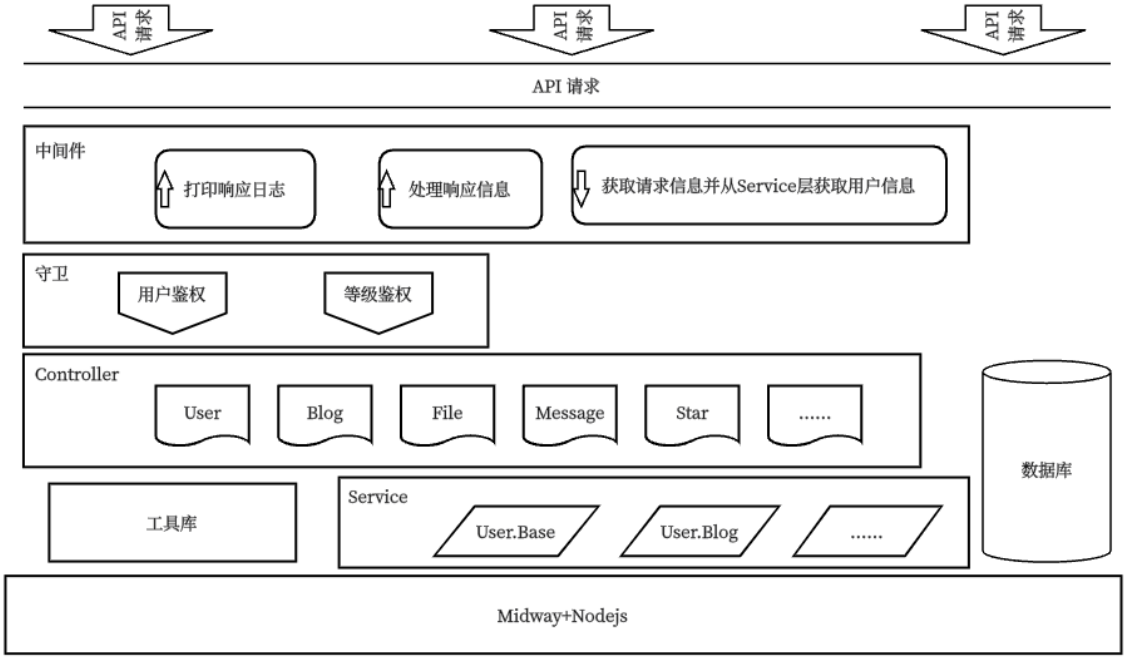


图 6.1 后端架构图

在图 6.1 中，外部的所有请求进入程序后，先经过中间件，中间件允许请求入站或者响应出站时作出额外的处理，在此后端架构中请求入站会自动获取用户的身份，并动态请求 Service 层接口获取用户信息；在响应出站时，会自动格式化响应结构，以及向日志记录请求记录。接着请求便匹配到对应的 Controller 层方法，即 API，如果该 API 添加了守卫，则会在中间件处理完成后，先经过守卫再次处理鉴权，如果鉴权成功，才会来到 API 方法，否则将不会进入 API 方法直接返回 403 拒绝错误；当进入 API 方法后，这里主要处理逻辑上的业务，当业务涉及到数据的时候，则会调用专门用于处理数据的 Service 层业务，在 Service 层中不会有太多的逻辑业务，它一般直接操作数据库，把一些功能封装成一个函数，为 Controller 层提供服务。

6.3 核心接口和路由设计

在后端程序中，也对接口划分成不同的类型，有用户相关的接口、博客相关的接口、消息相关接口等。

表 6-1 接口路由设计

接口类型	接口	描述
User	/User/isExist/{username}	查询用户是否注册
	/User/Signup	用户注册
	/User/Signin	用户登录

	/User/Delete	用户注销
	/User/GetUserInfo	获取用户信息
	/User/VerifyToken	用户验证 Token
	/User/UsersInfo	获取用户列表的信息
	/Blog/Publish	发布博客
	/Blog/Get	获取博客
	/Blog/List	获取博客信息
	/Blog/Comment	评论博客
	/Blog/GetComments	获取博客评论
	/Blog/Like	喜欢博客
Blog	/Blog/Share	转发博客
	/Blog/LikeComment	点赞评论
	/Blog/GetInteraction	获取互动信息
	/Blog/Delete	删除博客
	/Blog/Drop	彻底删除博客
	/Blog/DeleteComment	删除评论
	/Blog/HadRead	浏览过博客
	/Blog/GetUserBlogs	获取用户的博客
	/Star/Add	收藏博客
	/Star/Del	取消收藏
	/Star/DeleteFolder	删除收藏夹
Star	/Star/GetFolders	获取收藏夹
	/Star/Folder	收藏夹内容
	/Star/Search	搜索收藏夹
	/Star/Create	创建收藏夹
	/Star/Modify	修改收藏夹
	/Relation/Follow	关注用户
	/Relation/Unfollow	取消关注
	/Relation/GetInteraction	获取用户之间信息
Relation	/Relation/Block	拉黑名单
	/Relation/GetFollow	获取关注
	/Relation/GetFollower	获取粉丝
	/Message/StartChat	开始聊天
Message	/Message/Send	发送消息
	/Message/GetNew	获取新消息

Config	/Message/Get	获取消息
	/User/Config/PConf/Upload	上传个人配置文件
	/User/Config/PConf/Sync	同步个人配置文件
File	/File/Upload	上传文件
	/File/Delete	删除文件
	/File/{uid}	获取文件
	/File/Get	获取文件
	/File/GetInfo/{uid}	获取文件信息
Other	/VerifyMailCode	验证邮箱验证码
	/SendMailValidCode	发送邮箱验证码
Panel	/Panel/Get/{type}	获取配置值
	/Panel/Set/{type}	修改配置值
	/Panel/Delete/{type}	删除一个配置
	/Panel/GetAll/{type}	获取全部配置
Pull	/Pull/Blog/Order	获取推荐博客

