同济大学
TONGJI UNIVERSITY

实验教学管理系统

设计规约（说明书）

小组成员

1951107 毛健羽

1951098 赵兰德龙

1754100 赵中原

2021.12.30

目录

1.引言 3
1.1 概要设计依据 3
1.2 参考资料 3
1.3 假定和约束 3
2 概要设计 3
2.1 总体系统架构设计 3
2.1.1 系统总体架构图 3
2.1.2 系统总体架构说明 4
2.2 系统软件结构设计 4
2.2.1 系统技术架构图 4
2.2.2 系统技术架构说明 5
2.3 接口设计 6
2.3.1 全局约定 6
2.3.2 用户接口设计 6
2.3.3 课程相关接口设计 15
2.3.4 公告相关接口设计 32
2.3.5 成绩相关接口设计 35
2.4 界面设计 36
2.4.1 界面原型设计 36
2.4.2 登陆页面设计 37
2.4.3 注册页面设计 38
2.4.4 主页设计 39
2.4.5 课程页面设计 39
2.4.6 用户信息页面设计 44
2.4.7 账号信息页面设计 44
2.4.8 公告页面设计 45
2.4.9 成绩查询页面设计 45
2.5 数据库设计 46
2.6 系统出错设计 47
2.6.1 出错信息 48
2.6.2 补救措施 48

1.引言

1.1 概要设计依据

本系统的设计基于系统需求规约，详见《实验教学管理系统需求规约（说明书）》。

1.2 参考资料

[1]窦万峰.软件工程方法与实践[M].北京：机械工业出版社，2016.10
[2]窦万峰.软件工程实验教程[M].北京：机械工业出版社，2016.11
[3]Robert C. Martin.敏捷软件开发：原则、模式与实践[M] 北京：清华大学出版，2003.09
[4]需求概要设计文档格式标准[S].GB856D-1988.
[5]中国大学慕课清华大学课程《软件工程》

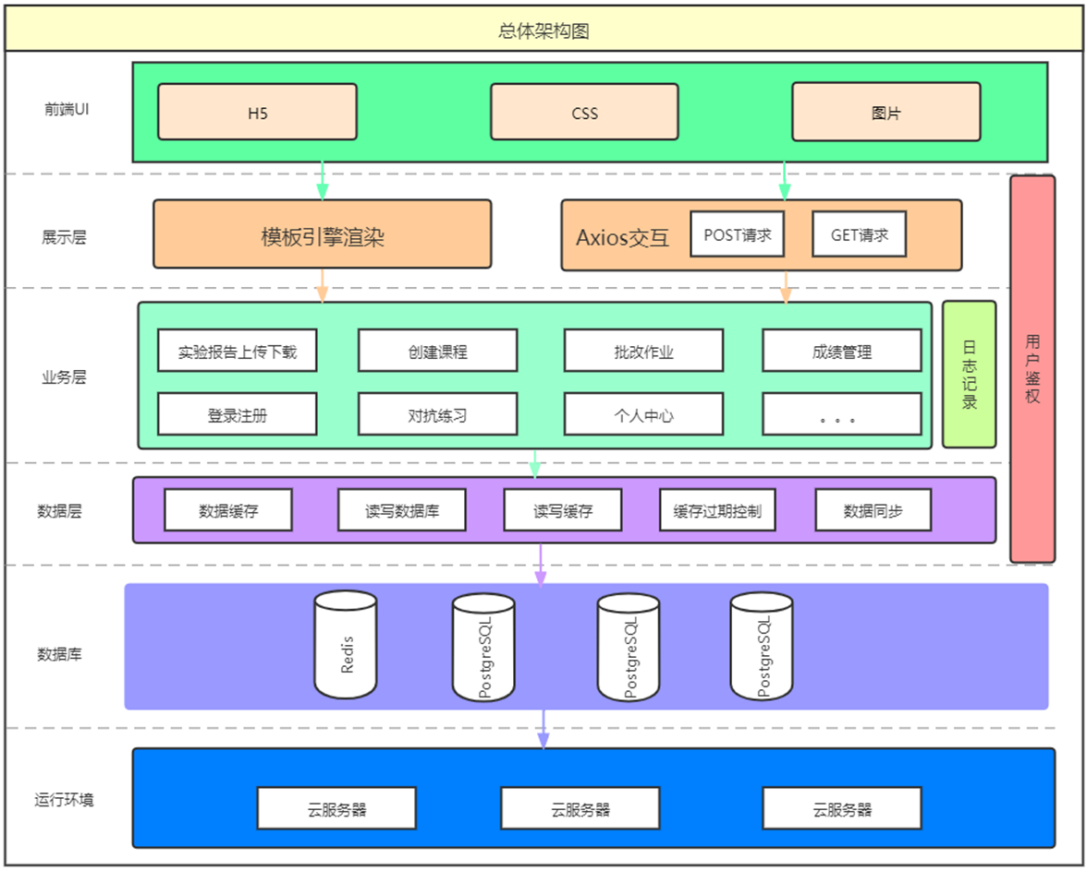
1.3 假定和约束

1. 开发周期约束：本项目的开发周期约为一个月
2. 规模约束：本项目的规模偏小，用户数量约为 2000 人
3. 资金约束：本项目的开发过程不考虑资金问题
4. 编程语言约束：本项目主要使用 HTML、CSS、JS 语言编写
5. 工具约束：本项目主要的开发工具为 VCode，主要的测试工具为 Postman
6. 性能约束：本项目的对抗练习模块具有潜在的高并发需求，需要尽量减少用户的等待时间，同时，要解决可能因为服务器响应速度慢而带来的不良影响与不公平现象。
7. 用户注册要求采用电子邮箱的方式
8. 用户基本信息由管理员提前导入，用户只执行激活流程
9. 责任教师是独立于教师的角色
10. 责任教师提前导入所有的课程以及课程教师和课程学生信息

2 概要设计

2.1 总体系统架构设计

2.1.1 系统总体架构图

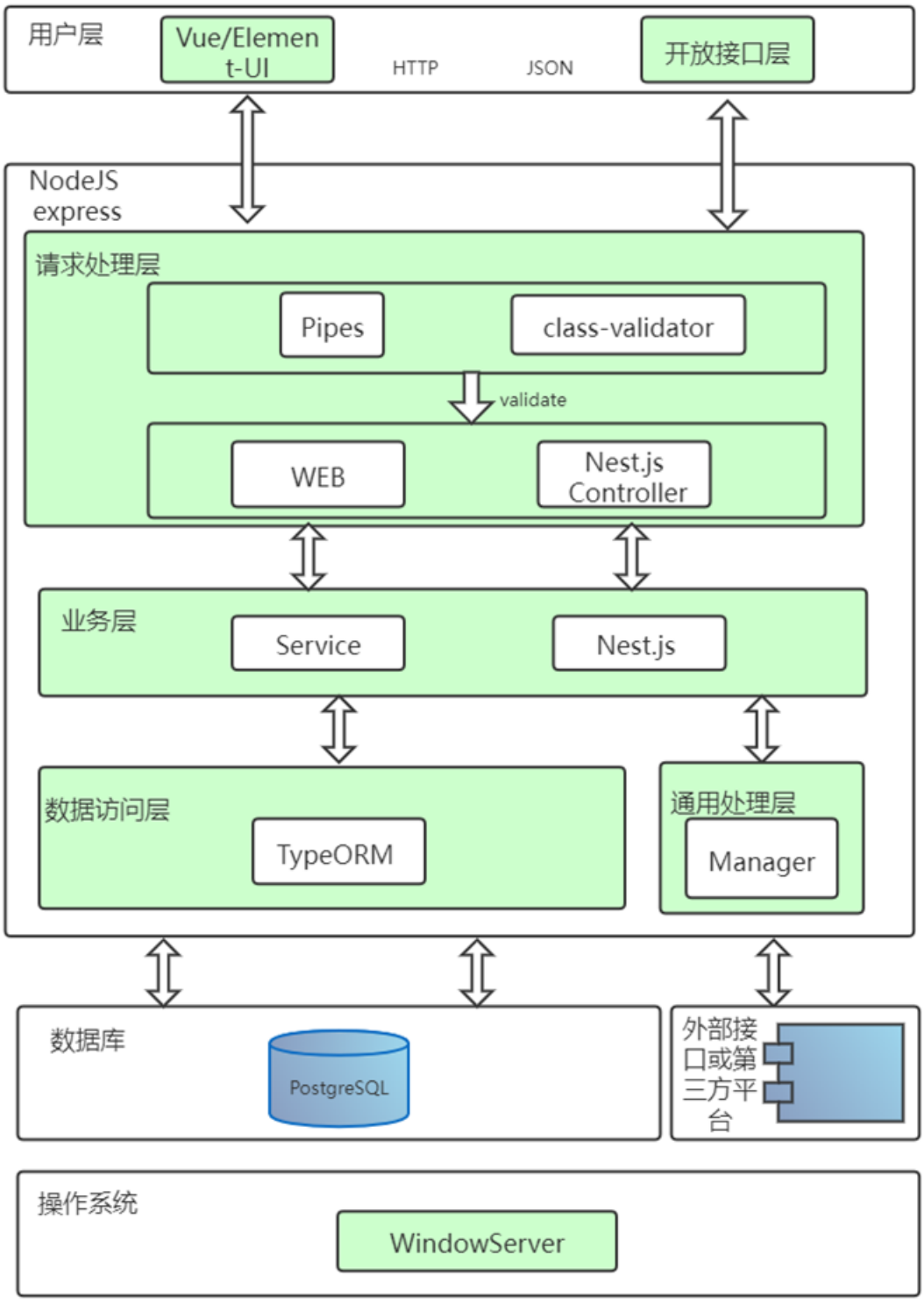


2.1.2 系统总体架构说明

我们的系统总体架构主要分为一下几个层次：前端 UI，展示层，业务层，数据层，数据库与运行环境。其中，前端 UI 是用户与系统交互的基础，包括前端的页面与各种展示信息以及可交互元素，这一层的主要编程语言是 HTML 与 CSS，用户从前端 UI 获取信息，与页面中的可交互元素交互。在前端 UI 下方是展示层，这一层主要由 Vue.js 组成，Vue 的主要功能在于数据绑定，它可以以数据绑定的方式动态地渲染数据，改变 UI 的样式，控制交互逻辑。同时数据绑定可以用于获取用户输入的信息，在这一层通过 Axios 库将信息发送给后端，也可以从后端获取数据用于动态渲染。因此，前端 UI 和展示层就是本系统的“前端”部分。再往下便是业务层，这一层囊括了数据处理与交互的所有逻辑，向上接受前端传递的数据和请求，在处理请求之后返回相应的结果；向下与数据库进行交互，包括但不限于对数据库数据的增删查改。这一层的业务按照功能模块进行了分类，通过接口路由暴露给前端访问，从而实现 API 的有效管理。数据层是沟通数据库与业务层的桥梁，在我们的系统中主要表现为 ORM 提供的数据库查询组件和相应的方法，业务层中不会手动写 sql 语句，而是调用 ORM 的方法进行数据操作，这样可以在编写接口的过程中始终保留面向对象的编写方法。数据层的下方是数据库，本系统中是 Postgre 数据库。最下层是整个系统的运行环境，即系统部署和运行的服务器，本项目中是 Windows10 平台。

