****

**实**

**验**

**报**

**告**

**课 程 Web开发**

**题 目 教育教学信息管理系统**

**分 数**

**学院名称 智能与计算学部**

**专 业 软件工程**

**学生姓名 刘兴宇**

**学 号 3017218063**

**年 级 2017**

**班 级 1**

目录

[一、实验目的 1](#_Toc527352380)

[二、实验内容 1](#_Toc527352381)

[三、实验步骤 1](#_Toc527352382)

[四、实验结果 1](#_Toc527352383)

[五、实验结论 1](#_Toc527352384)

[六、源代码 1](#_Toc527352385)

## 实验目的

本次实验题目为综合性的Web项目练习，自选题目实现全栈开发。我的作业题目是《教育教学信息管理系统》

## 二、实验内容

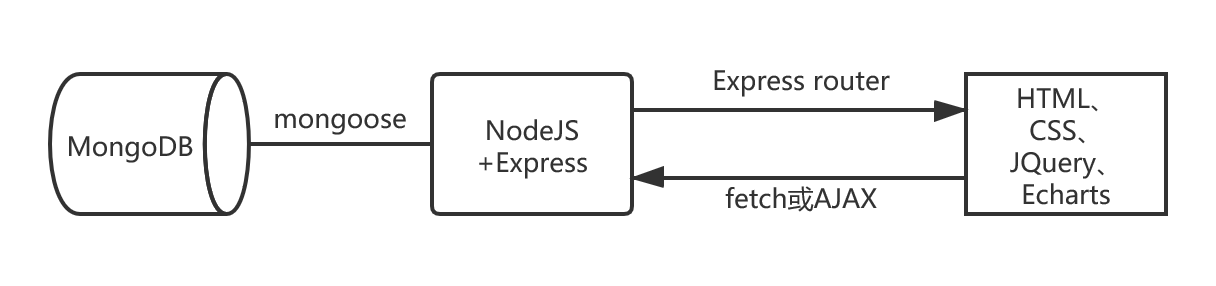
开发全栈Web项目天津大学教育教学信息管理系统，包括数据库设计、后端、前端

## 三、实验步骤

1. 确定需求

* 系统分为管理员端和用户端。在登录界面可以选择学生登录或者教师登录、注册学生账号
  + 注册学生账号：只有注册了学籍的学生才可以使用学号注册账户
  + 学生登录需要先注册
  + 管理员登录拥有固定的密码
* 学生端分为四个功能
  + 主页面：显示欢迎语、时间信息（入学天数和周数）、今日课程、教务通知
  + 个人信息：查看自己的学籍信息、课程表、修改密码
  + 成绩管理：查看各学期的加权成绩折线图、选择年份查看具体年份的成绩及雷达图
  + 选课管理：选课未开始不能选课，选课开始可以选课和退课，并且显示实时课程表
* 管理员端分为五个功能
  + 主页面：显示欢迎语、时间信息、修改管理员密码、修改教务通知
  + 学生管理：添加学籍、修改学籍信息、删除学籍
  + 课程管理：添加课程、修改课程信息、删除课程
  + 成绩管理：登课程成绩、修改学生成绩、按课程或学生查看成绩统计信息
  + 选课管理：开始/停止选课，为特定学生选课并实时显示课程表

2. 体系结构设计



* 数据库：MongoDB
* 后端：NodeJS环境+Express框架+mongoose操作MongoDBAPI
* 前后端交互：fetch
* 前端：HTML+CSS+JQuery+Echarts