在表 6-1 中展示了后端程序的所有接口，通过这些接口可以实现前端几乎所有功能。在处理一些特别的业务时，可能需要额外地添加接口，或者是优化现有接口。不过后面的接口设计也是遵循现有的架构，首先把需求拆解为逻辑业务和数据操作，如果有额外的处理，需要添加工具库，或者添加中间件和守卫，接着在对应的模块进行开发就好。

需要特别说明的是，如何防止恶意注册刷号，目前非常常见的方案是使用手机号注册，这能有效防止恶意注册，但是这会产生额外的短信费用。还有一种比较常见的方案是使用邮箱验证码，这也能有效防止恶意注册，所以后端程序需要添加 nodemailer 第三方库用于发送邮箱验证码，但效果肯定不如手机号注册。

6.4 文件数据存储逻辑

在本系统中用了比较少见的文件存储方式，即直接将文件的二进制数据存储到数据库。普遍的文件存储方式一般都是存放在服务器本地目录，或者文件云端，云端存储自然不会成为本系统的选择，如果把文件存储到服务器本地目录的话，这就忽视了 MongoDB 数据库的特点，所以我想把文件存到 MongoDB 数据库中，这样本系统的所有数据都会统一在一个地方，非常的便于管理。

在了解 MongoDB 的存储逻辑后，发现可以将文件分片然后以二进制存储到文档中，这就至少需要两个集合去管理文件。

表 6-2 文件信息集合

字段	类型	描述

fid	String	文件的唯一标识符
fileName	String	文件名，只保存第一次给定的名称
fileSize	Number	文件大小，单位为字节
fileType	String	文件类型，如 jpg、mp3 等
uploadTime	Date	上传时间
level	Number	用户级别
data	[ObjectId]	文件的数据
author	[ObjectId]	上传文件的用户
persitent	Boolean	文件是否持久存在，如果为 true，则文件不会被删除

表 6-3 文件数据集合

字段	类型	描述
_id	ObjectId	数据唯一 ID
data	Buffer	二进制数据

表 6-2 是文件信息集合 File.Info，存储的是一个文件的基本信息，且只存储信息，文件的唯一描述符 fid 是文件的散列值，它能保证文件的唯一性，也是获取文件的必要条件；表 6-3 是文件数据集合 File.Data，存储的是分片的二进制数据，且每一片数据都有一个唯一 ID；文件分片的大小可以在配置中更改。

在后端接收到上传文件请求后，会经过守卫对用户鉴权，只有为本站用户才能够上传文件。当鉴权通过后，会检查请求头中的基本文件信息，包括文件尺寸，如果文件大于后端程序所设置的最大上传尺寸，则会拒绝请求，接着查询请求头中的文件唯一标识符 fid，如果 File.Info 集合中存在该 fid，则直接返回成功，否则会通过请求提取拼接出完整文件，用给定长度分片好后，按顺序把文件片存储到 File.Data 中的 data 二进制字段，同样按顺序把文件片的 _id 依次插入 File.Info 中的 data 数组字段，把文件的其他信息写入数据库就完成了一个文件的上传。File.Info 和 File.Data 集合的对应关系要么是一对一，要么是一对多。

需要获取文件时，需要 fid 去请求相应的 API，在鉴权成功后，则从数据库获取 File.Info 集合中指定的文档，通过该文档获取 File.Data 集合中的的文件片，然后组成二进制数据返回给客户端。

但是获取大文件存在性能问题，该问题已经在 4.2 小节的问题四详细描述，以及给出了解决方案，接下来将详细描述处理办法。

由于数据库存储的是分片后文件，且拥有严格的顺序要求，我们可以每次获取一片数据，然后将这一片数据响应给客户端后，再请求下一片数据。所以问题

就变为了如何多次进行响应。经过查阅 MDN Web Docs[9]文档发现可以在响应头中添加 Transfer-Encoding 值为 chunked 实现多次响应。

现在可以多次响应数据给客户端了，但是数据操作 Service 层和业务处理 Controller 层不在同一个方法里，就不太好一次一次去请求文件片数据了。可以添加两个 Service 层方法，来实现在 Controller 层分片请求数据，即用一个 Service 方法返回文件片信息，然后再用一个 Service 方法请求单个片数据，但是这样做不是特别优雅，使 Controller 层业务略显复杂。不过我们可以在 Service 层请求文件时，只请求文件信息，不立即请求文件片数据，而是把请求每片文件数据的过程封装到函数，并添加到数组中，返回给 Controller，这样我们在 Controller 就得到了一个函数数组，在发送时依次执行函数就可以实现请求完文件片后立即发送，在发送下一片文件数据时，上一片数据已经被垃圾回收机制清理了，这将大大减少大文件响应时占用的内存空间。

