《Java 企业级应用》期末考核报告

2023-2024 学年第 2 学期（CST31215）

<table>
<tr><td colspan="4">《Java 企业级应用》期末考核报告</td></tr>
<tr><td>名称</td><td colspan="3">大型企业级应用软件系统的设计</td></tr>
<tr><td>类型</td><td colspan="3">☐验证性　☐设计性　☑综合性</td></tr>
<tr><td>学号</td><td>2021xxxx</td><td>姓名</td><td></td></tr>
<tr><td>设计内容</td><td colspan="3">利用 Java 企业级应用技术设计一个大型企业级软件系统，设计内容要求如下：
（1）一世界一流大学，在校生 10 万余人，每学期有三天时间集中开展下一学期的选课工作；现有系统在使用过程中经常发生卡死宕机现象，无法满足业务需求；计划开发一个新系统，要求做到满足 10 万学生同时选课。
（2）进行文献调研，完成核心业务需求分析；
（3）进行文献调研，完成大型企业级应用的技术现状分析；
（4）完成软件系统的设计，包括：技术方案选型；业务功能设计，软件架构设计，模块设计，界面设计，接口设计，数据库设计，安全设计，核心业务流程设计，类的设计(提供类图，设计模式的使用）等。
（5）只需提供设计方案，不需要提供实现代码。</td></tr>
<tr><td>报告要求</td><td colspan="3">设计报告包括：
（1）项目背景介绍
（2）大型企业级应用的技术现状分析或综述。
（3）需求分析，总体设计，详细设计等
（4）总结和创新点。
（5）报告正文要求格式规范。请参考重庆大学毕业设计论文格式规范（一级标题，二级标题，小标题，段落格式，字体大小，行距，参考文献格式。）。请在文件名中包含自己的学号姓名。</td></tr>
</table>

任务时间	2024 年 4 月 26 日至 2024 年 5 月 26 日

评分表：			
序号	评分项	分值	得分
1	需求分析	20 分	
2	系统设计	40 分	
3	创新性	10 分	
4	技术综述	20 分	
5	文档规范	10 分	
评分人：杨瑞龙		考核报告总得分：	

课程项目评分标准（总分 100 分）

序号	评分项目	完成情况	得分
1	需求分析（20）	分析合理、充分	**20 分**
		分析比较合理、充分	15 分
		分析有缺陷	10 分
		分析有大缺陷	5 分
2	系统设计（40）	设计合理、充分	**40 分**
		设计比较合理、充分	30 分
		设计有缺陷	20 分
		设计有大缺陷	10 分
3	创新性（10）	具有较强创新性	**10 分**
		有一定创新性	8 分
		创新性一般	6 分
		没有创新性	0 分
4	技术综述（20）	技术综述完整	**20 分**
		技术综述比较完整	15 分
		技术综述有大缺陷	10 分
		没有技术综述	0 分
5	文档规范（10）	格式规范、逻辑清楚、论述完整	**10 分**
		格式比较规范、逻辑清楚、论述完整	8 分
		格式比较规范、逻辑和论述有欠缺	6 分
		格式混乱、逻辑和论述不清	0 分

【本页开始为报告正文，要求排版格式美观规范，请参考重庆大学毕业论文规范格式，此括号之后的部分为报告正文。此页内容皆可以删除。】

设计报告的主要提纲和内容要求（下面给出的不是报告的格式，是主要提纲。可以在此基础上进行细化。）：

1. 项目背景与研究现状

项目背景，经文献调研后，分析国内外研究现状，相关技术综述。

2. 需求分析

经过文献调研后，形成的核心业务需求。

3. 系统总体设计

根据首页设计内容要求合理进行总体设计，合理分配总体设计和详细设计的内容。

4. 系统详细设计

根据首页设计内容要求合理进行详细设计，合理分配总体设计和详细设计的内容。

5. 总结与创新点

总结设计成果，提炼创新点。

6. 参考文献

不少于 10 篇中英文参考文献，并且在报告正文中按照学术规范予以正确的标注。不得列入正文中没有引用的参考文献。

7. 自评分

给出分数并说明理由。

目录

1 项目背景介绍

建设世界一流大学和一流学科（以下简称“双一流”建设）是党中央、国务院作出的重大战略部署。“双一流”建设实施以来，各项工作有力推进，改革发展成效明显，推动高等教育强国建设迈上新的历史起点。为着力解决“双一流”建设中仍然存在的高层次创新人才供给能力不足、服务国家战略需求不够精准、资源配置亟待优化等问题。 为响应党的号召，建设世界一流大学，亟需提升学校软件服务水平，开发符合新质生产力的大型学生选课系统。

1.1 学校概况

该大学是世界一流大学，具有悠久的历史和卓越的学术声誉。在国家的大力支持下，学校不断发展壮大，教学科研能力显著提升，已成为国际知名的高等学府。随着学校的发展，入学人数逐年增加，目前在校生人数已上升到 10 万余人 。

1.2 现有系统问题

每学期有三天时间集中开展下一学期的选课工作。现有选课系统在高并发情况下经常发生卡死和宕机现象，导致选课工作效率低下，无法满足庞大的业务需求。生产力没有适应生产关系的迅猛发展。

1.3 项目目标

利用 Java 企业级应用技术设计一个大型企业级软件系统，计划开发一个新选课系统，要求能够支持 20 万学生同时选课，确保系统的稳定性和高效性，提升选课体验。

1.4 大型企业级应用的技术现状分析

1.4.1 文献调研

关于”大型网站”的解释，在学术上并没有精确的定义。但是作为一个大型网站，数据量和访问量二者是缺一不可的，此外，除了海量数据和高并发的访问量，网站本身的业务逻辑和系统的复杂度也是一个很重要的考量方面。大型网页要解决解决静态资源访问、分布式缓存解决网站部分数据的读取效率、搜索引擎解决全站检索性能、并发架构缩短多任务执行的时间等待、消息中间件解耦应用并有效应对大并发下数据库写、数据访问中间层或分布式数据库解决数据写压力与存储 6 个问题。企业应用架构模式包括分层、分割、分布式、集群、缓存、异步、冗余、自动化、安全。

1.4.2 大型企业级应用的技术现状分析

下面从上述的 6 个问题，综述大型企业级应用的技术现状。

（1）解决静态资源访问问题

对于静态资源访问的优化，在应用层面，一般采用动静分离的模式，即将网站静态资源（html、js、css、jpeg、gif 等文件）与后台应用分开部署，采用 HTTP Server 来提高用户访问静态代码的速度，降低后台 Application Server 访问的压力。对于静态资源的访问，在不做特殊调优的情况下，HTTP Server 和 Application Server 在处理静态资源文件上的差别较大，如 Tomcat 一般支持并发数百的级别，而 Nginx 在静态文件访问方面支持并发可轻松达数万。

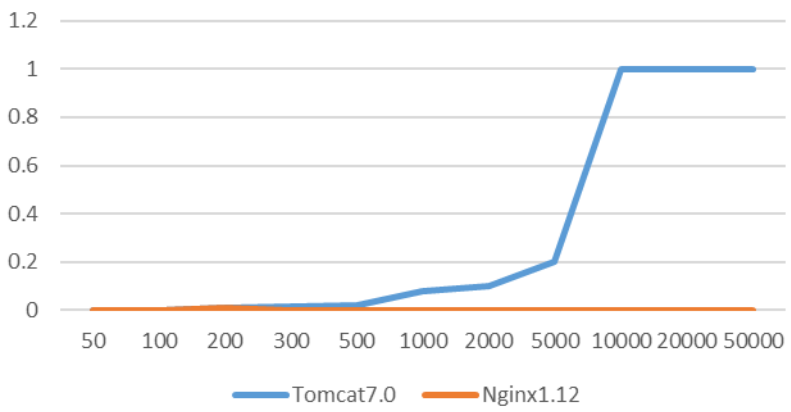

图 1 Tomcat 和 Nginx 静态文件请求错误率对比

同时，网络及系统整体部署架构层面，可通过利用 CDN（Content Delivery Network）技术来加速静态内容访问。CDN 其实质是一种网络缓存技术，它依靠部署在各地的分支节点服务器，通过中心平台的负载均衡、内容分发、调度等功能模块，使用户就近获取所需内容，从而降低网络拥塞，提高访问响应速度及命中率。网站引入 CDN 以后，通过浏览器访问网站的过程也将发生一些变化。

图 2 未引入 CDN 前浏览器访问网站的流程

引入 CDN 后浏览器访问网站的流程

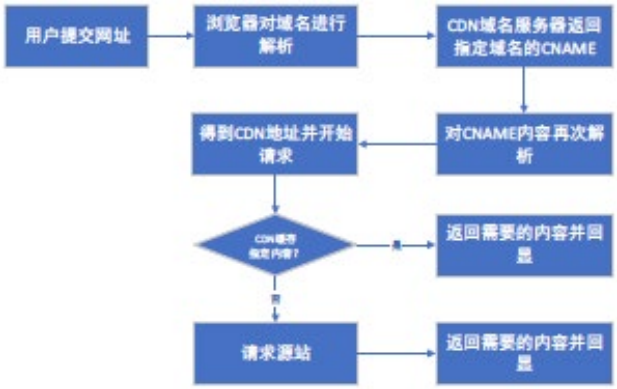

图 3 引入 CDN 后浏览器访问网站的流程

（2）分布式缓存解决网站部分数据的读取效率问题

对于网站变化频次低、读取频次高的动态数据，通常通过分布式缓存系统对数据进行缓存处理，以此加速数据读取速度，减少查询请求直接冲击数据库。目前主流分布式缓存系统有 Redis、Memcached 等。分布式缓存系统的部署一般与本地应用隔离[5]，数据存放在服务器内存。多个应用可直接共享缓存系统数据，实际应用场景中通常根据业务周期设置缓存数据的过期策略，达到缓存数据最佳使用效果。

（3） 搜索引擎解决全站检索性能问题

大型网站中全站搜索的功能，分布式缓存系统往往从技术上无法有效应对。然而，搜索引擎则可以作为读库天然的满足海量数据的检索。在实际生产环境

中，通过正确创建索引、适时的更新索引、构建一致的索引库，完全可以使应用在海量数据中快速捞取目的数据，而非通过数据库进行整库 like 匹配查询。常用的搜索引擎产品如 Elasticsearch、Solr 等。