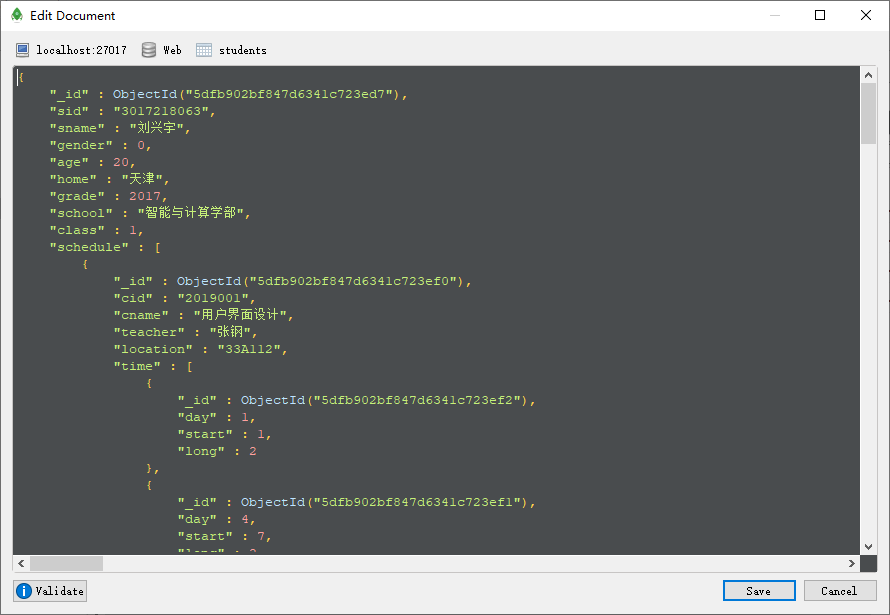
3. 数据库设计

数据库采用MongoDB数据库，它是一个基于文件的分布式非关系型数据库。采用MongoDB的主要原因是它采用了类似于JSON的Binary-JSON数据存储格式，便于数据的传输和处理。

MongoDB数据库的结构

* 数据库database：所有数据及配置的集合
* 集合Collection：表，定义并存储数据项
* 文档Document：一条数据记录
* 属性Key：属性名称
* 值Value：属性值

数据库截图



本次项目中数据库共设计了六张表

* Admins：存储管理员信息
* Users：存储用户信息
* Courses：存储课程信息
* Students：存储学籍信息
* Records：存储成绩信息
* Systems：存储系统信息

启动数据库的方法：开命令行，进入数据库bin文件夹下，输入命令：

mongod –dbpath 数据文件路径

即可启动数据库





数据库端口号为：27017

4. 后端

后端采用NodeJS编程环境，使用JavaScript编写后端环境

* 链接并操作数据库：mongoose

MongoDB数据库的API使用起来比较繁琐，因此使用Mongoose框架来操作数据库。

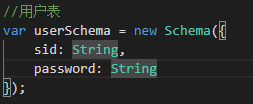
* + 配置mongoose信息（mongoose.js）



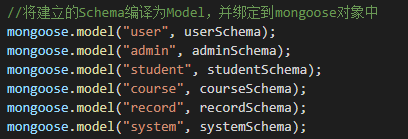
* + 链接数据库（db.js）



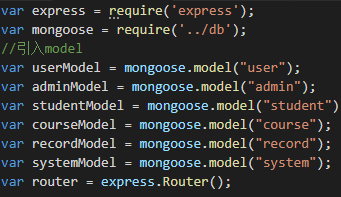
* + 建立Schema，用于定义Collection（db.js）



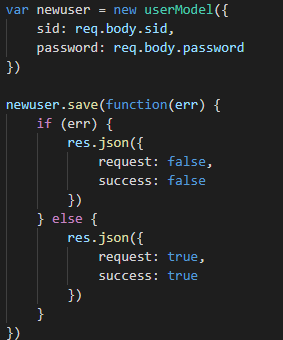
* + 编译建立的Schema，形成Model，并绑定到mongoose对象中



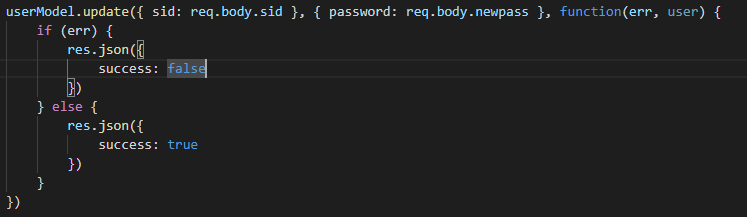
* + 在路由中引入Model，并进行操作（index.js）



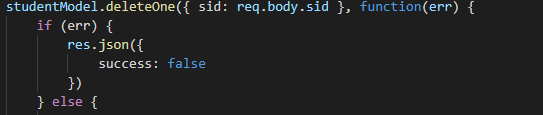
增加：



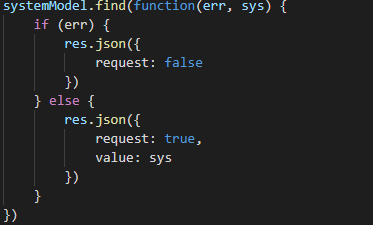
修改：



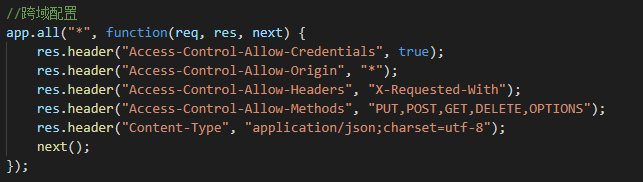
删除：



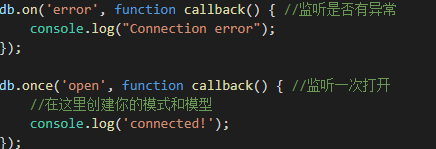
查询：



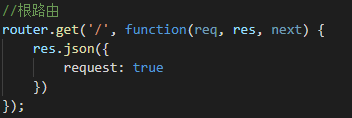
* 配置服务器：使用默认端口：3000
  + 跨域资源访问CORS：（app.js）



