课程主页及答疑系统

概要设计说明书

中国科学技术大学

2020 年春季学期 软件工程

幼儿编程小队（第七组）

当前版本	V2.0		
保密级别	内部级别		
作者	幼儿编程小队	编写日期	2020-08-16
审核人	幼儿编程小队	审核日期	2020-08-16

目录

1 简介

1.1 目的

本文档的书写旨说明对软件系统的设计，包括软件系统的基本处理流程，软件系统的组织结构、模块划分、功能分配、接口设计、运行设计、数据结构设计、数据表设计和出错处理设计等，为软件的详细设计奠定基础。

1.2 范围

1.2.1 软件名称

本软件名称暂定为中国科大课程管理及答疑系统。

1.2.2 软件功能

本项目希望实现一个面向教务管理人员、学生、老师及助教的在线课程平台，可以满足课程相关的各类用户对于课程的不同需求，既能清晰地显示课程信息，又能整理同学提出的问题，方便学生、老师和助教在线交流，提高效率。

本软件为每个课程提供一个网页平台，包括课程信息、教师及助教信息、课程日历、课程资源、课程公告和答疑区，并实现多学期支持。除答疑区需进行身份认证外，其他课程信息及资源均为公开内容，提升课程的开放程度。在答疑区中，学生可以提出问题、讨论问题、编辑答案、根据标签查找问题，老师和助教可以回答问题，也可以删除重复问题和无意义问题。答疑区记录问题答案的编辑版本历史，在需要时可进行回滚操作。

同时，软件提供统一的课程后台管理系统，以便教务管理人员和老师管理课程。

课程主页

课程主页旨在为学生提供一个了解课程内容、跟进课程进度的网页平台，包含了课程介绍、教师和助教的信息、课程公告，同时提供课程表和课程大纲，以及课程相关文件的下载。同时，课程主页也提供了过往学期主页的链接和本课程对应答疑系统的链接。课程主页向所有浏览者开放，无论其是否为本课程的学生，促进知识的自由传播，方便社会各界人士对本课程进行学习，或是提出意见或建议。

答疑系统

答疑系统提供了一个为同学们答疑解惑的在线平台，帮助同学们快速解决问题，更好地掌握知识点。

答疑系统支持用户提问，提问的内容不仅可以是纯文字，还能插入图片、链接、代码块、公式等，方便各个学科的同学在这个平台提出问题。提问中可以添加问题标签便于系统自动分类 (例如针对 “作业二” 的提问)。问题发布后，其它用户 (学生、老师、助教) 均可以回答问题。在问题已有回答后，其它用户仍可更改问题与回答，每次编辑的记录都会被详细记录，用户甚至可以通过时间轴，查看不同版本的回答，以及答案不断完善改进的过程。

管理系统

管理系统主要包括两个功能: 课程主页信息的修改、答疑系统的浏览权限管理。对于前者，管理人员的身份必须为教师或者助教，他们可以修改信息或者上传文件。对于后者，管理人员的身份是系统管理人员，负责管理课程对应的教师、助教、学生，其主要目的是限制答疑系统的访问权限，即只有选课的同学或相关的教师和助教才有权限访问该课程的答疑页。

1.2.3 软件应用

本软件可以应用到各学校，方便师生在线获取课程信息和互动交流。

1.3 实现系统环境

1.3.1 开发环境

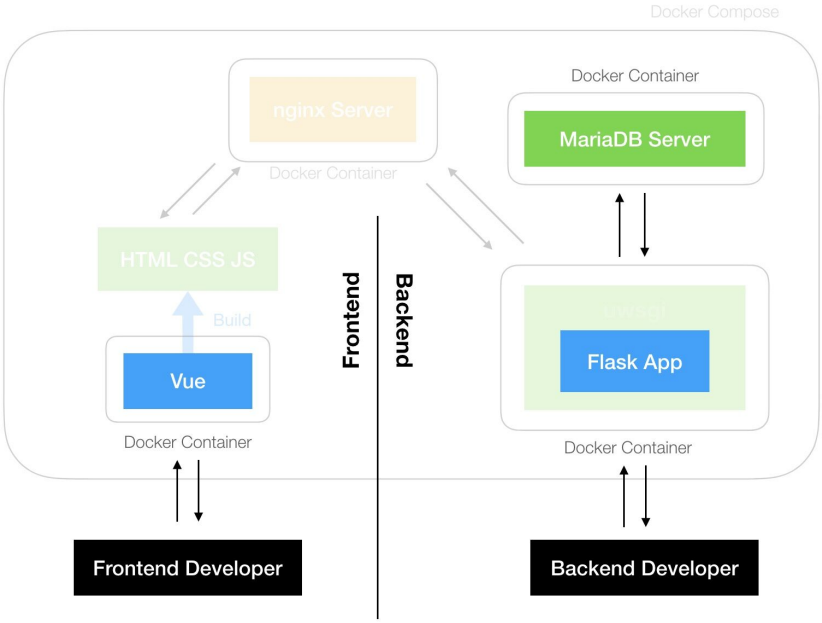

图 1: 开发架构图

使用前后端分离的方法进行开发：在第一阶段，前端仅完成静态页面的设计，不进行任何数据请求，后端根据事先商定的 API 和功能需求设计数据库；在第二阶段，前端使用 mock 自行生成数据，并在本地完成 HTTP 请求的开发和测试，后端开发 API 接口；在第三阶段，后端部署服务，前端将 mock 改为向实际服务器请求。

前端部分的开发环境以主要使用 ES5 作为开发语言，使用的框架是 Vue，node 版本是 10.x，同时因为 yarn 高效快速的特点，包管理器采用 yarn。在 UI 组件上，选择了和 Vue 高度契合并且简洁丰富的 Element-ui 组件作为 UI 组件库。为了便于前端开发，减轻因机器本身环境的不同造成的工作量，额外使用了 docker 进行开发工作环境的统一工作，为了减少流量消耗，使用了 alpine 作为宿主环境。由于在工作中使用了 Vuex 作为状态保存工具，使用了 vue-persist 持久化应用产生的数据。

后端部分的开发环境是以 Python 作为主要编程语言，基于 Flask 框架的 Docker 环境或 venv 环境。考虑到小组成员的开发机的硬件，操作系统，以及软件版本各有不同，为了避免版本和兼容问题，我们使用 Docker 或 venv 进行开发环境的统一。此外，考虑到 Python 是一种解释型的脚本语言，开发效率高，开发小组的成员更熟悉，可拓展性强，故选择 Python3 作为主要编程工具。Python 有还有上百种 Web 开发框架，最终选择了我们选择使用更简洁优雅，凸显 Pythonic 风格的 Flask。

1.3.2 客户端运行环境

现代 Web 浏览器。

1.3.3 服务器端运行环境

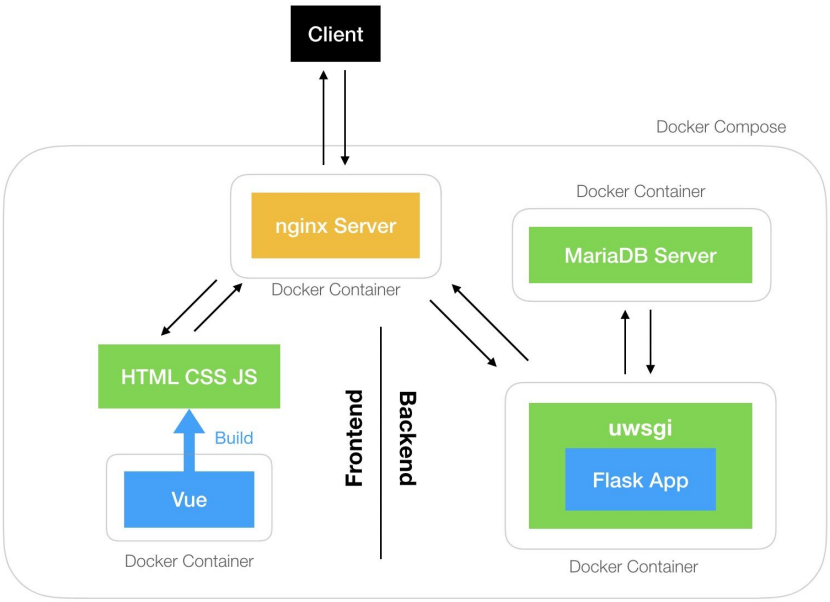

图 2: 部署架构图

前端应用打包后，会生成静态资源文件夹，包含该应用所需的全部资源，因此，对于前端应用部署，只需要将其通过 Nginx 静态代理即可。

至于服务器端的后端，数据库，以及负载均衡服务器均被部署在 Docker 环境中，各个模块各自的 Docker 环境中安装有它们各自所需的依赖，并通过 Docker 网桥进行通信。Docker 容器解决了各个模块的依赖和环境的统一问题，简化了部署的过程，并使开发环境和生产环境有更好地统一。完整的服务端系统可被部署在安装有 Docker 的 Linux，Windows 或 MacOS 上，并让 nginx 服务器暴露相应的端口以便客户发起请求。