（4）并发架构缩短多任务执行的时间等待

应用技术架构优化的目的是在安全可靠的情况下尽可能的榨取和高效的利用服务器资源。对于多库取数用于聚合处理、多处远程过程调用获取数据并整合处理的场景，使用串行执行的方式必然会降低数据获取效率、延长调用周期[5]，因此，在数据获取来源数量可控的情况下，采用并发架构设计任务执行流程，为各数据调用过程分配不同处理线程，并行的执行 DB 查询或远程过程调用，从一定程度上能够提高服务器资源利用率，缩短响应等待时间。

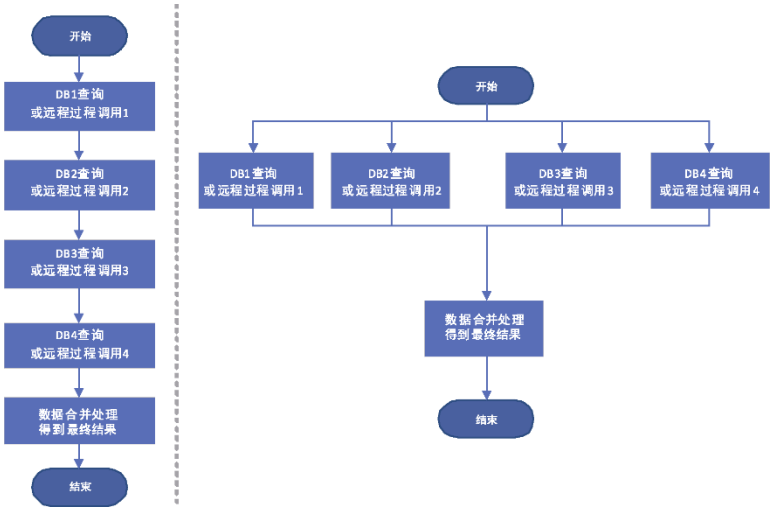

图 4 数据获取串行与并行执行流程的对比

（5）消息中间件解耦应用并有效应对大并发下数据库写的问题

分布式缓存系统、全文搜索引擎从一定程度上缓解了数据库读压力，多线程并发调用获取数据也充分提高了应用服务器的资源利用率。然而，数据库的写压力也是在随着业务量的增大而继续飙升。因此，要解决这个问题，我们开始考虑首先从应用层通过消息中间件对迅猛的数据流进行削峰处理。消息中间件适用于需要可靠数据传送的分布式环境[6]。采用消息机制的系统中，不同的对象之间通过传递消息来触发对方的服务事件，完成相应的操作指令。发送者将消息采用异步方式发送给消息服务器，消息服务器将消息存放在若干队列中，适时将消息异

步转发给接收者。消息中间件产品一般都具有较好的消息堆积能力，主流中间件产品通常能够提供数亿消息的堆积能力。因而，使用消息中间件集群系统能够降低前端业务层数据写请求的峰值，使数据能够按序堆积并投递给目的应用去消费，避免直接冲击应用和数据库，最终实现数据可靠流转，较好的应对高并发场景[7]。

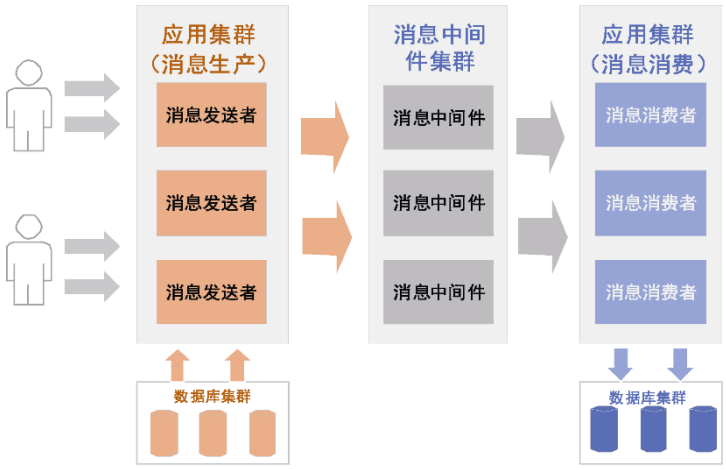

图 5 引入消息中间件后的数据流示意图

（6） 数据访问中间层或分布式数据库解决数据写压力与存储问题

在业务量持续增长的情况下，采用一系列的前端请求削峰与数据读写分离，虽然对数据库写压力能够较好的缓解，但依然对数据库系统自身的写性能没有实质性的提高。因此，采用数据库拆分，在此基础上搭建数据库访问中间层[8-9]或直接为应用系统引入分布式数据库产品的架构思路，成为解决性能问题所考虑的重点。数据库拆分包括水平拆分和垂直拆分，水平拆分即数据库在业务层横向的分库分表，垂直拆分即对数据库按照业务进行拆分。水平拆分解决了单表中数据量增长出现的压力，但是无法解决表与表之间的 io 争夺。垂直拆分虽然解决了表与表之间的 io 竞争，但是无法应对单表中数据量增长出现的压力。因此，在常规的架构设计中，往往采取水平拆分和垂直拆分相结合的方式，根据实体单元的业务特性和实际入库场景来进行合理拆解。数据访问中间层解决了数据库分库分表对数据访问所带来一些共性问题，使其便于研发的统一落地和数据的高效访问，而且为不同来源的数据提供一致性的访问方式和接口处理。目前，常用的数据访

问中间层产品有 Cobar, MyCAT, TDDL, DRDS 等。伴随着数据访问中间层技术的不断发展，业界也在不断涌现出一些基于开源数据库产品研发构建的分布式数据库产品。此类产品对于海量数据的高性能读取和高并发写操作进行了针对性优化，通常能够实现数千亿条记录、数百 TB 数据上的跨行跨表事务、T 级或 P 级数据的分布式存储等，如 OceanBase、TDSQL。这些产品从一定程度上解决了大型网站数据库读写压力与可靠性存储，有效支撑了各类复杂业务场景带来的数据架构问题。

2 业务需求分析

业务需求通过项目视图与项目文档转化为用户需求。用户需求通过使用实例文档进一步细化为具体的功能需求和质量属性。质量属性和其它非功能需求、约束条件共同构成了软件需求规格说明。下面分析选课系统的功能需求和非功能需求。

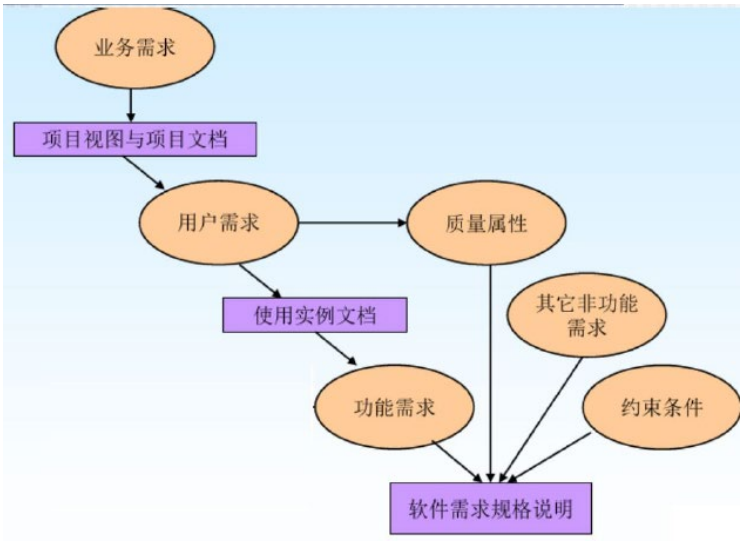

图 6 业务需求分析考虑的要素

2.1 功能需求分析

功能需求是基于用户需求的具体化，详细描述了系统需要具备的功能。功能需求包括系统各个模块的具体功能描述，如账号注册、学生账号、教师账号、教务账号等。下面详细列出系统各个功能模块的具体需求。

2.1.1 账号管理功能

① 账号注册

功能描述：提供新用户注册账号的功能。

输入要求：用户需提供用户名、密码、邮箱、手机号等基本信息。

验证机制：用户名唯一性检查，密码强度检查，邮箱和手机号格式验证。

成功反馈：注册成功后，向用户邮箱发送验证邮件以确认账号。

错误处理：若用户名已存在或输入信息不符合要求，提示相应错误信息。

② 账号登录

功能描述：用户通过已注册的账号进行登录。

输入要求：用户名和密码。

验证机制：验证用户名和密码匹配是否正确。

成功反馈：登录成功后，跳转至用户对应的主页（学生主页、教师主页或教务管理主页）。

错误处理：用户名不存在或密码错误时，提示相应错误信息。

③ 账号注销

功能描述：用户可以选择注销其账号。

输入要求：用户需确认注销操作。

验证机制：二次确认以防止误操作。

成功反馈：注销成功后，清除用户相关数据，并反馈注销成功信息。

错误处理：若用户在注销过程中遇到问题，提示相应错误信息，并提供客服支持信息。

2.1.2 教师账号功能

① 学生成绩管理

功能描述：教师可以录入、查看和修改学生的成绩。

输入要求：选择班级和课程，输入学生成绩。

验证机制：检查输入成绩的格式和范围，确保数据有效。

成功反馈：成绩录入或修改成功后，显示操作成功信息。

错误处理：输入数据不符合要求时，提示相应错误信息，并允许教师重新输入。

2.1.3 学生账号功能

① 查看成绩

功能描述：学生可以查看自己的各科成绩。

输入要求：学生需选择学期或课程进行查询。

验证机制：确保学生只能查看自己的成绩。

成功反馈：显示学生各科成绩的详细信息。

错误处理：若查询过程中出现问题，提示相应错误信息。

② 选择课程

功能描述：学生可以根据开课情况选择课程进行学习。

输入要求：学生选择学期和课程进行选课。

验证机制：检查选课人数限制和选课条件（如前置课程要求）。

成功反馈：选课成功后，显示选课结果并更新课程列表。

错误处理：若选课不成功（如选课人数已满或不符合条件），提示相应错误信息。

2.1.4 教务管理功能

① 班级管理

功能描述：教务人员可以添加、删除和修改班级信息。

输入要求：提供班级名称、年级、专业等信息。

验证机制：检查班级名称的唯一性和输入信息的完整性。

成功反馈：操作成功后，更新班级信息并显示操作结果。

错误处理：若输入信息不符合要求或操作失败，提示相应错误信息。

② 课程管理

功能描述：教务人员可以添加、删除和修改课程信息。

输入要求：提供课程名称、课程编号、课程描述等信息。

验证机制：检查课程编号的唯一性和输入信息的完整性。

成功反馈：操作成功后，更新课程信息并显示操作结果。

错误处理：若输入信息不符合要求或操作失败，提示相应错误信息。

③ 学生选课管理

功能描述：教务人员可以管理学生的选课情况，进行选课审批。

输入要求：查看学生的选课申请，进行审批操作。

验证机制：检查选课条件和选课人数限制。

成功反馈：审批成功后，更新学生选课状态并通知学生。

错误处理：若审批过程中出现问题，提示相应错误信息。

2.2 非功能需求分析

2.2.1 性能要求

① 系统需同时处理 10 万学生的选课请求：确保系统能够在高并发情况下保持稳定的性能和快速响应。

② 资源分配与隔离：如果将选课系统与现有的网站应用部署在一起，必须进行资源分配和隔离，以避免影响现有业务的正常运行。一旦资源争夺导致瓶颈，可能会导致整个网站瘫痪。

