深 圳 大 学 实 验 报 告

课程名称：基于 Web 的编程

实验项目名称：帐号管理

学院：计算机与软件学院

专业：软件工程

指导教师：尹剑飞

报告人：洪继耀 学号：2014150120 班级：02

一、实验要求

制作一个用户管理系统，功能包括：用户注册，用户资料修改，用户分级管理，用户权限管理等。要求本实验者做出系统不低于以下要求：

1.数据库设计要求

表有：

用户表 user，字段有 user_id（流水号），username,email（需要字段校验），sex（一个字符长度）,age(需要字段校验),class(所在班级),password(加密后的密文，加密算法可用 sha1),role_id(外键，参照完整性约束要实现，可通过 SQL 数据表定义约束或是通过编程实现；

用户角色表 role，字段有 role_id（流水号），role_name，role_rights（角色权限：类型为数组），parent_role_id（父角色）。角色表记录一个角色树，父角色的权限涵盖所有子角色权限。角色有：班长、团支书、学习委员、课代表、班级成员等。

角色权限与角色的关系由角色 - 权限矩阵表示（以下为样例，仅供参考）：

	发布开班会帖	收班费帖	评优等生帖	发布考勤结果帖	回复帖
班长 4	班长	班长	班长	班长	班长
团支书 3		团支书	团支书	团支书	团支书
学习委员 2			学习委员	学习委员	学习委员
课代表 1			课代表	课代表	课代表
班级成员 0					班级成员

上述权限管理是现实权限管理的简化版（真实的权限管理不是一棵树，而是一个无环有向图，鼓励实验者思考这个加分问题）。

开班会帖表 meeting：class_id（班级 ID）、time、place、numbers_required

（应到人数），numbers_gotten（实到人数）、topic（主题）、pool_agrees（投票同意名单、user_id 数组）、pool_disagrees（投票不同意名单、user_id 数组、可为动态生成字段）、发帖人（user_id）。

收班费帖 class_money：claass_id（班级 ID）、deadline（截止时间）、howmuch（缴费金额）、payed_members（已缴费名单、user_id 数组）、unpayed_members（未缴费名单、user_id 数组、可为动态生成字段）、发帖人（user_id）

评优等生帖 good_student：claass_id（班级 ID）、title（优等头衔）、user_id、bond（奖金）、发帖人（user_id）

发布考勤结果帖 check_in：claass_id（班级 ID）、user_id（旷课学生 Id）、course_name（课程名）、missing_date（旷课日期）

2．系统权限管理功能

当一个用户登录后，系统自动在界面上给出他可执行的功能：发布开班会帖　收班费帖 评优等生帖　发布考勤结果帖　回复帖。

给出权限管理的树型编辑的界面设计和实现、包括 CRUD 权限。

3．系统统计功能

系统为每个用户自动统计他交过的班费情况列表（包括总缴费金额、总欠费金额）、考勤情况表（包括旷课总数、旷课天数）。

4．系统智能预测功能（选做）

系统根据某学生的旷课情况，自动预测他下次旷课的概率。可用多个预测机来做(ensembling learning)，一种简单的算法是：一个朴素预测器（根据历史旷课次数就可得出）、另一个是神经网络预测器（输入：学生开班会旷的次数、累计旷课此次、

课程选课人数等；输出旷课概率），将多个预测器的输出进行投票，权重初始相等，每次得到新的旷课结果自动更新权重。

二、设计说明

数据库设计文档

如何实现数组字段的 CRUD？

由于 MongoDB 不必遵从第一范式，所以增删查改自需要一次调用接口就行了。

数组字段 vs 多字段的性能差别？

数据字段读取快，可以一次性读取全部需要的信息，有效地减少硬盘读写，但是在硬盘空间消耗上较多，冗余度很高，是典型的时空权衡，比较适合 MongoDB 这种需要减少硬盘读写的文档型数据库。

多字段查询上稍微慢，但是大大减少了空间冗余度。但是在保持数据一致性上有少许困难。但是现代关系型数据库软件非常成熟，因此多字段查询在大型项目里非常常见。

REST API 是怎样设计用于 CRUD 操作的

RESTFUL API 是一种接口设计风格，一般用 HTTP 幂等动词 GET 和 PUT 对应幂等的数据库操作查和改，用 HTTP 操作 POST 和 DELETE 对应数据库操作增加和删除。

