**河 南 科 技 学 院**

**2022届本科毕业论文（设计）**

**基于Golang+Vue的博客论坛的设计与实现**

|  |  |  |
| --- | --- | --- |
| **学 号：** |  | **20191544119** |
|  |  |  |
| **姓 名：** |  | **胡超** |
|  |  |  |
| **专 业：** |  | **通信工程** |
|  |  |  |
| **学 院：** |  | **信息工程学院** |
|  |  |  |
| **指导教师：** |  | **蔡磊(教授)** |
|  |  |  |
| **完成时间：** |  | **2023年5月4日** |

**摘 要**

**随着互联网的普及和信息化社会的发展，博客和论坛已经成为人们获取信息、分享观点和交流思想的重要平台。为了满足现代社交互动需求，本文以Golang和Vue为技术基础，设计并实现了一个具有现代化特点的博客论坛系统。系统采用前后端分离的架构，Golang作为后端语言，提供高性能、高并发的API接口服务，而Vue作为前端框架，提供丰富的用户界面和交互体验。**

**本系统具备用户注册、登录、发表文章、点赞等功能。本文详细阐述了系统的需求分析、架构设计、数据库设计、接口设计、系统实现、测试与部署等方面的内容，以期为类似项目的开发提供参考。通过本系统的设计与实现，旨在为现代博客论坛的发展提供一种高效、实用的解决方案。**

**关键词：Golang；Vue.js；前后端分离；MySQL; Redis**

**ABSTRACT**

**With the popularization of the Internet and the development of the information society, blogs and forums have become important platforms for people to obtain information, share views, and exchange ideas. To meet the needs of modern social interaction, this paper designs and implements a modern blog forum system based on Golang and Vue technologies. The system adopts a front-end and back-end separation architecture, with Golang as the back-end language to provide high-performance and high-concurrency API interface services, and Vue as the front-end framework to provide a rich user interface and interactive experience.**

**The system has functions such as user registration, login, posting articles, comments, and likes. This paper elaborates on the aspects of system requirement analysis, architecture design, database design, interface design, system implementation, testing, and deployment, aiming to provide a reference for the development of similar projects. The design and implementation of this system aim to provide an efficient and practical solution for the development of modern blog forums.**

**Keywords: Golang; Vue.js; Front-end and Back-end Separation; MySQL; Redis**

**目 录**

[1 绪论 1](#_Toc11035)

[1.1 课题背景 1](#_Toc28323)

[1.2 国内外研究现状 1](#_Toc17862)

[2 技术支持 3](#_Toc15197)

[2.1 SSM框架 3](#_Toc2308)

[2.1.1 Spring 4](#_Toc30331)

[2.1.2 Spring MVC 4](#_Toc17863)

[2.1.3 Mybatis 4](#_Toc32764)

[2.2 MySQL数据库 4](#_Toc19327)

[2.3 tomcat服务器 5](#_Toc25409)

[2.4 B/S架构 5](#_Toc13072)

[3 系统需求分析 5](#_Toc11410)

[3.1 用户需求分析 5](#_Toc15024)

[3.2 角色功能分析 5](#_Toc16410)

[4 系统设计 7](#_Toc385)

[4.1功能模块设计 7](#_Toc25995)

[4.1.1报名注册 7](#_Toc10526)

[4.1.2学生管理 8](#_Toc5633)

[4.1.3查询统计 8](#_Toc21624)

[4.1.4审核 8](#_Toc17887)

[4.2数据库设计 8](#_Toc30391)

[5 系统角色功能实现 9](#_Toc15774)

[5.1登陆页面 9](#_Toc17684)

[5.2报名注册页面 10](#_Toc23566)

[5.2.1报名学生 10](#_Toc8444)

[5.2.2录取名单 11](#_Toc16077)

[5.2.3交费注册 12](#_Toc16639)

[5.3学生管理页面 13](#_Toc3899)

[5.3.1课程管理 13](#_Toc11376)

[5.3.2考试管理 13](#_Toc19577)

[5.3.3成绩管理 14](#_Toc23021)

[5.4查询统计页面 15](#_Toc16360)

[5.4.1查询信息 15](#_Toc23138)

[5.4.2信息统计 16](#_Toc32612)

[5.5审核页面 16](#_Toc26110)

[结论 18](#_Toc22652)

[参考文献 19](#_Toc12980)

[致谢 20](#_Toc13168)

# 

# 1 绪论

## 1.1 课题背景及意义

随着互联网技术的发展和移动互联网的普及，人们在获取信息、交流思想和分享知识方面的需求不断增长。传统的博客和论坛在满足这些需求方面已经显得力不从心，用户对于内容质量、操作便捷性和交互体验等方面的要求越来越高。因此，构建一个具有现代化特点的博客论坛系统，以满足用户在网络时代的社交需求变得尤为重要。

近年来，前后端分离的架构逐渐成为web开发的主流方向，它能够实现前端与后端的解耦，提高开发效率和系统维护性。同时，随着编程语言和框架的不断创新，Golang和Vue.js等新兴技术日益成熟，为构建高性能、易用的现代化博客论坛系统提供了技术支持。

本课题旨在设计并实现一个现代化的博客论坛系统，以满足用户在网络时代的社交需求。通过采用Golang和Vue.js技术，提供高性能、高并发的服务，同时实现丰富的用户界面和交互体验，以吸引更多用户参与社交互动。此外，本课题通过实际项目的开发，探讨这些新技术在现代博客论坛系统中的应用方法和优化策略，为今后相关领域的技术研究和实践提供有益参考。

随着技术的不断进步和市场竞争的加剧，博客论坛行业亟需创新和变革。本课题通过研究和实现一个基于Golang和Vue.js的现代化博客论坛系统，为行业发展提供新的技术思路和实践经验，有助于推动行业创新和升级。此外，现代博客论坛系统的设计与实现涉及前后端技术、数据库设计、系统架构等多个方面的知识和技能。通过本课题的研究与实践，可以培养学生掌握相关技术，提高解决实际问题的能力，为社会培养具备专业技能的技术人才。

在开发过程中，将关注如何提高系统的性能和稳定性。Golang作为一种高性能的编程语言，具备出色的并发处理能力，能够有效地应对高负载场景。在系统设计阶段，将着重研究如何利用Golang的特性来实现高性能、高并发的后端服务。同时，将关注如何利用Vue.js框架来构建用户友好、响应迅速的前端界面，以提升用户体验。

综上所述，基于Golang和Vue.js的博客论坛系统设计与实现具有重要的研究意义。该课题既满足了现代用户的社交需求，推动了行业创新，又为相关技术研究和实践提供了宝贵的经验。同时，这一课题对于培养具备实际开发经验的技术人才具有积极的促进作用。通过本课题的研究与实践，将探讨新技术在现代博客论坛系统中的应用，促进行业创新，为相关领域的技术研究和实践积累宝贵经验，同时为社会培养具备专业技能的技术人才。

## 1.2 研究现状

1.2.1 国内研究现状

近年来，随着互联网技术的迅速发展和普及，国内关于博客论坛系统设计与

实现的研究日益受到重视。诸多研究者从不同角度对现有博客论坛系统进行了深入探讨，涉及前后端技术、数据库设计、系统架构等方面。在技术选型方面，许多研究着力于探究如何利用新兴技术如Golang和Vue.js等来构建高性能、易用的现代化博客论坛系统。

在前后端分离架构方面，国内研究者已经取得了一定的成果。通过引入RESTful API设计原则，实现前后端的解耦，从而提高了开发效率和系统维护性。此外，为了提升用户体验，部分研究关注了如何利用前端框架实现响应式设计、组件化、模块化等特性，简化前端开发流程。

在安全性方面，国内研究者针对博客论坛系统的数据安全和用户隐私保护问题展开了广泛的研究。相关研究成果涵盖了加密策略、安全认证机制和权限控制手段等多个方面，为提高系统的安全性做出了有益尝试。同时，针对常见网络攻击，如SQL注入、跨站脚本攻击（XSS）等，部分研究者提出了相应的防范策略，以确保系统的稳定运行。

1.2.2 国外研究现状

相较于国内研究现状，国外在博客论坛系统设计与实现方面的研究具有较长的历史和较为丰富的经验。国外研究者在前后端技术、数据库设计、系统架构等方面取得了一系列重要成果。同样，国外研究者也关注于如何应用新兴技术，如Golang和Vue.js，以构建高性能、易用的现代化博客论坛系统。

在前后端分离架构方面，国外研究者已经形成了较为成熟的理论体系和实践经验。许多国外研究成果关注于如何优化RESTful API设计和实现，以提高系统的性能和可维护性。此外，国外研究者在前端框架的应用方面取得了显著的成果，如响应式设计、组件化、模块化等方面的实践经验，为提升用户体验提供了有益参考。