2 第零层设计描述

2.1 软件系统上下文定义

2.1.1 课程主页

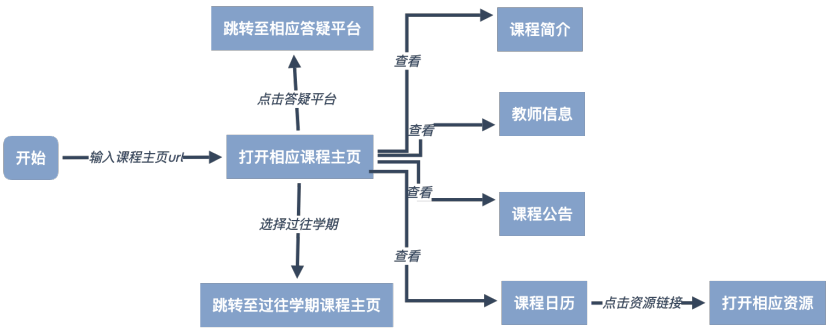

图 3: 课程主页上下文定义

每个课程主页有不同的 url，根据课程对应的 url 即可进入其课程主页。打开课程主页后，可以查看课程简介、教师信息、课程公告和课程日历。其中课程日历中包含每节课对应的课程资源链接，点击即可打开相应资源。除了课程信息与资源外，课程主页中包含答疑系统和过往学期课程主页的链接，点击答疑系统或选择过往学期即可跳转。

2.1.2 答疑系统

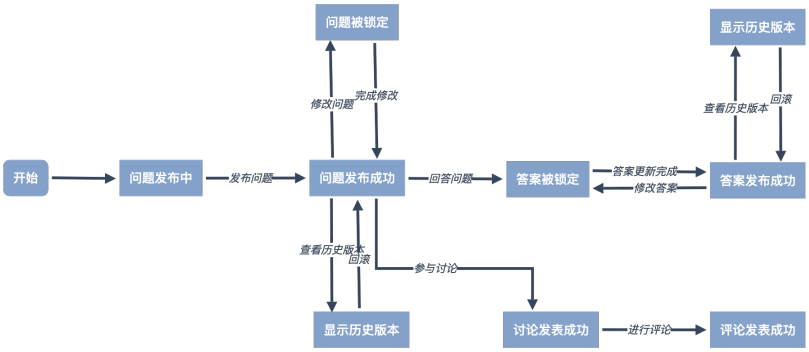

图 4: 答疑系统上下文定义

进入答疑系统后，输入问题内容后即可发布新问题。

对每一个问题而言，问题发布成功后，所有用户均可以对问题进行修改。当有用户在修改问题时，问题将被锁定，其他用户此时无法修改该问题。当修改成功后问题将被重新发布、解除锁定。对每一个问题，可以查看其历史版本，并可选择其中某一版本进行回滚。

每个问题将包括两个回答：学生答案、教师答案。当用户回答问题时，对应答案将被锁定，其他用户无法编辑此答案。当答案更新完成后，将重新发布答案、解除锁定。同样地，每个回答也将记录其历史版本，用户可查看答案的历史版本并进行回滚。

除回答问题外，用户还可以在每个问题的讨论区中参与讨论。对于用户感兴趣但无法确定答案的问题，用户可在讨论区中发布讨论，并可对所有已发布的讨论进行评论。

2.1.3 管理系统

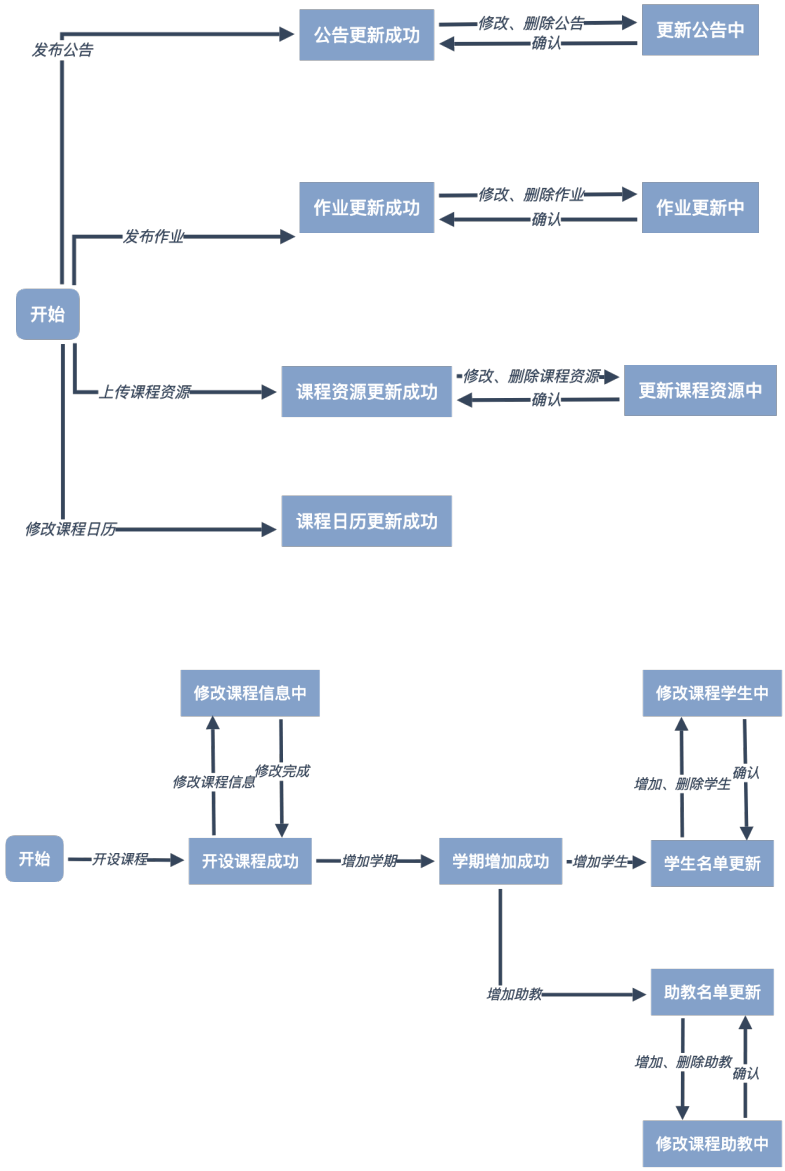

图 5: 管理系统上下文定义

管理系统根据权限分为管理员和教师两部分。

教师进入管理系统后，在选定某一个课程后即可对课程信息及资源进行管理。教师可在后台管理系统中管理课程公告。教师可在输入公告标题和信息后发布新公告。公告发布成功后可以再对其进行修改或删除，确认后公告将进行更新。类似地，教师可在后台管理系统中发布、修改或删除作业，可在后台管理系统中上传、修改或删除课程资源，也可在后台管理系统中修改课程日历。

管理员进入后台管理系统后，可以开设新的课程。课程开设成功后，可以修改其课程信息进行更新。对每一个开设成功的课程，管理员可以为其增加学期。对每一个学期，管理员可以对该学期的课程增加学

生和助教。增加学生或助教后，学生和助教名单将进行更新。管理员也可以对当前学生名单和助教名单中的学生或助教进行修改、删除。

3 前端第一层设计描述

3.1 系统架构

3.1.1 课程主页

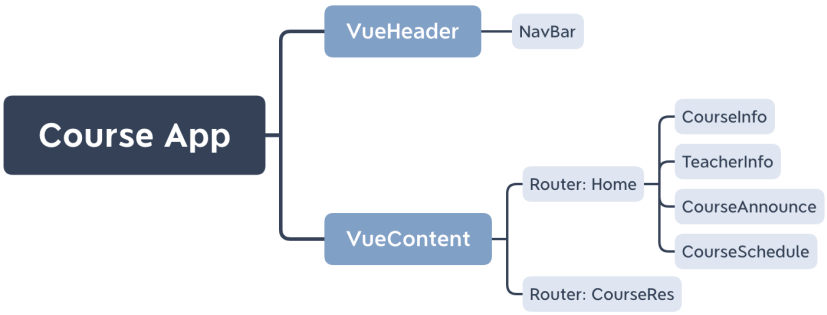

图 6: 课程主页模块结构图

- VueHeader 模块：界面的顶部栏，包括了课程名称、学期、NavBar 组件。
- NavBar 模块：菜单栏，可触发 VueContent 模块界面的切换，同时还有链接到对应答疑系统的按钮和过往学期的按钮。
- VueContent 模块：界面的内容主体部分，提供路由功能，根据路由，可在 Home 组件和 CourseRes 组件之间切换显示。
- Home 模块：包含本课程所有相关信息的组件，包括课程介绍信息、教师和助教的信息、公告、教学日程。
- CourseInfo 模块：包括课程中英文名、学期、简介。
- TeacherInfo 模块：包括教师和助教的姓名、邮箱、手机号。
- CourseAnnounce 模块：包含课程公告的标题、内容、发布者、发布日期、截止日期等信息。
- CourseSchedule 模块：教学日程，每一个教学课次包含了周数、日期、本课次相关资源、作业等。
- CourseRes 模块：课程相关的资源。