③ 防止频繁刷新带来的负载：选课开始前，学生通常会通过频繁刷新页面以确保不遗漏选课机会。一般的网站架构下，这些请求会对应用服务器和数据库造成极大负载压力。因此，需要采取措施，如使用缓存和队列来减轻直接访问数据库和应用服务器的压力。

2.2.2 可靠性要求

① 防止高峰期卡死或宕机：系统应在高并发情况下保持高稳定性，防止卡死或宕机。采用负载均衡、分布式架构和故障转移机制，以保障系统的高可用性。

② 订单提交检查：在用户提交选课请求时，系统应实时检查是否已有订单提交，防止重复提交和数据冲突。

③ 控制下单页面入口：为减轻服务器负载压力，可以通过限流机制控制进入下单页面的用户数量。仅允许部分用户进入下单页面，其余用户则进入等待响应页面，避免过多的请求直接冲击服务器。

2.2.3 运行环境要求

系统应支持多种设备和平台，包括 PC、手机和平板电脑，以方便学生使用。

2.2.4 安全性

系统必须保护学生的个人信息和选课数据，防止数据泄露和非法操作。

2.2.5 性能监控和运维

系统应提供性能监控和运维工具，便于管理员实时监控系统状态并及时处理故障。

2.2.6 备份与恢复

系统应具有完善的备份与恢复机制，确保数据在突发情况下的安全和完整性。

2.2.7 功能需求图

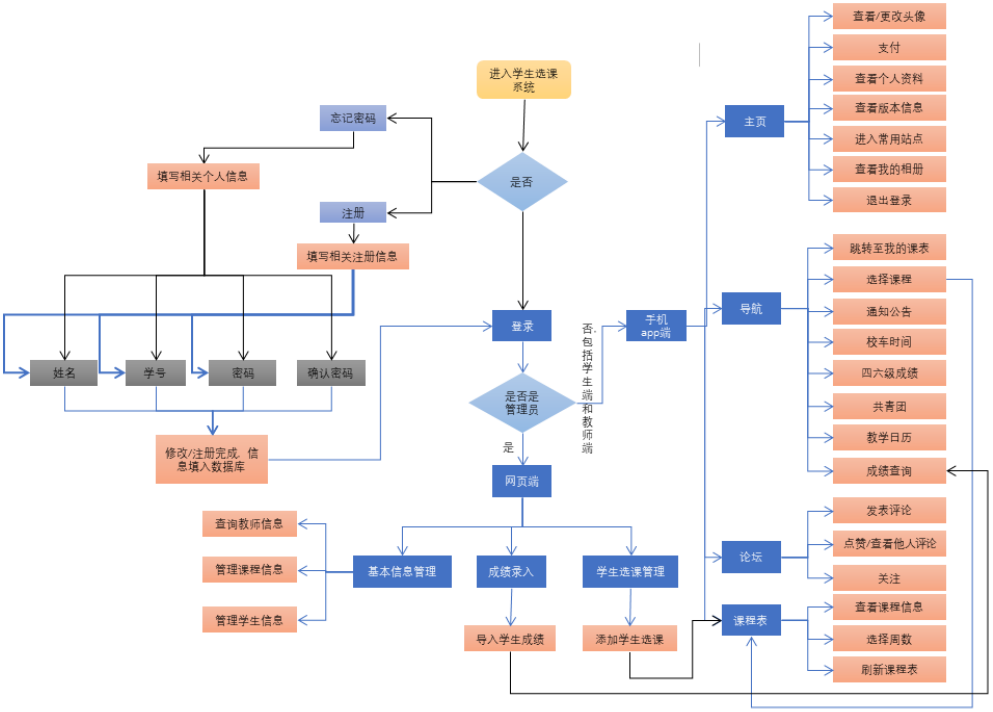

图 7 功能需求图

3 系统总体设计

3.1 系统概述

选课系统主要用于高校学生选课操作，确保用户能快速刷新课程页面并抢先进入下单页面，提供用户友好且高效的选课体验。除此之外，系统还涵盖用户登录/注册、教务管理、教师录入成绩等业务功能。

3.2 技术架构

服务器：云服务器

数据库：分布式数据库（如 MySQL Cluster）

前端技术：Vue.js 框架

后端技术：Spring Boot

反向代理及本地缓存：Nginx、CDN

3.3 分层架构

展示层（前端）：负责用户界面的展示和交互。

业务逻辑层（后端）：处理应用程序的业务逻辑。

数据访问层：负责与数据库的交互

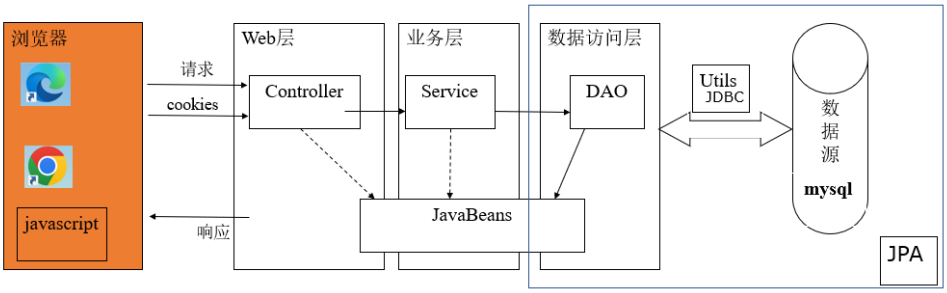

图 8 MCV 分层结构图

3.4 微服务架构

将不同功能模块独立部署和管理，提高系统的灵活性和可扩展性。包括（1）用户管理服务 （2）认证授权服务（3）选课服务（4）课程管理服务（5）查询服务

3.5 模块设计

模块设计包括三个模块：用户模块、选课模块、查询模块

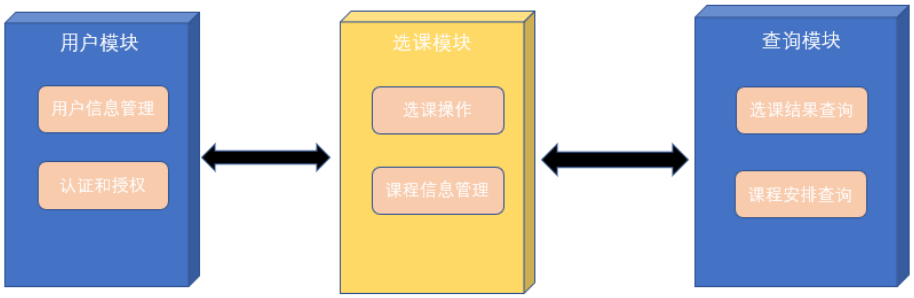

图 9 模块设计图

3.5.1 用户模块

用户信息管理：用户的基本信息维护。

认证和授权：基于 OAuth2.0、JWT 的用户身份验证和权限管理。

3.5.2 选课模块

选课操作：实现选课、退课等功能。

课程信息管理：课程的添加、修改和删除。

3.5.3 查询模块

选课结果查询：用户可以查询自己的选课结果和课程安排。

3.6 界面设计

用户友好的界面布局，响应式设计以适应不同设备，并提供原型设计。

3.7 接口设计

采用 RESTful API 设计，数据传输格式为 JSON，包括 GET,POST,PUT,DELETE 四种请求类型。

3.8 数据库设计

数据库读写分离，分布式数据库设计。

3.9 安全设计

安全设计包括：用户身份验证（采用 OAuth2.0）；数据加密（包括 HTTPS、数据库加密）；日志记录与安全审计（日志记录，安全审计）；安全应急响应等。

3.10 核心业务流程设计

单点登录流程：登录页面->用户登入->生成 JWT 令牌->存储到数据库->储存 JWT 到 cookie

单点登录流程：选课页面->检查 coolie 的 token->登录页面

选课流程：用户登录 -> 选课操作 -> 选课确认 -> 结果反馈

3.11 设计模式

（1）桥接模式 （2） 工厂模式 （3）装饰器模式（4） 单例模式 （5） 依赖注入等

4 详细设计

4.1 技术架构

本项目的技术架构设计旨在构建一个高性能、高可用、可扩展的系统。整体架构采用前后端分离的模式，前端使用 Vue.js 框架进行开发，后端采用 Spring Boot 构建 RESTful API，数据存储使用分布式数据库（如 MySQL Cluster），并通过 Nginx 进行反向代理及本地缓存处理。以下是详细的技术架构说明：

4.1.1 服务器

教务系统分为常驻系统和选课系统两个子系统。常驻系统包括学生查看课程，教务管理课程，教师录入学生成绩等基本功能，具有压力小、持久化使用的特点。选课系统负责学生选课，具有高并发，压力大，时间短的特点（一学期选课只有 3 天时间）。