在安全性方面，国外研究者对博客论坛系统的数据安全和用户隐私保护问题进行了深入研究。通过分析各种安全威胁和攻击手段，他们提出了一系列有效的加密策略、安全认证机制和权限控制方法。针对常见的网络攻击，如SQL注入、跨站脚本攻击（XSS）等，国外研究者提出了多种防范策略和解决方案，为保障系统的稳定运行提供了有力支持。

此外，在功能和模块方面，国外研究者针对博客论坛系统的实用性和扩展性进行了深入探讨。他们关注于如何实现丰富的功能和模块，以满足不同用户的需求。相关研究成果涉及用户注册、登录、文章发布、评论、点赞、私信、关注等功能，以及社区管理、标签分类、热门推荐等模块。这些研究为今后博客论坛系统的设计与实现提供了有益的借鉴和启示。

综合国内外研究现状来看，尽管博客论坛系统设计与实现方面的研究已经取得了一定的成果，但仍存在许多值得进一步探讨的问题。例如，在新兴技术如Golang和Vue.js的应用方面，如何发挥其优势以实现更高的性能和更好的用户体验仍有待深入研究。同时，在安全性、功能和模块设计等方面，如何根据实际需求进行优化和完善，也是今后研究的重要方向。

1.3 论文各章节安排

论文共分为七个章节，各章节内容如下：

第一章介绍了基于ROS的机械臂设计与控制研究的研究背景及设计本结构的意义，分析了国内外的发展现状和本系统的优势所在。

第二章对系统组成进行介绍，包括系统的模型仿真分析，系统的各部分硬件及其功能，整体系统结构的集成步骤，软件开发环境的介绍。

第三章着重对系统组成中的十字滑台硬件性能进行测试分析，以及进行对十字滑台的硬件装配，程序设计，以及效果验证分析。

第四章对系统结构组成中的伸缩杆硬件性能和精确度进行测试分析，以及进行对伸缩杆的硬件装配，程序设计，以及效果验证分析。

第五章是实验部分，先是对坐标系进行转化，矩阵变换为机械臂可执行的参数，通过ROS系统执行，对机械臂进行实时控制，以及结果验证和分析处理，包括执行效果和误差分析等。

第六章是对前期工作总结和实验过程的优势和不足之处进行分析记录，以及实验结果和整体系统结构进行分析，最后基于本系统的优势和不足之处对未来进行展望。

# 2 技术支持

## 2.1 Gin框架

框架Gin框架是一款基于Golang编程语言的高性能、轻量级的Web应用开发框架。自从发布以来，Gin框架已经逐渐成为Golang Web开发领域中广受欢迎的选择。其优势在于简洁的API设计、高性能的处理能力以及对中间件的支持，为开发者提供了一个便捷、灵活的Web开发工具。Gin框架因其简洁、高性能和易于使用而受到许多开发者的欢迎。

## 2.2 MySQL数据库

MySQL数据库是一款开源的关系型数据库管理系统，自1995年发布以来，已逐渐成为全球广泛使用的数据库解决方案之一。基于其高性能、稳定性、易用性和跨平台支持等特点，MySQL数据库在Web应用、企业信息系统、数据仓库等领域得到了广泛应用。此外，MySQL数据库提供了多种高可用性解决方案，如主从复制、集群等，以实现数据库的容错和负载均衡。在众多应用领域中，MySQL数据库已成为值得信赖的数据库解决方案。

## **2.3 Redis**

Redis（Remote Dictionary Server）是一款高性能的开源键值存储数据库，由意大利程序员Salvatore Sanfilippo于2009年首次发布。Redis支持多种数据结构，如字符串（String）、列表（List）、集合（Set）、有序集合（Sorted Set）以及哈希表（Hash）。得益于其高速缓存能力、数据持久化特性和分布式支持等优势，Redis在Web应用、分布式缓存、消息队列等领域得到了广泛应用。作为一个内存数据库，Redis能够实现高速的数据读写，大大降低了访问延迟。Redis作为一款高性能的键值存储数据库，在高速缓存能力、数据持久化、丰富的数据结构以及分布式支持等方面具有显著优势。

## 2.4 JWT

JSON Web Token（JWT）是一种开放标准（RFC 7519），用于在各方之间安全地传输和验证信息。JWT通常应用于身份验证和授权场景，例如，在用户登录一个Web应用后，服务器会生成一个JWT，并将其返回给客户端。之后，客户端在每次访问受保护资源时，都会携带这个JWT作为访问凭证。服务器接收到请求后，会验证JWT的有效性，并根据其携带的信息决定是否授权访问。JSON Web Token是一种简洁、自包含且安全的信息传输和验证方法。在现代Web应用和分布式系统中，JWT作为身份验证和授权的常用手段，具有广泛的实践应用价值。

## 2.5 Docker

Docker是一个开源的容器化平台，可以将应用程序和其依赖项打包成轻量级、可移植的容器。Docker容器提供了一种隔离环境，使得应用程序可以在任何地方以相同的方式运行，无论是在开发机、测试环境还是生产环境中。Docker具有快速启动、高效利用资源、易于管理和部署的优势，因此被广泛应用于云计算、DevOps、持续集成/部署等场景。

## 2.6 Vue框架

Vue是一种现代化的JavaScript框架，用于开发Web应用程序和单页应用。它提供了一种基于组件的架构，允许将UI拆分成独立的、可复用的组件，并通过模板语法将它们组合成完整的页面。Vue拥有许多特性，包括响应式数据绑定、计算属性、指令、事件处理、组件通信、生命周期函数等等。这些特性使得开发者可以更加方便地管理应用的状态和行为，从而提高开发效率并降低代码复杂度。Vue还提供了一个强大的插件系统，允许开发者将第三方库和工具集成到应用中。这些插件可以扩展Vue的功能，例如添加路由功能、数据管理、国际化等等。由于Vue的轻量级和易用性，其在前端开发领域得到了广泛的应用和支持，不少公司和团队选择使用Vue来构建商业级别的Web应用程序。

## 2.7 前后端分离

前后端分离是一种现代Web应用的架构模式，通过将用户界面（前端）与业务逻辑、数据处理（后端）进行分离，实现开发团队的解耦合与协作，提高开发效率和应用的可维护性。随着Web技术的发展，特别是HTML5、CSS3和JavaScript等前端技术的普及，以及RESTful API、微服务等后端技术的推广，前后端分离架构逐渐成为主流。总的来说，前后端分离架构为Web应用开发带来了诸多优势，有助于提高开发效率、应用性能和可维护性，已成为当今Web应用开发的主流模式。

## 2.8 B/S架构

B/S架构是指浏览器/服务器架构，它是一种客户端与服务器端相互配合的软件系统结构。在B/S架构中，用户通过浏览器向服务器发送请求并获得响应，而服务器则处理并返回请求的结果。相比于C/S架构，B/S架构具有更好的跨平台性和可维护性。B/S架构是一种全新的软件系统构造技术，是当今应用软件的首选体系结构。

# 3 网站总体设计

## 3.1 网站设计架构

如今高校的学生招生数量不断增加

## 3.2 功能模块设计

### 3.2.1报名注册

在双学位招生管理系统中，首先是报名注册模块。分别有报名学生、录取名单和交费注册三个功能。在允许报名的时间段内，符合双学位招收条件的学生可以先填写报名表进行报名，然后学生登录系统后查看自己的录取状态，看看是否被录取。管理系统后台的录取名单统计了本次双学位招收的所有学生，授课教师和教务管理员也可以查看录取的所有名单，教务管理员进行查阅修改后再次确认录取人数。最后被录取的学生看到自己的基本信息自行确认后可以进行交费注册，说明确定自己已选修双学位。教务管理员可以查阅学生的交费/欠费情况，导入已交费/欠费的信息，以便确认学生的注册/欠缴状态。

### 3.2.2学生管理

学生管理模块是本系统最重要的部分，学生用户在规定时间内完成报名和交费注册，确认过自己的已录取状态后，就可在下个学期参与课程选修、考核。学生管理子模块分别是课程管理、考试管理和成绩管理。教务管理员登录双学位管理系统后，负责在课程管理子模块发布课程科目和授课教师；学生查询课程，知道上课时间、地点、授课教师；授课教师查询自己教授的科目名称、上课时间、地点。每学期学习结束后会进行相关的考核，考核的方式由授课教师决定。学生最后可点击考试管理部分，查询个人考试地点和时间。考试结束后，老师拥有对考试成绩录入的权限。最后点击成绩管理子模块，教务管理员有查阅所有学生考试成绩的权力，学生也可查阅自己的成绩通过/未通过情况。