3.1.2 答疑系统

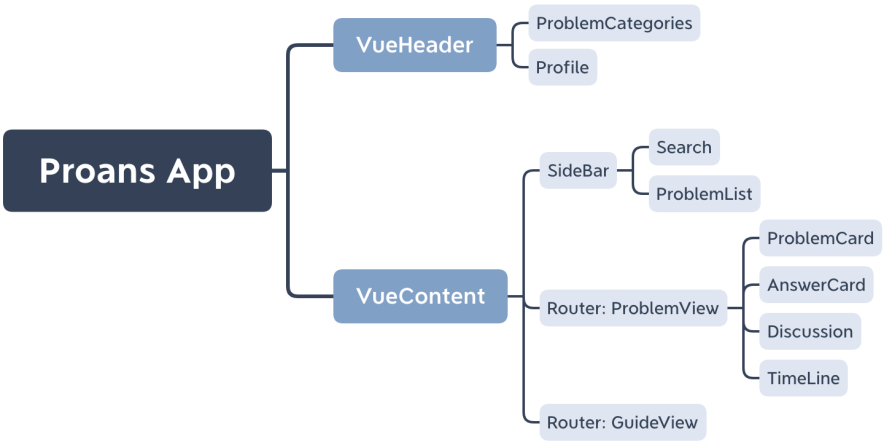

图 7: 答疑系统模块结构图

- VueHeader 模块：界面的顶部栏，包括了 ProblemCategory 模块、profile 模块。
- ProblemCategory 模块：标签分类选择列表，显示答疑系统中当前课程的所有问题标签，通过标签可以筛选出含有该标签的所有问题。
- Profile 模块：用户资料，提供用户头像、密码、个人资料的修改。
- VueContent 模块：界面的主体部分，包含 SideBar 模块及主体界面，主体界面根据路由在 ProblemView 组件和 GuideView 组件之间切换。
- SideBar 模块：侧边栏，包含 Search 组件和 ProblemList 组件。
- Search 组件：搜索组件，根据标签和标题同时搜索问题，搜索结果显示在 ProblemList 组件上。
- ProblemList 组件：问题列表，显示在当前限制条件下的所有问题，对于每一个问题显示问题标题及问题描述，并提供新增问题接口。
- ProblemView 组件：单个问题的详细内容界面，包含 ProblemCard 组件、AnswerCard 组件、Discussion 组件、TimeLine 组件。
- ProblemCard 组件：包含问题标题及问题描述，并对问题提供点赞、编辑以及删除按钮。
- AnswerCard 组件：包含针对问题的学生回答以及教师/助教回答，并分别提供编辑按钮。
- Discussion 组件：包含针对问题的所有讨论主题以及相应的回复，针对某一主题提供编辑以及删除按钮。
- Timeline 组件：时间轴，包含针对问题的历史编辑记录，可选择回退到之前的某一答案版本。
- GuideView 组件：操作教程，在用户未选择问题显示时显示系统操作的部分简单提示。

3.1.3 管理系统

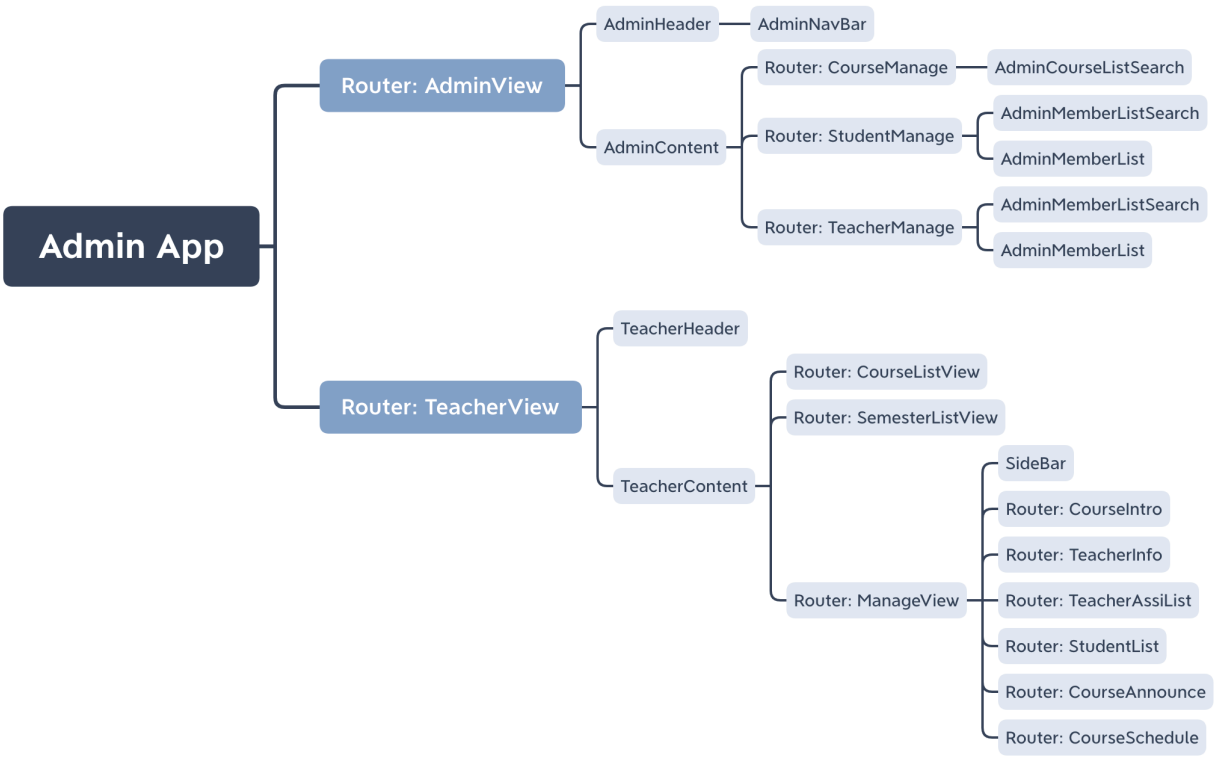

图 8: 后台系统模块结构图

- AdminHeader 模块：管理员界面顶部导航，另外还附带管理员个人信息查看和修改。
- AdminContent 模块：用于显示管理界面的一个路由视图。
- CourseMange 模块：管理员课程管理界面，提供课程的增删改查功能。
- TeacherManage 模块：管理员教师管理界面，提供教师的增删改查功能。
- StudentManage 模块：管理员学生管理界面，提供教师的增删改查功能。
- DataTable 模块：用于渲染数据量较大的数据，分页展示，用户可以自己选择跳转的页面或者是选择单词渲染的条数。
- DataTableSearchBar 模块：用于提供搜索功能，用户输入功能后，可以在 DataTable 中的数据中查找条目。
- TeacherHeader 模块：教师界面顶部导航，另外还附带教师个人信息查看和修改。
- TeacherContent 模块：教师内容区的路由组件。
- CourseListView 模块：用于渲染某个教师所教授的全部课程。
- SemesterListView 模块：用于渲染某个教师下面的某个课程对应的全部学期。
- ManageView 模块：用于渲染教师对某个课程的详细管理界面。

3.2 功能实现与模块/子系统的关系

功能需求	课程主页	答疑系统	管理系统	功能需求	课程主页	答疑系统	管理系统
注册/注销账户			√	增加新学期			√
开设/删除课程			√	修改课程基本信息			√
查看课程公告	√		√	发布课程公告			√
下载课程资源	√			上传/删除课程资源			√
查看课程作业	√		√	发布/删除课程作业			√
修改课程日历			√	修改主页显示模块			√
授权课程学生			√	授权课程老师/助教			√
跳转过往学期	√			进入答疑平台	√		
登录答疑平台		√		修改平台账户密码		√	
新建一个问题		√		查看某一问题		√	
点赞某一问题		√		关注某一问题		√	
编辑某一回答		√		锁定编辑/恢复编辑		√	
查看关注问题		√		查看问题历史版本		√	
删除某一问题		√		在讨论区发布评论		√	