* + 监听打开与异常



* 编写路由：req为请求报文，res为响应报文
  + 根路由



* + 其它路由

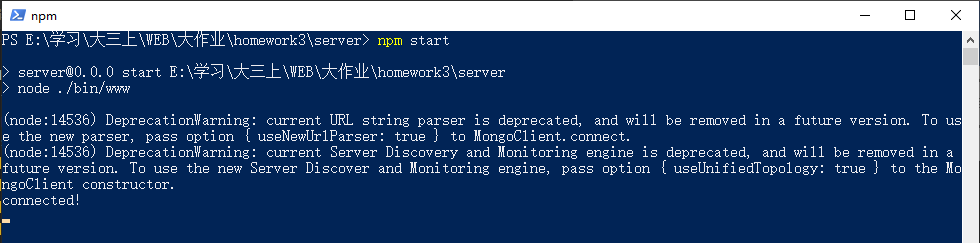


* 启动服务器的方法：

在server目录下打开命令行，输入命令

npm start

即可成功启动Express服务器



5. 前端设计

前端主要采用的技术是HTML+CSS+JQuery，使用fetch进行前后端交互

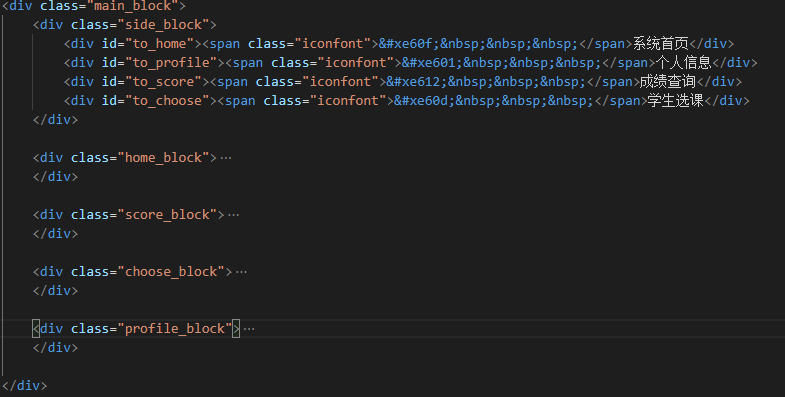
* 页面结构：（具体布局请参考代码）

为了解决子页面过多导致HTML文档过多，且切换时页面刷新和重新加载对用户体验的影响问题，将同一主页面中所有子页面压缩到一个HTML文档，并且使用JQuery进行路由切换，显示需要展示的子页面，而将不需要展示的子页面隐藏(display：none)