图 6.2 22M 文件原生响应

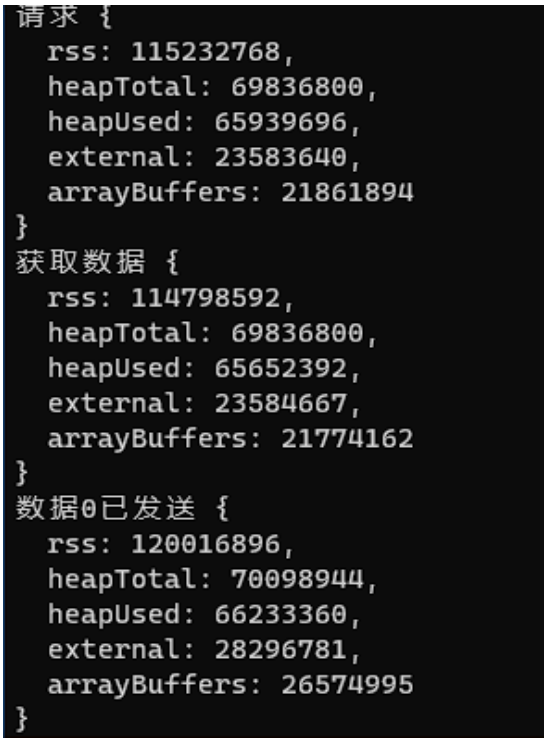


图 6.3 22M 文件优化响应

在图 6.2 和图 6.3 中，是 Node.js 中的内存监测函数返回的数据，其中 rss 表示操作系统分配的实际物理内存；heapTotal 表示 Node.js 总内存量；heapUsed 为使用的内存量；external 为额外占用的内存量；arrayBuffers 为缓存区分配的内存量。其中测试文件的大小为 22M，文件单片尺寸为 1.6M，我们只需要关心响应前后 arrayBuffer 的变化即可。可以从图 6.2 中明显看出，原生请求即直接把整个文件响应，响应前后的 arrayBuffer 内存差第一次为 57M 左右，第二次为 33M 左右，都远超文件尺寸 22M；而图 6.3 中，优化后响应前后 arrayBuffer 内存差仅为 5M 左右，其中有请求时内存使用量、获取数据时内存量、数据发送后内存量，

发现请求时和获取数据时内存几乎无差别，是因为在获取数据时，并没有真正获取文件片数据，只是构造了获取文件的函数数组。请求结束和发送后的内存获取依旧在函数内，所以不存在垃圾回收导致的数据错误。

通过这种方法，哪怕再大的文件，在响应时依旧只占用大概一片文件数据尺寸的内存，这极大地提升了程序的性能。

6.5 安全性和性能优化

虽然后端 API 设置有守卫鉴权，能够阻止一些没有权限的用户，但是当有权限的用户信息被截获泄露出去，那么守卫就无法判断哪个才是安全的请求。

所以需要考虑到请求中敏感信息的加密，这能有效阻止中间人网络攻击。加密的信息需要在后端解密，所以加密必须是可逆的，否则失去了传递信息的作用。本系统使用时间加 Token 来实现信息的加密和解密，并且能够保证加密信息的时效性、可逆性、安全性。

在图 6.4 中，加密算法是使用 Date 去计算新的 Token，不同的请求时间得到的 Token 是不一致的，在中间人不知道加密算法的情况下，破解难度是非常困难的。客户端的原生 Token 是通过登录成功后，服务器返回用于鉴权的 Token。

如果加密算法过于简单，那么中间人破解的概率就会非常高，如果过于复杂混乱，那服务端解密就比较困难了。本系统是通过请求时，获取时间戳，通过时间戳去加密原生 Token。