4 前端第二层设计描述

4.1 课程主页

图 9: 课程主页示意图

4.1.1 App 根模块

子组件

VueHeader、VueContent

功能

通过 GET 请求，从后端 api 获取与本课程相关的所有信息，赋值给名为 allinfo 的对象，将这个对象传给子组件 VueHeader、VueContent，以供其后续使用。也可以把数据存储在 Vuex，由子组件直接从中提取。

4.1.2 VueHeader 模块

参数

allinfo：包含了本课程相关的所有信息

子组件

NavBar

功能

界面的顶部栏，左侧显示课程名称、当前学期，右侧显示 NavBar 的内容。接收 App 组件传入的 allinfo 对象，将它传给 NavBar。

4.1.3 NavBar 模块

参数

allinfo：包含了本课程相关的所有信息

功能

界面的导航栏，包括“课程主页”和“课程资源”按钮，用来切换路由，以达到切换主界面显示内容的目的。还包括本课程对应的答疑系统的外链、过往学期的链接。这里，过往学期为一个下拉菜单，显示学期，如 2020 年春季。

4.1.4 VueContent 模块

参数

allinfo：包含了本课程相关的所有信息

子组件

Home、CourseRes

功能

界面的主要内容显示区域，用于根据路由来切换所要显示的内容，如果路由为`/course/:cid/home`或`/course/:cid` 则显示 Home 组件的内容，若路由为`/course/:cid/resourse` 则显示 CourseRes 组件的内容。其中 cid 为当前课程的唯一 id 号，与后端一致。同时，会把 allinfo 对象传给子组件。

4.1.5 Home 模块

路由

`/course/:cid/home` 或`/course/:cid`

参数

allinfo：包含了本课程相关的所有信息

子组件

CourseInfo、TeacherInfo、CourseAnnounce、CourseSchedule

功能

展示课程相关的信息的布局，包括 CourseInfo 组件、TeacherInfo 组件、CourseAnnounce 组件、CourseSchedule 组件的内容，把 allinfo 对象直接下传给这些组件。

4.1.6 CourseInfo 模块

参数

allinfo：包含了本课程相关的所有信息

功能

以单张卡片的形式展示课程的中英文名称、学期、简介，其中，左侧为名称和和学期，中文名称字体稍大，右侧为简介垂直居中，水平左对齐。

4.1.7 TeacherInfo 模块

参数

allinfo：包含了本课程相关的所有信息

功能

以单张卡片的形式展示教师和学生的相关信息，包括身份、姓名、邮箱、手机号，卡片内以表格形式呈现，不显示单元格边框。

4.1.8 CourseAnnounce 模块

参数

allinfo：包含了本课程相关的所有信息

功能

以单张卡片的形式展示公告内容，为了避免课程主页中信息杂乱、难以找到有效信息，课程公告中将默认只显示最近一个公告，其余公告将会被折叠。若需要查看历史公告，需要点击以显示更多公告。没有公告时，显示“暂时没有任何公告哦 ~”。当没有公告或者只有一条公告时，不显示折叠展开的按钮。

4.1.9 CourseScedule 模块

参数

allinfo：包含了本课程相关的所有信息

功能

课程日程中将显示各周课程的时间、课程内容，同时集成了课程作业和课程资源，以方便同学快速找到某节课对应的资料。对每一节课都将在日程中显示其作业和课程资源，如 PPT 等，课程资源将显示为链接形式。

4.1.10 CourseRes 模块

路由

`/course/:cid/resourse`

参数

allinfo：包含了本课程相关的所有信息

功能

卡片列表形式展现各个资源的标题、内容、下载链接等，这部分功能与 CourseSchedule 模块的部分功能重复，但又面向不同范围的资源，此功能为选做。

4.2 答疑系统

图 10: 答疑系统示意图

4.2.1 App 根模块

子组件

VueHeader、VueContent

功能

通过 GET 请求，从后端 api 获取与本课程相关的所有问题，并通过 Vuex 存储存储所有问题信息，子组件直接与 Vuex 交互获取数据。

4.2.2 VueHeader 模块

子组件

ProblemCategory、profile

功能

界面的顶部栏，左侧显示 ProblemCategory 模块的内容，右侧显示 Profile 模块的内容。

4.2.3 ProblemCategory 模块

参数

categories：本课程问题中使用过的所有标签的集合

功能

通过与 Vuex 交互获取标签集合，将其显示为按钮集合，用户选择某一标签后，通过路由跳转到对应包含该标签的问题列表界面，将该标签的 id 通过路由 query 传递，key 为 tid。

4.2.4 Profile 模块

参数

userinfo: 包含该用户的所有信息

功能

通过 GET 请求，从后端 api 获取与该用户相关的所有信息，显示给用户。点击提交按钮后，将用户修改的信息通过请求发送给后端，进行用户信息修改。

4.2.5 VueContent 模块

功能

左侧显示 sidebar 模块的内容，右侧为问题的主体部分，用于根据路由来切换所要显示的内容，如果路由为 `/proans?tid=<int>` 则显示 GuideView 组件的内容；如果路由为`/proans?tid=<int>&qid=<int>` 则显示 ProblemView 组件的内容。

4.2.6 SideBar 模块

功能

侧边栏，顶部显示 Search 组件的内容，然后显示 ProblemList 组件的内容。通过与 Vuex 交互，获取本课程问题中使用过的所有标签的集合，传递给 Search 组件；获取对应标签的问题列表，传递给 ProblemList 组件，其中，标签从路由中获取。通过监听 Search 模块传回的查询信息，对问题列表进行更新。

4.2.7 Search 模块

参数

tags：本课程问题中使用过的所有标签的集合

功能

查询框，将用户选择的标签以及输入的标题相关文本通过 emit 方法传递给父组件 SideBar。

4.2.8 ProblemList 模块

参数

problems：需要显示的问题列表

功能

显示可选择的问题列表，选择问题后，通过路由跳转至为`/proans?tid=<int>&qid=<int>`，其中 tid 为标签 id，从路由中获取，qid 为问题 id，从选择的问题信息中获取。同时提供添加问题按钮，跳转至`/proans/addproblem/?tid=<int>`。

4.2.9 ProblemView 模块

功能

显示问题详情，自上而下分别显示 Timeline、ProblemCard、AnswerCard、Discussion 组件的内容。通过从路由中获取的问题 id，将从 Vuex 中获取到问题的历史信息、问题描述信息和答案信息、通过 GET 请求从后端获取到的讨论信息，分别传递给以上四个组件。

4.2.10 ProblemCard 模块

参数

problem：包含一个问题的基本信息

功能

显示问题基本信息，选择编辑后跳转至/proans/editView/tid=<int>, 并通过 params 传递问题信息。选择删除后向后端发送 DELETE 请求，删除问题。选择点赞后，向后端发送 POST 请求，进行点赞/取消。

4.2.11 AnswerCard 模块

参数

problemId：问题 ID teacherAnswer：该问题的教师/助教回答 studentAnswer：该问题的学生回答

功能

分别显示问题的教师/助教回答以及学生回答，在对应回答卡片上选择编辑会弹出答案编辑器，在编辑完成提交后，向后端发送 PUT 请求，提供编辑信息，对相应的回答进行编辑。

4.2.12 Discussion 模块

参数

commentList：该问题的讨论主题的列表

功能

问题的评论区，显示用户对于该问题及其回答的讨论。可新增讨论主题或对已有讨论主题进行修改或者回复。新增讨论主题将向后端发送 POST 请求，提供讨论主题的标题及内容。修改已有讨论时，向后端发送 PUT 请求，当用户为该主题创建者时可修改。

4.2.13 Timeline 模块

参数

history：问题的编辑历史

功能

版本时间轴，以时间轴形式显示该问题及其回答的版本编辑记录。通过点击时间轴上的点可浏览过往回答版本，同时在浏览之前版本后提供版本回退功能，此功能只对教师用户生效。

4.2.14 GuideView 模块

功能

提示界面，在用户未选择问题时显示系统操作提示。显示查看某一问题的基本操作步骤。

4.3 管理系统

图 11: 管理员课程管理示意图

4.3.1 AdminHeader 模块

功能

管理员界面顶部导航，链接指向管理员的个人信息页面，支持对信息的修改。

4.3.2 AdminNavBar 模块

功能

导航栏三个标签分别为课程管理，教师管理和学生管理, 其通过路由链接到不同的管理界面。

4.3.3 AdminContent 模块

子组件

CourseManage，TeacherManage，StudentMnage

功能

用于显示管理界面的一个路由视图，根据路由分别显示组件。从后端获取所有课程、教师、学生数据并保存到 Vuex 中。

4.3.4 CourseManage 模块

参数

allCourses：所有课程的信息。

子组件

DataTable, DataTableSearcgBar.