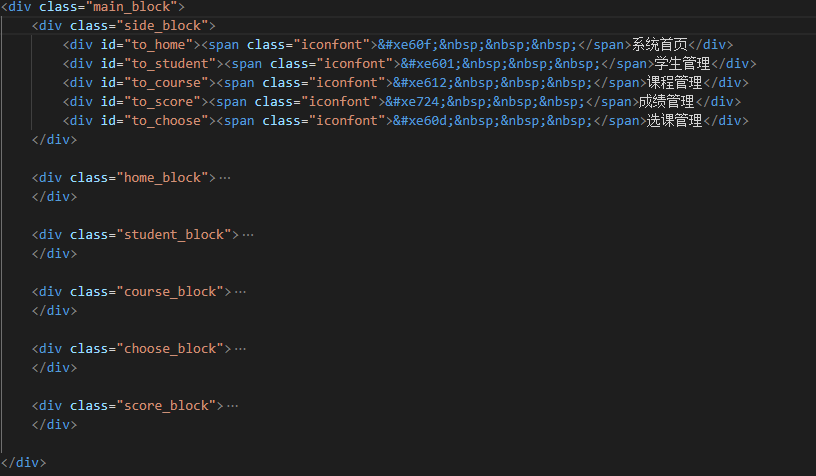
* + login.html：



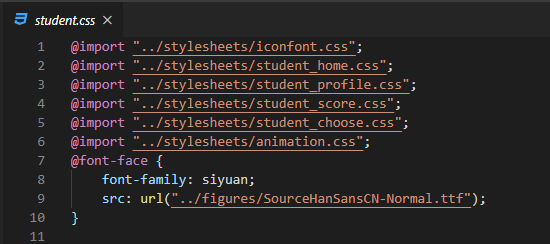
* + student.html



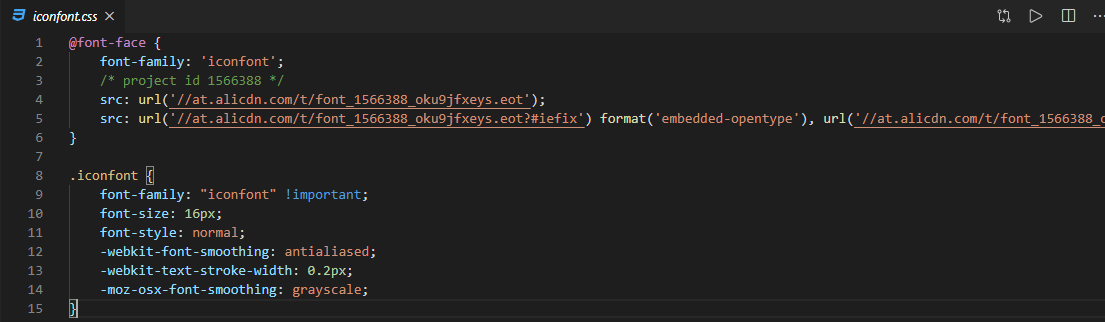
* + admin.html



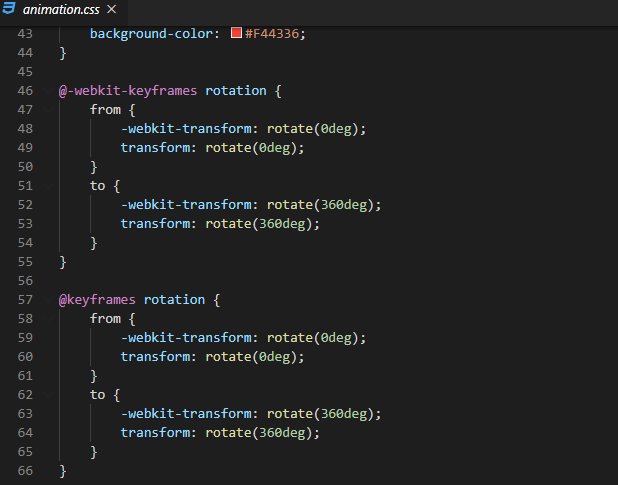
* 样式设计：采用了部分响应式设计和极简主义的设计风格。将子页面的CSS样式写在分开的文件中，虽然页面链接降低了部分性能，但更易于维护，并且使用了思源黑体的字体和阿里巴巴矢量图库iconfont的矢量图作为图标。还学习了加载动画的制作