### 3.2.3查询统计

查询统计是双学位招生管理系统中的第三个模块，有两个功能，分别是查询信息和信息统计。教务管理员可以通过查询信息功能查看未通过课程考试的学生人数名单，通过页面的一个搜索栏查看更加具体的课程名称，通过或者未通过情况，方便发布通知。学生用户也可以查询自己的课程通过情况，或者通过页面的搜索栏检索自己所有的课程情况。使用信息统计这个功能，教务管理员可以更加清晰明了地看出本校参与选修双学位课程的学生人数以及相对于全校总人数的占比情况还有考试通过率等因素，从而对学校接下来的双学位课程开设设置计划有更好的建议。

### 3.2.4审核

审核模块是双学位招生管理系统后台的最后一个模块。只有教务管理员拥有审核权限，进入系统后，管理员可以查询、修改所有学生的毕业审核情况，负责对学生毕业信息进行管理、归档。预毕业学生也可以进入审核模块查看自己的毕业审核结果。

## 3.3 数据库设计

数据库是按数据结构来存储和管理数据的计算机系统，是应用程序的根基和软件设计的起点，数据库的功能是组织、存储、管理数据，使信息系统可以方便准确地获取信息。下面列出数据库中的用户表、社区表和帖子表。

用户表的设计信息如表4.1所示，在数据库表主要包含主键id、用户id、用户名、密码、邮箱、性别、创建时间、更新时间等基本情况。

**表4.1用户表**

|  |  |  |
| --- | --- | --- |
| 字段名称 | 类型 | 说明 |
| id  user\_id  username  password  email  gender  create\_time  update\_time | int  bigint  varchar  varchar  varchar  tinyint  timestamp  timestamp | 主键id  用户id  用户名  密码  邮箱  性别  创建时间  更新时间 |

帖子表的设计信息如表4.2所示，在数据库表中主要包含主键id、帖子id、标题、内容、作者的用户id、所属社区、帖子状态、创建时间和更新时间字段等基本情况。

**表4.2帖子表**

|  |  |  |
| --- | --- | --- |
| 字段名称 | 类型 | 说明 |
| id  post\_id  title  content  author\_id  community\_id  status  create\_time  update\_time | int  bigint  varchar  varchar  bigint  bigint  tinyint  timestamp  timestamp | 主键id  帖子id  标题  内容  作者的用户id  所属社区  帖子状态  创建时间  更新时间 |

社区表的设计信息如表4.3所示，在数据库表中主要包含主键id、社区id、社区名称、简介、创建时间和更新时间字段等基本情况。

**表4.3社区表**

|  |  |  |
| --- | --- | --- |
| 字段名称 | 类型 | 说明 |
| id  community\_id  community\_name  introduction  create\_time  update\_time | int  int  varchar  varchar  timestamp  timestamp | 主键id  社区id  社区名称  简介  创建时间  更新时间 |

# 4 论坛前端

## 4.1 注册

在本次研究中，项目开发了一个具有用户注册功能的前端主页。为了满足现代用户对于数据安全和隐私的需求，为注册过程设置了严格的字段验证，确保收集到的用户信息准确且符合规范。用户可以通过填写用户名、邮箱、密码、确认密码和性别这些必填字段进行注册。

注册表单的设计如下：

1. 用户名：要求用户输入一个具有唯一性的用户名，以便于识别和管理。
2. 邮箱：要求用户输入一个有效的电子邮件地址，以便用于接收系统通知、找回密码等功能。
3. 密码：要求用户设置一个安全的密码，以保护账户信息。
4. 确认密码：要求用户再次输入密码，以确保两次输入的密码一致。系统会对两次输入的密码进行比较，只有在它们完全一致时，注册过程才会继续进行。
5. 性别：要求用户选择自己的性别。提供了“男性”、“女性”两个选项，以满足不同用户的需求。在选择性别时，用户只能选择一个选项。

如下图4-1所示。



图 4-1 注册页面

在用户完成表单填写并提交后，系统对所有字段进行前端验证。如果发现任何不符合要求的字段，系统会立即提示用户进行修改。只有在所有字段都满足验证规则后，注册请求才会被发送到后端服务器进行进一步处理。

总之，通过设计一个严谨的注册表单和实现相应的前端验证功能，确保了用户注册过程的安全性和有效性。这有助于提升用户体验，同时降低因为错误信息导致的客户支持成本。

## 4.2 登录

登录页面的设计注重易用性，同时兼顾安全性。页面上仅包含两个主要字段：用户名和密码。

用户名字段要求用户输入在注册过程中所创建的唯一用户名，该用户名用于唯一标识用户。密码字段则要求用户输入相应的账户密码。为了确保登录过程的安全性，系统对密码字段进行了掩码处理，以防止他人窥视。用户填写完毕后，只需点击登录按钮，即可提交表单。

在后端处理登录请求时，系统会验证用户名和密码的正确性。只有在用户名和密码匹配的情况下，用户才能成功登录并访问受保护的资源。该登录页面不仅简化了用户登录过程，还为整个应用提供了一道安全屏障。如下图4-2所示。



图 4-2 登录页面

## 4.3 首页

## 4.4 网站状态

## 4.5 帖子列表

## 4.6 发表帖子

## 4.7 帖子详情

## 4.8 社区详情

## 4.9 Github热门项目排行榜

## 4.10 Golang热门项目排行榜

## 4.11 每日一言

# 5 论坛后端

## 5.1 JWT认证

JWT就是一种基于Token的轻量级认证模式，服务端认证通过后，会生成一个JSON对象，经过签名后得到一个Token（令牌）再发回给用户，用户后续请求只需要带上这个Token，服务端解密之后就能获取该用户的相关信息了。

下面演示在Gin框架中使用JWT：

首先，需要定制自己的需求来决定JWT中保存哪些数据，本项目规定在JWT中要存储user\_id、username信息，那么就定义一个MyClaims结构体，如图5-1所示。



图 5-1 自定义Claims

然后定义JWT的过期时间，这里定义为7天，如图5-2 所示。



图 5-2 定义JWT的过期时间

接下来还需要定义一个用于签名的字符串,如图5-3 所示。

]BUV[~{T{{7IC$]BA{%_UJC图 5-3 定义用于加密的字符串

据自己的业务需要封装一个生成 token 的函数，如图5-4 所示。

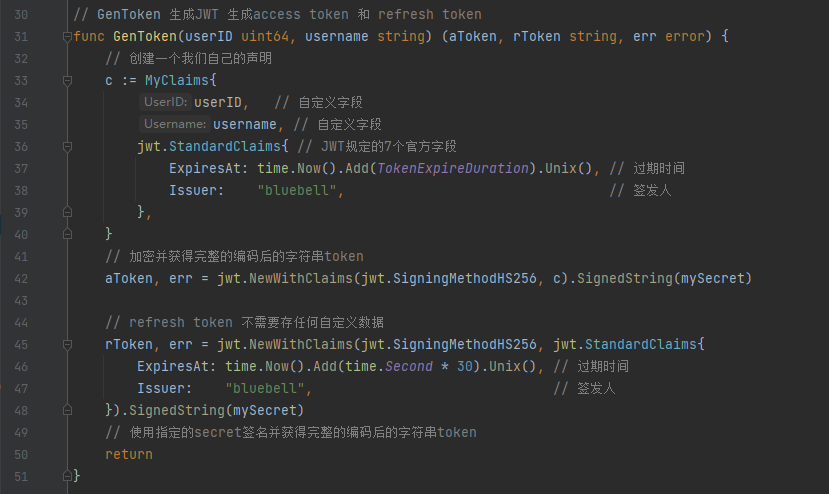


图 5-4 生成access\_token和refresh\_token的函数

根据给定的 JWT 字符串，解析出数据,如图5-5所示。

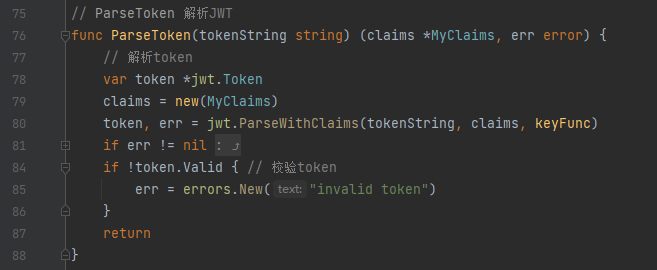


图 5-5 解析JWT的函数

当access\_token过期时,可以选择用refresh\_token来进行刷新,函数如图5-6所示。

****

图 5-6 **刷新access\_token的函数**

## 5.2 分布式ID：雪花算法

Sonyflake是一个可扩展的、高性能的、全局唯一ID生成算法，旨在满足分布式系统中ID生成的需求。该算法基于Twitter的雪花算法（Snowflake）设计，通过组合时间戳、机器ID和序列号来生成64位的整数ID。这些ID具有以下特性：