功能

管理员课程管理界面，以列表形式显示所有课程信息，并提供课程的增删改查功能。

4.3.5 TeacherManage 模块

参数

allTeachers：所有教师的信息。

子组件

DataTable, DataTableSearcgBar.

功能

管理员教师管理界面，以列表形式显示所有教师信息，提供教师的增删改查功能。

4.3.6 StudentManage 模块

参数

allStudents：所有学生的信息。

子组件

DataTable, DataTableSearcgBar.

功能

管理员学生管理界面，以列表形式显示所有教师信息，提供学生的增删改查功能。

4.3.7 DataTable 模块

参数

renderData: 需要被渲染的数组。

tableForm：表格的格式控制，对象数组。

功能

用于渲染数据量较大的数据，分页展示，用户可以自己选择跳转的页面或者是选择单词渲染的条数。

4.3.8 DataTableSearchBar 模块

功能

用于提供搜索功能，用户输入功能后，可以在 DataTable 中的数据中查找条目。

图 12: 教师课程管理示意图

4.3.9 TeacherHeader 模块

功能

教师界面顶部导航，另外还附带教师个人信息查看和修改。

子组件

TeacherNavBar。

4.3.10 TeacherNavBar 模块

功能

教师顶部的导航条，用于跳转到课程选择或者学期选择。

4.3.11 TeacherContent 模块

子组件

CourseListView，SemesterListView，ManageView。

参数

AllCoursesForTeacher：某个教师对应的全部课程，数组。

功能

教师内容区的路由组件。通过路由显示不同的组件。

4.3.12 CourseListView 模块

功能

用于渲染某个教师所教授的全部课程。通过 GET 请求从后端获取该教师教授的全部课程信息，并保存到 Vuex 中。

4.3.13 SemesterListView 模块

功能

用于渲染某个教师下面的某个课程对应的全部学期。从 Vuex 中获取对应课程的学期列表。

4.3.14 ManageView 模块

子组件

MemberList，CourseInfo，CourseAnnounces，CourseAssignment，CourseResource，CourseSchedule。

功能

用于渲染教师对某个课程的详细管理界面。选择左侧标签后，通过路由跳转至不同的组件进行编辑。

4.3.15 MemberList 模块

功能

显示助教和学生的列表，显示助教和学生的详细信息，同时提供增删改查功能。

4.3.16 CourseInfo 模块

功能

显示课程信息，同时提供课程信息修改功能，修改后的课程信息将会在对应的课程主页上更新。

4.3.17 CourseAnnounces 模块

功能

显示课程通知，同时提供增删改查功能，课程通知将会显示在对应的课程主页上面。

4.3.18 CourseAssignment 模块

功能

显示课程对应的作业信息，提供发布修改功能，发布和修改后均会在对应的课程主页上更新。

4.3.19 CourseResource 模块

功能

显示该课程对应的教学资源信息，教师可以在此上传 Slides 等资源。

4.3.20 CourseSchedule 模块

功能

显示该课程对应的日常安排，教师新建日程后，可以在该日程上选择对应的作业，资源和参考资料，最终显示在课程主页上。

5 后端第一层设计描述

5.1 系统架构

后端系统主要划分为：主页系统自成模块，权限控制模块，答疑系统中的问答模块、问题历史模块、收藏模块，和管理系统中的选课管理模块、主页信息管理模块。其中，权限控制系统仅作用于答疑系统和管理系统，于用户访问前对其权限等级作出划分。

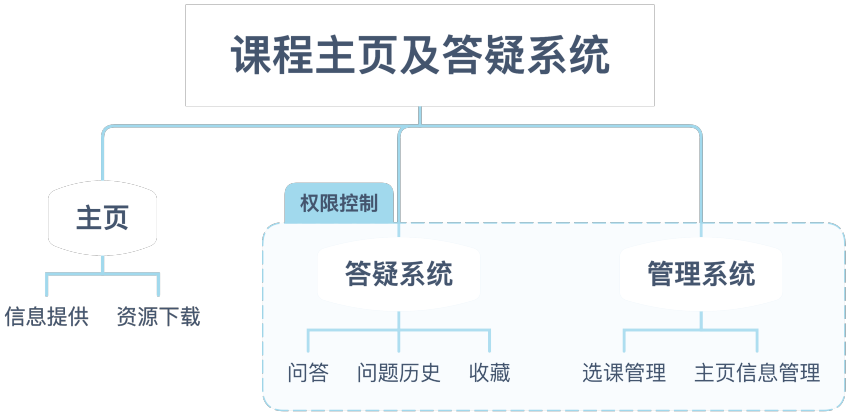

图 13: 后端系统架构图

5.2 功能实现与模块/子系统的关系

	课程模块	权限模块	问答模块	问题历史记录模块	收藏模块
通知功能	✓				
回答功能			✓	✓	✓
作业功能	✓				
课程功能	✓				
讨论功能			✓		
提问功能			✓		
资源功能	✓				
日程功能	✓				
用户功能		✓			✓

6 后端第二层设计描述

6.1 主页模块

6.1.1 信息提供模块

模块架构

主页信息提供模块负责返回用户请求的，与课程主页相关的信息，包括课程的基本信息，日程安排，作业安排等。该模块由课程，日程，作业等子模块中的信息查询接口构成。

相关数据表设计

Course 课程

字段	描述	是否必须	是否唯一	默认值	备注
cid	课程主键	是	是		主键
name_zh	课程中文名	是	否		长度最大为 40
name_zh	课程英文名	是	否		长度最大为 40
intro	课程简介	否	否	null	长度最大为 200
pre_course	前置课程	否	否	null	长度最大为 50
textbooks	该课程课本	否	否	null	长度最大为 50
semester	课程学期	否	否	null	长度最大为 50
series	课程系列	否	否	null	长度最大为 50

Schedule 日程

字段	描述	是否必须	是否唯一	默认值	备注
id	日程主键	是	是		主键
topic	日程话题	是	否		长度最大为 255
reference	日程参考	是	否		长度最大为 255
course_cid	所属课程的 cid	是	否		外键
week_id	周数	是	否	null	整数

Assignment 作业

字段	描述	是否必须	是否唯一	默认值	备注
id	作业主键	是	是		主键
schedule_id	所属日程主键	是	否		外键
title	作业标题	是	否		长度最大为 63
due	截止日期	是	否		整数

模块 API

获取课程的全部信息

请求方法：`GET`

请求地址：`/v1/courses/<int:cid>`

成功返回：

```
{
    "name_zh": String,
```

```
"name_en": String,
"intro": String,
"textbooks": [String]
"semesters": [
    "name": String,
    "link": String
],
"series":String,
"series_courses":[CourseObject],
"teachers": [
    {
        "name": String,
        "email": String,
        "phone": String
    }
],
"assistants": [
    {
        "name": String,
        "email": String,
        "phone": String
    }
],
"announces": [
    {
        "id": Number
        "title": String,
        "author": AuthorObject,
        "create_time": DateTime,
        "content": String
    }
],
"schedules": [
    "week": Number,
    "topic": String,
    "reference": String,
    "assignments": [
        "title": String,
        "due": String,
        "attachments": [
            "name": String,
            "link": String
        ]
    ]
```

```
        ],
        "resources": [
            {
                "title": String,
                "date": Date,
                "content": String,
                "attachments": [
                    {
                        "name": String,
                        "link": String
                    }
                ]
            }
        ]
    }
```

获取全部课程信息

请求方法：GET

请求地址：/v1/courses

成功返回：

```
    [
        {
            "id": Number,
            "name_zh": String,
            "name_en": String,
            "intro": String,
            "pre_course": String,
            "textbooks": String,
            "semester": String,
            "series": String,
            "teachers_gid": [String],
        },
    ]
```

获取全部老师信息

请求方法：GET

请求地址：/v1/users/teachers

成功返回：

```
    [
        {
            "gid": Number
            "name": String,
            "email": String,
            "phone": String,
```

```
            "school": String,
        }
    ]
```

获取用户对应的课程信息

请求方法：GET

请求地址：/v1/users/<gid>/courses

```
    [
        {
            "cid": Number
            "name_zh": String,
            "name_en": String,
            "intro": String,
            "pre_course": String,
            "textbooks": String,
            "semester": String,
            "teachers_gid": [String],
            "students_gid": [String],
            "tas_gid": [String]
        },
    ]
```