2.2 系统软件结构设计

2.2.1 系统技术架构图



2.2.2 系统技术架构说明

在用户层，我们所采用的技术主要是 Vue+Element UI，Vue 的主要功能在于数据绑定，它可以以数据绑定的方式动态地渲染数据，改变 UI 的样式，控制交互逻辑。同时数据绑定可以用于获取用户输入的信息，在这一层通过 Axios 库将信息发送给后端，也可以从后端获取数据用于动态渲染。Element UI 是一套成熟的、高可用度的 UI 组件库，

可以帮助我们快速地搭建我们的前端页面，同时保证前端页面的交互与展示水平，Vue+Element UI 这样的组合方式是一套受到广泛欢迎和高度评价的实践之一。同时，我们使用 Axios 作为请求发送技术，Axios 是一套封装良好的 js 请求库，其优势在于良好的结构化和优秀的全局设置能力。接下来是请求处理层，我们没有使用常用的 SpringBoot 作为这一层的技术栈，而是选择了 Nest.js 作为我们的开发技术，在技术栈的选取上我们是有所考量的，Nest.js 主要使用 JavaScript 和 TypeScript 语言来进行开发，它运行在 node.js 的基础上，拥有详实的参考文档、活跃的社区和丰富的插件，这些构成了一个可快速迭代的、灵活的、高拓展性的开发平台，同时它拥有出色的编译速度，可以给开发者良好的开发体验和完备的支持。如图中所示，我们使用了 Pipe（管道）和 class-validator（类验证器）来充当前端请求数据过滤与预处理机制，使用 JWT 生成 token 保护接口和减轻服务端压力。在数据访问层，我们使用的是 TypeORM，它可以让我们在编写业务逻辑时始终保持面向对象，而不需要考虑数据库的操作。数据库方面我们使用的是 Postgre，PostgreSQL 是一种特性非常齐全的自由软件的对象-关系型数据库管理系统，他可以更好地与 ORM 合作，提供更好的开发体验。最后，我们的运行环境为 Windows 平台。

2.3 接口设计

2.3.1 全局约定

在编写接口时，我们对接口的状态码进行了规定，以更方便地区分相应的结果。

状态码	说明
200	成功
400	无法识别的接口
401	未授权
500	服务端错误

2.3.2 用户接口设计

接口路由：/users

---

login

简要描述
- 登陆

请求 URL
- http://localhost:3000/auth/login

请求方式
- post

Header

header	必选	类型	说明
Token	否	string	token 字符串，用于验证

请求 Body 参数

参数名	必选	类型	说明
emailOrJobNumber	是	string	邮箱地址或学工号
password	是	string	密码

成功返回示例

```
JSON
{
   "token":
"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJqb2JOdW1iZXIiOiIxOTUxMDk4Iiwic3ViIjoxNywiaWF0IjoxNjQwNzQzNzI3LCJleHAiOjEwMjgwNjU3MzI3fQ.qjSpS8na1JBzTHpECoFwRoDNcYdXui-wlHD-TAkamVc"
}
```

成功返回示例的参数说明

参数名	类型	说明
token	string	token 字符串

失败返回示例

```
JSON
{
   "statusCode": 401,
   "message": "用户名或密码错误",
   "error": "Unauthorized"
}
```

失败返回示例的参数说明

参数名	类型	说明
statusCode	string	错误码
message	string	信息
error	string	错误说明

备注

userEnable

简要描述

- 用户通过邮箱激活账号

请求 URL

- http://localhost:3000/auth/enable

请求方式

- post

Header

header	必选	类型	说明
Token	否	string	token 字符串，用于验证

请求 Body 参数

参数名	必选	类型	说明
email	是	string	激活邮箱
password	是	string	密码
jobNumber	是	string	学工号

成功返回示例

```
Plain Text
{“邮件已发出”}
```

失败返回示例

```
JSON
{
  "statusCode": 400,
  "message": "用户信息未录入",
  "error": "Bad Request"
}
```

失败返回示例的参数说明

参数名	类型	说明
statusCode	string	状态码
message	string	消息
error	string	错误类型

备注

getUserByToken

简要描述

- 根据 token 获取 user 信息

请求 URL

- http://localhost:3000/users/userByToken

请求方式

- post

Header

header	必选	类型	说明
Token	否	string	token 字符串，用于验证

成功返回示例

JSON


```
{
    "id": 17,
    "email": "1951098@tongji.edu.cn",
    "enablingEmail": "1951098@tongji.edu.cn",
    "name": "赵兰德龙",
    "password": "1114514",
    "enablingPassword": "1114514",
    "subject": "",
    "description": "",
    "avatar": "",
    "age": 18,
    "jobNumber": "1951098",
    "grade": 2019,
    "enabled": true,
    "createdTime": "2021-11-24T09:46:03.861Z",
    "type": "student",
    "gender": "unknown"
}
```


成功返回示例的参数说明

参数名	类型	说明
id	string	id
email	string	邮箱
enablingEmail	string	激活邮箱
name	string	姓名

password	string	密码
subject	string	专业
description	string	基本描述
avatar	string	用户头像
age	string	年龄
jobNumber	string	学工号
grade	string	年级
enabled	string	是否激活
createdTime	string	创建时间
type	string	用户类型
gender	string	性别

失败返回示例

```
JSON
{
   "statusCode": 401,
   "message": "未找到 token 对应的用户"
}
```

失败返回示例的参数说明

参数名	类型	说明
statusCode	string	状态码
message	string	消息

备注

---

userCreate

简要描述

- 用于提前录入用户的信息

请求 URL

- http://localhost:3000/users/create

请求方式

- post

Header

header	必选	类型	说明
Token	否	string	token 字符串，用于验证

请求 Body 参数

参数名	必选	类型	说明
name	是	string	真实姓名
jobNumber	是	int	学工号
gender	是	string	性别
type	是	string	用户类型
grade	否	string	年级（学生）
subject	是	string	专业
age	否	string	年龄

成功返回示例

JSON


```
{
    "jobNumber": "1919190",
    "name": "",
    "id": 18,
    "email": "",
    "enablingEmail": "",
    "password": "",
    "enablingPassword": "",
    "subject": "",
    "description": "",
    "avatar": "",
    "age": 18,
    "grade": 2019,
    "enabled": false,
    "createdTime": "2021-12-29T02:08:04.312Z",
    "type": "student",
    "gender": "unknown"
}
```


成功返回示例的参数说明

参数名	类型	说明
jobNumber	string	学工号
name	string	姓名
id	string	用户 id
email	string	用户邮箱
enablingEmail	string	激活邮箱
password	string	密码
subject	string	专业
description	string	描述
avatar	string	头像
age	string	年龄
grade	string	年级
enabled	string	是否激活
createdTime	string	创建时间
type	string	用户类型
gender	string	性别

备注

---

userVerify

简要描述

- 用户邮箱验证

请求 URL

- http://localhost:3000/users/varify

请求方式

- post

Header

header	必选	类型	说明
Token	否	string	token 字符串，用于验证

请求 Body 参数

参数名	必选	类型	说明
userId	是	string	无

成功返回示例

```
Plain Text
邮件已发送
```

备注

---

editEmail

简要描述

- 用户换绑邮箱

请求 URL

- http://localhost:3000/users/email

请求方式

- put

Header

header	必选	类型	说明
Token	否	string	token 字符串，用于验证

请求 Body 参数

参数名	必选	类型	说明
userId	是	string	用户 id
email	是	string	新邮箱

成功返回示例

```
Plain Text
验证邮件已发出，请注意查收
```

备注

---

editPassword

简要描述

- 更改密码

请求 URL

- http://localhost:3000/users/password

请求方式

- put

Header

header	必选	类型	说明
Token	否	string	token 字符串，用于验证

请求 Body 参数

参数名	必选	类型	说明
uesrId	是	string	用户 id
password	是	string	新密码
oldPassword	是	string	旧密码

成功返回示例

```
Plain Text
更改成功
```

备注

editInfo

简要描述

- 无

请求 URL

- http://localhost:3000/users/info

请求方式

- put

Header

header	必选	类型	说明
Token	否	string	token 字符串，用于验证

请求 Body 参数

参数名	必选	类型	说明
userId	是	string	用户 id
avatar	是	string	头像 url
description	是	string	用户简介

成功返回示例

```
Plain Text
修改成功
```

备注

2.3.3 课程相关接口设计

接口路由：/course

courseCreate

简要描述

- 创建课程

请求 URL

- http://localhost:3000/course/create

请求方式

- post

Header

header	必选	类型	说明
Token	否	string	token 字符串，用于验证

请求 Body 参数

参数名	必选	类型	说明
name	是	string	课程名称
creatorId	是	string	创建者 id
startTime	是	string	课程开始时间
endTime	是	string	课程结束时间
teachers	是	array	教师 id 数组

students	是	array	学生 id 数组
weight	是	string	各个部分的权重

成功返回示例

```
Plain Text
创建成功
```

失败返回示例

```
JSON
{
   "statusCode": 400,
   "message": "创建失败",
   "error": "Bad Request"
}
```

备注

---

getCourseById

简要描述

- 根据 id 获取参与的课程列表

请求 URL

- http://localhost:3000/course?id=1

请求方式

- get

Header

header	必选	类型	说明
Token	否	string	token 字符串，用于验证

请求 Query 参数

参数名	必选	类型	说明
id	是	string	用户 id

成功返回示例

```
JSON
[
   {
      "name":"计算机网络",
      "courseId":1,
      "startTime":"",
      "endTime":""
```

```
    }
]
```

成功返回示例的参数说明

参数名	类型	说明
name	string	课程名称
courseId	string	创建者 id
startTime	string	开始时间
endTime	string	结束时间

失败返回示例

```
JSON
{
    "statusCode": 400,
    "message": "获取失败"
}
```

失败返回示例的参数说明

参数名	类型	说明
statusCode	string	状态码
message	string	信息

备注

---

getCourseDetail

简要描述

- 无

请求 URL

- http://localhost:3000/course/detail?courseId=1

请求方式

- get

Header

header	必选	类型	说明
Token	否	string	token 字符串，用于验证

请求 Query 参数

参数名	必选	类型	说明
courseId	是	string	课程 id

成功返回示例

```
JSON
{
  "name": "计算机网络",
  "creator": {},
  "teachers": [],
  "startTime": "2021-03-15T16:00:00.000Z",
  "endTime": "2021-06-30T16:00:00.000Z",
  "id": 24,
  "isCourseOpen": true,
  "createdTime": "2021-12-29T02:08:04.312Z"
}
```

成功返回示例的参数说明

参数名	类型	说明
name	string	课程名称
creator	object	课程创建者
teachers	array	教师（可能有多个）
startTime	string	开始时间
endTime	string	结束时间
id	string	课程 id
isCourseOpen	string	是否开放
createdTime	string	课程被创建的时间

备注

---

createPractice

简要描述

- 无

请求 URL

- http://localhost:3000/course/practice/create

请求方式
- post

Header

header	必选	类型	说明
Token	否	string	token 字符串，用于验证

请求 Body 参数

参数名	必选	类型	说明
practiceName	是	string	练习名称
timeLimitation	是	string	时间限制
questions	是	array	问题数组