1. 全局唯一：确保在分布式环境下生成的ID不会发生冲突。
2. 时间有序：生成的ID随时间单调递增，便于排序和检索。
3. 高性能：算法具有低延迟和高吞吐量，可在高并发场景下使用。

首先，在项目中安装sony/sonyflake库，执行如下命令。

**go get -u github.com/sony/sonyflake**

然后，在需要使用ID生成器的Go文件中引入库，如图5-7所示。

****

**图5-7 系统结构图**

初始化ID生成器在代码中创建一个新的Sonyflake实例，用于生成全局唯一ID，如图5-8所示。

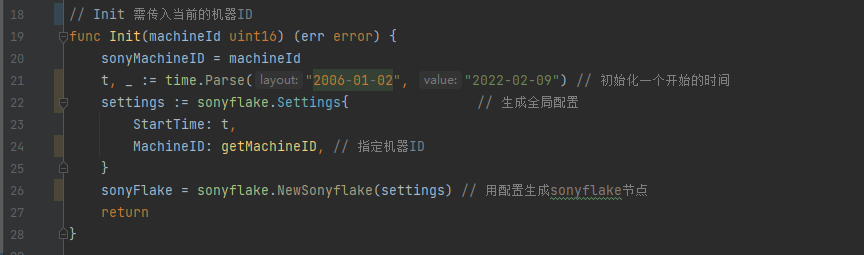


图 5-8 生成全局唯一ID

使用NextID()方法从Sonyflake实例生成新的ID，如图5-9所示。

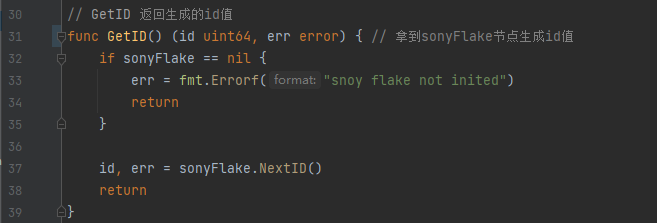


图 5-9 从Sonyflake实例生成新的ID

总结：Sonyflake是一个适用于分布式环境的高性能全局唯一

## 5.2 配置文件管理：viper

[Viper](https://github.com/spf13/viper" \t "_blank)是适用于Go应用程序的完整配置解决方案。它被设计用于在应用程序中工作，并且可以处理所有类型的配置需求和格式。在构建现代应用程序时，你无需担心配置文件格式；你想要专注于构建出色的软件。Viper的出现就是为了在这方面帮助你的。

安装，执行如下命令。

go get github.com/spf13/viper

读取配置文件。下面是一个如何使用Viper搜索和读取配置文件的示例。不需要任何特定的路径，但是至少应该提供一个配置文件预期出现的路径。如图5-10所示。

8G8{G$VCX`S_E@N[VO9LHFD

图 5-10 读取配置文件

在加载配置文件出错时，可以像下面这样处理找不到配置文件的特定情况，如图5-11所示。

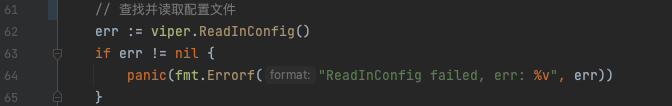


图 5-11 查找并读取配置文件

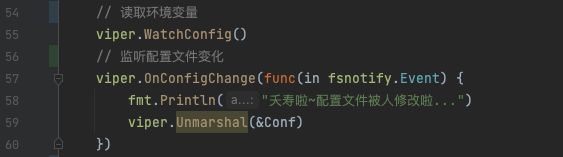
需要重新启动服务器以使配置生效的日子已经一去不复返了，viper驱动的应用程序可以在运行时读取配置文件的更新，而不会错过任何消息。只需告诉viper实例watchConfig。可选地，你可以为Viper提供一个回调函数，以便在每次发生更改时运行。如图5-12所示。

图 5-12 监控并重新读取配置文件

当需要将viper读取的配置反序列到我们定义的结构体变量中时，一定要使用mapstructure tag，如图5-13所示。



图 5-13 把读取到的配置信息反序列化到Conf变里中

## 5.3 参数校验：validator

在web开发中一个不可避免的环节就是对请求参数进行校验，通常会在代码中定义与请求参数相对应的模型（结构体），借助模型绑定快捷地解析请求中的参数，例如 gin 框架中的Bind和ShouldBind系列方法。本文就以 gin 框架的请求参数校验为例，介绍一些validator库的实用技巧。

validator库本身是支持国际化的，借助相应的语言包可以实现校验错误提示信息的自动翻译。下面的示例代码演示了如何将错误提示信息翻译成中文，翻译成其他语言的方法类似。如图5-14、5-15所示。

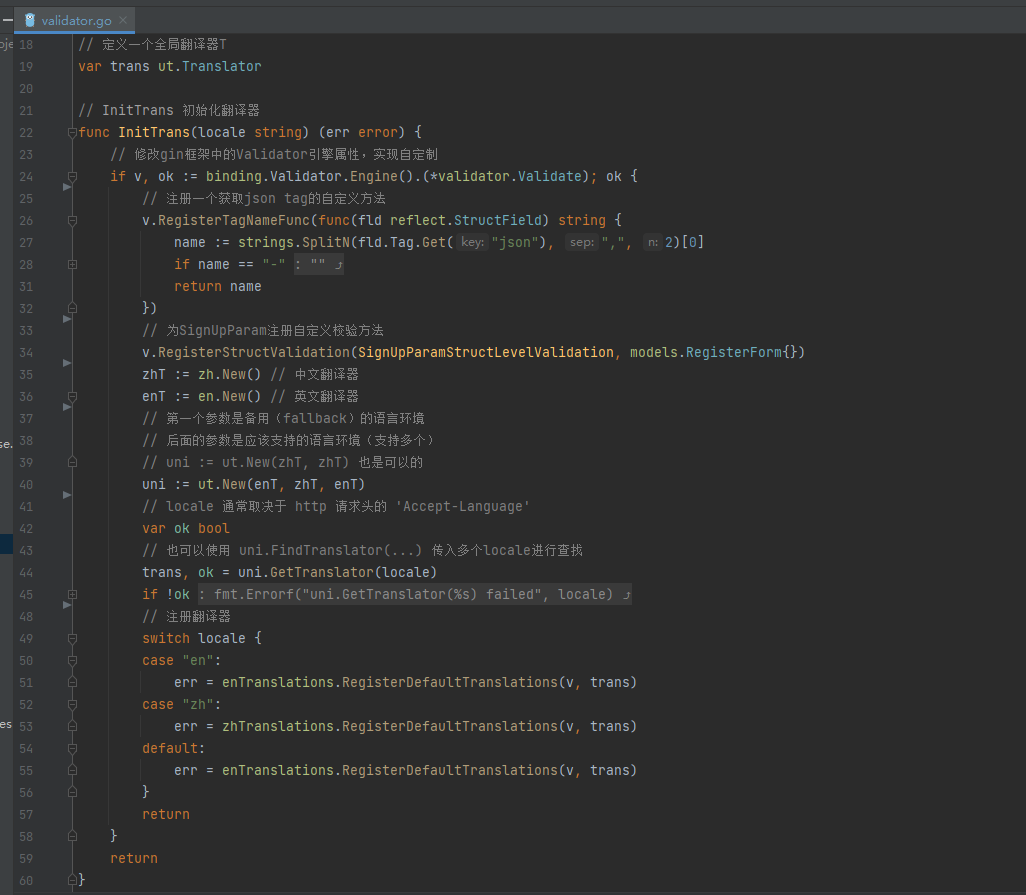


图 5-14 初始化翻译器

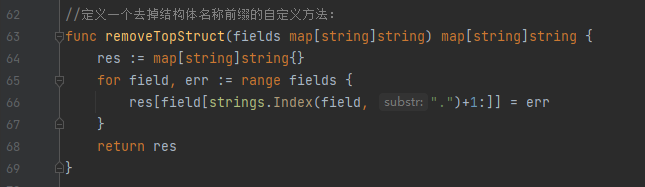


图 5-15 定义一个去掉结构体名称前缀的自定义方法

最后，在代码中使用上述函数将翻译后的errors做一下处理即可。如图5-14所示。

图 5-16 翻译errors

上面的校验还是有点小问题，就是当涉及到一些复杂的校验规则，比如re\_password字段需要与password字段的值相等这样的校验规则，上面自定义错误提示字段名称方法就不能很好解决错误提示信息中的其他字段名称了。此时如果想要追求更好的提示效果，将上面的Password字段也改为和json tag一致的名称，就需要自定义结构体校验的方法。例如，为SignUpParam自定义一个校验方法如下图5-17所示。