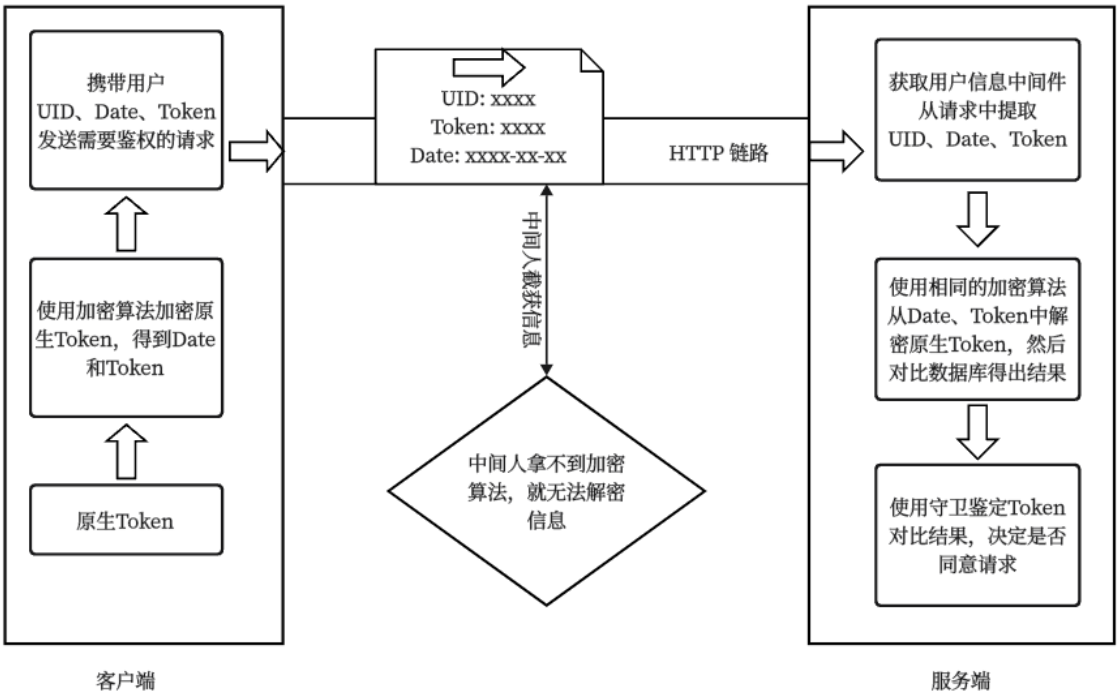


图 6.4 信息加解密流程

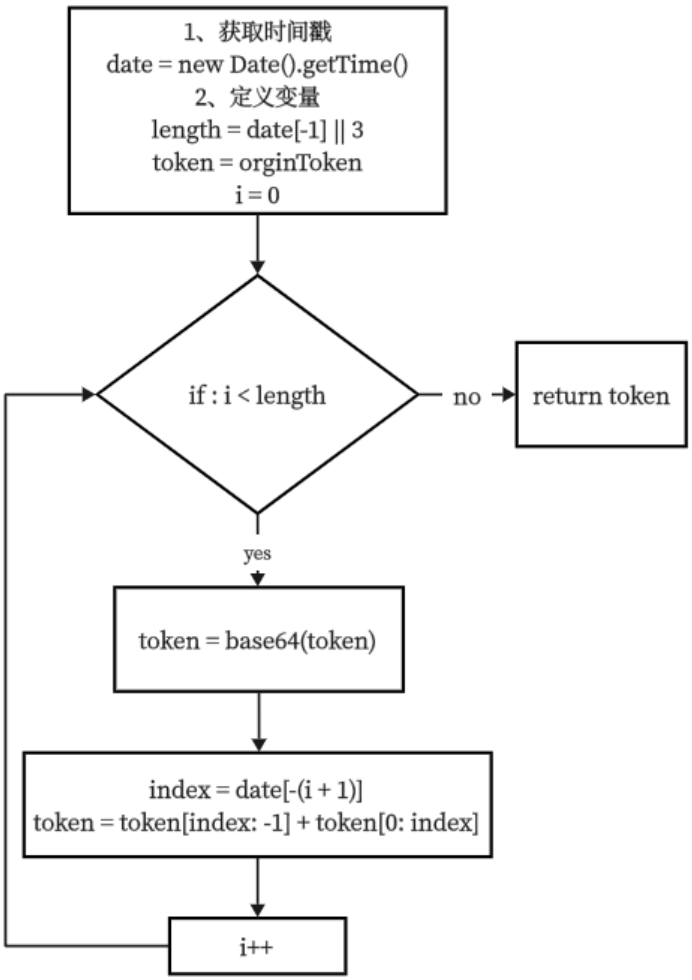


图 6.5 加密算法流程图

在图 6.5 中，加密算法开始会获取时间戳的最后一位数作为加密循环，这一位是毫秒级单位，由于请求发出时间的不确定，所以能够保证循环是比较随机的，如果该数小于 3 则会赋值为 3；接着进入循环，第一步会使用 base64 算法给 token 编码，然后选取当前循环的迭代次数作为索引指针，给编码完成的 token 分片，再调换前后分片位置；迭代循环至加密循环次数 length 就完成了 token 加密。解密时把算法逆推就好了，其中 base64 是将任意字节重新编码为 ASCII 码，是可逆的。该加密算法一般可生成原码 1 到 10 倍长度的密文，使密文更难破解。

7 部署与测试

7.1 环境配置

在 CentOS 官网下载主机合适架构的系统镜像，刻录到 U 盘中便可在主机上安装该系统。在安装完成后设置固定 ip 并开启 ssh 端口，以便在工作电脑中远程连接主机。

使用 ssh 连接主机，放入提前下载好的 Nginx 程序包和 Node.js 程序包，解压到指定路径即可完成安装。需要注意一点，Nginx 启动后，需要使用 firewall 工具

放行 WEB 访问端口，否则在局域网中无法访问 WEB 服务。安装好 Node.js 后，运行命令 npm install pm2 用以安装 Node 程序的守护进程，它可以在后端服务奔溃时自动重启服务。使用 pm2 启动守护进程后，也需要放行后端程序端口。

7.2 部署

当前后端程序开发完成后，就可以打包上传到主机部署了。

Vite+Vue 的打包命令是 npm run build，它会在项目路径生成 dist 目录，里边就是需要部署的前端项目，它是经过混淆和优化的代码。

```
PS C:\Users\xazht> cd .\Documents\GitHub\XAZH\xazh\
PS C:\Users\xazht\Documents\GitHub\XAZH\xazh> npm run build

> xazh@0.0.0 build
> vue-tsc && vite build
```

图 7.1 前端打包执行

图 7.1 是正在执行打包命令的图片，当打包完成后，便会出现图 7.2 中的各个模块文件详细列表，我们为了节约网络带宽，在前端代码中路由懒加载组件，让打包后的项目分离成小文件，这样可以使客户端按需请求前端资源。如果不使用懒加载办法，打包后的 js 文件将会是一个独立的非常巨大的文件。

```
✓ 4154 modules transformed.
dist/index.html                                          0.50 kB │ gzip:   0.33 kB
dist/assets/fontello-febc4ea8.woff2                      7.76 kB
dist/assets/fontello-8aa3e26e.woff                       9.07 kB
dist/assets/fontello-37711755.ttf                       15.40 kB
dist/assets/fontello-0286fb90.eot                       15.57 kB
dist/assets/fontello-022bf1a3.svg                       16.29 kB │ gzip:   5.72 kB
dist/assets/push-7f41af53.css                            0.04 kB │ gzip:   0.06 kB
dist/assets/Lottie-a7e35617.css                          0.05 kB │ gzip:   0.07 kB
dist/assets/Logo-add3b49a.css                            0.06 kB │ gzip:   0.08 kB
dist/assets/SignupPage-76dbb5fb.css                      0.15 kB │ gzip:   0.13 kB
dist/assets/404-9c6ddc5c.css                             0.17 kB │ gzip:   0.15 kB
dist/assets/403-ad85d8d4.css                             0.17 kB │ gzip:   0.15 kB
dist/assets/FnNotice-9821bb34.css                        0.39 kB │ gzip:   0.22 kB
dist/assets/UploadPic-f5c745a6.css                       0.44 kB │ gzip:   0.25 kB
dist/assets/WEBPanel-85ac1b10.css                        0.45 kB │ gzip:   0.28 kB
dist/assets/Blogs-280d39e6.css                           0.65 kB │ gzip:   0.31 kB
dist/assets/UserInfoCard-5148919b.css                    1.08 kB │ gzip:   0.38 kB
dist/assets/ShowBlog-6c864675.css                        1.41 kB │ gzip:   0.47 kB
dist/assets/BlogViewList-4aca2ec4.css                    1.42 kB │ gzip:   0.53 kB
dist/assets/SelfHome-8403e7f4.css                        1.52 kB │ gzip:   0.56 kB
dist/assets/blogoptions-4a3f704d.css                     1.63 kB │ gzip:   0.44 kB
dist/assets/EditBlog-62518663.css                        2.06 kB │ gzip:   0.77 kB
dist/assets/Editor-435cef1b.css                          2.39 kB │ gzip:   0.85 kB
dist/assets/Signup-b6368d6a.css                          2.71 kB │ gzip:   0.75 kB
dist/assets/Main-8dc6374e.css                            2.73 kB │ gzip:   0.78 kB
dist/assets/index-87fabf6f.css                           3.01 kB │ gzip:   0.88 kB
dist/assets/injectionKey-4a0c8518.js                     0.09 kB │ gzip:   0.09 kB
dist/assets/_plugin-vue_export-helper-c27b6911.js        0.09 kB │ gzip:   0.10 kB
dist/assets/index-aa0dc98b.js                            0.13 kB │ gzip:   0.14 kB
dist/assets/useFlexGapSupport-bf208b33.js                0.13 kB │ gzip:   0.13 kB
dist/assets/level.user.type-ca969d38.js                  0.16 kB │ gzip:   0.15 kB
dist/assets/AdminPanel-a3404b53.js                       0.19 kB │ gzip:   0.17 kB
dist/assets/test-d04de4e9.js                             0.21 kB │ gzip:   0.18 kB
```