因此常驻系统的部署选用学校本地服务器，选课系统使用云服务器。

学校本地服务器足以应对日常的低频访问，且成本低。

云服务器（如阿里云、腾讯云或 AWS）具有高可用性、弹性扩展以及按需付费等优势，可以根据业务需求灵活调整资源配置。选课期间，进行部署，按需付费，降低平时对服务器设备的开销，且拓展性强。

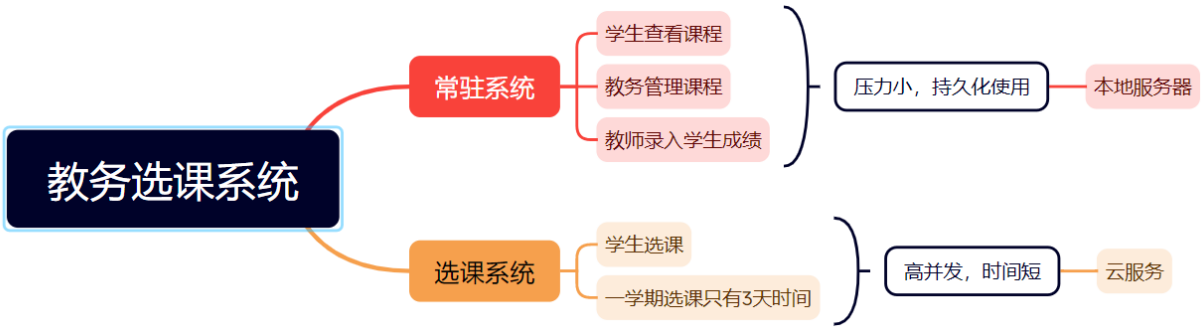

图 10 教务选课系统

云服务器的选择主要考虑以下几点：计算能力：根据业务需求选择合适的 CPU 和内存配置，确保服务器具有足够的计算能力。存储：采用 SSD 云硬盘，提高数据读写速度，保障数据库的高性能。网络：选择高速、稳定的网络环境，保证系统的网络吞吐量和低延迟。

4.1.2 数据库

为了满足系统高可用性和高扩展性的要求，数据库选用分布式数据库 MySQL Cluster。MySQL Cluster 通过数据分片和冗余备份等机制，实现了数据的分布式存储和管理，具备以下优点：

·高可用性：通过多节点冗余备份，确保数据库在节点故障时仍能正常运行。

·高扩展性：支持水平扩展，可以根据业务增长灵活增加数据库节点，提升系统的处理能力。

·高性能：分布式架构和内存存储机制，使得 MySQL Cluster 在处理大规模数据时具有高效的读写性能。

4.1.3 前端技术

前端部分采用 Vue.js 框架进行开发，Vue.js 是一款轻量、高效的渐进式 JavaScript 框架，适用于构建复杂的单页面应用（SPA）。选择 Vue.js 的主要原因包括：

·组件化开发：通过组件化开发，提升代码复用性和可维护性。

·响应式数据绑定：实现视图和数据的双向绑定，简化前端开发流程。

·丰富的生态系统：拥有丰富的插件和工具支持，如 Vue Router、Vuex 等，方便实现复杂功能。

此外在选课开始前，要提前将选课列表网页做成静态网页部署到 javascript 服务器上，避免选课时查询，与数据库进行连接额外查询。

4.1.4 后端技术

后端部分使用 Spring Boot 框架进行开发，Spring Boot 是基于 Spring 框架的快速开发框架，简化了 Spring 应用的配置和部署过程。Spring Boot 具有如下优势：

快速开发：提供开箱即用的配置，简化项目的初始化和配置过程，加快开发速度。

微服务支持：方便构建微服务架构，支持 Spring Cloud 等微服务组件，适合大规模系统的开发。

良好的生态系统：与 Spring 全家桶无缝集成，拥有丰富的第三方库和插件支持，方便扩展功能。

4.1.5 反向代理及本地缓存

系统的反向代理和本地缓存功能由 Nginx 和 CDN（内容分发网络）实现[1]：

Nginx：作为反向代理服务器，Nginx 负责处理客户端的请求，将请求转发到后端服务器，并进行负载均衡。此外，Nginx 还可以缓存静态资源，减轻后端服务器的压力，提高响应速度。

CDN：通过 CDN 加速，缓存和分发静态资源（如图片、CSS、JavaScript 等）到全球各地的节点，缩短用户访问静态资源的时间，提升用户体验。

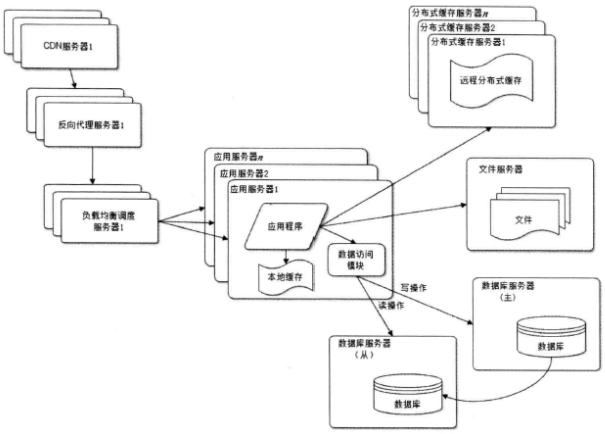

图 11 反向代理及本地缓存原理

4.1.6 技术架构图

综合以上技术组件，本项目的技术架构图如下：

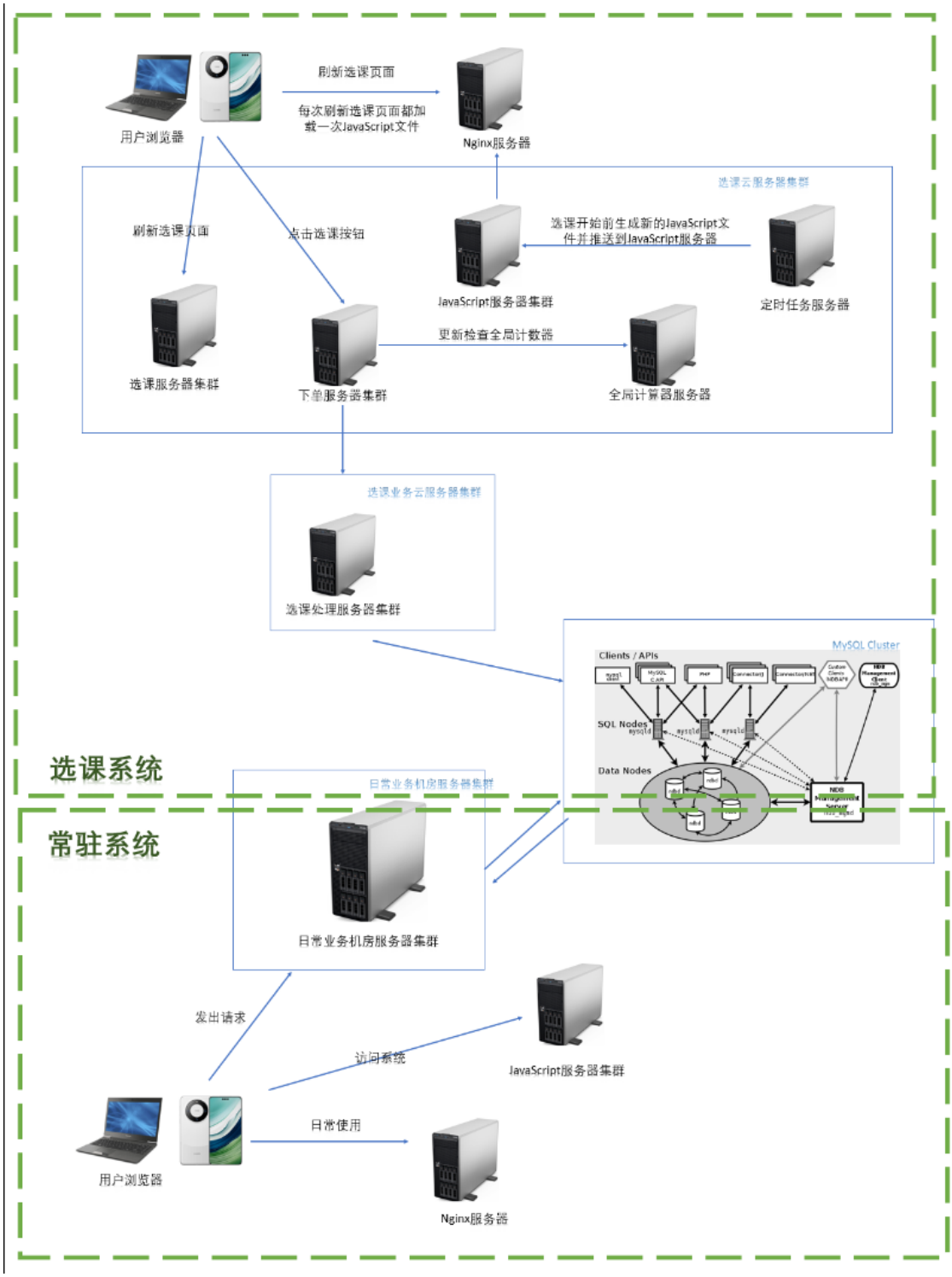

图 12 技术架构图

这种架构设计能够充分利用各技术组件的优势，实现系统的高性能、高可用和可扩展性。

4.2 分层架构

为了实现系统的高可维护性和可扩展性，本项目采用分层架构设计。分层架构将系统功能划分为不同的层次，每一层专注于特定的职责，减少层间耦合，提高系统的可维护性和扩展性。具体包括展示层、业务逻辑层和数据访问层。

4.2.1 展示层（前端）

展示层负责用户界面的展示和交互。向用户提供友好、响应迅速的界面，收集用户的输入并展示处理结果。

技术栈如下：

·Vue.js 框架：用于构建动态单页面应用。

·Vue Router：实现前端路由控制。

·Vuex：用于管理应用状态。

主要功能：用户认证与授权界面（登录、注册、密码管理），课程信息展示与操作界面（课程列表、课程详情、选课操作），用户信息管理界面（用户资料查看与修改），响应用户操作，调用后端 API 获取数据并更新界面。