而 RESTFUL API 的路径则对应数据库的某张表或者某个文档(在 NoSQL 中)。

进一步更加复杂的操作可用子路径和 URL query 参数来区分，实现参数化查询。

各种表的参考完整性是如何保证的

本次实验的参照完整性是借助 NodeJS 模块 mongoose-auto-increment 实现的，它有自动增长和模拟外键关联两种功能，同时也保证了参照完整性。

如何用 SQL 的 AO / window function / subquery 解决查询问题？

MongoDB 在 NodeJS 里的查询主要是通过异步控制和参数查询来解决查询问题的。

对于多个查询条件，我们可以一步一步查出结果来。

如何防止外部导入数据时流水号的冲突

mongoose-auto-increment 同时保证了没有流水号冲突，而且也不存在外部导入数据这种功能啊

各种数据表是如何备份的

没有数据表备份功能。

pool_disagrees 这样的字段用静态字段表示和用动态字段表示有何差别？

动态字段消耗时间性能，静态字段消耗空间性能，且不好保证一致性。在 MongoDB 这种数据库中，我推荐使用静态字段，虽然更加浪费空间，但是能够减少硬盘读写，减小对硬盘的伤害。

使用 db adapter 还是 db ORM？

后者，ORM 模型的简单性简化了数据库查询过程。使用 ORM 查询工具，用户可以访问期望数据，而不必理解数据库的底层结构。我们这只是一个作业而已。

关于真正的权限管理是一个有向无环图而不是树

因为现实环境中，父角色如公司老总不一定有子角色如公司 CEO 的权限，这种权限分离是很常见的，如果是树的话，父角色包裹了子角色，应该有子角色的所有权限才对，两者矛盾。因此是一个有向无环图，而不是退化为树。

关于精简掉的那些信息

有些信息如注册时的性别等信息，实际系统中根本用不到，属于数据库的冗余设计，所以我精简掉了。

另外，党费帖老师没有给出对应的模型，所以我也精简掉了。

关于其他问题的设计说明

HTTP redirect 该使用何种 status code，各有何种优劣性

常用的重定向方式有：301 redirect、302 redirect 与 meta fresh。

- 301 redirect

301 代表永久性转移，301 重定向是网页更改地址后对搜索引擎友好的最好方法，只要不是暂时搬移的情况，都建议使用 301 来做转址。

- 302 redirect

302 代表暂时性转移，在前些年，不少 Black Hat SEO（黑帽 SEO）曾广泛应用这项技术作弊，目前，各大主要搜索引擎均加强了打击力度，像 Google 前些年对域名之王(Business ）以及近来对 BMW 德国网站的惩罚。即使网站客观上不是 spam，也很容易被搜寻引擎容易误判为 spam 而遭到惩罚。

- meta fresh

这在 2000 年前比较流行，不过现在已很少见。其具体是通过网页中的 meta 指令，在特定时间后重定向到新的网页，如果延迟的时间太短(约 5 秒之内)，会被判断为 spam。

另外还有另外一些情况

- 302（Found）

请求的资源现在临时从不同的 URI 响应请求。除非指定了 Cache-Control 或 Expires，否则该响应不可缓存。 如果当前请求非 HEAD 或 GET，浏览器需取得用户确认，再进行重定向。这很好理解，因为上下文发生了变化，比如 POST 请求不是幂等的。

- 303（See Other）

对应当前请求的响应可以在另一个 URI 上被找到，而且客户端应当采用 GET 的方式访问那个资源。 这个方法的存在主要是为了允许由脚本激活的 POST 请求输出重定向到一个新的资源。 303 响应禁止被缓存。303 会使得浏览器直接 GET 那个资源，不需用户同意。这是 Web 应用中最常见的重定向方式。

- 304（Not Modified）

如果客户端发送了一个带条件的 GET 请求且该请求已被允许，而文档的内容（自上次访问

以来或者根据请求的条件）并没有改变。 304 响应禁止包含消息体。

304 响应也是一种缓存机制。Web 服务器对静态资源文件通常会采取缓存，因此在 Web 开发中你可以看到大量的 304 响应。 服务器给出的相应中通常会包含 Etag 来标识资源 ID

控制器与功能之间的是如何对应的？

在这个作业中，控制器主要体现在 express 的路由上，路由负责调用数据库操作的接口，也就是调用 Model，各种 Model 则负责与数据库沟通，操作数据，体现各种业务逻辑。