图 7.2 前端打包成功部分列表

后端程序一般不需要打包，直接放在主机的 node 环境里，安装依赖后直接运行就好，但是查阅官方文档[10]后，发现在部署前先编译 TypeScript 可能会对性能有一定提升，也能更好管理代码。图 7.3 是执行构建后端代码并移除开发依赖的命令。构建好后端代码后，会在项目目录里生成 dist 目录，该目录里会生成 TypeScript 编译好的 JavaScript 文件。现在我们需要复制 dist 目录和项目目录下的 package.json、bootstrap.js、以及 node_modules 文件夹到主机，就可以直接使用 pm2 命令执行 bootstrap.js 文件来启动项目。

```
PS C:\Users\xazht\Documents\GitHub\XAZH\xazh-back> npm run build

> xazh-back@1.0.0 build
> midway-bin build -c

   - Copy Complete
Build Complete!
PS C:\Users\xazht\Documents\GitHub\XAZH\xazh-back> npm prune --production
npm WARN config production Use `--omit=dev` instead.

up to date, audited 209 packages in 3s

23 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

图 7.3 后端代码构建并移除开发依赖

7.3 功能测试

在部署成功后，需要对核心功能做一个简单的测试。测试环境是 Edge 浏览器，版本为：125.0.2535.51 (正式版本) (64 位)。

表 7-1 核心功能测试表

测试项	测试步骤	期望结果	实际结果
用户注册	打开注册页面，按照注册要求注册新账号。	能够成功注册用户。	与预期相符
用户登录	打开登录页面，使用注册的账号登录。	能够成功登录用户。	与预期相符
发布博客	点击进入编辑页面，插入一系列富文本，并发布。	能够成功上传图片，发布博客。	与预期相符
编辑博客	在个人中心，重新编辑已发布的博客。	能够编辑已有博客。	与预期相符
浏览博客	选择一篇博客，检查各种富文本及图片是否正常显示。	能够正确阅览博客，包括图片等富文本。	与预期相符

表 7-1 是个人主页系统的核心功能的测试结果，其中的各个测试项是处于用户视角的测试，一个测试项会包含很多细分功能，这些细分功能在前端实际上就是基础操作和 API 接口。前端操作和后端接口功能性测试在开发过程中会实时地进行，这些细分功能对于用户来说一般并不关心，但能够影响该表的测试结果。

7.4 性能测试

前端程序性能一般根据客户端浏览器以及客户端宿主机的配置而不同，一般不作为性能测试的测试项目；后端 API 请求时间测试、并发请求测试才是性能测试的主要方法。

图 7.4 后端 API 响应记录

由于使用的是 Node.js 作为后端运行时，在单线程程序中，高 IO 处理是非常轻松容易的，所以后端 API 请求时间基本都在毫秒级，如图 7.4。程序的并发测试也能抗住比较大的请求量。

附录一是使用 Apifox 进行的压力测试截图，可以看到在一分钟内请求了近 5000 次，平均每秒请求约 71 次，平均响应时间为 15 毫秒，这个响应速度是非常可观的，最大响应时间是 867 毫秒，不过依旧在 1 秒之内，90%的请求都在 20 毫秒内响应完成的，这充分说明该系统的性能非常优异。

实际的 API 请求时间不仅受程序影响，最重要的还是互联网的影响，比如主机的上行下行带宽，网络质量等。但这些都是可以根据实际需求去提升网络配置的。

8 结论

8.1 系统优缺点分析

整个个人网站系统比较庞大，其中运用了很多技术框架以及第三方库，部分功能是我结合需求原生开发的，所以无法做到十全十美。我将分析系统中所用技术带来的优缺点。

优点：

1. 使用 Vue3 和 Vite 框架，能够为开发带来灵活、快速、简单的特点；
2. 使用 Ant Design UI 库，统一了界面设计语言，提升了界面美观性；
3. 博客编辑使用比较主流的富文本编辑器 Markdown，拥有美观、易学、易编辑的文本处理；
4. 将系统产生的数据和文件，全部存储在数据库中，使系统数据更方便管理；
5. 后端使用单线程运行时 Node.js 让程序响应速度非常的快，也能轻松负载比较高的并发场景；
6. 整个系统架构设计非常清晰、合理、具有层次，不管是前端还是后端都能快速扩展新功能和特性。

缺点：

1. Vue3 虽然很主流，但是很多大型、稳定、企业化的库并没有得到很好的支持，对未来扩展有些许影响；
2. 为了使网站 UI 更加统一，不得不破坏性地修改原生 UI 库，有一些高度自定义的组件还需要特别定制；
3. 文件虽然和数据可以一起管理，但是数据库备份将会更加庞大；
4. 后端使用的是对称加密，在不知道加密算法时，系统安全性非常有保障，一旦加密算法泄露，那么系统将变的不安全；
5. Node.js 处理 IO 密集型任务非常具有优势，但是处理 CPU 密集型任务就没有太多优势了，因为它是单线程的。

8.2 开发总结和展望

该个人主页系统非常简洁和易用，可以作为极客的技术记录站点，也可作为各类人群的日常生活记录站点。

个人网站系统的开发让我收获了丰富的经验，涉及信息传输安全、性能调优、架构设计等诸多方面。在此过程中，我查阅了大量的资料，同时结合学业中所学的知识，成功地解决了这些问题。我也衷心希望我的解决方案能够为更多的开发者带来切实的帮助。