iconfont：



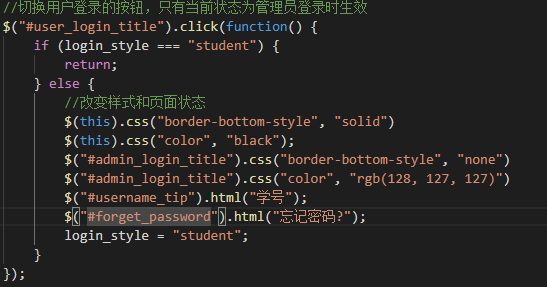
加载小动画：（部分代码）



* 交互代码设计：采用Jquery中监听器和选择器，更便于对DOM文档结构、样式进行改变、进行网络请求等操作

关键技术：

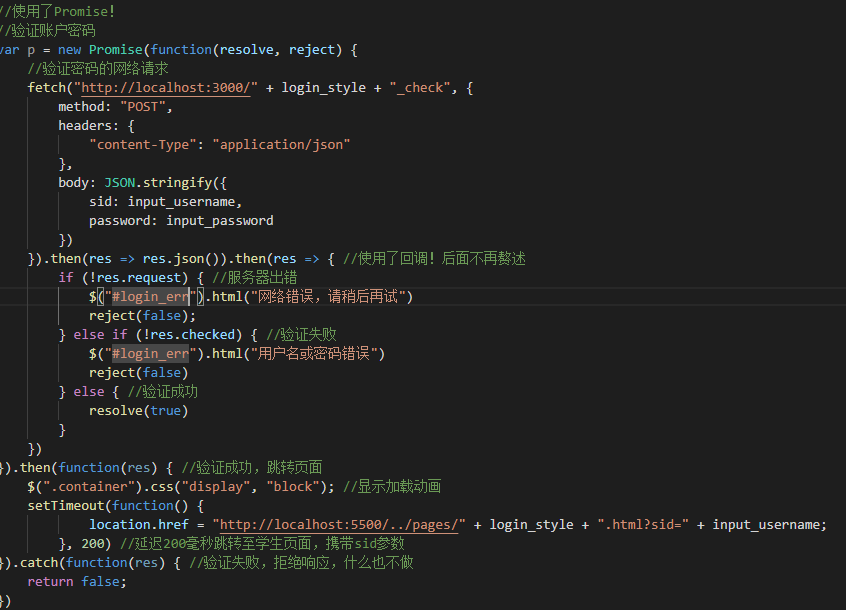
* + 点击事件监听器



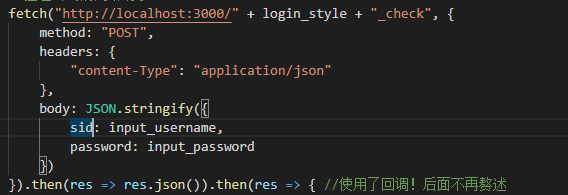
* + 网络请求：fetch是基于Promise的网络请求API，与AJAX有着相同的作用，且相比AJAX可以使用链式回调



* + Promise：promise是一个用于确保异步操作后执行某些操作的API，使用then、catch回调函数捕捉异步程序的执行结果。并且可以将嵌套回调变为链式回调