图 5-17 自定义SignUpParm结构体校验函数

然后在初始化校验器的函数中注册该自定义校验方法即可。

## 5.4 日志库：zap

在许多Go语言项目中，需要一个好的日志记录器能够提供下面这些功能：

1. 能够将事件记录到文件中，而不是应用程序控制台。
2. 日志切割-能够根据文件大小、时间或间隔等来切割日志文件。支持不同的日志级别。例如INFO，DEBUG，ERROR等。
3. 能够打印基本信息，如调用文件/函数名和行号，日志时间等。

[Zap](https://github.com/uber-go/zap" \t "https://www.liwenzhou.com/posts/Go/zap/_blank)就是是非常快的、结构化的，分日志级别的Go日志库。

本节将介绍在基于gin框架开发的项目中如何配置并使用zap来接收并记录gin框架默认的日志和如何配置日志归档。

安装，执行如下命令。

go get -u go.uber.org/zap

创建一个全局的 Logger 对象，如下图5-18所示。



图 5-18 全局Logger对象

然后编写一个初始化日志记录器的函数。它接收一个日志配置对象（cfg）和一个模式字符串（mode），并返回一个错误。如下图5-19所示。

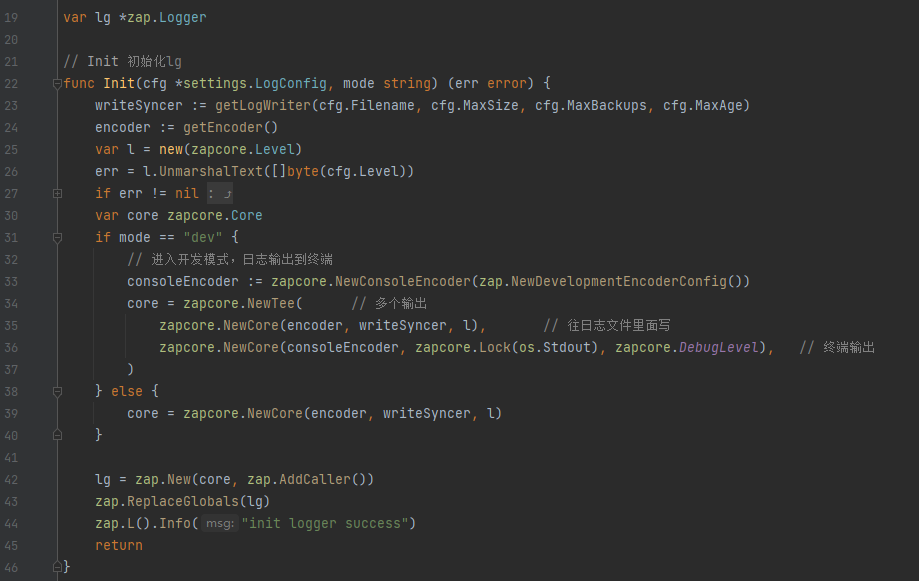


图 5-19 初始化日志记录器

这段代码的作用是根据给定的配置和模式创建一个 Logger 对象，并将其设置为全局的默认 Logger。根据不同的模式，日志可以同时输出到终端和文件中，或者只输出到文件中。

接下来创建一个 JSON 编码器，用于格式化日志并创建一个日志写入器。如图5-20、5-21所示。

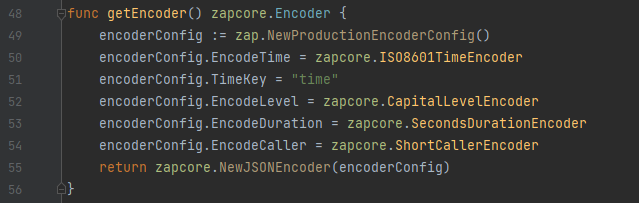


图 5-20 JSON编码器

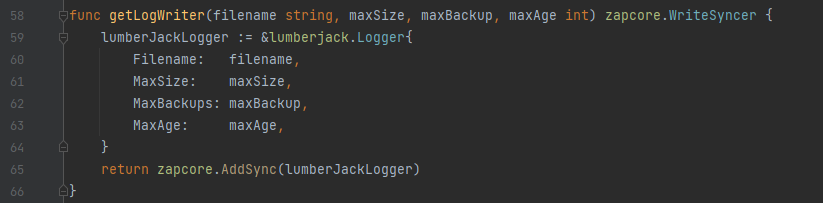
它使用 Zap 提供的生产环境编码器配置，并将时间编码为 ISO8601 格式、设置时间键名为 "time"、将日志级别编码为大写形式、将持续时间编码为秒，并使用短格式编码调用者信息。

图 5-21 日志写入器

它使用 Lumberjack 库创建一个具有文件名、最大大小、最大备份数和最大保留时间的日志写入器，并将其封装为一个 WriteSyncer 对象。

最后，使用zap实现两个自定义中间件GinLogger用于接收gin框架默认日志、GinRecovery用于recover掉项目中可能出现的panic。如图5-22、5-23所示。



图 5-22 自定义中间件GinLogger

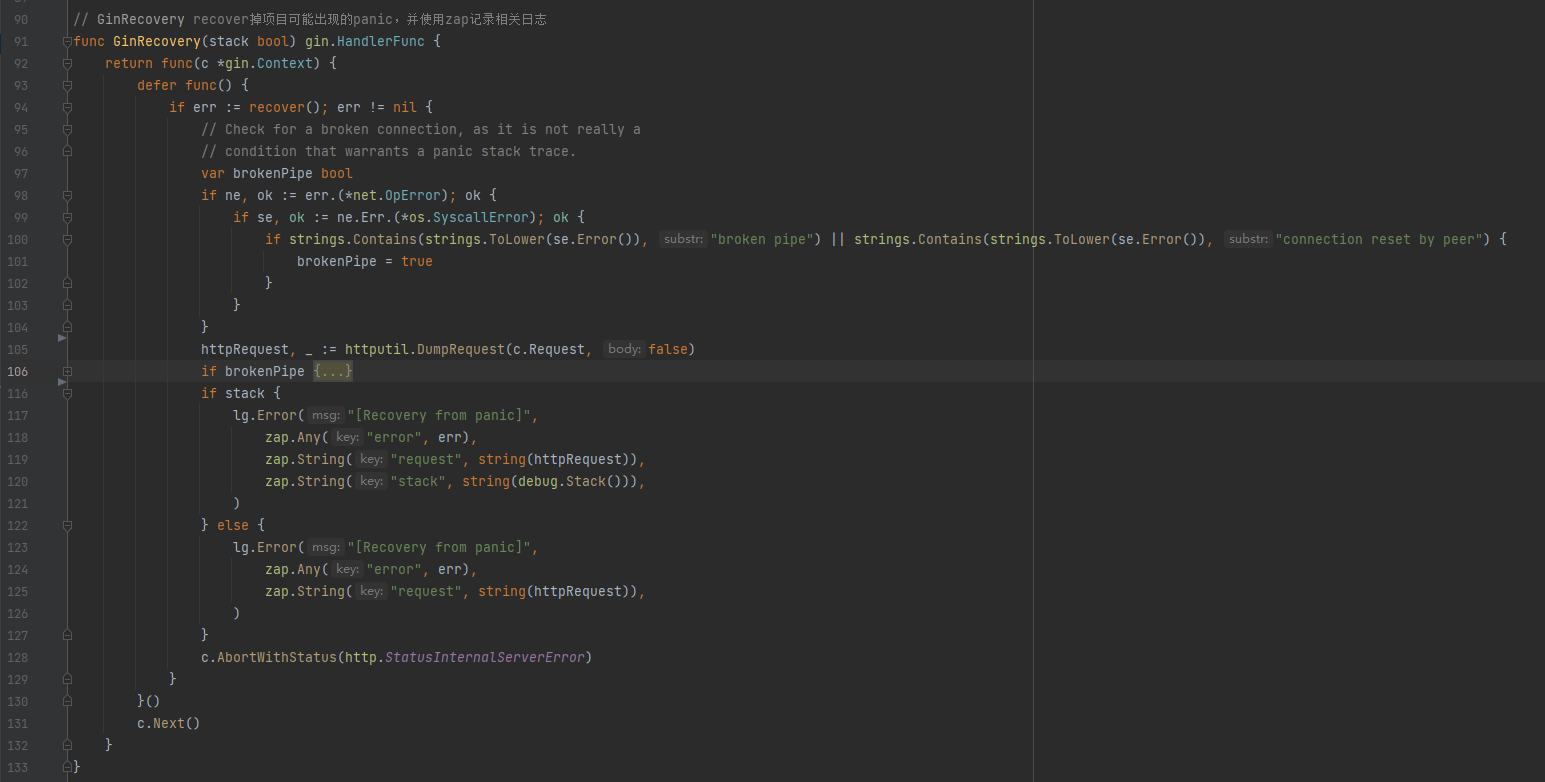
它在处理请求之前记录请求的起始时间，并在请求处理完成后计算请求处理的耗时。然后，使用 Logger 对象记录请求的相关信息，包括请求路径、状态码、请求方法、查询参数、客户端 IP、用户代理、错误信息和处理耗时。