从前期的规划设计到后期的开发实施，我投入了大量的心血和精力。幸运的是，我取得了较为满意的成果。根据开发软件的统计数据，前端程序总共编写了约 7800 行代码，后端程序总计写了约 6500 行代码，整个系统的代码总量约为 14300 行。

在未来，我会进一步扩展、完善和维护这个系统，使其成为我生活中乐趣、实验中成果、实践中技巧的分享记录平台。

我能够独立完成这个系统，离不开学校的悉心培养以及各位老师的教导，尤其要感谢指导老师的精心指导。我将运用所学的知识和技术，为建设社会贡献力量，为人们提供更优质的服务。同时，也希望能为人类计算机事业的发展贡献自己的一份力量。

参考文献

[1] Microsoft.TypeScript for JavaScript Programmers[DB/OL].https://www.typescriptlang.org/docs/handbook/typescript-in-5-minutes.html.2012

[2] 朴灵.深入浅出 Node.js[M].北京:人民邮电出版社,2013.12.

[3] 许会元,何利力.NodeJS 的异步非阻塞 I/O 研究[J].工业控制计算机,2015,28(03):127-129.

[4] 刘博文.深入浅出 Vue.js[M].北京:人民邮电出版社,2019.3.

[5] 季甜甜,刘冬冬.基于 Vue 前端性能的研究与分析[J].阜阳师范大学学报(自然科学版),2024,41(01):15-22.

[6] Brad Dayley.MongoDB 入门经典[M].北京:人民邮电出版社,2020.2.

[7] MongoDB.MongoDB Limits and Thresholds[DB/OL].https://www.mongodb.com/docs/manual/reference/limits.2023.

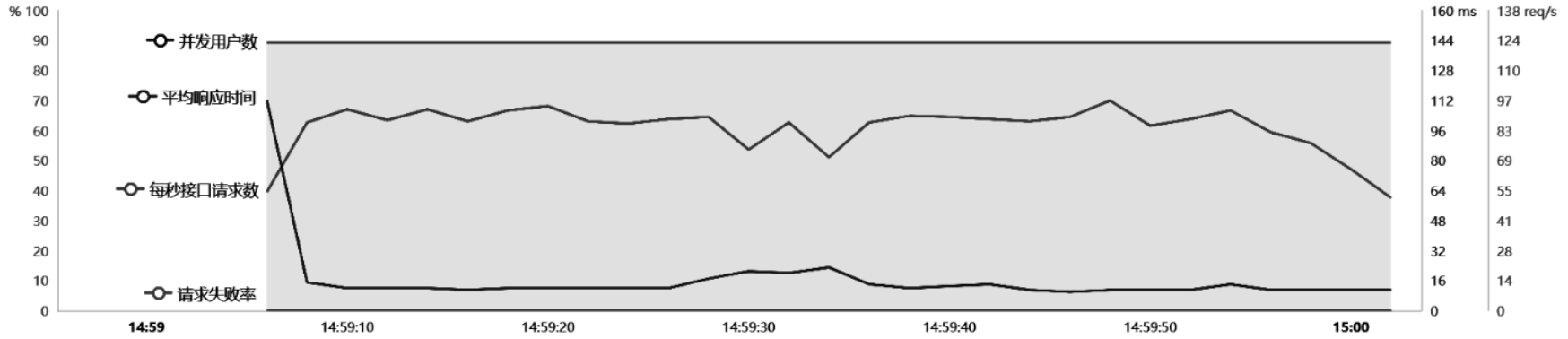
[8] 朱俭.面向方面编程(AOP)介绍[J].计算机工程,2004(S1):170-172.

[9] MDN Web Docs.Transfer-Encoding Header[DB/OL].https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Transfer-Encoding.2022.

[10] Midway.启动和部署 Midway[DB/OL].https://www.midwayjs.org/docs/deployment.2022.

← | 正式环境 压力测试

总请求次数	每秒接口请求数	平均响应时间	请求失败率
4,842	71.27	15 ms	0.00 %



接口请求	总请求数	每秒接口请求数	平均响应时间(ms)	最小响应时间(ms)	最大响应时间(ms)	90% 响应时间(ms)	失败率
POST 获取推荐博客 /Pull/Blog/Order	4,842	71.27	15	6	867	20	0

渭南师范学院本科毕业论文（设计）任务书

设计题目		基于Vue+Midway的个人主页设计与开发			
学生姓名	郭佳龙	院系、专业、班级	计算机学院 计专升本2022级2班	学号	221321060
指导教师	哈渭涛		职　称	教授	

一、文献查阅指引

[1]Martin.CleanCode:A Handbook of Agile Software Craftsmanship[M].Prentice Hall.

[2]唐灿.下一代Web界面前端技术综述[J].重庆工商大学学报:自然科学版.

[3]张海藩.软件工程导论(第6版)[M].北京:清华大学出版社.

[4]刘博文.深入浅出Vue.js[M].北京:人民邮电出版社.

[5]曹刘阳.编写高质量代码:Web前端开发修炼之道[M].北京:机械工业出版社.

[6]王卓,李春葆.现代软件工程[M].北京:清华大学出版社.

[7]赵达,张磊.Web前端开发技术[M].机械工业出版社.

[8]李娟.基于Vue+Node的高职院校学生成绩管理系统设计与实现[J].现代信息科技.

[9]张涵诚.JavaScript高级程序设计(第4版)[M].北京:机械工业出版社.

[10]李战,孙恒达..Vue.js权威指南[M].北京:电子工业出版社.

[11]陈锋,陈卫星.全栈开发：从前端到后端的实践[M].北京:清华大学出版社.

二、内容要求

1、本论文围绕基于 VUE+Midway 的个人主页设计与开发，需阐述开发背景等；

2、技术背景要分析技术栈，详介 Vue、Midway 及 MongoDB 特性；

3、系统设计做好需求、模块与架构分析；数据库设计涵盖技术分析、数据汇总与模型设计；

4、前端开发含 Vue 环境搭建等；

5、后端开发有 Midway 环境与架构等设计；

6、部署与测试包括各种环境配置等；

7、结论要分析优缺点与总结展望。

需深入研究和准确阐述这些重点以成高质量论文。

三、 进度安排

1.动员：2023年10月10日

2.毕业论文撰写起止时间（2023年10月11日—2024年4月21日）

（1）选题与完成开题报告（2023年10月11日—2024年1月10日）

确定选题下任务书：2023年10月11日—2023年10月31日

完成开题并提交开题报告：2023年11月1日—2024年1月10日

（2）论文撰写及修订（2024年1月11日—2024年4月21日）

（3）论文中期检查（2024年3月11日—2024年3月15日）

（4）论文定稿打印（2024年3月16日—2024年4月21日）

3.毕业论文审查与答辩（2024年4月22日—2024年5月29日）