备注

---

getProblems

简要描述
- 获取对抗练习的题目

请求 URL
- http://localhost:3000/course/practice?practiceId=

请求方式
- get

Header

header	必选	类型	说明
Token	否	string	token 字符串，用于验证

请求 Query 参数

参数名	必选	类型	说明
practiceId	是	string	对抗练习 id

成功返回示例

JSON
```
{
   "practiceId":0,
   "practiceName":"",
```

```
  "timeLimitation":1200,
  "questions":[
    {
      "problem":"",
      "problemType":0,
      "options":[]
    }
  ]
}
```


成功返回示例的参数说明

参数名	类型	说明
practiceId	string	练习 id
practiceName	string	练习名称
timeLimitation	string	时间限制
questions	array	所有题目
problem	string	题干
problemType	string	单选/多选
options	array	选项

备注

---

submit

简要描述

- 提交对抗练习答案

请求 URL

- http://localhost:3000/course/practice

请求方式

- post

Header

header	必选	类型	说明
Token	否	string	token 字符串，用于验证

请求 Body 参数

参数名	必选	类型	说明
userId	是	string	无
practiceId	是	string	无
answers	是	array	无

成功返回示例

```
Plain Text
提交成功
```

备注

---

uploadMaterial

简要描述

- 上传教学资料

请求 URL

- http://localhost:3000/course/material

请求方式

- post

Header

header	必选	类型	说明
Token	否	string	token 字符串，用于验证

请求 Body 参数

参数名	必选	类型	说明
userId	是	string	用户 id
dataType	是	string	教学资料的类别（课程资料，项目资料，对抗练习资料）
fileName	是	string	文件名称
fileBody	是	file	文件主体
courseId	是	string	课程 id

成功返回示例

```
Plain Text
上传成功
```

备注

---

getMaterials

简要描述

- 获取教学资料列表

请求 URL

- http://localhost:3000/course/material

请求方式

- get

Header

header	必选	类型	说明
Token	否	string	token 字符串，用于验证

成功返回示例

```
JSON
[
    {
        "fileName":"",
        "url":"",
        "uploadTime":"",
        "userName":""
    }
]
```

成功返回示例的参数说明

参数名	类型	说明
fileName	string	文件名称
url	string	文件 url
uploadTime	string	上传时间
userName	string	上传者名称

备注

---

getGrades

简要描述

- 学生查看本门课程的成绩情况

请求 URL

- http://localhost:3000/course/grade?userId=&courseId=

请求方式

- get

Header

header	必选	类型	说明
Token	否	string	token 字符串，用于验证

请求 Query 参数

参数名	必选	类型	说明
userId	是	string	用户 id
courseId	是	string	课程 id

成功返回示例

JSON

```
{
   "usual":{
      "score":0,
      "weight":0.2
   },
   "project":[
      {
         "projectId":0,
         "projectName":"",
         "score":0,
         "weight":0.1
      }
   ],
   "practice":{
      "score":0,
      "weight":0
   }
}
```

成功返回示例的参数说明

参数名	类型	说明
usual	object	平时成绩
score	string	得分

weight	string	权重
project	array	项目成绩（可能有多个）
projectId	string	项目 id
projectName	string	项目名称
practice	object	对抗练习成绩

备注

---

getAnnounceByCourseId

简要描述

- 根据课程 id 获取课程所有公告

请求 URL

- http://localhost:3000/course/announce?courseId=

请求方式

- get

Header

header	必选	类型	说明
Token	否	string	token 字符串，用于验证

请求 Query 参数

参数名	必选	类型	说明
courseId	是	string	课程 id

成功返回示例

```
JSON
[
  {
    "announceName":"",
    "time":"",
    "announceId":0
  }
]
```

成功返回示例的参数说明

参数名	类型	说明

announceName	string	公告标题
time	string	公告发布时间
announceId	string	公告 id

备注

---

getDetailByAnnounceId

简要描述

- 根据公告 id 获取公告详情

请求 URL

- http://localhost:3000/course/announce?announceId=

请求方式

- get

Header

header	必选	类型	说明
Token	否	string	token 字符串，用于验证

请求 Query 参数

参数名	必选	类型	说明
announceId	是	string	公告 id

成功返回示例

JSON

```
{
   "announceName":"",
   "time":"",
   "announceId":0,
   "content":"",
   "files":[
      {
         "fileName":"",
         "url":""
      }
   ]
}
```

成功返回示例的参数说明

参数名	类型	说明

announceName	string	公告名称
time	string	时间
announceId	string	公告 id
content	string	公告正文
files	array	公告附件
fileName	string	附件名称
url	string	附件 url

备注

---

createAnnounce

简要描述

- 创建公告

请求 URL

- http://localhost:3000/course/announce/create

请求方式

- post

Header

header	必选	类型	说明
Token	否	string	token 字符串，用于验证

请求 Body 参数

参数名	必选	类型	说明
userId	是	string	用户 id
title	是	string	公告标题
content	是	string	公告正文
files	否	array	公告附件（可以是多个或空）

成功返回示例

```
Plain Text
创建成功
```

备注

---

createProject

简要描述

- 无

请求 URL

- http://localhost:3000/course/project/create

请求方式

- post

Header

header	必选	类型	说明
Token	否	string	token 字符串，用于验证

请求 Body 参数

参数名	必选	类型	说明
projectName	是	string	项目名称
endTime	是	string	项目截止日期
description	是	string	项目要求
files	是	array	项目文件，内部是 file 类型数据

成功返回示例

```
Plain Text
创建成功
```

---

getProjectDetail

简要描述

- 无

请求 URL

- http://localhost:3000/course/project/detail?projectId=1

请求方式

- get

Header

header	必选	类型	说明
Token	否	string	token 字符串，用于验证

请求 Query 参数

参数名	必选	类型	说明
projectId	是	string	项目 id

成功返回示例

```
JSON
{
   "projectName":"",
   "projectId":0,
   "endTime":"",
   "desctiption":"",
   "files":[
      {
         "fileName":"",
         "url":""
      }
   ]
}
```

成功返回示例的参数说明

参数名	类型	说明
projectName	string	项目名称
projectId	string	项目 id
endTime	string	项目截止日期
desctiption	string	项目要求描述
files	array	项目文件
fileName	string	文件名称
url	string	文件 url

备注

getProjectListByCourseId

简要描述

- 根据课程 id 获取课程项目列表

请求 URL

- http://localhost:3000/course/project?courseId=1

请求方式

- get

Header

header	必选	类型	说明
Token	否	string	token 字符串，用于验证

请求 Query 参数

参数名	必选	类型	说明
courseId	是	string	课程 id

成功返回示例

```
JSON
[
  {
    "projectName":"",
    "projectId":0,
    "endTime":""
  }
]
```

成功返回示例的参数说明

参数名	类型	说明
projectName	string	项目名称
projectId	string	项目 id
endTime	string	项目截止时间

备注

---

markHomework

简要描述

• 为项目作业评分
请求 URL
• http://localhost:3000/course/project/homework
请求方式
• post
Header

header	必选	类型	说明
Token	否	string	token 字符串，用于验证

请求 Body 参数

参数名	必选	类型	说明
projectId	是	string	项目 id
uesrId	是	string	用户 id
score	是	int	评分

成功返回示例

```
Plain Text
评分已保存
```

备注

---

getHomeworks

简要描述
• 获取某一个项目下提交的作业
请求 URL
• http://localhost:3000/course/project/homework?projectId=
请求方式
• get
Header

header	必选	类型	说明
Token	否	string	token 字符串，用于验证

请求 Query 参数

参数名	必选	类型	说明

projectId	是	string	项目 id

成功返回示例

```
JSON
[
    {
        "userName":"",
        "jobNumber":"",
        "fileName":"",
        "url":"",
        "marked":true,
        "score":80
    }
]
```

成功返回示例的参数说明

参数名	类型	说明
userName	string	用户真实姓名
jobNumber	string	学工号
fileName	string	文件名称
url	string	文件 url
marked	string	是否已评分
score	string	得分

备注

---

submitHomework

简要描述

- 提交作业

请求 URL

- http://localhost:3000/course/project/submit

请求方式

- post

Header

header	必选	类型	说明
Token	否	string	token 字符串，用于验证

请求 Body 参数

参数名	必选	类型	说明
fileName	是	string	文件名称
fileBody	是	file	文件主体
projectId	是	string	项目 ID
userId	是	string	用户 id

成功返回示例

```
Plain Text
上传成功
```

备注

2.3.4 公告相关接口设计

接口路由：/announce

createAnnounce

简要描述

- 发布全局公告

请求 URL

- http://localhost:3000/announce

请求方式

- post

Header

header	必选	类型	说明
Token	否	string	token 字符串，用于验证

请求 Body 参数

参数名	必选	类型	说明
uesrId	是	string	发布者
title	是	string	标题

content	是	string	内容
files	是	array	附件

成功返回示例

```
Plain Text
发布成功
```

备注

---

getDetail

简要描述

- 获取公告细节

请求 URL

- http://localhost:3000/announce/detail?announceId=

请求方式

- get

Header

header	必选	类型	说明
Token	否	string	token 字符串，用于验证

请求 Query 参数

参数名	必选	类型	说明
announceId	是	string	公告 id

成功返回示例

```
JSON
{
  "announceName":"",
  "time":"",
  "announceId":0,
  "content":"",
  "files":[
    {
      "fileName":"",
      "url":""
    }
  ]
}
```

成功返回示例的参数说明

参数名	类型	说明
announceName	string	公告名称
time	string	时间
announceId	string	公告 id
content	string	公告正文
files	array	文件
fileName	string	文件名称
url	string	文件 url

备注

---

getAnnounce

简要描述

- 获取系统全局公告

请求 URL

- http://localhost:3000/announce

请求方式

- get

Header

header	必选	类型	说明
Token	否	string	token 字符串，用于验证

成功返回示例

```
JSON
[
  {
    "announceName":"",
    "time":"",
    "announceId":0
  }
]
```

成功返回示例的参数说明

参数名	类型	说明

announceName	string	公告名称
time	string	时间
announceId	string	公告 id

备注

2.3.5 成绩相关接口设计

接口路由：/grade

getGradesByCourseId

简要描述

- 根据课程 id 获取这门课所有人的成绩信息

请求 URL

- http://localhost:3000/grade?courseId=

请求方式

- get

Header

header	必选	类型	说明
Token	否	string	token 字符串，用于验证

请求 Query 参数

参数名	必选	类型	说明
courseId	是	string	课程 id

成功返回示例

```
JSON
[
   {
      "userName":"",
      "jobNumber":"",
      "grade":{
         "usual":80,
         "practice":80,
         "project":80,
         "overall":80
      }
   }
```

]