图 5-23 自定义中间件GinRecovery

它使用 defer 和 recover 捕获可能出现的 Panic，并进行处理。如果发生 Panic，它会检查是否是由于连接中断导致的，并记录相关的错误信息和请求内容。如果是连接中断导致的 Panic，则记录错误并中止请求处理。如果不是连接中断导致的 Panic，则根据传入的 stack 参数决定是否记录堆栈信息。最后，将请求的处理状态设置为 500，并中止请求处理。

综上，这些函数提供了一个简单而强大的日志记录功能，可以方便地集成到基于 Gin 的 Web 应用程序中。

## 5.5 令牌桶限流

限流又称为流量控制（流控），通常是指限制到达系统的并发请求数。生活中也会经常遇到限流的场景，比如：某景区限制每日进入景区的游客数量为8万人；沙河地铁站早高峰通过站外排队逐一放行的方式限制同一时间进入车站的旅客数量等。限流虽然会影响部分用户的使用体验，但是却能在一定程度上报障系统的稳定性，不至于崩溃（大家都没了用户体验）。

本项目使用令牌桶限流策略来保证系统后端的稳定性。

令牌桶的原理，令牌桶按固定的速率往桶里放入令牌，并且只要能从桶里取出令牌就能通过，令牌桶支持突发流量的快速处理。对于从桶里取不到令牌的场景，可以选择等待也可以直接拒绝并返回。如图5-24所示。



图 5-23 自定义令牌桶限流中间件

虽说是令牌桶，但是没有必要真的去生成令牌放到桶里，只需要每次来取令牌的时候计算一下，当前是否有足够的令牌就可以了，具体的计算方式可以总结为下面的公式：

当前令牌数 = 上一次剩余的令牌数 + (本次取令牌的时刻-上一次取令牌的时刻)/放置令牌的时间间隔 \* 每次放置的令牌数

对于该限流中间件的注册位置，我们可以按照不同的限流策略将其注册到不同的位置，例如：

1. 如果要对全站限流就可以注册成全局的中间件。
2. 如果是某一组路由需要限流，那么就只需将该限流中间件注册到对应的路由组即可。

## 5.6 投票排名算法

根据帖子的发布时间以及帖子获得的赞成票数和反对票数实现帖子的实时动态排名功能，是本项目的核心。下面讲解本项目的帖子投票排名算法。

如果文章A有100张赞成票、5张反对票，文章B有1000张赞成票、950张反对票，谁应该排在前面呢。怎样才能将赞成票和反对票结合起来，计算出一段时间内最受欢迎的文章呢。本项目设计的算法规则如下。

投票分为四种情况：1.投赞成票、2.投反对票、3.取消投票、4.反转投票。

记录文章参与投票的人，谁给哪个帖子投了什么票，将此信息记录至Redis数据库。

本项目使用简化版的投票分数，用户点击投票后，更新文章分数：赞成票要加分，432分、反对票减分，现用v表示。

当v=1时，有两种情况：

1.之前没投过票，现在要投赞成票，更新分数和投票记录，差值的绝对值为：1，帖子+432分

2.之前投过反对票，现在要改为赞成票，更新分数和投票记录，差值的绝对值为：2，帖子+432\*2分

当v=0时，有两种情况：

1.之前投过反对票，现在要取消，更新分数和投票记录，差值的绝对值为：1，帖子+432分

2.之前投过赞成票，现在要取消，更新分数和投票记录，差值的绝对值为：1，帖子-432分

当v=-1时，有两种情况

1.之前没投过票，现在要投反对票，更新分数和投票记录，差值的绝对值为：1，帖子-432分

2.之前投过赞成票，现在要改为反对票，更新分数和投票记录，差值的绝对值为：2，帖子-432\*2分

投票的限制：

每个帖子子发表之日起一个星期之内允许用户投票，超过一个星期就不允许投票了。

1、到期之后将Redis中保存的赞成票数及反对票数存储到Mysql表中

2、到期之后删除Redis记录用户及投票类型的key：KeyPostVotedZSetPrefix

核心代码逻辑如图5-24所示。

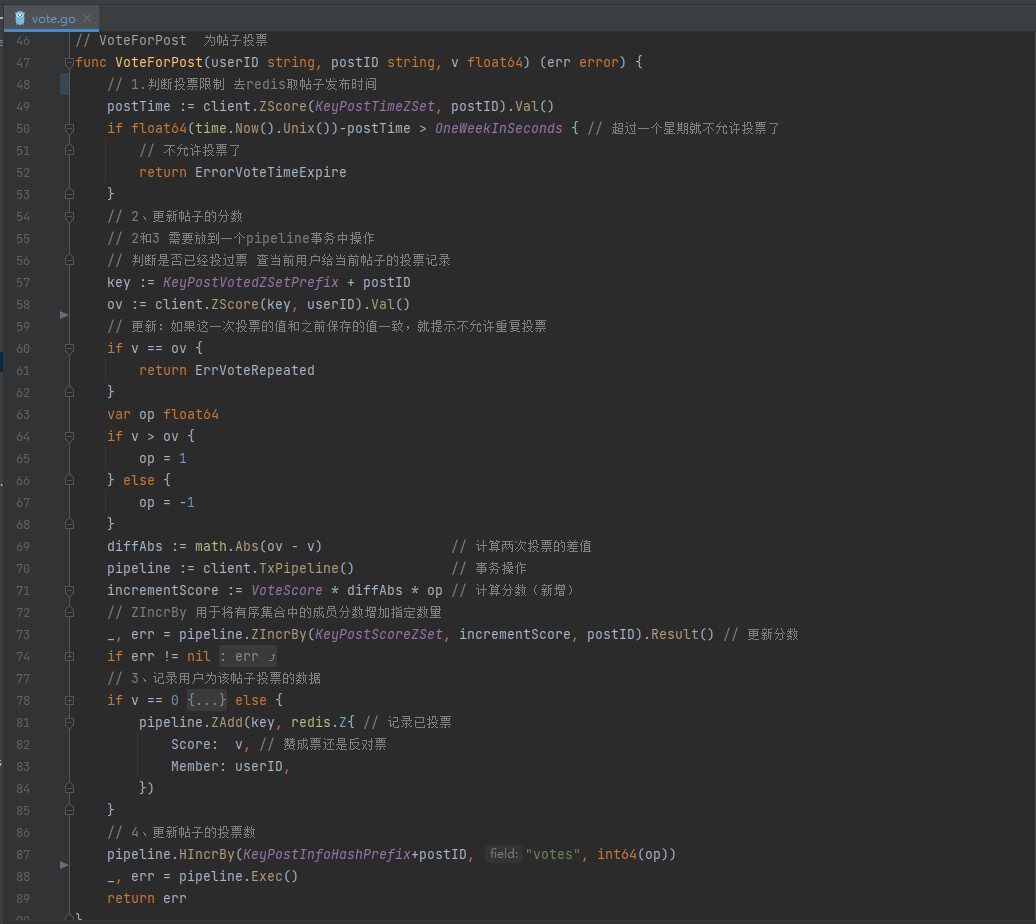


图 5-24 帖子投票算法函数

这段代码考虑了这样几个因素：

1. 帖子的新旧程度postTime

postTime的单位为秒，用unix时间戳计算。不难看出，一旦帖子发表，postTime就是固定值，不会随时间改变，而且帖子越新，postTime值越大。

1. 赞成票与反对票的差diffAbs。diffAbs = 赞成票 - 反对票

表示赞成票与反对票之间差额的绝对值。如果对某个帖子的评价，越是一边倒，diffAbs就越大。如果赞成票等于反对票，diffAbs就等于1。

1. 投票方向op。op是一个符号变量，表示对文章的总体看法。如果赞成票居多，op就是+1；如果反对票居多，op就是-1；如果赞成票和反对票相等，op就是0。

op的作用是产生加分或减分。当赞成票超过反对票时，这一部分为正，起到加分作用；当赞成票少于反对票时，这一部分为负，起到减分作用；当两者相等，这一部分为0。这就保证了得到大量净赞成票的文章，会排在前列；赞成票与反对票接近或相等的文章，会排在后面；得到净反对票的文章，会排在最后（因为得分是负值）。