获取某一课程对应的学生/助教列表

请求方法：GET

请求地址：/v1/courses/<cid>/students 或/v1/courses/<cid>/tas

请求参数：

```
    {
        "ids": [String]
    }
```

成功返回：

```
    [
        {
            "id": String,
            "name_zh": String,
            "name_en": String,
            "email": String,
            "phone": String,
            "school": String,    // 学院
        }
    ]
```

获取某课程的日程安排

请求方法：GET

请求地址：/v1/courses/<cid>/schedules

成功返回：

```
[
    {
        "id": Number,
        "week_id": Number,
        "lectures": [
            "title": String,
            "resources": [
                {"url": String, "title": String}
            ],
            "assignments": [
                {"url": String, "title": String}
            ]
        ]
    }
]
```

查询某个的日程安排

请求方法：GET

请求地址：/v1/schedules/<int:sid>

成功返回：

```
[
    {
        "id": Number,
        "week_id": Number,
        "topic": String,
        "reference": String,
        "assignments": [
            "id": Number,
            "schedule_id": Number,
            "title": String,
            "due": Number
        ]
    }
]
```

查询作业信息

请求方法：GET

请求地址：/v1/assignments/<int:aid>

成功返回：

```
[
    {
        "id": Number,
        "schedule_id": Number,
        "title": String,
        "due": Number
    }
]
```

6.1.2 资源下载模块

模块架构

该模块负责处理课程资源的下载，由一个单独的课程资源模块进行处理。

获取课程资源

请求方法：GET

请求地址：/v1/courses/<cid>/resources

上传资源 请求方法：PUT

请求地址：/v1/resources/<int:fid>

下载资源 请求方法：GET

请求地址：/v1/resources/<int:fid>

6.2 权限控制模块

模块架构

权限控制模块采取面向切面编程的思路，由三种颗粒度的子控制模块组成：分别为选课关系鉴别模块、师生身份鉴别模块、个体身份鉴别模块。在实际应用中，对用户的请求，后端在访问视图函数前，依次进行这三个层级的验证，从而保障资源安全。

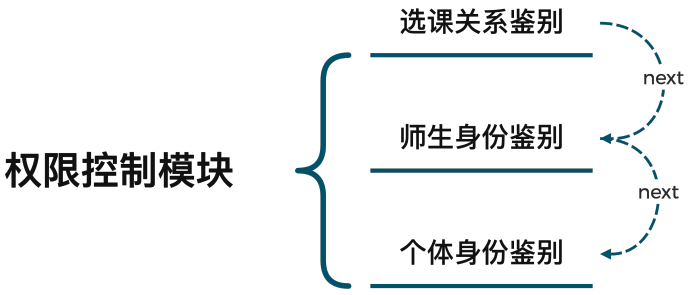

图 14: 资源控制模块架构图

功能实现与子模块的关系

	选课关系鉴别	师生身份鉴别	个体身份鉴别
提问	✓		
提交学生回答	✓	✓	
提交老师回答	✓	✓	
删除问题	✓	✓	
获得个体用户信息			✓
收藏问题	✓		✓
新建 tag	✓	✓	

分解描述

选课关系鉴别

接口：`login_required`

输入：token

输出：若选课，则将用户写入全局变量 g.uid，否则进入错误处理

师生身份鉴别

接口：`role_required`

输入：g.uid

输出：将用户身份写入全局变量 g.role

个体身份鉴别

接口：`identity_required`

输入：当前访问资源号 sid 和用户的 uid

输出：若该资源属于该用户，则返回 True，否则为 False

相关数据表设计

User 用户

字段	描述	是否必须	是否唯一	默认值	备注
gid	用户 gid 作为主键	是	是		主键
auth	用户类型	是	否		Small Integer
email	邮箱地址	否	否	null	邮箱格式长度最大为 63
nickname	用户昵称	是	是		唯一用户名长度最大为 24
name	用户姓名	否	否	null	长度最大为 24
phone	用户手机号	否	否	null	长度最大为 24
school	用户所属学院	否	否	null	长度最大为 63

6.3 答疑系统

6.3.1 问答模块

模块架构

该模块负责完成用户提问和回答问题，由提问 API 和回答 API 两部分组成。

相关数据表设计

Question 问题

字段	描述	是否必须	是否唯一	默认值	备注
id	问题主键	是	是		主键
course_cid	所属课程主键	是	否		外键
title	问题标题	是	否		长度最大为 127
content	问题内容	是	否		Text
author_gid	作者 gid	是	否		外键
teacher_aid	教师回答外键	否	否	null	外键
student_aid	学生回答外键	否	否	null	外键
update_time	更新时间	否	否	null	整数

Answer 回答

字段	描述	是否必须	是否唯一	默认值	备注
id	回答主键	是	是		主键
question_id	所属问题主键	是	否		外键
content	回答内容	是	否		Text
author_gid	作者 gid	是	否		外键

DiscussionTopic 讨论主题

字段	描述	是否必须	是否唯一	默认值	备注
id	讨论主题主键	是	是		主键
question_id	所属问题主键	是	否		外键
title	讨论主题标题	是	否		长度最大为 127
content	讨论主题内容	是	否		Text
author_gid	作者 gid	是	否		外键

DiscussionAnswer 讨论回答

字段	描述	是否必须	是否唯一	默认值	备注
id	讨论回答主键	是	是		主键
topic_id	所属讨论主题主键	是	否		外键
content	讨论主题内容	是	否		Text
author_gid	作者 gid	是	否		外键
reply_to	回复讨论回答主键	否	否	0	整数，若是 0 表示回复主楼

Announce 通知

字段	描述	是否必须	是否唯一	默认值	备注
id	通知主键	是	是		主键
course_cid	所属课程主键	是	否		外键
title	通知标题	是	否		长度最大为 127
content	通知内容	是	否		Text
author_gid	作者 gid	是	否		外键

CourseResource 课程资源

字段	描述	是否必须	是否唯一	默认值	备注
id	文件主键	是	是		主键
course_cid	所属课程主键	是	否		外键
name	文件名	是	否		长度最大为 63
file	文件位置	是	否		文件位置

Enroll 学生课程关系

字段	描述	是否必须	是否唯一	默认值	备注
id	关联主键	是	是		主键
course_cid	课程主键	是	否		外键
enroll_type	关系类型	是	否	1	1 学生，2 助教，3 老师
user_gid	用户 gid	是	否		外键

History 历史

字段	描述	是否必须	是否唯一	默认值	备注
id	历史主键	是	是		主键
root_cid	所属问题主键	是	否		外键
qid	历史问题主键	是	否		外键
student_aid	历史学生回答主键	是	否		外键
teacher_aid	历史教师回答主键	是	否		外键

HistoryQuestion 历史问题

字段	描述	是否必须	是否唯一	默认值	备注
id	历史问题主键	是	是		主键
title	历史问题标题	是	否		长度最大为 127
content	历史问题内容	是	否		Text
author_gid	作者 gid	是	否		外键

HistoryAnswer 历史回答

字段	描述	是否必须	是否唯一	默认值	备注
id	历史回答主键	是	是		主键
content	历史回答内容	是	否		Text
author_gid	作者 gid	是	否		外键

Tag 标签

字段	描述	是否必须	是否唯一	默认值	备注
id	标签主键	是	是		主键
name	标签名	是	是		长度最大为 20

QuestionTagTable 问题标签关系

字段	描述	是否必须	是否唯一	默认值	备注
tag_id	标签主键	是	是		外键
question_id	问题主键	是	否		外键

QuestionUpVote 问题点赞

字段	描述	是否必须	是否唯一	默认值	备注
id	投票主键	是	是		主键
question_id	问题主键	是	否		外键
author_gid	作者 gid	是	否		外键

模块 API

获取课程所有问题

请求方法：GET

请求地址：/v1/courses/<int:cid>/questions

请求参数：

```
[
    {
        "id": Number,
        "stars": Number,
        "title": String,
        "tags": [
            {
                "id": Number,
                "name": String
            }
        ],
        "author": {
            "auth": String,
            "email": String,
            "gid": String,
            "nickname": String
        }
        "content": String,
        "create_datetime": String,
        "update_datetime": String,
        "teacher_answer": {
            "id": Number,
            "content": String,
            "author": String,
            "update_datetime": String,
        },
        "student_answer": {
            "id": Number,
            "content": String,
            "author": String,
            "update_datetime": String,
        },
```

```
            "history": [
                {
                    "question": {
                        "id": Number,
                        "title": String,
                        "content": String,
                        "author": {}
                    },
                    "student_answer": {},
                    "teacher_answer": {}
                }
            ]
        }
    ]
```