成功返回示例的参数说明

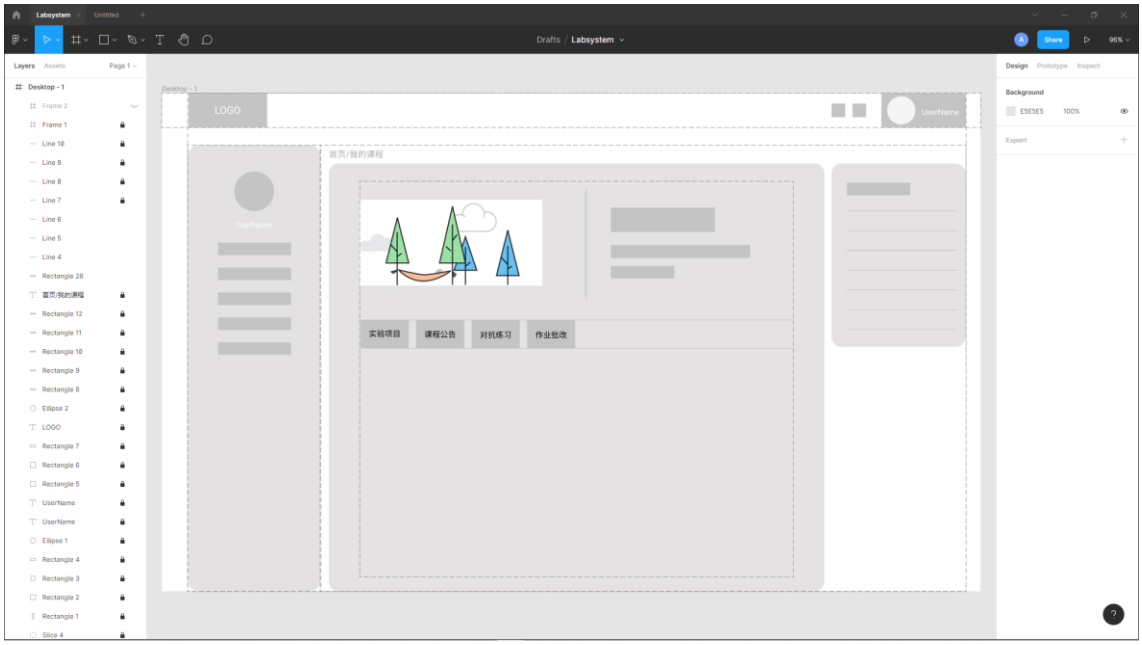
参数名	类型	说明
userName	string	学生姓名
jobNumber	string	学工号
grade	object	成绩
grade.usual	string	平时成绩
grade.practice	string	练习成绩
grade.project	string	项目成绩
grade.overall	string	总成绩

备注

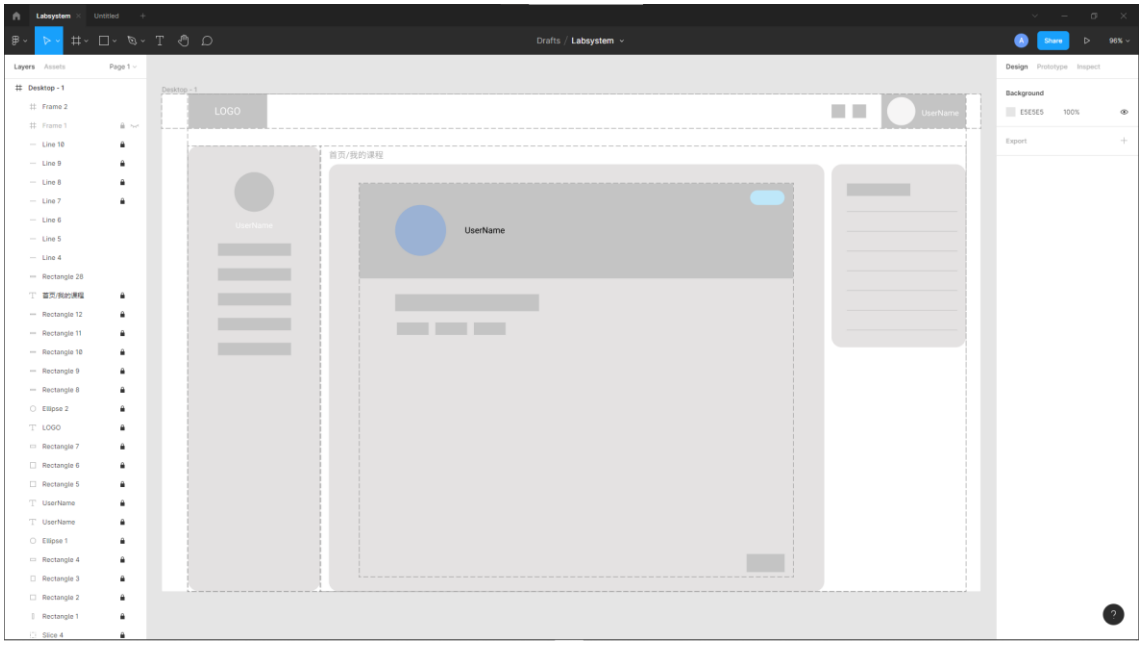
2.4 界面设计

2.4.1 界面原型设计

在进行了深入的需求分析之后，我们对页面进行了大致的规划，包括页面的布局，页面的风格以及页面的层次结构，我们对作为关键的两个部分：课程主页与个人信息主页进行了原型上的设计，通过模块化组件化的思想，细化页面层次布局，打造更加合理简洁的页面交互模式，争取构建出美观而实用的页面。



课程界面前端原型



用户中心前端原型

从上图中可以看出，本系统的原型分为以下几个部分：

头部导航栏

在系统页面的上方，包含系统 logo、基本的控制部件，包括消息提醒与设置，用户栏位。头部导航栏的主要职责是向用户展示实验管理系统最基本的信息，同时用户栏位的设置让用户清楚地了解这是自己的账户，从而明确系统范围，基本的控制部件可以方便用户的使用。

左侧侧边栏

在系统的左侧，包含用户头像，用户姓名，以及最重要的导航功能。同样地，用户栏位的设置让用户感受到这是服务于自己的页面，同时，侧边栏和头部导航栏中用户部分的占比不同，侧边栏中用户信息明显要比头部导航栏中的用户信息更能够吸引用户的视线，这样以来，信息的层级就有了主次之分，让用户的浏览动线更加清晰。导航栏是系统最重要的组成成分之一，导航栏的设计关系到整个系统页面的切换逻辑和使用逻辑。

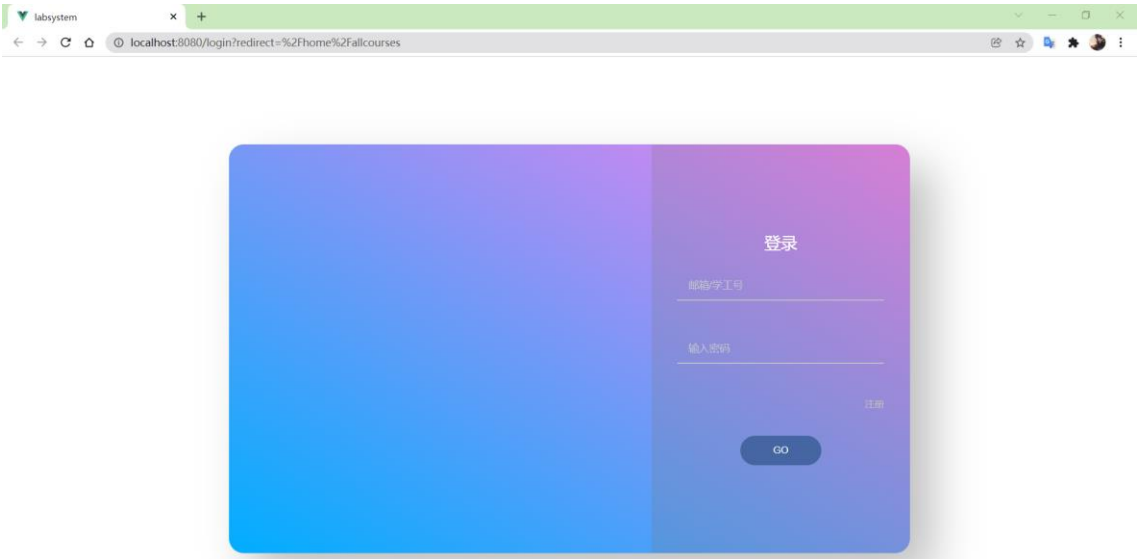
右侧栏位

在系统的右侧，它的高度与左侧的导航栏相比较小，可以很容易地看出，本栏位是一些信息的汇总补充，或者一些非主要需求的交互逻辑，把用户可能需要的信息放在这里，可以让系统的信息更加丰富，同时信息层级分明不杂乱。

内容主体

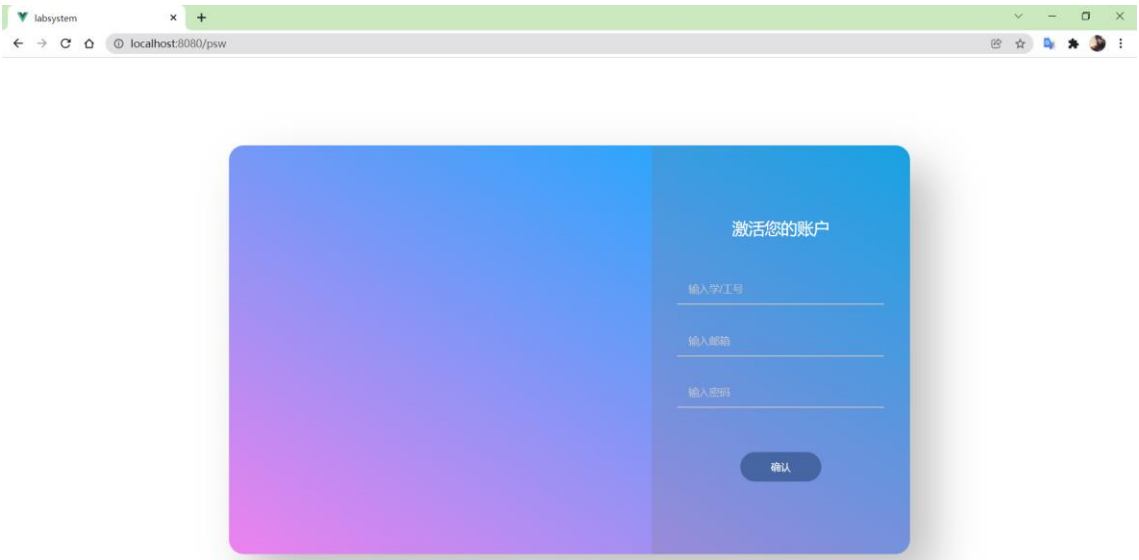
在系统页面的中心，也是所占面积最大的区块。这里是系统最主要展示的内容，课程、用户信息、账户信息、公告、成绩这些模块都展示在这个窗口之中，他们之前的切换逻辑和左侧的导航栏导航逻辑耦合，同时，在模块的内部也可能存在子页面，子页面之间的切换逻辑由模块内部的导航来决定，如课程界面的前端原型，模块内部还按照功能的区别划分了几个子模块，他们直接的切换由模块中的横向导航栏决定。