* + 回调函数：异步操作执行后执行，在多个地方都使用了回调





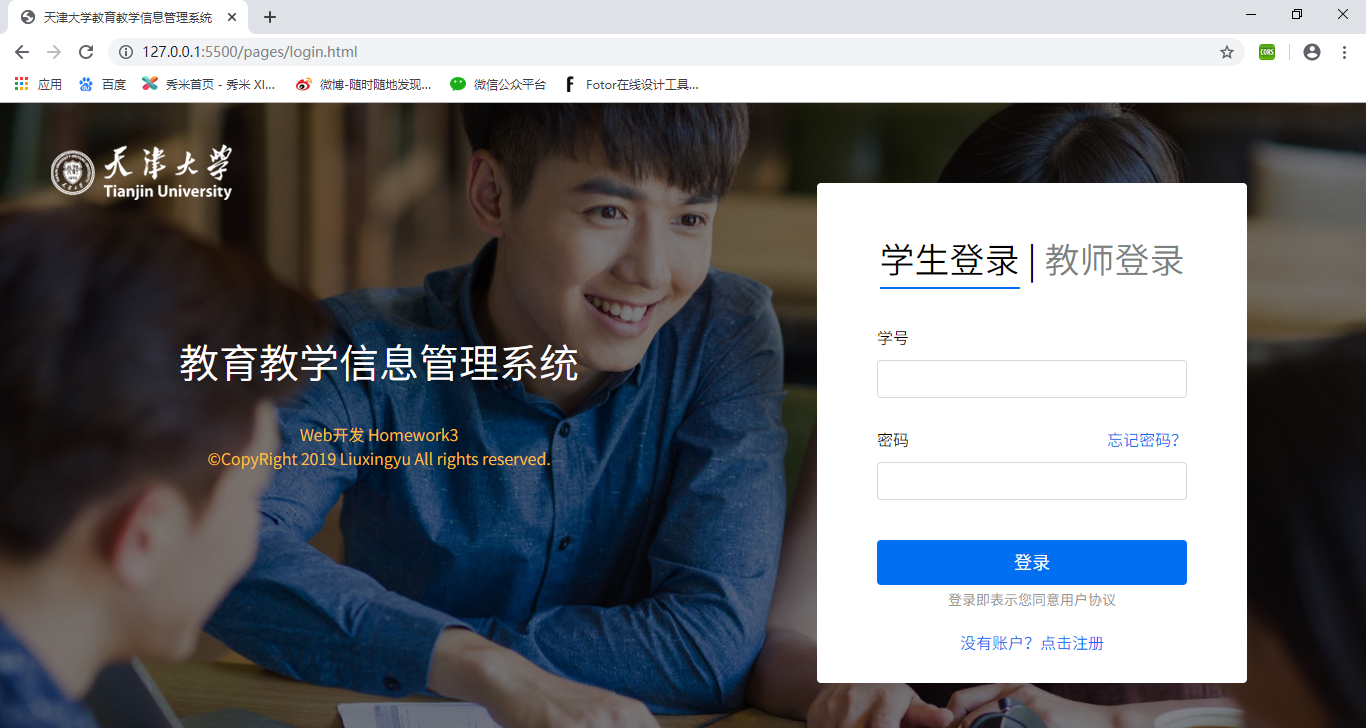
* + 闭包：在函数内部定义的函数，用于封装方法



其余技术在代码中有详细注释

## 四、实验结果

登录页面：（学生登录、教师登录、注册）



学生端：

* 主页面：