4.2.2 业务逻辑层（后端）

业务逻辑层处理应用程序的业务逻辑，实现各功能模块的业务处理，确保数据处理的正确性和一致性。

技术栈如下：

·Spring Boot 框架：用于快速开发和部署 RESTful API。

·Spring Security：用于实现安全控制和用户认证授权。

·Spring Cloud：用于构建和管理微服务。

主要功能包括用户管理（注册、登录、权限管理），课程管理（课程创建、修改、删除、查询），选课操作（选课、退课、选课记录查询），数据校验和业务规则处理。

4.2.3 数据访问层

数据访问层负责与数据库的交互，提供数据持久化服务，处理数据库的增删改查操作。

技术栈：

·Spring Data JPA：简化数据访问层的开发。

·MySQL Cluster：提供高可用性和高扩展性的分布式数据库解决方案。

主要功能：定义数据访问对象（DAO），实现数据的增删改查操作，管理数据库连接和事务，确保数据操作的原子性和一致性，数据缓存（如使用 Redis）以提高数据访问速度。

4.3 微服务架构

为了提高系统的灵活性和可扩展性，本项目采用微服务架构。将系统功能模块化，按不同业务需求拆分为多个独立的服务，每个服务独立部署和管理。这样不仅提高了系统的灵活性，还能在不同功能模块之间实现松耦合，提高系统的容错性和可维护性。

4.3.1 微服务设计原则

微服务设计原则如下：

·将系统的服务层抽象为一个个服务

·每个服务只负责简单的功能

·服务通常用 RESTfull 等轻量级协议实现

4.3.2 主要微服务

（1）用户管理服务

处理用户注册、登录、权限管理等操作。支持用户信息存储、认证授权、密码管理功能。

（2）认证授权服务

构建独立的认证授权服务，CAS 中心，支持单点登录。

（3）选课服务

处理用户选课、退课等操作。支持选课记录存储、选课冲突检测、选课结果返回功能。

（4）课程管理服务

管理课程信息，包括课程创建、修改、删除、查询等。支持课程信息存储、课程查询、课程更新功能。

（5）查询服务

提供各种查询功能，如用户信息查询、课程信息查询、选课记录查询等。支持处理前端发来的查询请求，返回查询结果。依赖其他服务的数据接口，不直接管理数据库。

4.3.3 微服务架构图

微服务架构图如下

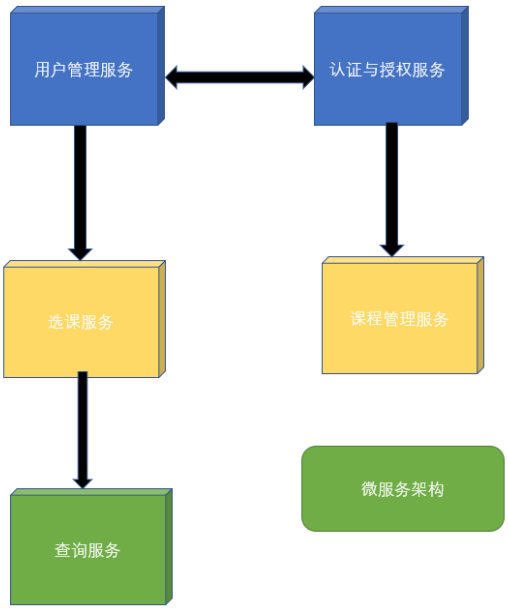

图 13 微服务架构图

每个微服务都有自己的独立数据库和业务逻辑，可以独立部署和扩展。服务之间通过 API 进行通信，实现松耦合的架构设计，提高了系统的灵活性和可维护性。

4.4 模块设计

4.4.1 用户模块

（1）用户信息管理

功能描述：管理用户的基本信息，包括注册、修改个人信息、查看个人信息等。

主要功能：

用户注册：新用户通过注册接口提交信息创建账号。

信息修改：用户可以修改个人资料，如姓名、邮箱、联系方式等。

信息查看：用户可以查看自己的基本信息。

（2）认证和授权

功能描述：基于 OAuth2.0 和 JWT（JSON Web Token）实现用户身份验证和权限管理。

主要功能：

用户登录：用户通过登录接口获取 JWT 令牌。

用户注销：用户可以注销登录状态。

权限管理：根据用户角色和权限，控制用户对不同资源和操作的访问权限。

4.4.2 选课模块

功能描述：实现选课和退课功能，用户可以选择或退选课程。

主要功能：

选课：用户可以选择自己感兴趣的课程。

退课：用户可以退选已选择的课程。

4.4.3 课程管理模块

功能描述：课程的添加、修改和删除。

主要功能：

添加课程：教务可以添加新课程。

修改课程：教务可以修改课程信息。

删除课程：教务可以删除课程。

4.4.4 查询模块

功能描述：用户可以查询自己的选课结果和课程安排。

主要功能：

查询选课结果：用户可以查看自己已经选的课程和课程安排。

查询课程安排：用户可以查看具体课程的时间和地点安排。

接口设计：

GET /api/courses/enrollments：查询用户的选课结果。

GET /api/courses/schedule：查询课程安排。

4.5 界面设计

用户友好的界面布局，响应式设计以适应不同设备。参考重庆大学教务网和软件综合设计的课设，原型设计如下。

4.5.1 PC 端

CAS 认证中心

图 15 CAS 认证中心原型设计

课程管理页面（软件综合设计的课设）

图 16 课程管理页面原型设计

选课页面，为解决高并发问题，这个页面为静态页面（不与数据库建立连接，抢课开始前即可访问）

图 17 选课页面原型设计

具体选课页面，只允许少数人与服务器建立连接后访问的秒杀页面，此时才与数据库建立连接。

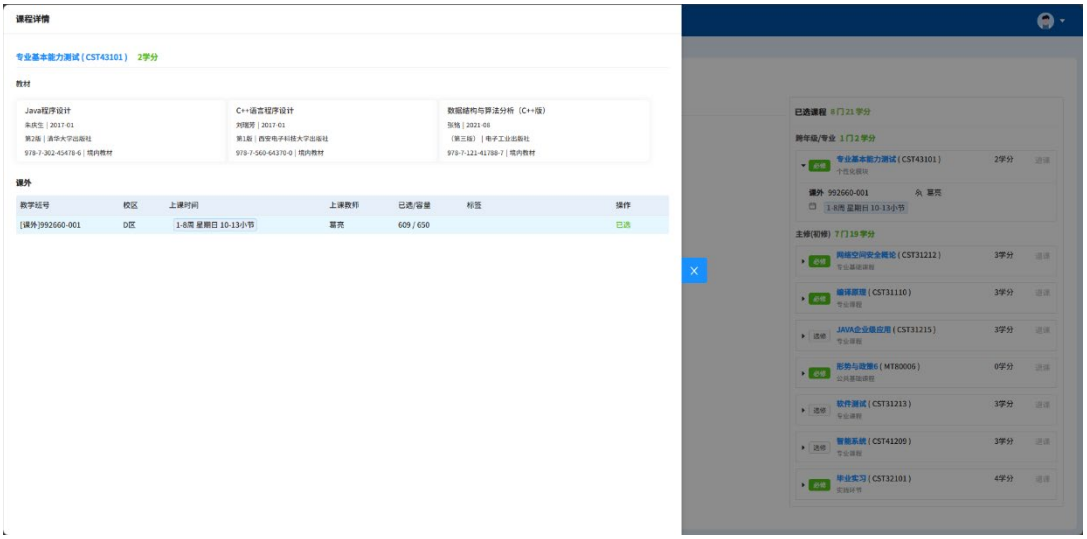

图 18 具体选课页面（秒杀页面）原型设计

4.5.2 手机端

选课页面，为解决高并发问题，这个页面为静态页面（不与数据库建立连接，抢课开始前即可访问），来源软件综合设计课设

图 19 手机端选课页面原型设计

具体选课页面，只允许少数人与服务器建立连接后访问的秒杀页面，此时才与数据库建立连接。

图 20 手机端具体选课页面（秒杀页面）原型设计

4.6 接口设计

RESTful API 设计

表 1 RESTful API 设计

请求类型	接口	功能	传输格式
POST	/api/users/register	用户注册数据	JSON
PUT	/api/users/{userId}	修改用户信息	JSON
GET	/api/users/{userId}	查看用户信息	JSON
POST	/api/auth/login	用户登录，返回 JWT 令牌	JSON
POST	/api/auth/logout	用户注销	JSON
GET	/api/auth/refresh	刷新 JWT 令牌	JSON
POST	/api/courses/enroll	选课操作	JSON

DELETE	/api/courses/unenroll	退课操作	JSON
POST	/api/courses	添加新课程	JSON
PUT	/api/courses/{courseId}	修改课程信息	JSON
DELETE	/api/courses/{courseId}	删除课程	JSON
GET	/api/courses/enrollments	查询用户的选课结果	JSON
GET	/api/courses/schedule	查询课程安排	JSON

4.7 数据库设计

4.7.1 数据库读写分离

应用服务器在写数据的时候，访问主数据库，主数据库通过主从复制机制将数据更新同步到从数据库，这样当应用服务器读数据的时候，就可以通过从数据库获得数据。为了便于应用程序访问读写分离后的数据库，通常在应用服务器端使用专门的数据访问模块，使数据库读写分离对应用透明。

4.7.2 分布式数据库设计

分布式数据库是网站数据库拆分的最后手段，只有在单表数据规模非常庞大的时候才使用。不到不得已时，网站更常用的数据库拆分手段是业务分库，将不同业务的数据库部署在不同的物理服务器上。