2.4.2 登陆页面设计



本项目中，登录页面是用户进入到系统见到的第一个页面，用户需要在本页面进行邮箱和密码的验证，登录-注册页面构成了整个页面逻辑的最上层，这两个页面是可以互相切换并且紧密联系的页面。登录的逻辑是多样的，登录逻辑的选取往往取决于系统的属性，如果系统是一个比较开放的系统，开发者希望所有访问的用户都能看到自己展示的信息，而又希望保护部分信息，那么可以采用先让用户进入首页，等用户希望查看被保护的信息时再引导用户登录。这一逻辑最大的优点在于极大地降低了网站访问的门槛。而对于教学管理系统这一类偏向管理的系统，信息往往是私密的，因此选择先登录在进入系统的方法，这样可以最大限度地保证系统的安全性和健壮性。

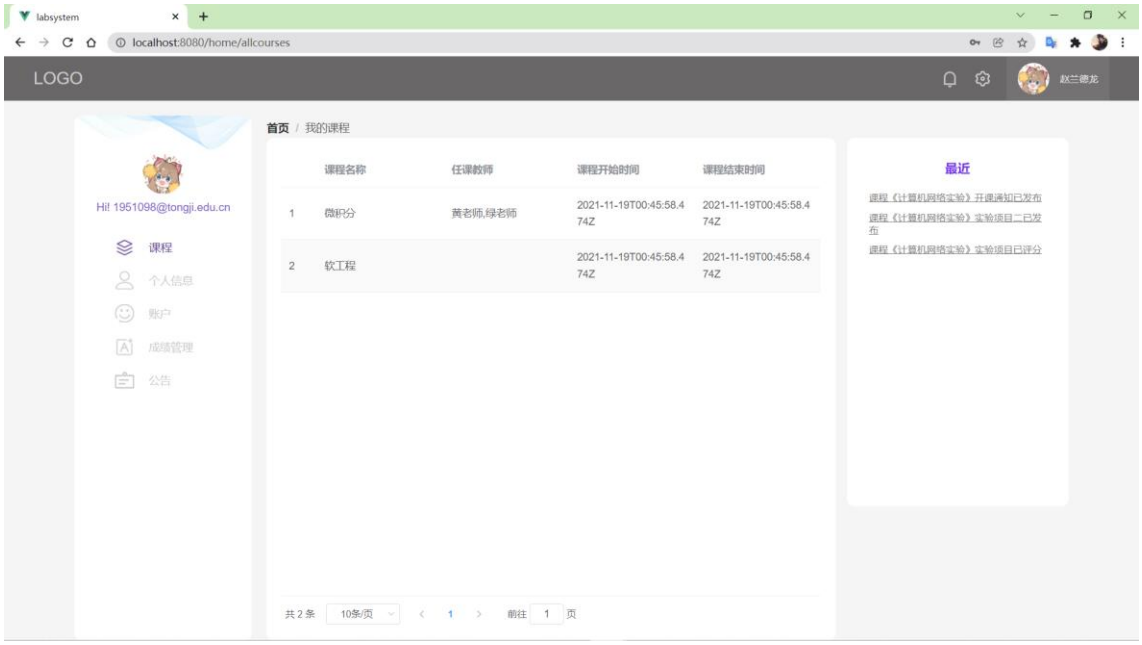
2.4.3 注册页面设计



本系统采用的注册方式为邮箱注册，因此需要用户输入邮箱，前面提到，登录和注册功能是紧密地联系在一起的，因此登录页面和注册页面往往采用相似的设计，从而提

高登录-注册功能模块的整体性与美观。在用户填写完成信息并且点击注册之后，系统会发送验证邮件，用户点击验证邮件即完成验证，注册成功，网页自动跳转进入系统主页。

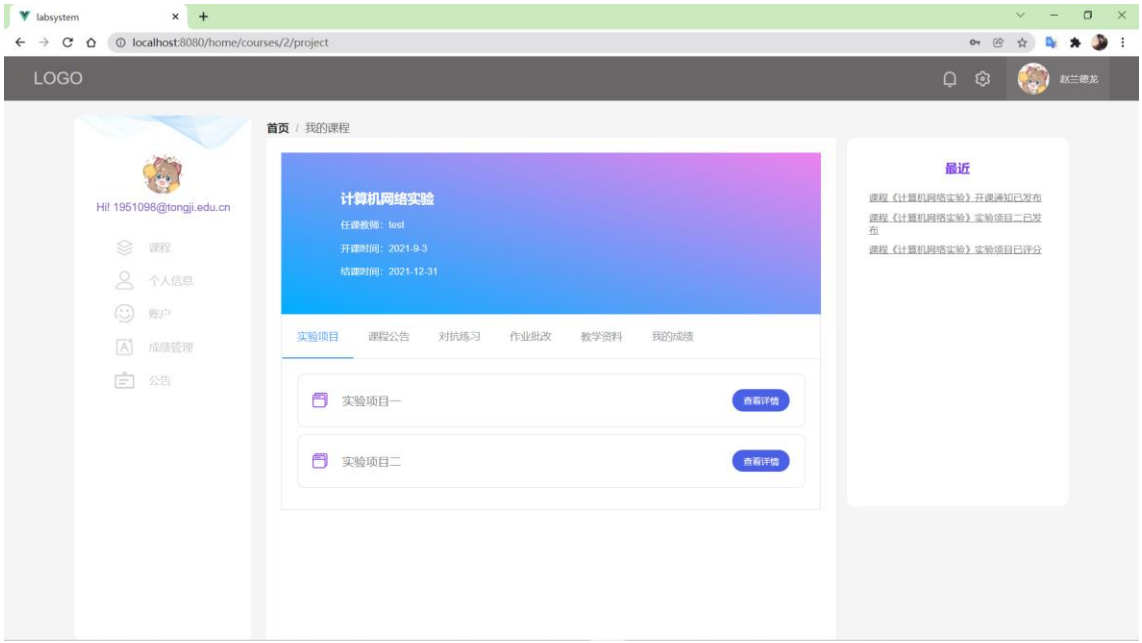
2.4.4 主页设计



本项目的主页设计按照前端原型的设计理念设计而成，在配色方面，本系统选择了蓝-紫色的渐变，希望在设计风格上能够更加地体现出活力和美感；在主页的布局上，合理地分配页面的空间，为页面划分出清晰的层次，包括头部导航栏，左侧侧边栏，右侧侧边栏和中间的主体内容，导航栏的格式为竖向横向交替，更合理地划分页面，让页面不会显得混乱，同时适当留白，不让页面的主体显得过于拥挤。

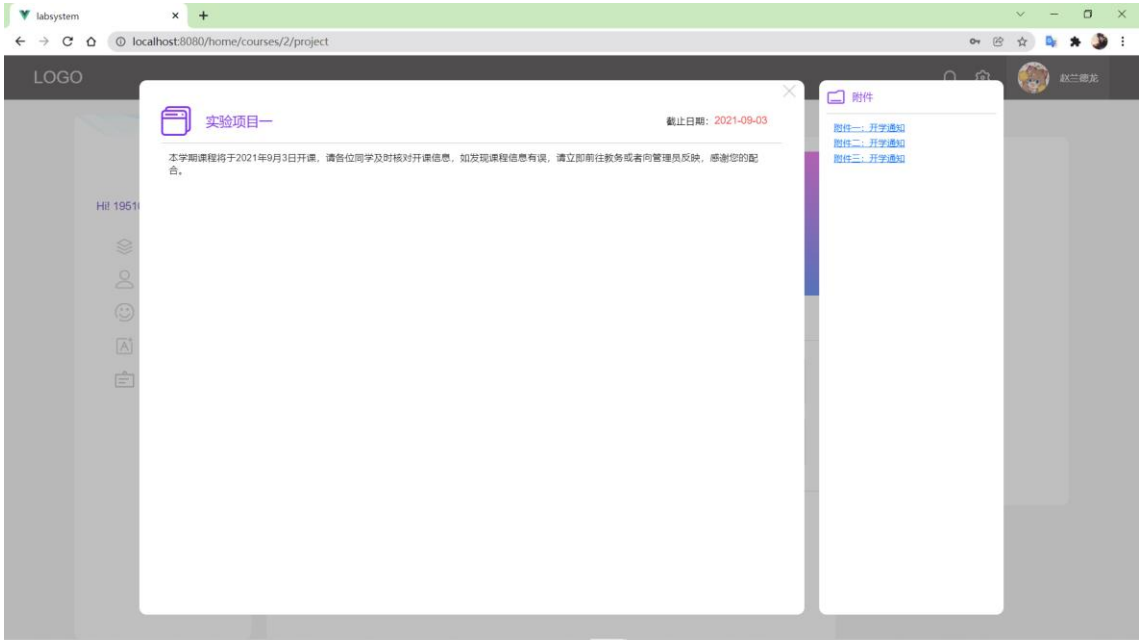
2.4.5 课程页面设计

课程详情-项目列表设计



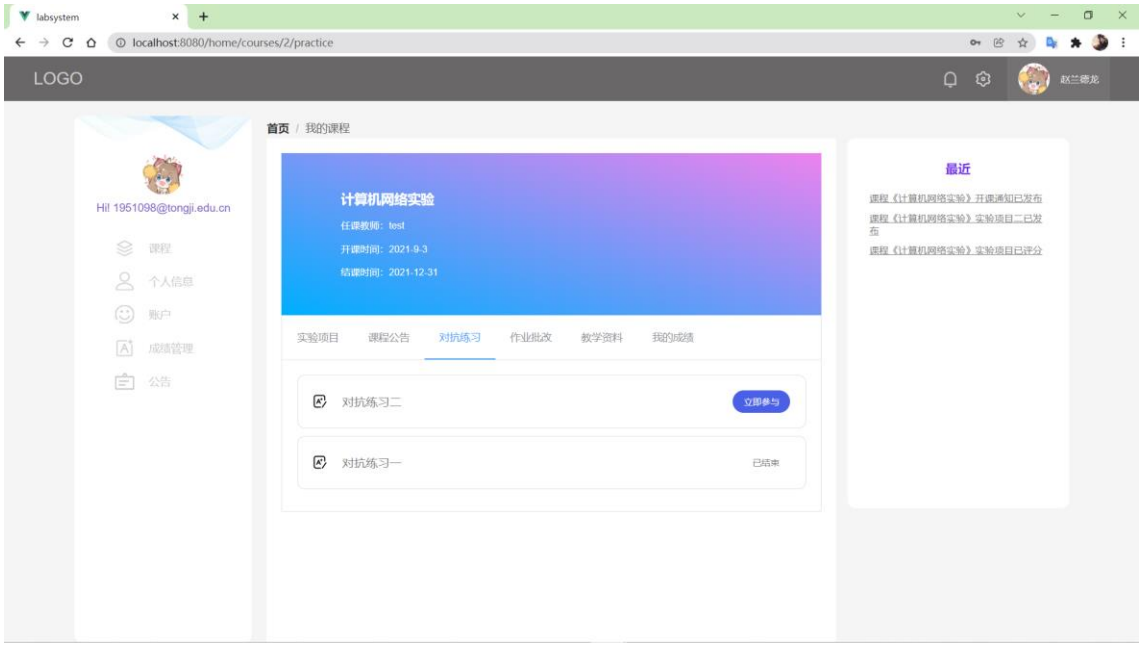
课程页面是我们本次项目的一个重点，在这一个模块之中包含了大量的功能、大量的交互逻辑以及大量的页面，想要把这些功能有序的组织好并不是一件轻松的任务，首先在布局方面，本项目采用了二级导航机制，从而把子功能模块划分出来，让条理更加清晰，同时按照信息的层级进行展示，最基本的信息放在最能吸引视线的位置，让页面更加美观。