（1）论文评阅（2024年5月10日前完成）

（2）答辩资格审查（2024年5月15日前完成）

（3）论文答辩（2024年5月18日-2024年5月29日）

4.论文成绩评定及各种材料整理上交（2024年5月30日—2024年5月31日）

指导教师（签名）_____

系 主 任（签名）_____

主管院长 （签名）_____ 年 月 日

注：1. 任务书由指导教师填写、经系主任及主管院长审批后，在第七学期末之前下达给学生。

2. 文献查阅指引，应是对查阅内容和查阅方法的指引，即查阅什么和怎样查阅。

渭南师范学院本科毕业论文（设计）开题报告

<table>
<tr><td>论文（设计）题目</td><td colspan="5">基于Vue+Midway的个人主页设计与开发</td></tr>
<tr><td>学生姓名</td><td>郭佳龙</td><td>院系、专业、班级</td><td colspan="2">计算机学院
计专升本2022级2班</td><td>学号</td><td>221321060</td></tr>
<tr><td>指导教师</td><td colspan="2">哈渭涛</td><td>职　称</td><td colspan="3">教授</td></tr>
</table>

一、拟开展研究的价值、意义

本项目鉴于个人主页对于部分人群所具有的必要性，进而开发出一款具备本地化、易用性且简约美观的个人主页系统。通过对数据的所有权以及高度的自定义化进行了分析，以此明确了个人主页系统给部分人群赋予了怎样特有的功能与目的。同时，也给那些想要独立开发个人主页的人群给予了一个开发模板，涵盖技术栈的剖析、系统架构的设计、数据库的规划、前后端的设计等等，特别是本项目中所运用到的一些特定的业务处理模式，为广大开发者提供了相应的一些解决办法。

二、研究步骤、方法及措施

首先进行需求分析，明确个人主页系统的需求；然后进行技术选型，选择合适的技术栈；接下来进行系统设计，包括系统架构和数据结构的设计；再按照设计进行前后端的开发；接着进行测试部署，包括功能测试、性能测试等；最后对系统进行总结展望，分析优缺点并提出改进方向。

研究方法包括文献研究、案例分析和需求导向，通过不断迭代开发来完善和优化系统。

三、论文拟定提纲

1.引言

2.技术背景

3.系统设计与架构

4.数据库设计

5.前端开发

6.后端开发

7.部署与测试

8.结论

四、主要参考文献

[1]朴灵.深入浅出 Node.js[M].北京:人民邮电出版社.

[2]唐灿.下一代 Web 界面前端技术综述[J].重庆工商大学学报:自然科学版.

[3]Brad Dayley.MongoDB 入门经典[M].北京:人民邮电出版社.

[4]刘博文.深入浅出 Vue.js[M].北京:人民邮电出版社.

[5]曹刘阳.编写高质量代码:Web 前端开发修炼之道[M].北京:机械工业出版社.

[6]冯爱花.基于 Vue 云管理平台 Web 前端性能优化的研究[J].长江信息通信.

[7]Smith,Performance Analysis of Node.js in Modern Web Development.Journal of Software Engineering Research.

[8]李娟.基于 Vue+Node 的高职院校学生成绩管理系统设计与实现[J].现代信息科技.

[9]刘洋.从零开始学 HTML5+CSS3+JavaScript[M].北京:清华大学出版社.

[10]张涛.Web 安全开发与防护[M].北京:机械工业出版社.

指导教师意见：_____　　主管院长意见：_____

指导教师签字：_____　　主管院长签字：_____

年　月　日　　年　月　日

注：开题报告是在导师的指导下，由学生填写。

渭南师范学院本科毕业论文（设计）中期检查表

<table>
<tr><td colspan="2">设计题目</td><td colspan="4">基于 Vue+Midway 的个人主页设计与开发</td></tr>
<tr><td>学生姓名</td><td>郭佳龙</td><td>院系、专业、班级</td><td>计算机学院
计专升本 2022 级 2 班</td><td>学号</td><td>221321060</td></tr>
<tr><td>指导教师</td><td colspan="2">哈渭涛</td><td>职　称</td><td colspan="2">教授</td></tr>
<tr><td colspan="6">完成情况及改进意见：
完成情况：通过对相关文献的深入研究和分析，已成功完成系统的设计与架构规划，以及数据库设计的相关工作。同时，通过实践验证了系统开发的可行性，并已开始着手进行系统的实际开发工作，前后端将同步推进。
改进意见：需要加快速度完成系统的开发，然后准备论文编写相关的事情。在论文编写方面，需要提前规划好论文的结构和内容，合理安排时间，确保论文能够高质量地完成。可以参考相关的学术论文和研究报告，学习优秀的写作技巧和方法，提高论文的学术水平和可读性。

指导教师签字：_____
年　月　日</td></tr>
<tr><td colspan="6">系部意见：

签字：_____
年　月　日</td></tr>
<tr><td colspan="6">主管领导签字：

签字：_____
年　月　日</td></tr>
</table>

注：完成情况及改进意见由指导教师填写。

渭南师范学院本科毕业设计登记表

<table>
<tr><td>论文（设计）题目</td><td colspan="5">基于Vue+Midway的个人主页设计与开发</td></tr>