4.8 安全设计

4.8.1 用户身份验证

用户身份验证是确保系统只允许授权用户访问的关键措施。本系统采用OAuth 2.0 协议进行用户身份验证。

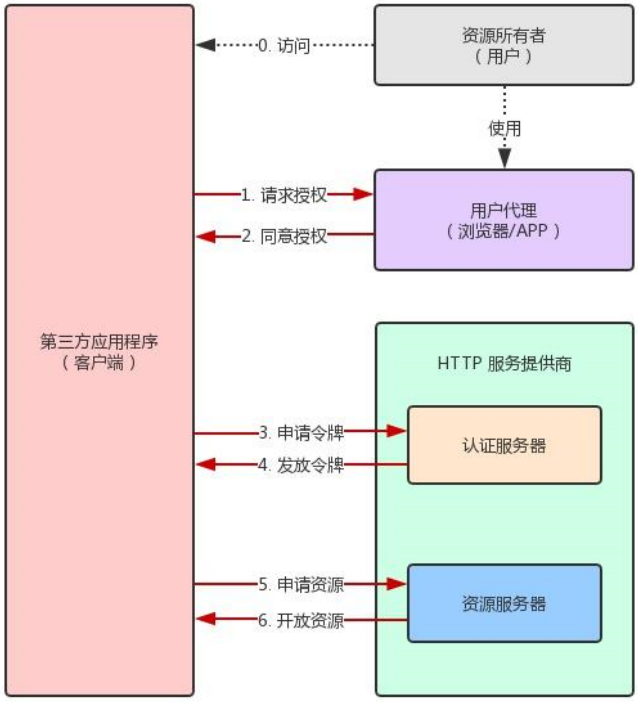

图 21 OAuth 2.0 协议原理

OAuth 2.0 认证流程：用户请求访问->客户端请求授权->用户同意授权->授权服务器发放授权码->客户端交换授权码->授权服务器发放访问令牌->客户端访问资源

此外还通过如下设计保证安全：

用 HTTPS：所有与授权服务器和资源服务器的通信均使用 HTTPS 加密，以防止中间人攻击。

令牌加密：访问令牌应采用 JWT（JSON Web Token）格式，并进行签名和加密，以防止篡改和伪造。

令牌过期时间：访问令牌应设置较短的过期时间，减少令牌被滥用的风险。

刷新令牌：使用刷新令牌机制，确保用户长时间使用系统时无需频繁登录，但也能在令牌失效时重新获取新的访问令牌。

4.8.2 数据加密

为了保护传输和存储中的数据，系统将采用多层次的数据加密措施。

（1）HTTPS 保证传输层安全和 SSL 证书

传输层安全：使用 TLS（Transport Layer Security）协议确保客户端与服务器之间的通信安全，防止数据在传输过程中被窃听或篡改。

SSL 证书：部署有效的 SSL 证书，确保服务器的身份真实性，并加密所有 HTTP 流量。

（2）数据库加密

静态数据加密：数据库中的敏感数据（如用户密码、个人信息）应采用 AES（Advanced Encryption Standard）加密，确保即使数据库被攻破，数据仍然是不可读的。

字段级加密：对于特别敏感的字段，如用户密码，应采用单向哈希算法（如 bcrypt）进行加密存储，确保不可逆。

数据库连接加密：数据库连接应使用 SSL/TLS 加密，确保数据在应用服务器和数据库服务器之间传输时的安全性。

4.8.3 日志记录与安全审计

日志记录和安全审计是监控系统安全状态和检测潜在安全威胁的重要手段。

（1）日志记录

访问日志：记录所有用户的登录、登出、访问资源等操作日志，以备追踪用户行为。

错误日志：记录所有系统错误和异常，以便及时发现和修复漏洞。

安全事件日志：记录所有安全相关事件，如登录失败、多次尝试登录、令牌过期等。

（2）安全审计

定期审计：定期进行安全审计，检查系统配置、访问控制、日志记录等方面的安全性。

自动化工具：使用自动化安全审计工具，及时发现并修复系统漏洞和配置错误。

审计报告：生成详细的审计报告，提供给安全团队和管理层，以便做出必要的安全改进决策。

4.8.4 安全应急响应

为了应对潜在的安全事件，系统应针对不同的软硬件故障指定策略。以下将详细描述针对各种故障类型的应急响应策略。[12]