项目详情设计



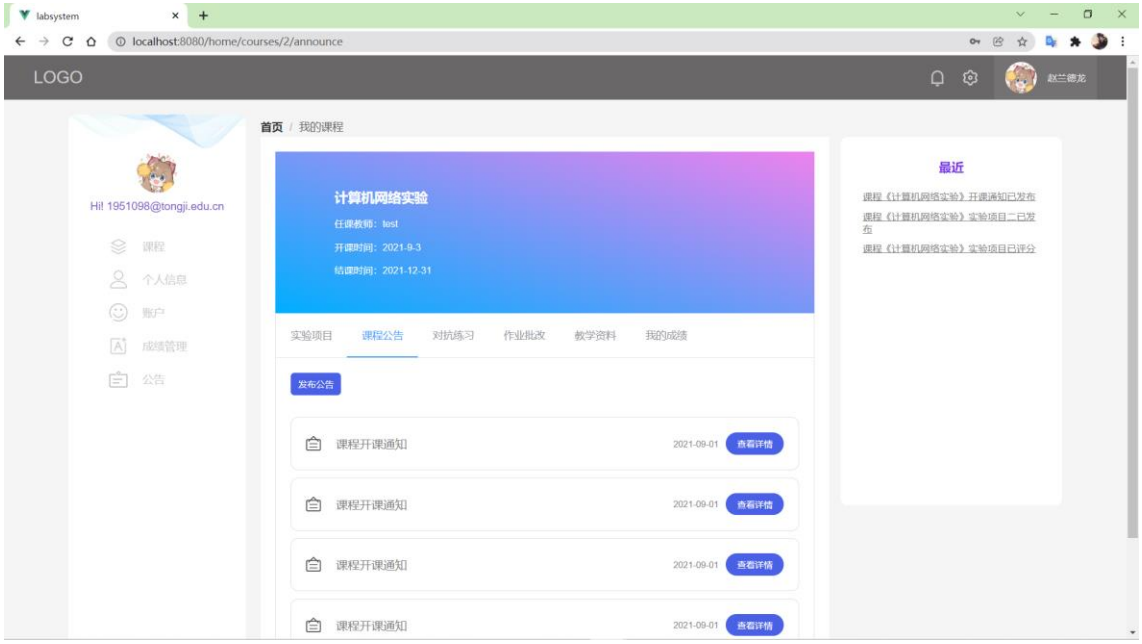
课程项目子模块中，我们采用了二级的信息展示，即：在第一级展示项目的列表，第二级展示项目的详情，这样可以有效地精简页面信息。在项目详情的页面中，首先我们为项目的内容留出了足够多的空间，同时，我们把截止日期在显眼的位置标注了出来，进行强调，最后针对附件功能，我们创新性地把附件和主体内容分割开来。这样设计的考量在于，当正文内容过长，导致出现滚动条，那么附件会被隐藏在正文的最下方而不容易被发现，这无疑降低了用户的体验，所以我们在页面设计方面将主体和附件解耦，从而为用户带来更好的体验。