结论就是，本项目的帖子排名，基本上由发帖时间决定，超级受欢迎的文章会排在最前面，一般性受欢迎的文章、有争议的文章都不会很靠前。

## 5.7 热重启：air

在使用Go语言的gin框架在本地做开发调试的时候，经常需要在变更代码之后频繁的按下Ctrl+C停止程序并重新编译再执行，这样就不是很方便。

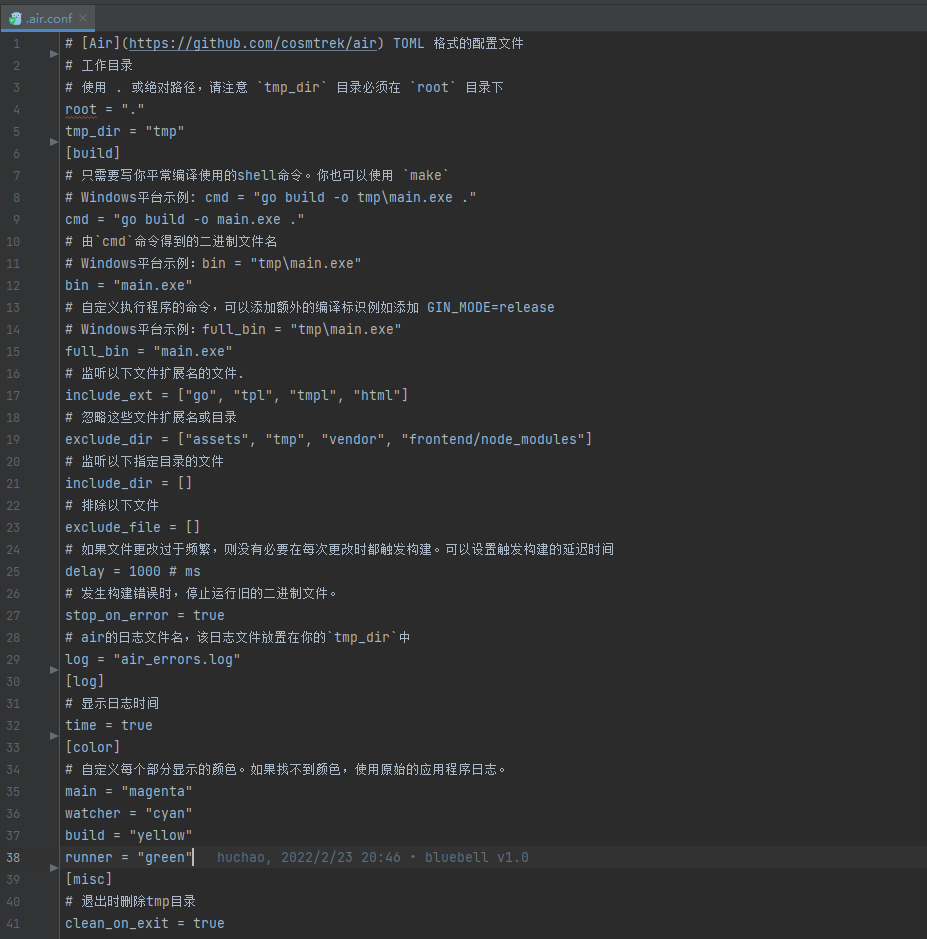
Air能够实时监听项目的代码文件，在代码发生变更之后自动重新编译并执行，大大提高gin框架项目的开发效率。因此，本项目使用Air来实时监听后端项目的代码文件。如图5-25所示。

图 5-25 air.conf配置

# 6 网站部署

## 6.1 安装Docker

在部署之前，请确保已经在服务器上安装了Docker。可以访问Docker官方文档([https://docs.docker.com/engine/install/](https://docs.docker.com/engine/install/" \t "_blank)) 获取安装教程。

## 6.1 编写Dockerfile

要创建Docker镜像（image）必须在配置文件中指定步骤。这个文件默认我们通常称之为Dockerfile。（虽然这个文件名可以随意命名它，但最好还是使用默认的Dockerfile。）

DockerFile构建镜像文件如图6-1所示。

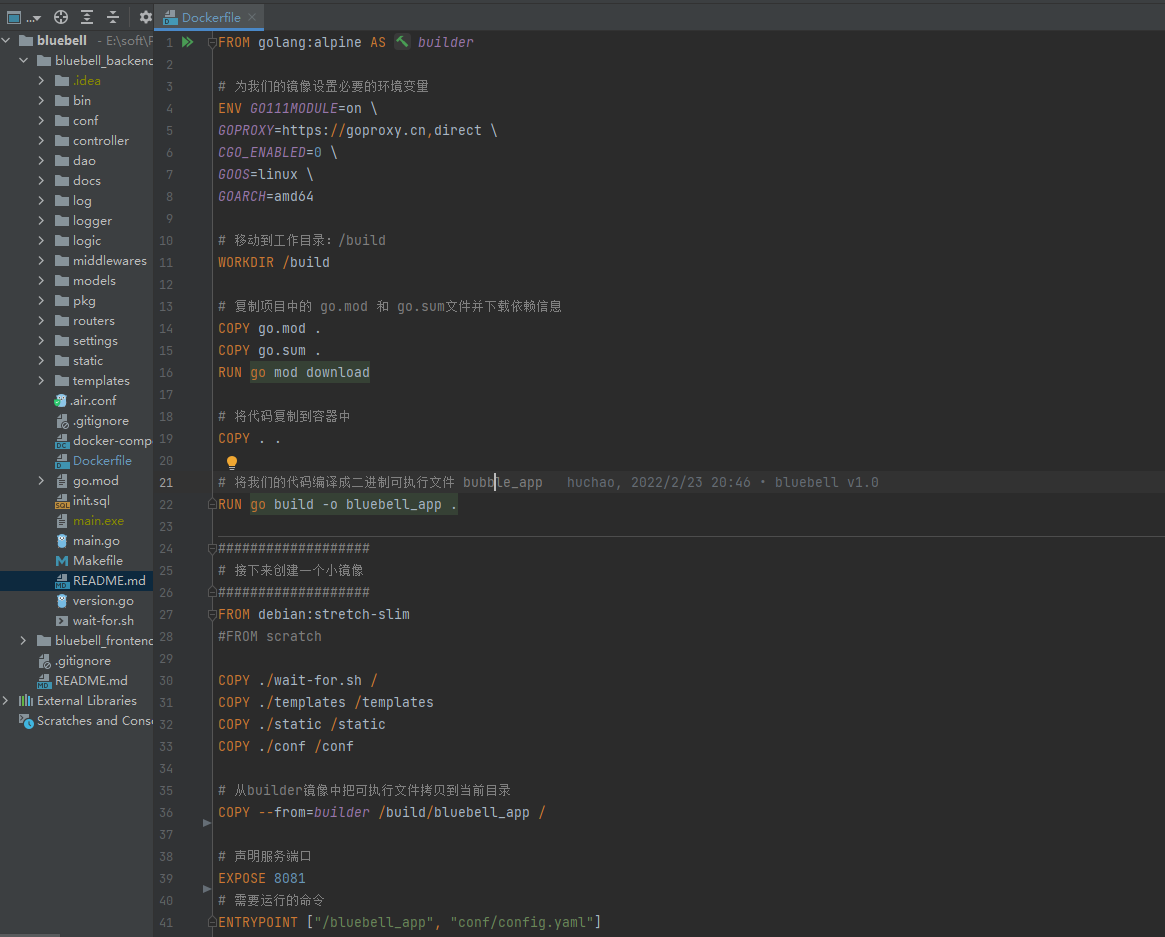


图 6-1 DockerFile构建镜像文件示意图

## 6.1 构建Docker镜像

在项目目录下，执行下面的命令根据Dockerfile构建Docker镜像，并指定镜像名称为bluebell\_app :

docker build -t bluebell\_app .