（1） 事务故障（Transaction Failure）

逻辑错误是指事务由于某些内部条件无法继续正常执行，例如非法输入、找不到数据、溢出或超出资源限制等。应急响应策略：

·输入验证：在事务开始之前进行全面的输入验证，确保数据合法。

·事务回滚：在检测到逻辑错误时，立即回滚事务，确保数据库的一致性。

·错误日志记录：详细记录错误信息，包含发生时间、用户、输入数据和错误类型，以便后续分析和修复。

·用户提示：向用户提供明确的错误提示，指导用户修正输入或进行其他操作。

·自动重试机制：对于某些可恢复的错误，系统可以自动重试事务。

系统错误是指系统进入一种不良状态（如死锁），导致事务无法继续正常执行，但该事务可以在以后重新执行。应急响应策略：

·死锁检测和解除：系统应具备死锁检测机制，并在检测到死锁时采取措施解除，如回滚某些事务。

·错误日志记录：记录系统错误的详细信息，包括涉及的事务、资源状态和错误时间。

·事务重试：在系统错误解除后，自动重新执行被中断的事务。

·系统监控：持续监控系统状态，及时发现和处理潜在的系统错误。

（2） 系统崩溃（System Crash）

系统崩溃是指由于硬件故障、数据库软件或操作系统漏洞，导致易失性存储器内容丢失，并使得事务处理停止，而非易失性存储器仍完好无损。应急响应策略：

·定期备份：定期对数据库进行完整备份，确保在系统崩溃时能够恢复数据。

·日志恢复：使用事务日志进行崩溃恢复，通过重做和撤销操作恢复数据库状态。

·硬件冗余：采用冗余硬件配置，如 RAID 技术，减少单点故障的风险。

·监控报警：设置系统监控和报警机制，及时发现并报告系统崩溃。

·快速恢复：建立快速恢复流程，确保在系统崩溃后能够迅速恢复服务。

（3） 磁盘故障（Disk Failure）

磁盘故障是指在数据传送操作过程中，由于磁头损坏或故障造成磁盘块上的内容丢失。应急响应策略：

·数据冗余：使用 RAID 技术或其他数据冗余技术，确保单个磁盘故障不影响数据完整性。

·备份恢复：定期将数据备份到其他磁盘或三级介质（如 DVD 或磁带），在磁盘故障时可以恢复数据。

·数据镜像：对重要数据进行镜像存储，确保在原始数据丢失时仍有可用副本。

·故障监控：监控磁盘健康状态，及时发现和预防潜在的磁盘故障。

·磁盘热备：配置热备磁盘，确保在磁盘故障时能快速切换和恢复数据。

（4） 其他事故其他事故包括天灾（如地震或火灾）、人祸（如人为破坏机房或数据），间谍或黑客攻击等。应急响应策略：

·灾难恢复计划（DRP）：制定并定期测试全面的灾难恢复计划，涵盖自然灾害和人为破坏。

·数据加密：对敏感数据进行加密存储，防止数据在被窃取或破坏时泄露。

·访问控制：严格控制物理和逻辑访问权限，防止未授权人员访问关键系统和数据。

·安全监控：部署安全监控系统，及时检测和响应安全威胁，如入侵检测系统（IDS）和防火墙。

·定期演练：定期进行应急响应演练，确保团队在实际事故发生时能够迅速有效地响应。

·备份存储：将备份数据存储在异地，确保在本地灾难发生时能够恢复数据。

通过以上多层次、多方位的应急响应策略，系统可以有效应对各种潜在的安全事件，确保在故障发生时能够快速恢复，减少对用户和业务的影响。

4.9 核心业务流程设计

4.9.1 单点登入

访问登录页面时序图如下

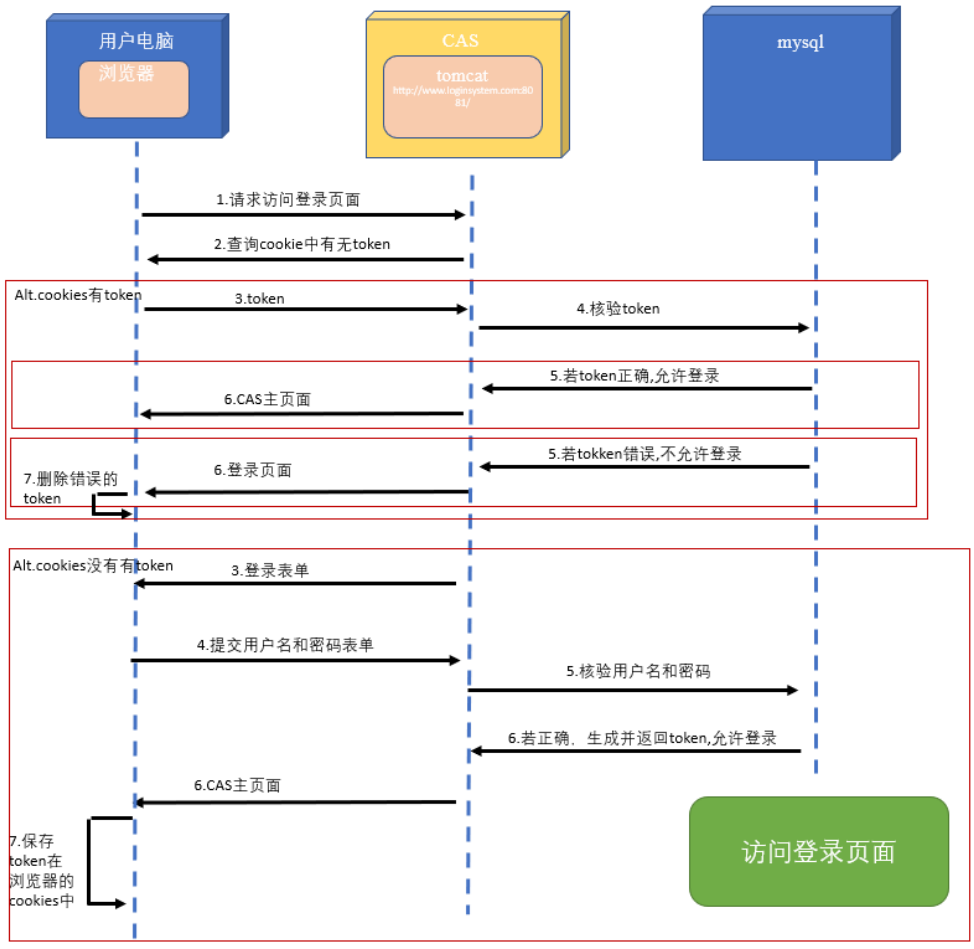

图 22 访问登录页面时序图

4.9.2 访问选课页面

访问选课平台的时序图如下

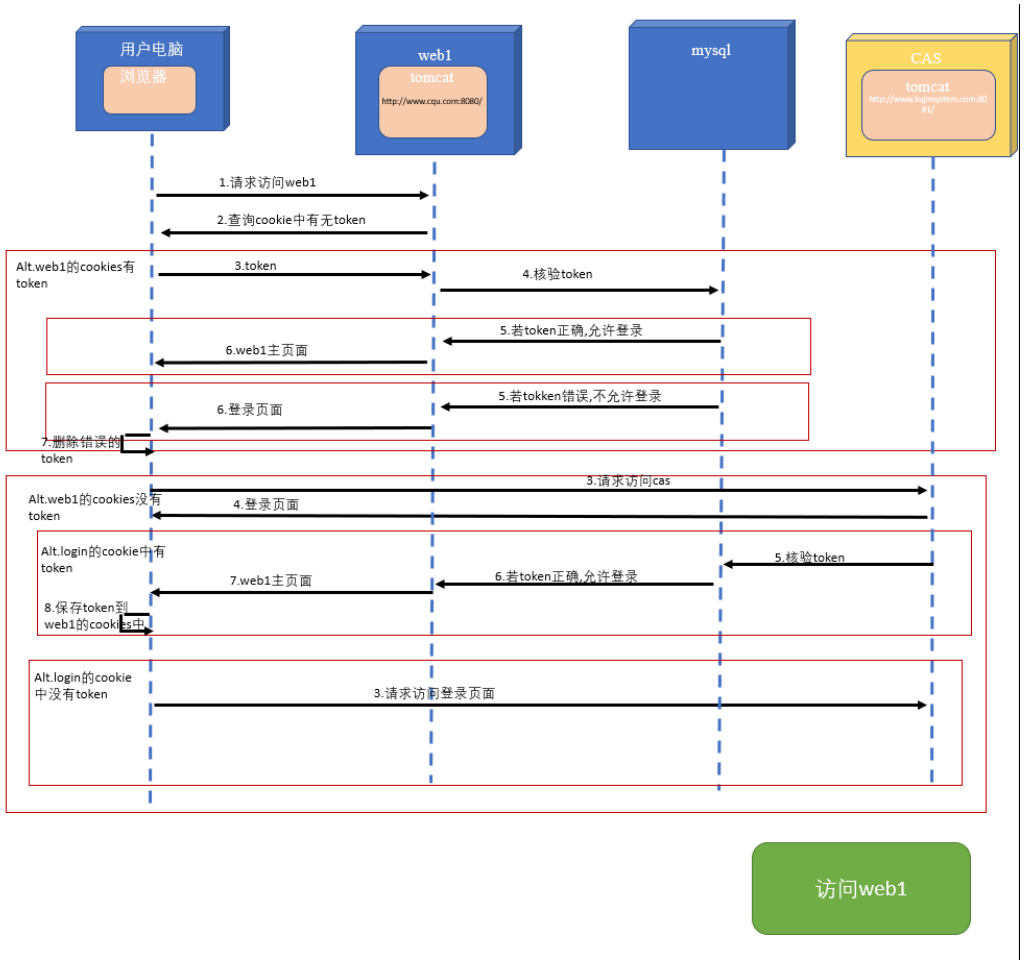

图 23 访问选课平台的时序图

4.9.3 程序主要流程

程序主要流程图如下

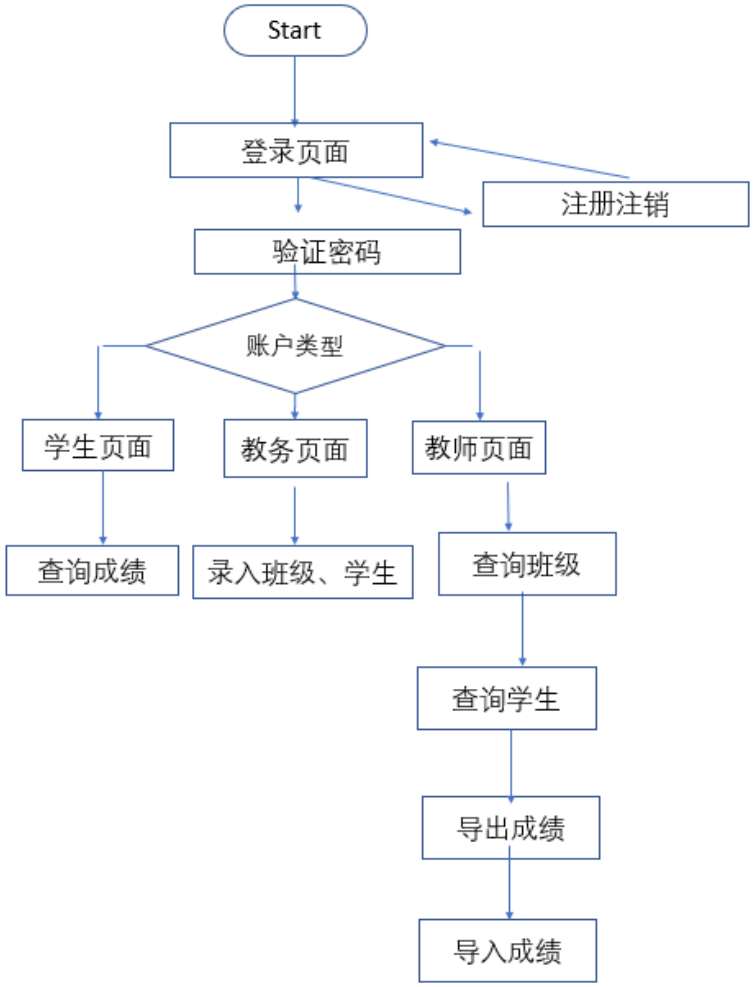

图 24 程序主要流程图

4.9.4 选课访问服务器控制图

选课访问服务器控制图如下

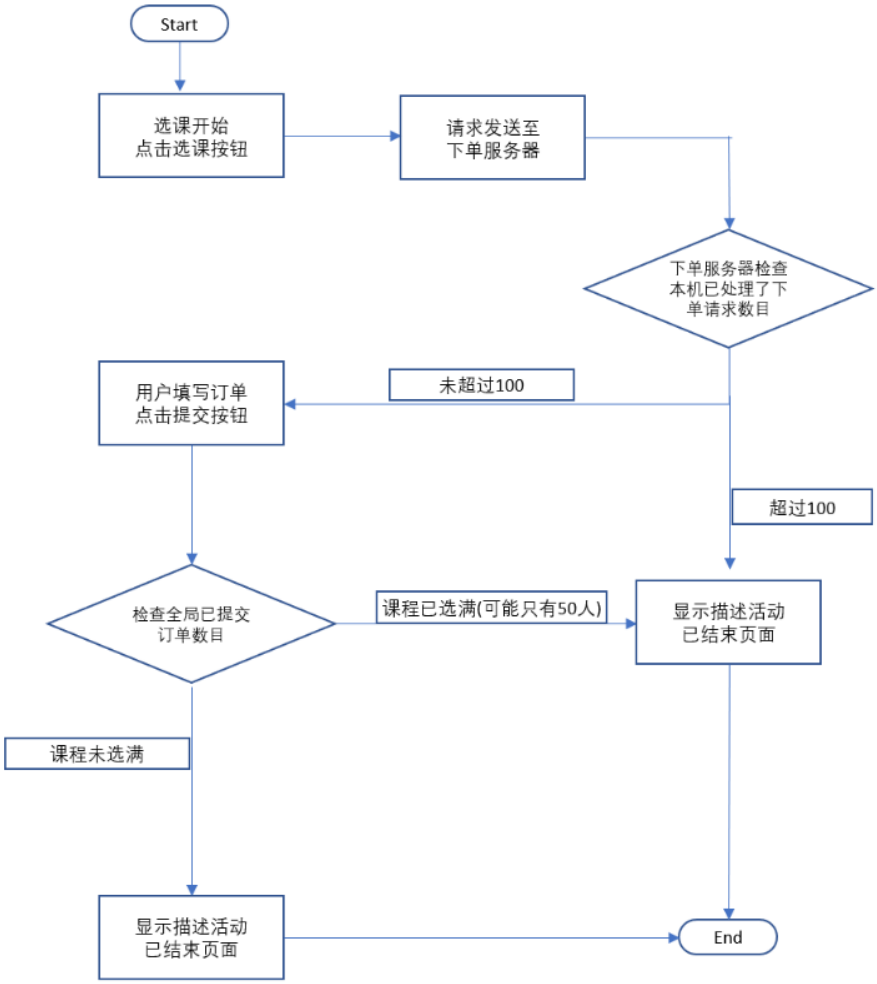

图 25 选课访问服务器控制图

4.10 类的设计

4.10.1 实体类

User 类，包括用户名、密码、账户类型

教师类，包括用户名，教师编号（主键），姓名

学生类,包括学号、平均分、学生 id,用户 id,排名，姓名

课程类，包括课程编号、课程名

班级类，包括班级编号、课程编号、教师编号、学生总数

分数类，也是学生选课的中间类，包括编号、班级 id，学生 id，综合成绩、期末成绩、平时成绩、期中成绩、实验成绩

4.10.2 接口

Jpa 使用接口查询数据库

以 GradeRepository 为例，只定义根据每个变量的值进行查询

（3）@Controller 的类

以 StudentController 为例，findStudentByUserid（）利用 userid 查询学生信息

4.10.3 类关系图

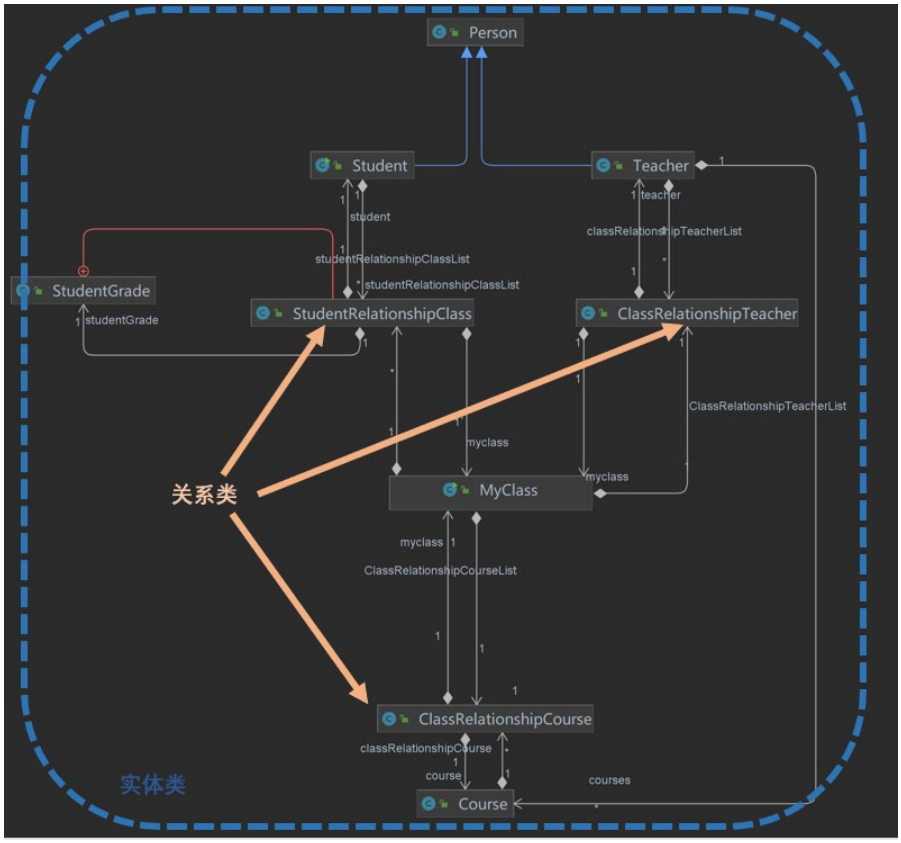

图 26 类关系图

4.11 设计模式

（1）桥接模式，桥接模式（Bridge Pattern）：JDBC 规范中使用了桥接模式。这是一种结构型设计模式，基于类的最小设计原则。通过使用封装、聚合及继承等行为，让不同的类承担不同的职责，DriverManager 是 JDBC 中的一个关键类，它负责加载数据库驱动并管理数据库连接。它充当了桥接模式中的桥接角色，将应用程序与具体的数据库驱动隔离开来。JDBC 使用反射方法加载数据库驱动。每个数据库厂商的驱动类必须实现 java.sql.Driver 接口，并在加载时将自身注册到 DriverManager 中。例如，对于 MySQL，我们可以通过 Class.forName（”com.mysql.jdbc.Driver”） 来加载驱动。

（2） 工厂模式 （Factory）：Spring 使用工厂模式创建 Bean 实例。Spring IoC 容器就是一个 Bean 工厂，通过配置元数据（XML 文件或注解），Spring 容器可以灵活地创建和管理 Bean。

（3）装饰器模式：装饰器模式（Decorator Pattern）允许向一个现有的对象添加新的功能，同时又不改变其结构。这种类型的设计模式属于结构型模式，它是作为现有的类的一个包装。实验中使用了@CrossOrigin、@ResponseBody、@RequestParam、@Autowired、@Controller、@Service

（4） 单例模式：userService 作为 service 的单例。单例模式（Singleton Pattern）这种类型的设计模式属于创建型模式，它提供了一种创建对象的最佳方式。这种模式涉及到一个单一的类，该类负责创建自己的对象，同时确保只有单个对象被创建。这个类提供了一种访问其唯一的对象的方式，可以直接访问，不需要实例化该类的对象。