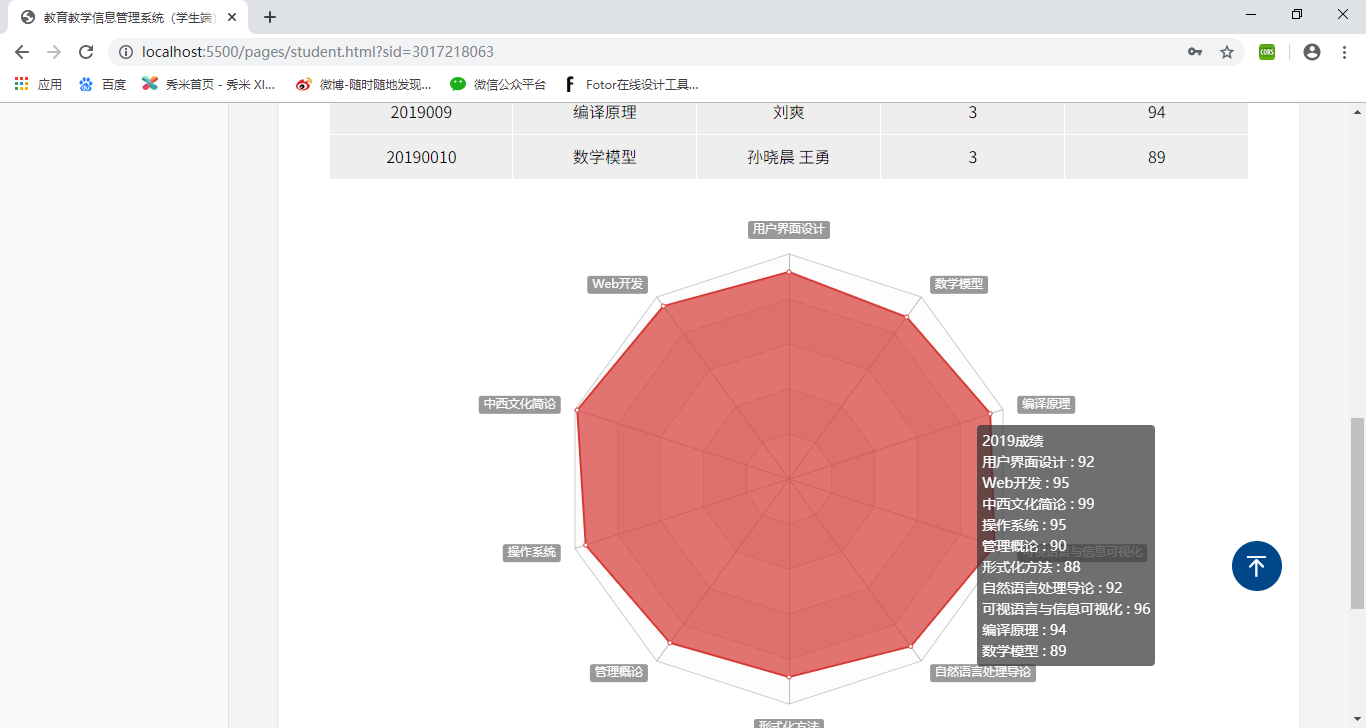
* 个人信息



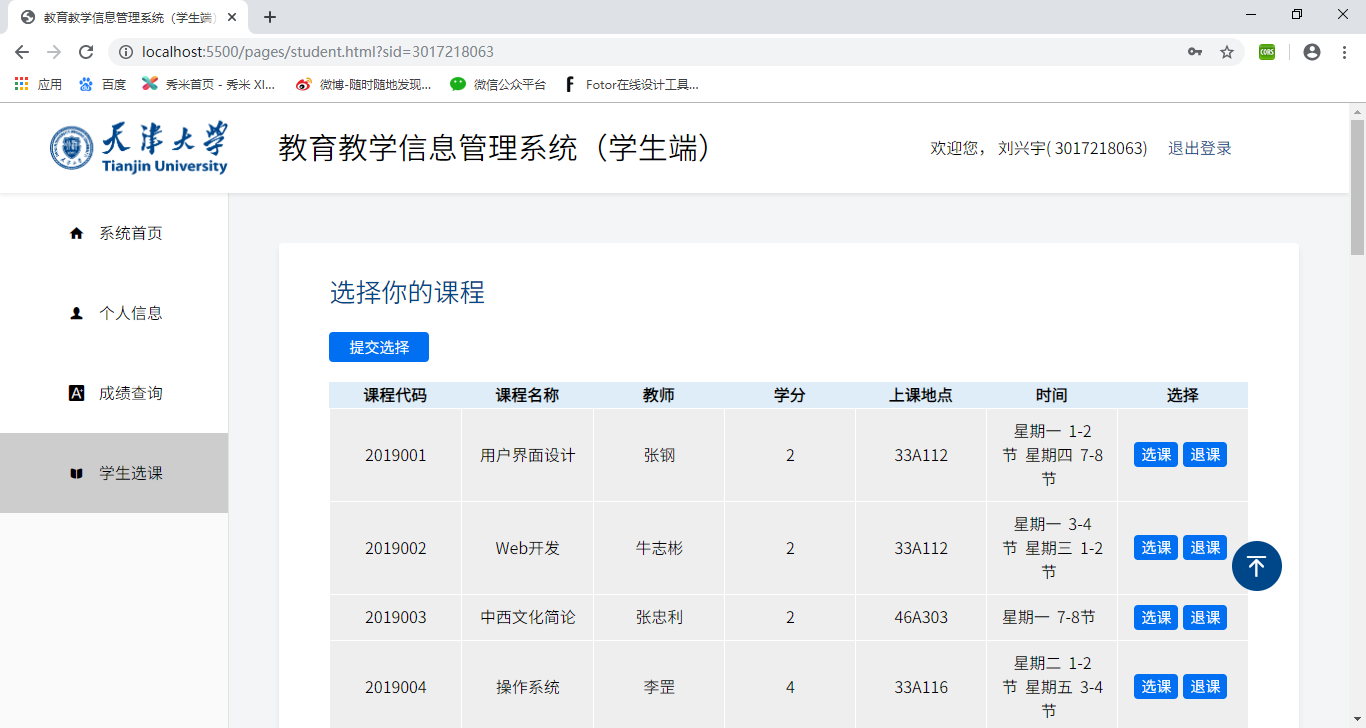


* 成绩页面





* 选课页面





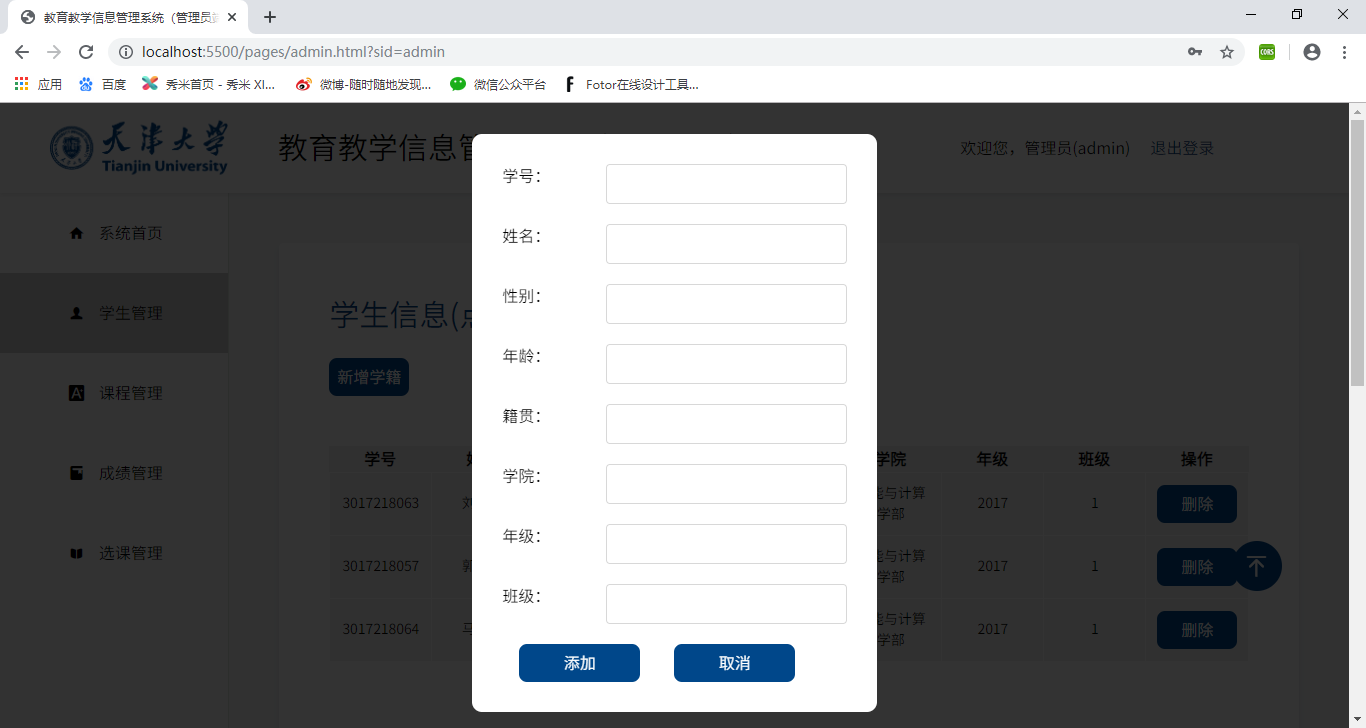
管理员端：

* 主页面



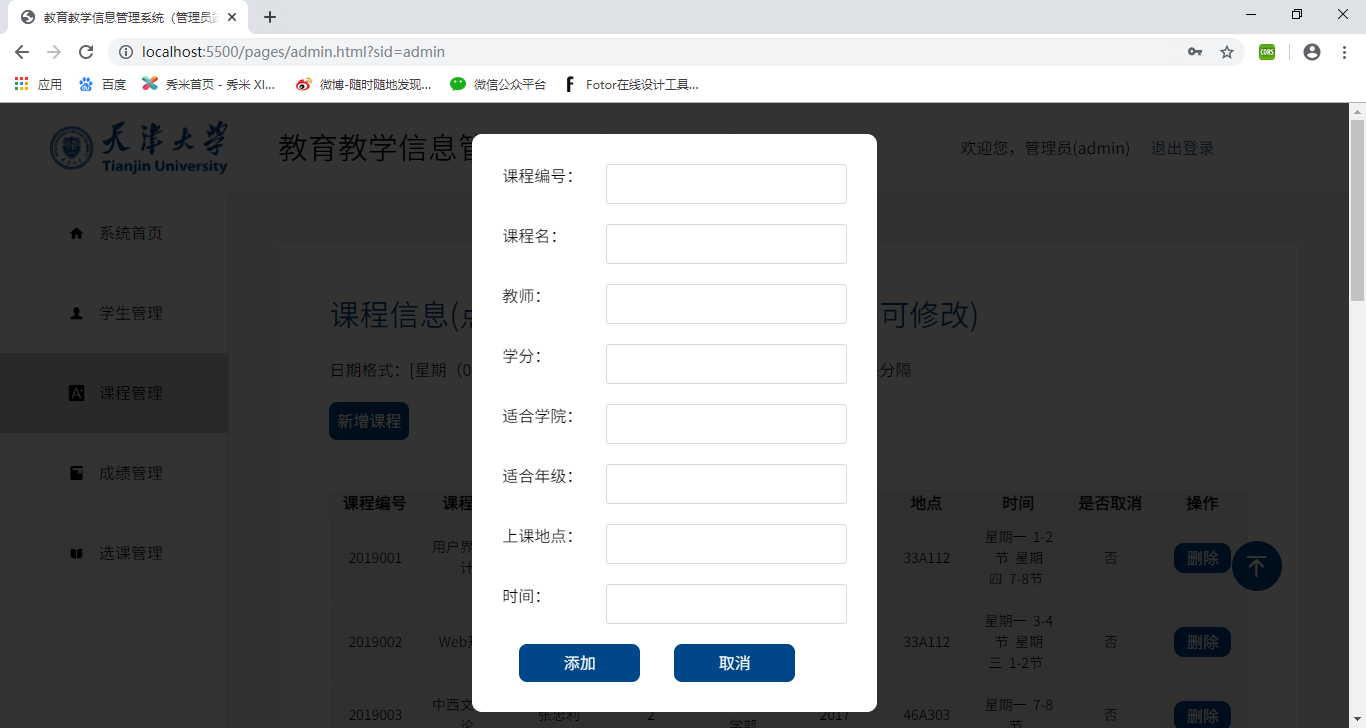
* 学生管理





* 课程管理





* 成绩管理



* 选课管理



## 五、实验结论

通过这次作业，学会并熟练掌握了MongoDB、Express的原理和使用方法，并且对全栈开发的模式有了更好的运用。并且熟练掌握了JQuery的使用动态页面渲染等技术。

## 六、源代码

源代码过多，详情见源代码文件