对抗练习设计

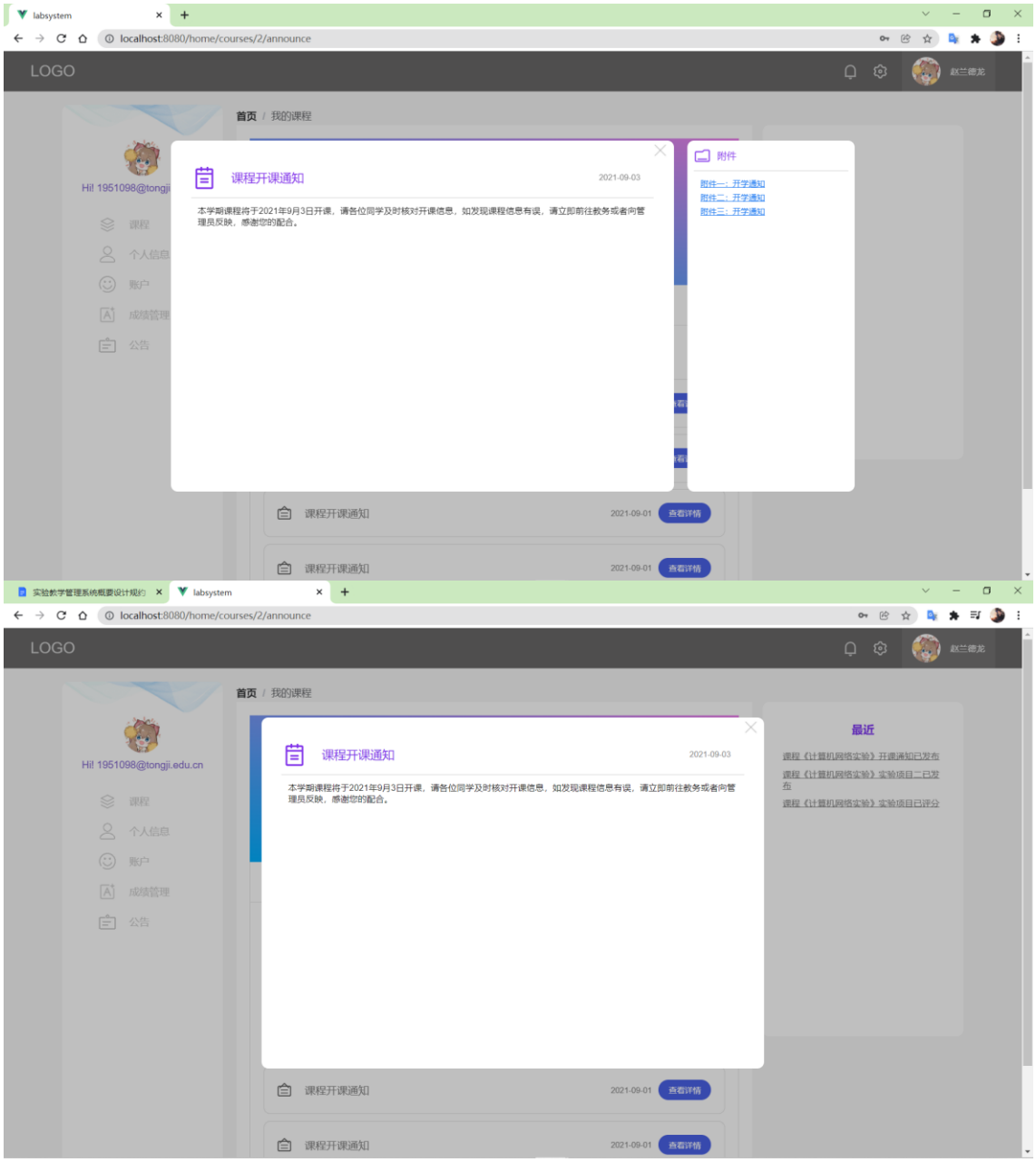


对抗练习模块，同样采用二级的信息列表，第一级展示对抗练习的列表，第二级展示对抗练习的详细信息。

公告设计

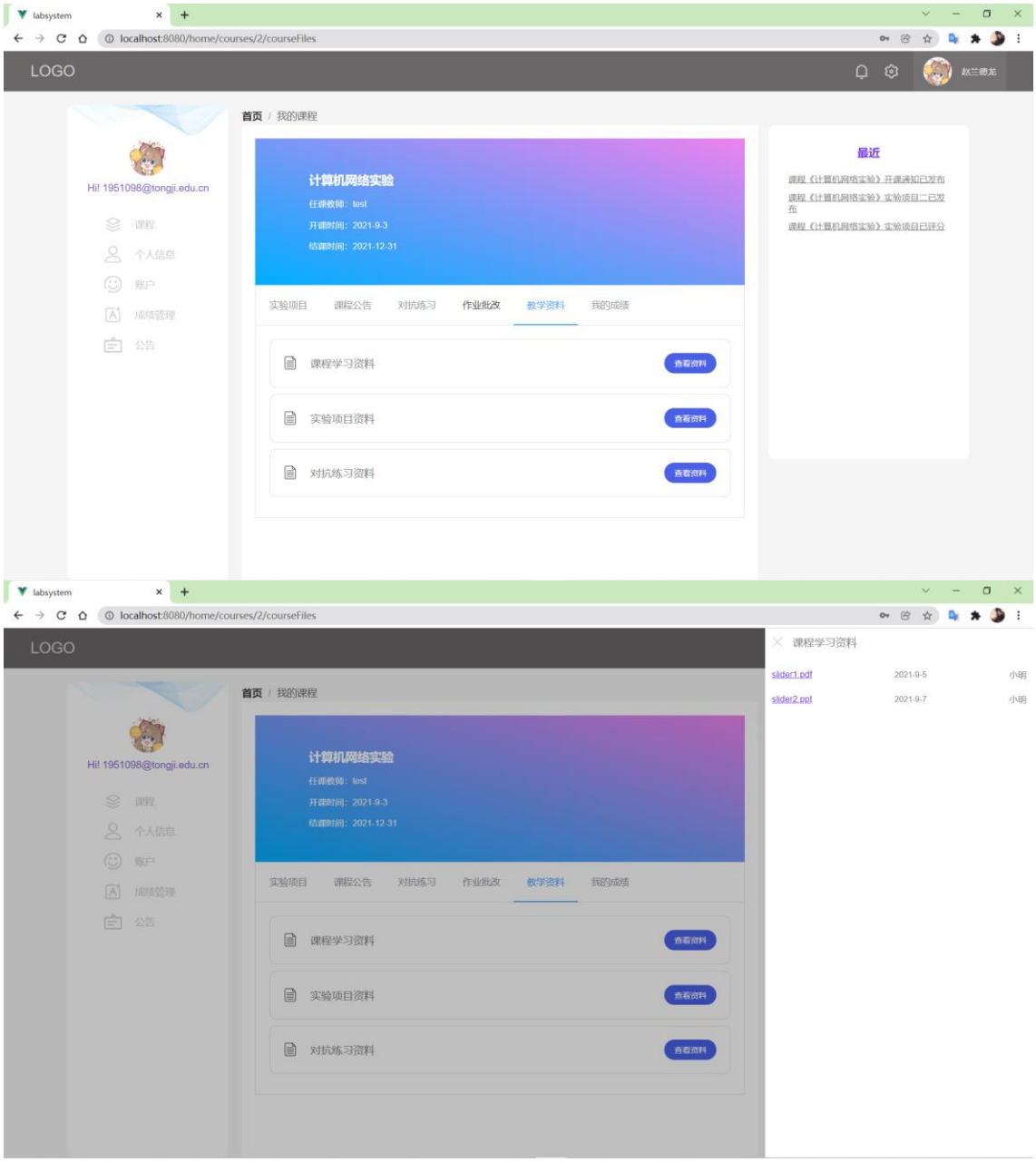


公告详情设计



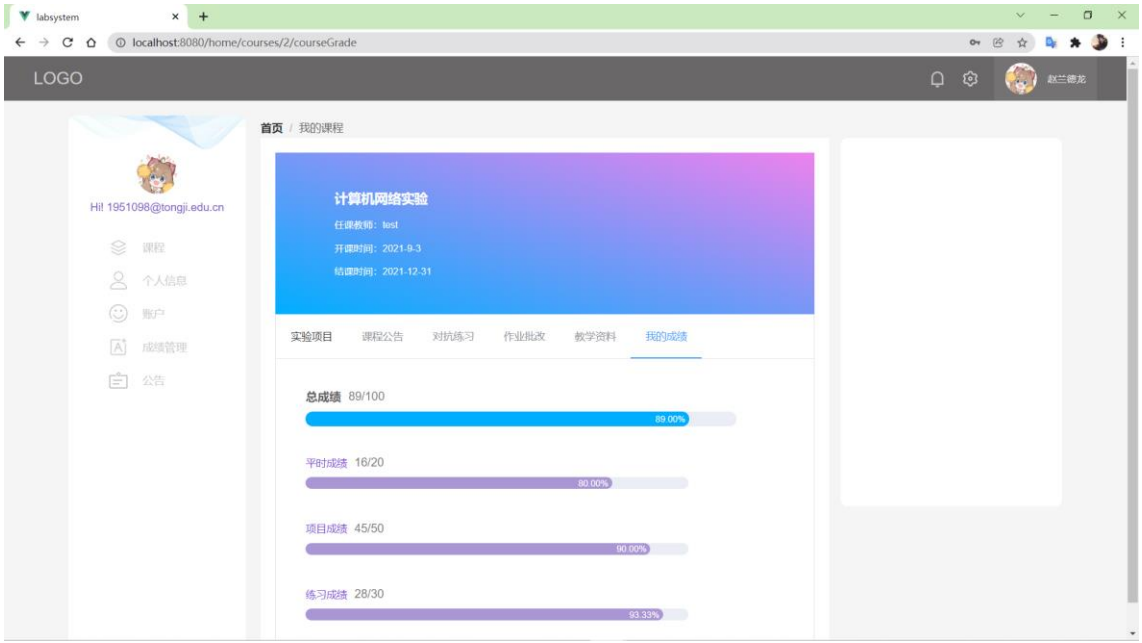
公告的设计和实验项目的设计相似，我们把主要的信息放在了公告的二级页面当中，同时把主体内容和公告附件分离，并且附件的窗口是动态变化的，当某一个公告没有附件时，附件的窗口会被隐藏，这样可以给用户带来更加优质的体验，让用户聚焦与主要内容。

教学资料库设计



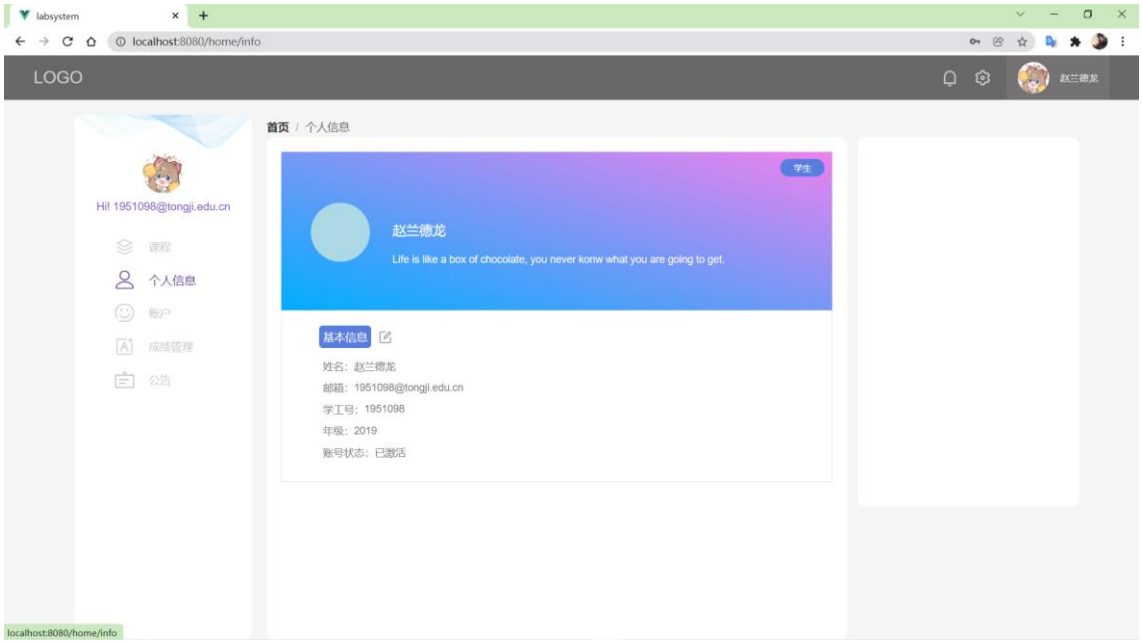
教学资料库同样采用二级设计，在需求中要求根据教学资料的归属对教学资料进行分类，包括课程学习资料，实验项目资料和对抗练习资料。在二级界面中给出资料的列表，用户可以自行选择下载。

成绩查看设计



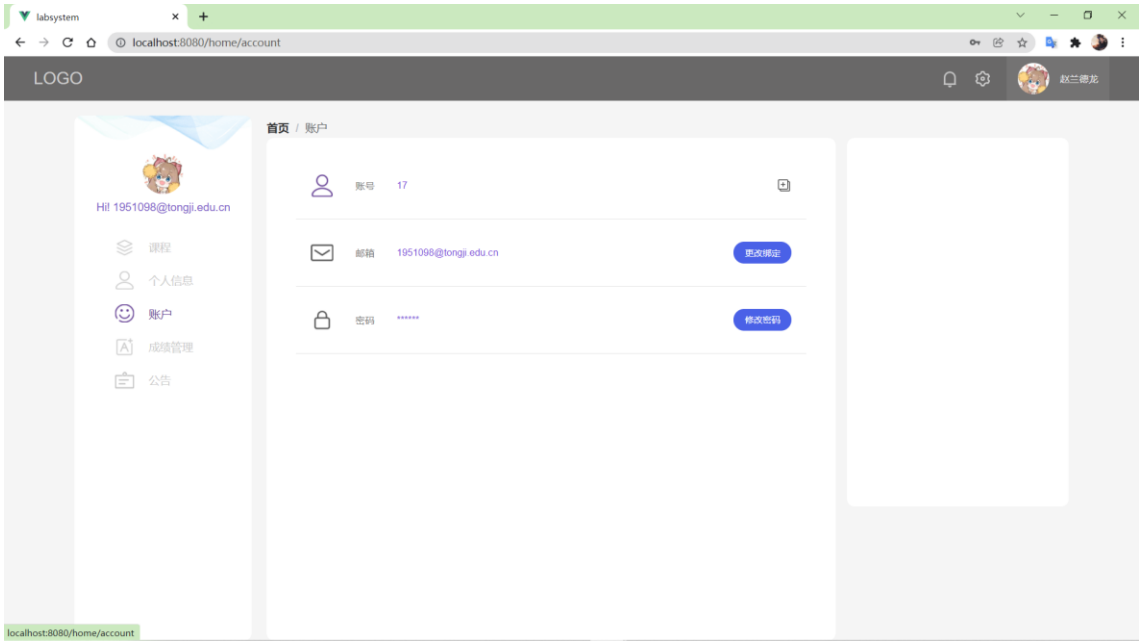
成绩方面，为了达到更好的展示效果，我们对成绩信息进行了可视化，通过进度条来表示用户得分的占比，让成绩信息更加直观。

2.4.6 用户信息页面设计



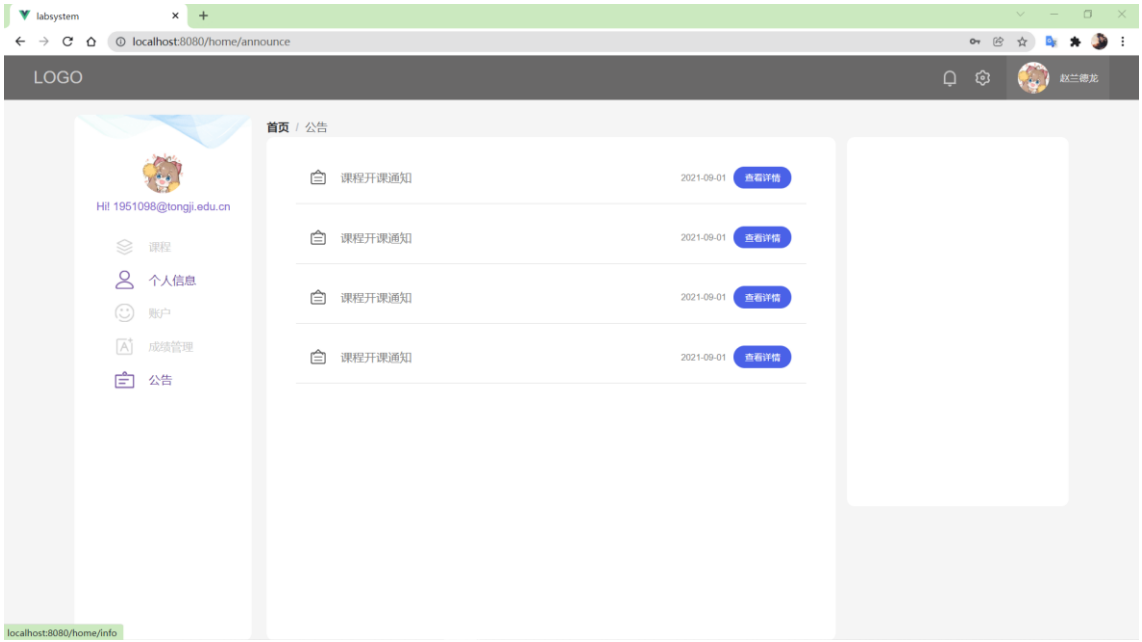
用户信息页面展示用户的基本信息，并且提供信息修改的功能，页面分为头部和主体，头部主要包括用户的头像与描述，右上角有用户类型的标识，而主体则规整地列出用户的基本信息，并且给出了修改信息的按钮，从而更好地展示和指引。