查询单个问题

请求方法：GET

请求地址：/v1/questions/<int:id>

请求参数：

```
    {
        "id": Number,
        "stars": Number,
        "title": String,
        "tags": [
            {
                "id": Number,
                "name": String
            }
        ],
        "author": {
            "auth": String,
            "email": String,
            "gid": String,
            "nickname": String
        }
        "content": String,
        "create_datetime": DateTime,
        "update_datetime": DateTime,
        "teacher_answer": {
            "id": Number,
            "content": String,
            "author": {},
            "update_datetime": DateTime,
        },
```

```
        "student_answer": {
            "id": Number,
            "content": String,
            "author": {},
            "update_datetime": DateTime,
        },
        "history": [
            {
                "question": {
                    "id": Number,
                    "title": String,
                    "content": String,
                    "author": {}
                },
                "student_answer": {},
                "teacher_answer": {}
            }
        ]
    }
```

查询讨论主题

请求方法：GET

请求地址：/v1/questions/<int:qid>/discussions

成功返回：

```
    [
        {
            "id": Number,
            "question_id": Number,
            "title": String,
            "content": String,
            "stars": Number,
            "author_gid": String
        }
    ]
```

查询讨论主题的所有回帖

请求方法：GET

请求地址：/v1/discussions/<int:did>/answer

成功返回：

```
    [
        {
            "id": Number,
            "title": String,
            "content": String,
```

```
            "author_gid": String,
            "reply_id": Number
        }
    ]
```

修改讨论主题
请求方法：PUT
请求地址：/v1/discussions/<int:did>
请求参数：

```
    {
        "title": String,
        "content": String
    }
```

删除讨论主题
请求方法：DELETE
请求地址：/v1/discussions/<int:did>

修改讨论主题
请求方法：PUT
请求地址：/v1/discussions/<int:did>
请求参数：

```
    {
        "title": String,
        "content": String
    }
```

创建问题的讨论主题
请求方法：POST
请求地址：/v1/questions/<int:qid>/discussions
请求参数：

```
    {
        "title": String,
        "content": String
    }
```

修改答案内容
请求方法：PUT
请求地址：/v1/answers/<int:aid>
请求参数：

```
    {
        "content": String
    }
```

修改问题

请求方法：PUT

请求地址：/v1/question/<int:qid>

请求参数：

```
{
    "title": String,
    "content": String,
    "new_tags": [String],
    "del_tags": [String]
}
```

新建问题

请求方法：POST

请求地址：/v1/courses/<int:cid>/questions

请求参数：

```
{
    "title": String,
    "content": String,
    "tags": [String]
}
```

删除问题

请求方法：DELETE

请求地址：/v1/questions/<int:qid>

为讨论主题新增回帖

请求方法：POST

请求地址：/v1/discussions/<int:did>/answer

请求参数：

```
{
    "title": String,
    "content": String
}
```

6.3.2 收藏模块

模块架构

该模块负责完成用户对问题对收藏。

模块 API

收藏/取消收藏

请求方法：POST

请求地址：/v1/questions/<int:qid>/like

6.4 后台管理系统

6.4.1 选课管理模块

模块架构

该模块负责处理选课信息，管理员通过该模块修改选课的学生，以及该课程的授课教师和助教。教师也可管理选修自己课程的学生。

模块 API

教师对某一课程增加学生

请求方法：POST

请求地址：/v1/courses/<cid>/students

请求参数：

```
{
    "id": Number
}
```

教师对某一课程增加助教

请求方法：POST

请求地址：/v1/courses/<cid>/tas

请求参数：

```
{
    "id": Number
}
```

修改课程内容

请求方法：PUT

请求地址：/v1/courses/<int:cid>

请求参数：

```
{
    "name_zh": String,
    "name_en": String,
    "intro": String,
    "pre_course": String,
    "series": String,
    "textbooks": String,
    "semester": String,
    "new_teachers_gid": [String],
    "new_students_gid": [String],
    "TAs_gid": [String],
    "del_teachers_gid": [String],
    "del_students_gid": [String],
    "del_TAs_gid": [String]
}
```

6.4.2 主页信息管理模块

模块架构

该模块负责对课程主页的各种信息进行修改，由课程，公告，日历，作业等模块中负责对相应信息进行增加，删除修改的 API 构成。

模块 API

开设新课程

请求方法：POST

请求地址：/v1/courses

请求参数：

```
{
    "name_zh": String,
    "name_en": String,
    "intro": String,
    "pre_course": String,
    "series": String,
    "textbooks": String,
    "semester": String,
    "teachers_gid": [String],
}
```

删除课程

请求方法：DELETE

请求地址：/v1/courses

修改某个课程

请求方法：PUT 或者 PATCH

请求地址：/v1/courses/<cid>

请求参数：

```
{
    "name_zh": String,
    "name_en": String,
    "intro": String,
    "series": String,
    "pre_course": String,
    "textbooks": String,
    "semester": String,
    "new_teachers_gid": [],
    "new_students_gid": [],
    "new_TAs_gid": [],
    "del_teachers_gid": [],
    "del_students_gid": [],
```

```
        "del_TAs_gid": [],
    }
```

创建课程日历

请求方法：POST

请求地址：/v1/courses/<cid>/schedules

请求参数：

```
    {
        "week_id": Number,
    }
```

对创建的日历添加 Lecture

请求方法：POST

请求地址：/v1/schedules/<sid>/lectures

请求参数：

```
    {
        "lectures": [{
            "title": String,
            "resource_ids": [String],
            "assignment_ids": [String]
        }]
    }
```

教师添加课程公告

请求方法：POST

请求地址：/v1/courses/<cid>/announces

请求参数：

```
    {
        "announce": {
            "title": String,
            "content": String
        }
    }
```

教师删除课程公告

请求方法：DELETE

请求地址：/v1/announces/id

教师修改课程公告

请求方法：PUT

请求地址：/v1/announces/id

请求参数：

```
{
    "title": String,
    "content": String
}
```

教师对某一课程增加学生

请求方法：POST

请求地址：/v1/courses/<cid>/students

请求参数：

```
{
    "id": Number
}
```

教师对某一课程增加助教

请求方法：POST

请求地址：/v1/courses/<cid>/tas

请求参数：

```
{
    "id": Number
}
```

教师创建作业

请求方法：POST

请求地址：/v1/schedules/<int:sid>/assignments

请求参数：

```
{
    "title": string,
    "due": Number
}
```

获取全部老师信息

请求方法：GET

请求地址：/v1/users/teachers

成功返回：

```
[
    {
        "gid": Number
        "name": String,
        "email": String,
        "phone": String,
        "school": String,
    }
]
```

修改某个老师信息

请求方法：PUT

请求地址：/v1/users/<gid>

请求参数：

```
{
    "name_zh": String,
    "name_en": String,
    "email": String,
    "phone": String,
    "school": String,
}
```

注册新老师

请求方法：POST

请求地址：/v1/teachers

请求参数：

```
[
    {
        "name": String,
        "email": String,
        "phone": String,
        "school": String,
    }
]
```

删除教师

请求方法：DELETE

请求地址：/v1/teachers

修改某个老师信息

请求方法：PUT

请求地址：/v1/users/<gid>

请求参数：

```
{
    "name_zh": String,
    "name_en": String,
    "email": String,
    "phone": String,
    "school": String,
}
```

获取某一课程对应的学生/助教列表

请求方法：GET

请求地址：/v1/courses/<cid>/students 或/v1/courses/<cid>/tas

请求参数：

```
{
    "ids": [String]
}
```

成功返回：

```
[
    {
        "id": String,
        "name_zh": String,
        "name_en": String,
        "email": String,
        "phone": String,
        "school": String,    // 学院
    }
]
```

注册新学生

请求方法：POST

请求地址：/v1/students

请求参数：

```
[
    {
        "gid": Number,
        "name": String,
        "email": String,
        "phone": String,
        "school": String,
    }
]
```

删除学生

请求方法：DELETE

请求地址：/v1/users/<gid>

修改某个学生信息

请求方法：PUT

请求地址：/v1/users/<gid>

请求参数：

```
{
    "name": String,
    "email": String,
    "phone": String,
    "school": String,
}
```