<tr><td>学生姓名</td><td>郭佳龙</td><td>院系、专业、班级</td><td>计算机学院
计专升本2022级2班</td><td>学号</td><td>221321060</td></tr>
<tr><td>指导教师</td><td colspan="2">哈渭涛</td><td>职　称</td><td colspan="2">教授</td></tr>
<tr><td colspan="6">成绩评定</td></tr>
<tr><td colspan="6">指导教师评语：
郭佳龙同学的本科毕业论文选题新颖，具有实用价值。论文中系统分析了需求，采用科学的研究方法，深入探讨了前后端技术。论文体现了多项创新，包括文件分片至数据库和对称加密鉴权方案，展示了较强的技术能力和严谨的研究态度。设计按时完成任务，整体上具备较高的理论和实用价值。
达到了本科学生毕业设计的要求，同意参加答辩。</td></tr>
<tr><td>成 绩</td><td></td><td colspan="4">指导教师签字：_____　　年　月　日</td></tr>
<tr><td colspan="6">评阅人评语：
这篇本科毕业论文质量较高。从材料综述来看，作者广泛查阅了相关文献，综述部分内容详实，覆盖了个人主页设计的最新技术和趋势。论证过程中，作者逻辑清晰，步骤严谨，能够有效地将理论与实际应用结合，展示了扎实的理论基础和实践能力。结论部分总结了系统的设计与实现，客观评估了项目的优缺点，具有较高的说服力。
达到了本科学生毕业设计的要求，同意参加答辩。</td></tr>
<tr><td>成 绩</td><td></td><td colspan="4">评阅人签字：_____　　年　月　日</td></tr>
<tr><td colspan="6">答辩小组评语：
郭佳龙同学在答辩过程中能够正确阐述自己的主要观点，流利清晰地介绍了论文的研究内容和成果。面对答辩小组老师的问题，他能够从容应对，回答准确，展现出对研究内容的深入理解和熟悉程度。在规定的时间内，他完整地完成了答辩任务，展示了良好的时间管理能力。论文质量较高，选题新颖，论证严密，具有较强的创新性和实用价值。
经答辩小组研究，一致同意郭佳龙同学通过毕业设计答辩。</td></tr>
<tr><td>成 绩</td><td></td><td colspan="4">答辩小组组长签字：_____　　年　月　日</td></tr>
<tr><td colspan="6">答辩委员会审核意见：

总成绩_____　等级_____
答辩委员会主席签字：_____　　年　月　日</td></tr>
</table>

注：综合成绩由指导教师成绩（40%）、评阅教师成绩（20%）和答辩小组成绩（40%）组成

渭南师范学院本科毕业论文（设计）答辩记录

<table>
<tr><td colspan="2">论文（设计）题目</td><td colspan="5">基于Vue+Midway的个人主页设计与开发</td></tr>
<tr><td>学生姓名</td><td>郭佳龙</td><td>院系、专业、班级</td><td colspan="2">计算机学院
计专升本2022级2班</td><td>学号</td><td>221321060</td></tr>
<tr><td>指导教师</td><td colspan="2">哈渭涛</td><td>职　称</td><td colspan="3">教授</td></tr>
<tr><td>答辩记录</td><td colspan="6">填写学生论文汇报情况，答辩提问、学生回答等答辩过程。

汇报情况：
学生设计的是基于Vue和Midway的个人主页网站，网站主要是与博客相关，能够发布、浏览、删除、收藏博客等操作，也有用户管理和文件管理等内容。在论文中，该生首先全面分析了个人主页的意义及目的。接着，着重介绍了开发所用的框架。在系统设计与架构部分，对系统的需求进行了分析，并将其划分为不同的功能模块，详细描述了技术选型和系统架构的设计过程，以确保系统的稳定性和可扩展性。同时，详细介绍了前端和后端的开发过程。此外，通过对数据和数据库技术的研究，设计出一套更适合个人主页系统的数据模型。
在答辩过程中，该生能够清晰地阐述论文的主要内容和研究成果，也展示了自己的系统。同时，该生能够回答评委提出的问题，并对系统的进一步改进和完善提出了自己的想法和建议。

答辩提问1：系统所用的对称加密具体用在哪里？
答：对称加密算法主要用在前后端鉴权部分，因为前后端交互时，如果不对数据进行处理，那么一些权限比较敏感的操作可能会遭到破解，所以使用对称加密算法对数据进行处理，以保护操作和数据的安全。
答辩提问2：在主页中用到的背景模糊是如何实现的？
答：背景模糊是使用UI框架的模态功能，但是原生UI框架的模态功能不具有美感，所以通过改写一部分CSS内容，以实现具有高斯模糊的背景效果。
答辩提问3：如何实现的文件分片？
答：文件分片主要是为了将文件存储在数据库中，因为在MongoDB数据库中单个文档限制最大为16M字节，所以在后端会将文件的二进制分割成较小的片存储在数据库中，在客户端下载文件时，也得益于文件分片技术，每次请求数据会把内存消耗降低至一个文件片，极高地减轻了服务器的内存压力。</td></tr>
<tr><td colspan="2">答辩日期</td><td colspan="5">年　　月　　日</td></tr>
<tr><td colspan="7">答辩小组组长签字：</td></tr>
<tr><td colspan="5">答辩小组成员签字：</td><td>记录人</td><td></td></tr>
</table>