2.4.7 账号信息页面设计



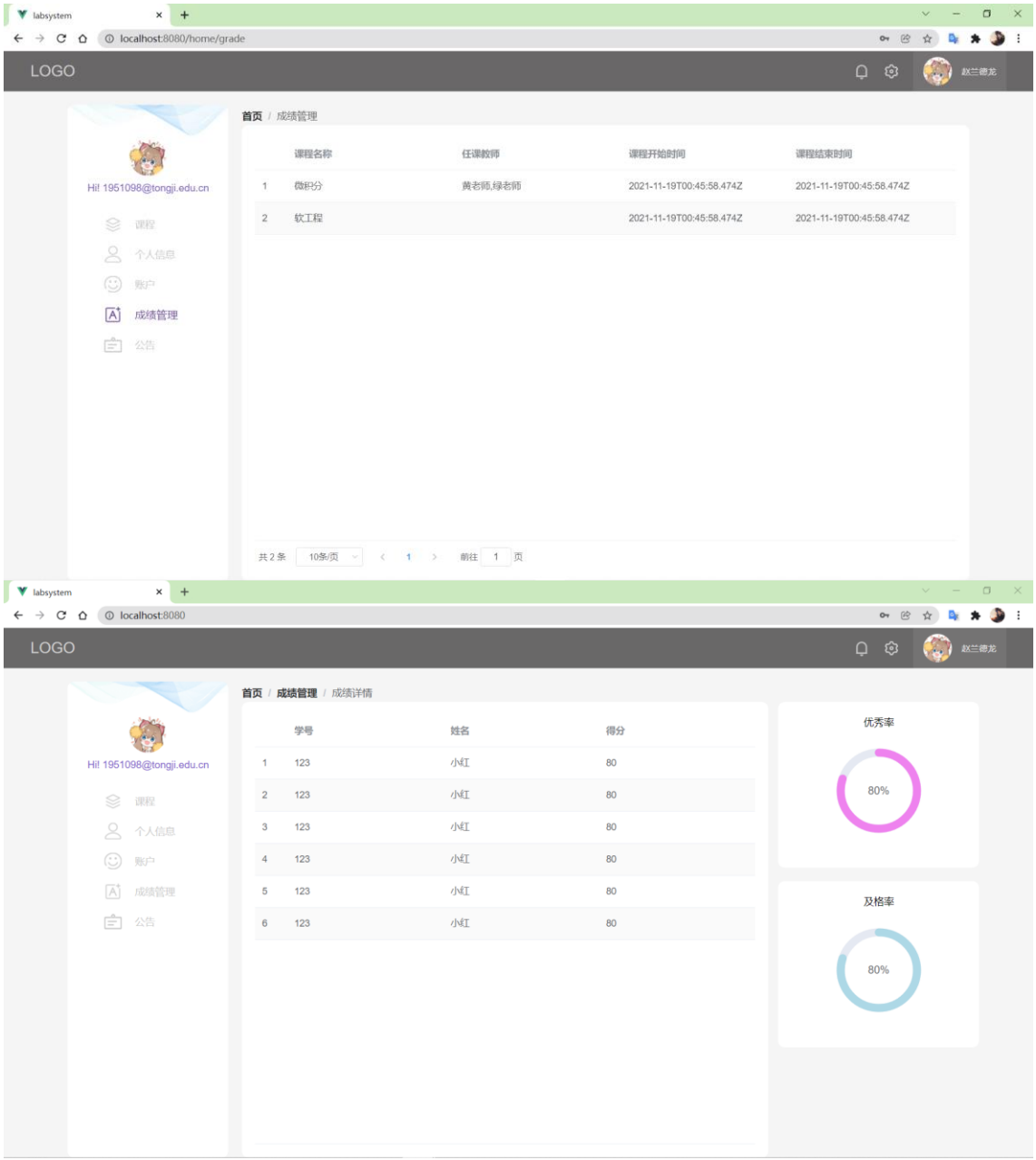
在账号信息页面，我们将账号相关的信息以列表的形式展示，并且在右方给出了可用操作，构成了项-取值-操作的基本结构，让界面更加美观，同时也方便了用户的操作。

2.4.8 公告页面设计



这是一个面向全体用户的公告机制，与之前课程模块中公告子模块十分相似，唯一的区别在于数据的作用范围，即这里的公告是面向整个系统的，而课程中的公告只面向本课程。

2.4.9 成绩查询页面设计

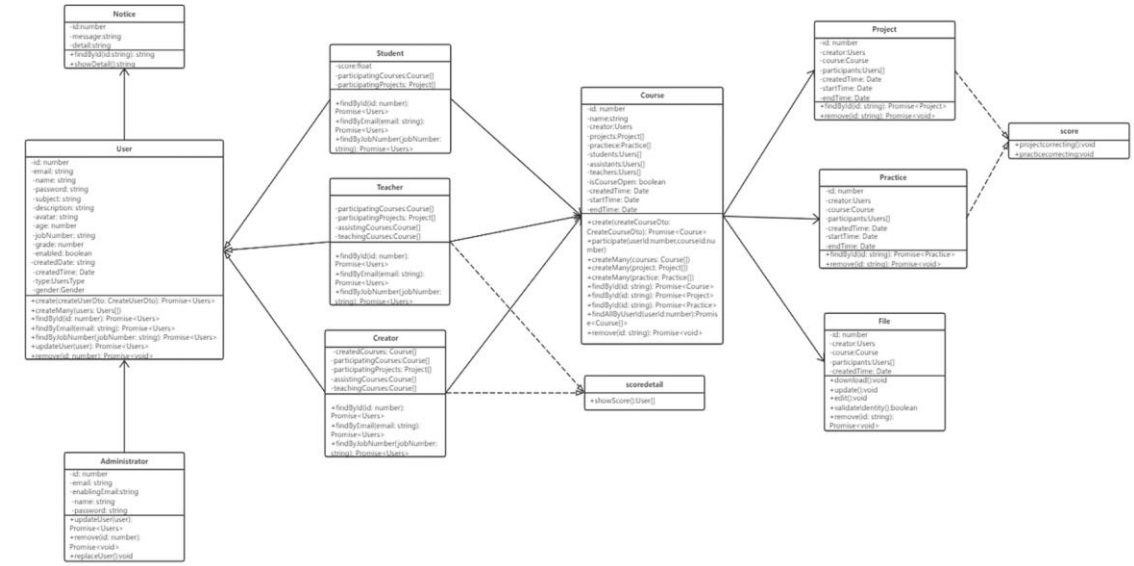


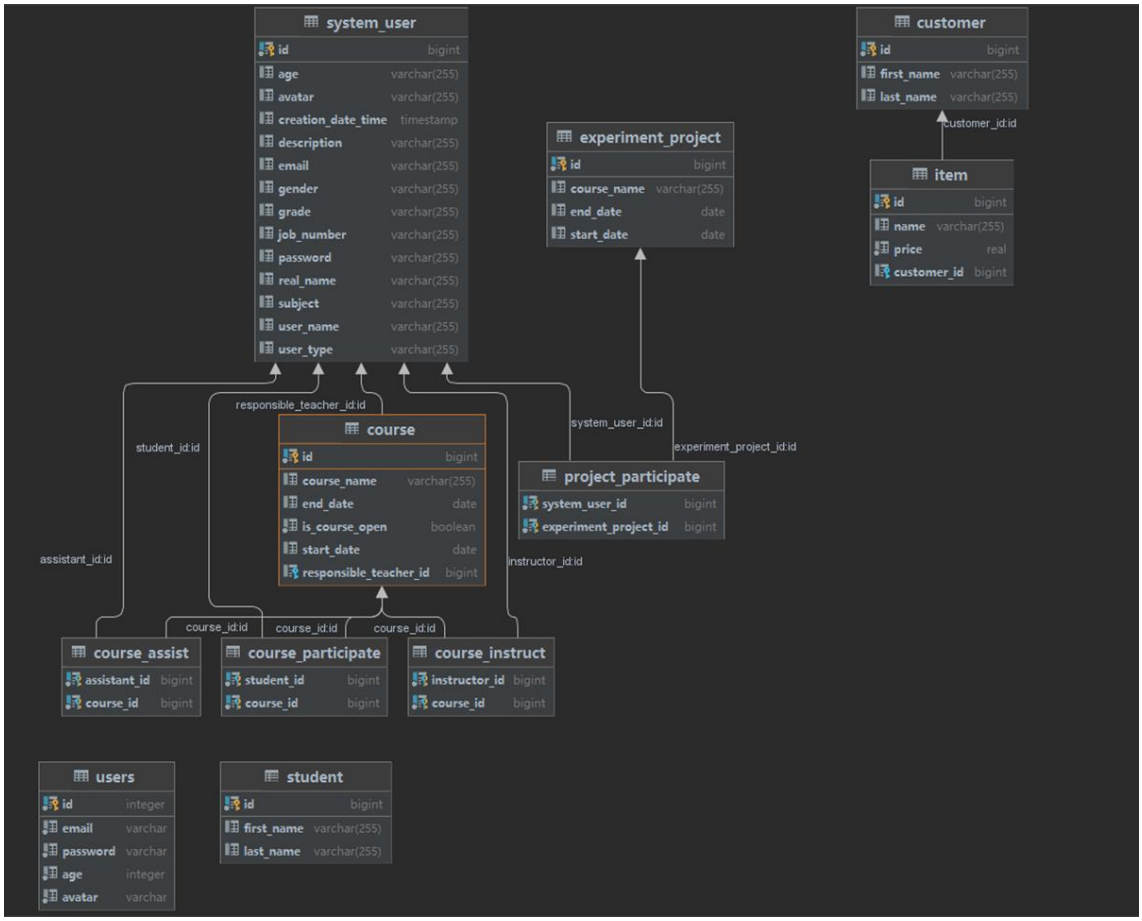
成绩查询服务于责任教师，首先采用了二级的页面展示形式，在详情页面中，我们不仅准备了数据展示的列表，同时，我们也在右方进行了数据的可视化，从而提供更加优秀的展示效果。

2.5 数据库设计

在数据库的构建方面，我们采用的是 ORM。对象关系映射（英语：Object Relational Mapping，简称 ORM，或 O/RM，或 O/R mapping），是一种程序设计技术，用于实现面向对象编程语言里不同类型系统的数据之间的转换。从效果上说，它其实是创建了一个可在编程语言里使用的“虚拟对象数据库”。我们主要用 ORM 来解决面向对象编程与数据库存储之间的不匹配问题。ORM 的构建方法也是丰富多样的，在研究和学习中大致总结出两种：第一种是自底向上构建，先根据需求在数据库中把所有表建好，使用自动化工具按照数据库中的表自动构建实体类，从而完成 ORM 的构建。这种方法下数据库对编写者不透明，相对直观，但需要手动建立数据库，设计联系集，并且数据

库改变时需要重新构建实体类；第二种方法是自顶向下的构建方法，直接编写实体类，由 ORM 工具自动创建数据库表，编写者直接在类的层面进行编写，从而完成 ORM 的构建。这种方法下数据库对编写者透明（相对），需要在实体类中明确指定类之间的关系（一对一，多对多等），改变实体类时数据库会自动同步更改。权衡之下，我们选择了第二种方法，即设计实体类，明确类与类之间的关系，由 ORM 工具自动生成数据库表。因此，我们数据库设计与类设计一脉相承。





2.6 系统出错设计

2.6.1 出错信息

出错信息	说明	操作
用户名或密码错误	登陆时没有输入正确信息	重新登陆
该账号已被激活	注册时可能输错学号或者账号被抢注	检查输入或者联系管理员
用户信息为录入	注册时可能输错学号或者信息未录入	检查输入或者联系管理员
未授权	请求时 token 丢失	重新登陆
获取 XX 失败	网络错误或服务端故障	重新登陆或者联系管理员

2.6.2 补救措施

系统中如果出现错误，会自动返回到登陆界面，此时用户可以：

1. 尝试重新登陆
2. 联系系统管理人员与维护人员