7 部署和维护

7.1 部署

7.2 整体部署

整个项目的整体部署架构如下图所示。

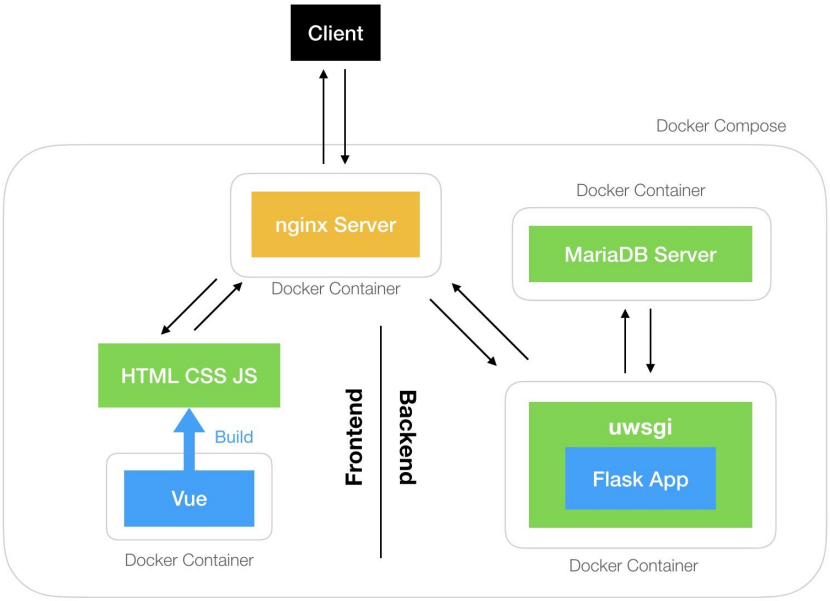

图 15: 整体部署图示

整个项目由前端，后端，数据库三个模块组成。前端基于 Vue.js 框架将生成网页文件，后端基于 Flask 框架处理业务逻辑，数据库基于 MariaDB 进行数据持久化。前端，后端，和客户之间使用 nginx 进行负载均衡。

7.3 高可用性部署方案

对于课程主页及答疑系统，当使用该系统的学校、课程规模较小，人数较少时，其业务流量比较小，且该系统的业务逻辑比较简单，故单台服务器便可以满足基本的需求；但随着使用该系统的学校、学生增加，业务流量越来越大，并且随着业务的拓展，业务逻辑也越来越复杂，单台机器的性能问题以及单点问题凸显了出来，因此需要多台机器来进行性能的水平扩展以及避免单点故障。

为了将不同的用户的流量分发到不同的服务器上面，使用 nginx 服务器进行负载均衡，如下图所示。

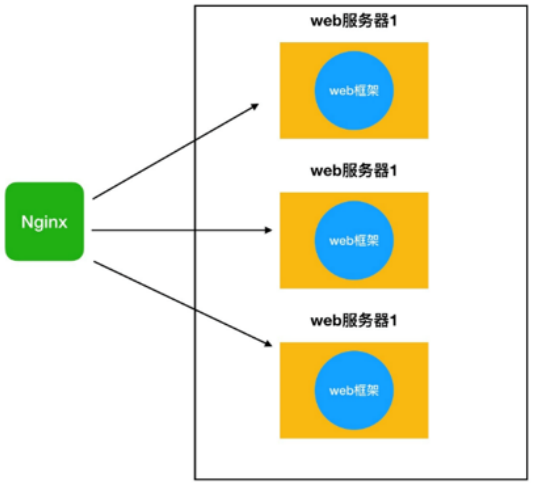

图 16: nginx 负载均衡图示

nginx 服务器负责前端模块，后端模块，以及客户之间的负载均衡。此时，客户端的流量首先会到达 nginx 负载均衡服务器，由负载均衡服务器通过一定的调度算法将流量分发到不同的应用服务器上面，同时负载均衡服务器也会对应用服务器做周期性的健康检查，当发现故障节点时便动态的将节点从应用服务器集群中剔除，以此来保证应用的高可用。

7.4 高存储可靠性部署方案

随着项目运营时间的增长，用户规模的增加和业务复杂度的增大，数据库中数据量会随之增大，数据出现丢失、损坏的可能性也随之增加。实际应用中，为了防止数据出现意外，应采用多备份数据服务器的架构。为了提高存储可靠性，保证数据的安全，本项目在实际部署中采用如下图所示的存储部署方案。

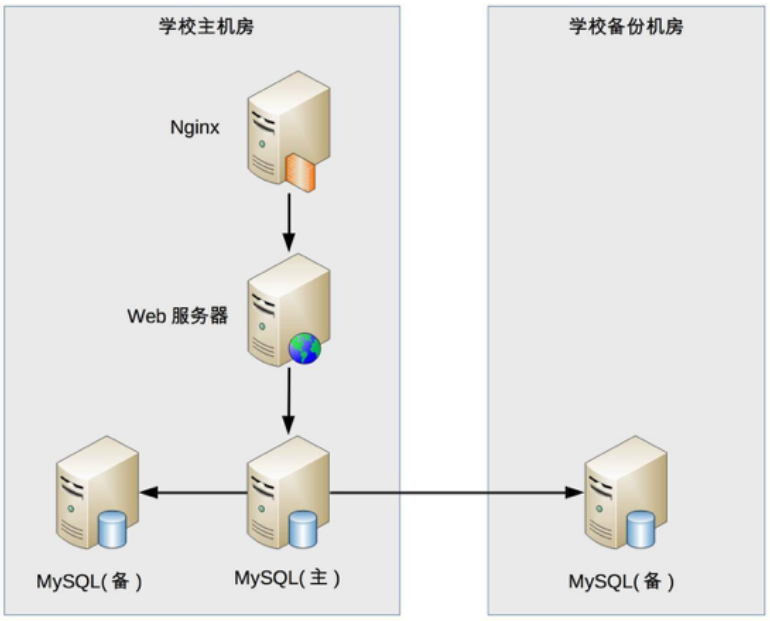

图 17: 多备份数据服务器

如图所示，本项目在本地机房中部署了两处 MySQL 服务，一处作为主服务，另一处作为备份服务。

此外，还在其他机房部署了一处远程 MySQL 服务。系统正常运行时，后端服务器和主 MySQL 服务进行交互。当主数据库服务发生意外时，由于同一机房中存在着备份的数据库服务，则系统可以轻松地切换至同一机房中的备份服务，不产生过大的开销。当主机房发生停电、火灾等意外，导致主机房中的服务全部失效时，我们还可以使用异地备份的数据库服务，用作后续的数据恢复。这样以来，我们最大程度地保证了数据的安全和数据存储的高可靠性。

7.5 维护

本部分是后端程序的维护说明。为了在软件已经交付使用之后，改正错误或者满足新的需要而修改软件，我们需要进行改正性维护，适应性维护，完善性维护等。软件维护是软件生命周期的最后一个阶段，需要的工作量非常大，编写维护软件的文档十分重要。本文档旨在给软件维护人员提供了一份完整，清晰的说明文档，便于其快速有效地进行维护工作。

7.5.1 后端维护准则

后端程序变更的准则

- 不降低程序质量。
- 检查可供选择的设计方案，寻找一种与程序的原始设计原理相容的变更设计。
- 努力使设计简化。
- 能满足可变性要求的设计。
- 用可测试的并具备测试方法的术语描述设计。
- 考虑处理时间，存储量和操作过程方面的变化。
- 考虑标更对用户服务的干扰以及实施变更的代价与时间。

维护中修改代码的准则

- 不要做不必要的修改。
- 不影响原始程序的风格和相容性。
- 记录所作过的修改。
- 审查软件质量是否符合标准。
- 更新程序文档以反映修改并保留修改前的程序代码版本。

重新验证程序的准则

- 首先测试程序的未改动部分，最后测试程序的修改部分。
- 不允许做修改的维护程序员成为唯一的重新验证程序的人。
- 鼓励终端用户参与到重新测试进程中来。
- 在重新验证进程中，记录出错的次数与类型，并把结果同所提供的测试功能进行比较，以便估量出程序是否退化。

7.5.2 后端维护流程

维护流程同开发流程一样，应该遵循如下开发流程规范。

图 18: 维护流程图示

每当用户报告错误，或有新功能需要增加时，开发人员按照根据需要对已有代码进行错误排查或新功能的设计，并根据以上后端维护准则进行代码开发。开发过程中应另外建立分支，及时更新文档，并做好程序的版本维护。后端开发人员在本地开发完成后，先在本地开发环境进行自测，然后将程序同步到测试环境和前端进行联调，再进行灰度，确认无误后完全同步到生产环境。

7.5.3 后端维护验证过程

每当软件被修改后，都要校验其正确性。维护员应该有选择地作些重新测试工作，不仅要证实新的逻辑的正确性，而且要校验实程序的为修改部分是否无损害，并且整个程序运行正确。若发现错误，则要马上进行修正。