（5） 依赖注入 （Dependency Injection） 这是 Spring 框架的核心模式，通过构造函数注入、setter 注入或字段注入，将依赖对象注入到目标对象中。

5 总结和创新点

5.1 项目总结

5.1.1 总结项目的主要实现过程和成果

(1) 主要实现过程

① 搭建常驻系统，常驻系统包括学生查看课程，教务管理课程，教师录入学生成绩等基本功能。搭建过程：

• 根据需求分析确定具体业务

• 前端服务器、日常业务服务器集群均选用学校机房服务器，搭建 MySQL Cluster 数据库服务器，反向代理服务器服务器选择 Nginx

• 根据业务和类，创建 MySQL Cluster，部署到数据库服务器

• 用 Vue.js 框架开发实现前端网页，部署到 JavaScript 服务器

• 后端部分使用 Spring Boot 框架进行开发，部署到日常业务服务器集群

• 实现 Nginx 服务器部署，负责处理客户端的请求，将请求转发到后端服务器，并进行负载均衡，还可以缓存静态资源，减轻后端服务器的压力，提高响应速度。

• 连接各台服务器

② 搭建选课系统，选课系统负责学生选课，具有高并发，压力大

• 根据需求分析确定具体业务

• 前端服务器、选课服务器集群、下单服务器集群、全局计算服务器集群、定时任务服务器集群均选用云服务器集群，使用①MySQL Cluster 数据库服务器，反向代理服务器服务器选择 Nginx

- 根据新增的业务和类，修改 MySQL Cluster，部署到数据库服务器
- 用 Vue.js 框架开发实现前端网页
- 后端部分使用 Spring Boot 框架进行开发
- 实现 Nginx 服务器部署，负责处理客户端的请求，将请求转发到后端服务器，并进行负载均衡，还可以缓存静态资源，减轻后端服务器的压力，提高响应速度。
- 在每学期的选课周，去租用云服务器集群，将上述内容部署到服务器，连接各服务器

③ 测试项目：单元测试、集成测试和系统测试，确保高并发不崩溃

④ 实际部署，持续监测、支持

(2) 成果

Java 企业级应用技术设计一个大型企业级软件系统，教务系统，除却常驻功能，选课功能做到满足 10 万学生同时选课。

5.1.2 评价新系统在性能和用户体验上的提升

(1) 性能

测试时，从响应时间、并发处理能力、资源利用率、稳定性与旧系统进行对比。

(2) 用户体验

测试时，从界面友好度、操作便捷性、反馈及时性、移动端适配性四个维度评价用户体验。

图 27 评价系统图

5.2 创新点

5.2.1 提出的技术方案和架构设计的创新点

架构图，见图 12

（1）本地服务器+云服务器，可拓展性强，性价比高

（2）分布式数据库 MySQL Cluster，高可用性、高扩展性、高性能

（3）后端部分使用 Spring Boot 框架进行开发，支持 Spring Cloud 等微服务组件，适合大规模系统的开发。

（4）系统的反向代理和本地缓存，使用 Nginx 负责处理客户端的请求，将请求转发到后端服务器，并进行负载均衡。此外，Nginx 还可以缓存静态资源，减轻后端服务器的压力，提高响应速度。

5.2.2 在高并发和系统稳定性方面的技术突破

（1） 使用缓存改善网站性能.

（2） 使用应用服务器集群改善网站的并发处理能力.

（3）数据库读写分离

（4）使用反向代理进行负载均衡加速网站响应.

（5）使用分布式数据库系统和分布式服务.，异步操作

特殊的应对策略

（6）选课下单系统单独部署，避免拖垮整个网站

（7）选课列表网页静态化，避免选课时查询，与数据库进行连接额外查询。

（8）租借选课活动的网络带宽

（9）动态生成随机选课下单 URL，避免用户直接访问。

（10）如图 25，只允许略大于课程人数的的用户端访问选课下单页面，减轻下单系统和数据库的压力。

5.2.3 在用户体验和安全性方面的优化措施

（1） 用户体验

提升反馈及时性，减少 http 请求、使用浏览器缓存、减少 cookies 传输，反向代理、分布式缓存、使用集群等策略

优化页面，原型设计的基础上不断优化

支持移动端和 PC 端，解决学生选课困难

（2）安全性

- 采用 OAuth 2.0 协议进行用户身份验证
- HTTPS 保证传输层安全和 SSL 证书
- 数据库加密
- 日志记录与安全审计
- 安全应急响应，定期备份、数据冗余、备份恢复、数据镜像

参考文献

[1] 李智慧, 大型网站技术架构-核心原理与案例分析[M]. 北京: 电子工业出版社, 2013.09.01.

[2] 林木.大型网站技术架构演进的分析与研究[J].软件,2020,41(3):197-200

[3] 刘艳. 基于 Eclipse RCP 的银行柜面软件架构可扩展性的研究[J]. 软件, 2018, 39(5): 18-21.

[4] 王旭峰, 王智立. 基于 SOA 的业务动态定制的网络管理系统设计与实现[J]. 软件, 2015, (1): 100-103.

[5] 徐云涛, 许武军, 翟梦琳. 基于 B/S 架构的高并发虹膜识别系统[J]. 计算机工程, 2019, 45(8): 102-106, 112.

[6] 马晓星, 刘譞哲, 谢冰, 等. 软件开发方法发展回顾与展望[J]. 软件学报, 2019, 30(1): 3-21.

[7] 王建民. 领域大数据应用开发与运行平台技术研究[J]. 软件学报, 2017, 28(6): 1516-1528.

[8] 李洋. 基于消息队列遥测传输协议的智能家居消息中间件设计[J]. 计算机应用, 2018, 38(z1): 162-164, 217.

[9] 黄芝龙, 徐莉莎, 瞿少成. 高并发 Web 电商系统的设计与优化[J]. 计算机与数字工程, 2019, 47(7): 1719-1724, 1775.

[10] 陈代旺. 高并发系统解决方案与分布式技术选型探讨[J].价值工程, 2019, 38(17): 251-253.

[11] JIANG Quan ZHU Chunling WANG Siqi. Qualitative analysisfor state/event fault trees using formal model checking[J].系统工程与电子技术(英文版), 2019, 30(5): 959-973.

[12] 张君施 等 数据库原理及应用[M].电子工业出版社.北京.2023.10

自评分

1	需求分析（20）	分析合理、充分，详尽程度如需求规格说明书 **20 分**
2	系统设计（40）	设计合理、充分，使用了许多有效的技术并考虑了成本,图文并茂 **40** 分
3	创新性（10）	具有较强创新性，见 5.2 创新点 **10** 分
4	技术综述（20）	技术综述完整，查阅了大量文献和参考资料 **20** 分
5	文档规范（10）	格式规范、逻辑清楚、论述完整，严格按照毕业论文要求 **10** 分

合计 100 分