等待构建过程结束，构建成功后输出如下图6-2所示：



图 6-1 根据DockerFile构建镜像成功示意图

## 6.1 运行Docker容器

准备好了镜像，但是目前它什么也没做。接下来运行镜像，以便它能够处理请求。运行中的镜像称为容器。执行下面的命令来运行镜像：

docker run -d -p 8081:8081 --name bluebell\_app bluebell\_app

标志位-p用来定义端口绑定。由于容器中的应用程序在端口8081上运行，将其绑定到主机端口也是8081。如果要绑定到另一个端口，则可以使用-p $HOST\_PORT:8081。例如-p 5000:8081。

启动后的容器如图6-3所示。

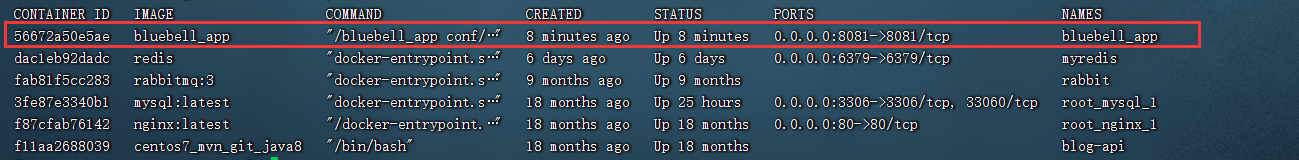


图 6-3 Docker容器示意图

## 6.1 访问网站应用

现在，可以通过访问服务器的IP地址和相应的端口来访问应用程序。例如，如果服务器IP地址是47.93.20.204，那么可以通过访问`http://47.93.20.204:8081/`来查看前端页面，如下图6-4所示。



图 6-4 前端网页

通过访问`http://47.93.20.204:8081/api/v1`来访问后端API,如下图6-5所示。

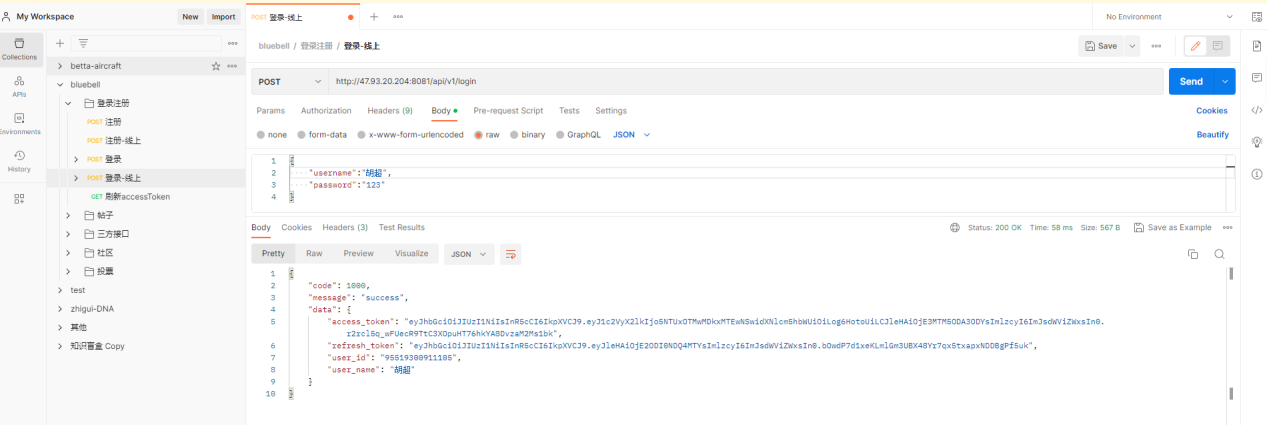


图 6-6 线上接口测试成功示意图

图6-6所示，接口返回正常，网站部署成功。

# 7 结论

本文主要研究了基于Golang和Vue.js的博客论坛设计与实现。通过深入分析了Golang的优势、Gin框架的特点和Vue.js的优势，我们成功地设计和实现了一个功能齐全、性能优越的博客论坛系统。

在本文中，详细介绍了后端和前端的设计与实现。后端采用Golang语言和Gin框架，实现了用户登录注册、文章管理、点赞管理、分类以及全文搜索等功能。前端采用Vue.js框架，实现了用户注册与登录、文章列表展示、文章详情展示、发表与编辑文章等页面。

本项目的实现充分体现了Golang和Vue.js技术的优势。Golang语言因其高性能、简洁的语法和丰富的库支持，使得后端开发变得高效而稳定。Vue.js则为前端开发提供了轻量、易学、高效的特点，有助于提升开发速度和用户体验。

此外，本项目的实践过程也为开发者提供了宝贵的经验，有助于加深对Golang和Vue.js技术的理解，为今后的Web开发奠定基础。当然，本项目还存在一定的改进空间，例如可以尝试引入更多的功能，例如用户间的私信功能、实时通知功能等；在性能优化方面，可以尝试引入缓存机制、分布式存储等技术，提高系统的性能和可扩展性。

总之，本文通过设计和实现一个基于Golang和Vue.js的博客论坛系统，充分展示了这两种技术的优势，为Web开发者提供了一个实践的范例。我们相信，随着Golang和Vue.js技术的不断发展和完善，它们将在Web开发领域发挥越来越重要的作用。

参考文献

1. 李洋,刘婷.基于MySQL的家电回收管理系统的数据库设计[J].科技与创新,2023(03):141-143+146
2. 王福兴,周明辉.基于Golang+Gin的技术运维系统设计与实现[J].现代电视技术,2022(10):134-137
3. 苏佳旭,白燕,温晓东.基于Web前端与MySQL数据库的自然灾害应急信息共享平台设计与实现[J].电脑知识与技术,2023,19(05):74-77
4. 张辉,李鹏.基于Golang的跨平台蜜罐框架系统的设计与实现[J].现代计算机,2022,28(21):87-91
5. 顾雅枫,葛静微.基于MVC的实验室低值易耗品管理系统的开发与实现[J].现代信息科技,2023,7(06)
6. 王晓峰.Golang语言实现的流水线模型[J].电子技术与软件工程,2020(01):53-54
7. 齐洋,原变青,刘颖,等.基于Gin和Vue.js的作业管理系统的设计[J].信息技术与信息化,2022(10):103-105+110
8. 徐健.基于Go和Vue.js的体育选课系统的设计与实现[J].电脑知识与技术,2022(08)
9. 卢云霞.浅谈个人博客网站的设计与实现[J].内蒙古科技与经济,2021(17):78-79+81
10. 丁岚,范开勇,王英明.基于Golang的网络爬虫系统设计与实现 [J] 电脑编程技巧与维护.2019,第006期
11. 王铮清,刘壮峰.基于Go语言的内容管理系统的设计与实现[J].电脑知识与技术,2022,18(24)
12. 宋云奎,吴文鹏,赵磊,等.基于Redis的分布式数据存储方法[J].计算机产品与流通,2020(08)
13. 肖睿.基于Gin框架的营销活动公共类库的设计与应用[D].武汉:华中科技大学，2019
14. 王雄.Golang或将统治人工智能下一个10年[J].计算机与网络,2020(20)
15. Wu Daiwen. The Application and Management System of Scientific Research Projects Based on PHP and MySQL[J].Journal of Interconnection Networks,2022,22(Supp02)

# 致谢

经过不懈的努力学习和不断摸索，在设计开发方面，老师和学长学姐的指导帮助我完成了我的毕业设计“基于Java的双学位招生管系统的设计”的开发。在开题阶段我对课题的国内外发展情况和现状及发展趋势有了更多深度的了解，对用户的需求和功能的需求分析，使我更加了解双学位系统的基本功能，在系统设计开发过程中，我先后学习了各方面的知识，例如Java的基础语法和学习HTML，实践是检验理论的唯一标准，开发这个系统使我更好的将学习的理论知识应用到实践中去，这对我今后的工作和学习会有很大的帮助。

感谢学校和学院的培养，给我们提供了非常好的学习环境和教学设施，让我们不受环境条件的限制，在学习的海洋任意翱翔。在课堂上任课老师们的认真负责讲授专业知识，使我能够很好的掌握和运用知识，才使我的毕业论文顺利完成；感谢我的家人，在我低沉缺乏信心时，给我鼓励，支持，为我打气，让我重拾信心，相信自己，在我的求学路上，父母作为我最有力的后盾，让我没有后顾之忧，推动着我向上学习，引导着我成长，教我做人的道理，虽然现在还没有对社会做出贡献，但也坚持努力的帮助身边的人。最后，感谢我的室友，在我的学习和生活上帮助我许多，在我课堂上不会的，难以理解的问题，课下会主动的给我讲清楚，帮助我理解，在生活上也非常友好，非常融洽，我们守望相助，感谢我们大学四年这一段相遇的日子。最后发自内心的祝愿我和我的同学，都能够顺利毕业，在今后的日子里，在面对挫折和困难的时候能够迎难而上，积极主动的努力生活，认真严谨的探索学习，成为对社会有益的人。

大学的四年时光过得很快，四年的学习，不仅让我学习到了专业的知识，还使我具备了一个比较完整的逻辑思维，做任何事情都要有理有据，眼界也比以前更加开阔，最重要的是结识了一群朝气蓬勃、志同道合的同学和传道受业解惑的老师，这将是我人生中最大的财富。时光白驹过隙，但仍历历在目。衷心希望我在大学期间遇到的所有的人们，都能够拥有光明的未来，做最好的自己。