权限管理的 UI 设计如何解决树的直观性与树难以二维显示的两难境地、而使用 HTML table 显示又存在不够直观的问题

HTML Table 也可以设计得很直观的，可以看我具体的设计，就是占用空间比较大 = =

前后端交互设计

概述

后端 API 是在 RESTful 风格的,发送和返回内容均为 json 数据。

若操作成功：

- GET 直接返回获取的内容
- POST 和 PUT 返回修改或创建的条目的 ID，以及成功消息
- DELETE 返回成功消息

正常返回数据：

```
{
    code : '0',             // '0' 代表成功
```

```
    data : {data}           // 按实际要求封装
}
```

无返回数据：

```
{
    code : '0',             // '0' 代表成功
    data : null
}
```

异常返回：

```
{
    code :  String          // 非 '0' 时表示异常代码
    msg  :  String          // 具体的错误信息
}
```

数据

学生用户登录注册相关

1001 注册

```
POST /register
Params {
    registerName    :  String （用户名）
    registerPass    :  String （加密后的密码）
}
Response {
```

```
    code : '0'
}

Response Excetion : { code : '', msg : ''}
code    msg
1001A   登录名已存在
1002 登录并获取统计数据
POST /login
Params {
    loginName   : String （用户名）
    loginPass   : String （加密后的密码）
}
Response {
    code : '0',
    data : {
      statistics : {              // 统计数据
        howMuch         : Number,   // 已经交了多少班费
        howMuchRemain   : Number,   // 还差多少要交
        numOfDayNotSign : Number,   // 旷课总数
        dayNotSign      : [Date]    // 旷课情况
      }
    }
```

```
}
```

Response Excetion : {code : '', msg : ''}

code msg

1002A 用户不存在

1002B 密码不正确

1003 登出

GET /logout

Params

```
    null
```

```
Response {
    code : '0'
}
```

Response Excetion : {code : '', msg : ''}

code msg

1003A 未知错误

管理员相关

2001 获取所有学生

GET /manager/students

Params

```
    null
```

```
Response {
    code : '0',
    data : {
      students : [{
        authorId : ObjectId, // 学生的流水号，也是学号 Id
        roleId   : Number    // 学生的 roleId
      }]
    }
}
```

```
Response Excetion : {code : '', msg : ''}
code     msg
2001A    数据库查询错误
2002 更新某个学生的权限
PUT /manager/update
Params {
  student : {
    authorId : ObjectId, // 学生的流水号，也是学号 Id
    roleId   : Number    // 学生的新 roleId
  }
}
Response
```

```
{
    code : '0'
}
```

```
Response Excetion : {code : '', msg : ''}
code     msg
2002A    找不到这个学生
2002B    没有这种权限等级
```

帖子相关

```
3001 获取某个班级全部开班会帖
GET /meeting
Params {
  classId : String // 哪个班
}
Response {
    code : '0',
    data : {
      meetings : [{
        classId     : String,   // 班级 ID
        date        : Date,     // 开会时间
        place       : String,   // 开会地点
        title       : String,   // 主题
```

```
        num         : Number,   // 应到人数
        gotten      : Number,   // 实到人数
        pAgree      : [Number], // 同意的人的 ID
        pDisagree   : [Number], // 不同意的人的 ID
        authorId    : Number    // 发帖人 ID
      }]
    }
}
```

```
Response Excetion : {code : '', msg : ''}
code     msg
3001A   数据库查询错误
3002 发表开班会帖
POST /meeting
Params {
  meeting : {
    classId     : String,   // 班级 ID
    date        : Date,     // 开会时间
    place       : String,   // 开会地点
    title       : String,   // 主题
    num         : Number,   // 应到人数
    gotten      : Number,   // 实到人数
```

```
    pAgree      : [Number], // 同意的人的 ID
    pDisagree   : [Number], // 不同意的人的 ID
    authorId    : Number    // 发帖人 ID
  }
}
Response {
  code : '0'
}

Response Excetion : {code : '', msg : ''}
code     msg
3002A   数据库查询错误
3003 获取全部收班费帖
GET /classMoney
Params {
  classId : String // 哪个班
}
Response
{
    code : '0',
    data : {
      classMoneys : [{
```

```
        classId         : String,   // 班级 ID
        date            : Date,     // 截止时间
        payedMembers    : [Number], // 已经缴费的人的 ID
        unpayedMembers  : [Number], // 尚未缴费的人的 ID
        authorId        : Number    // 发帖人 ID
      }]
    }
}
```

```
Response Excetion : {code : '', msg : ''}
code     msg
3003A   数据库查询错误
3004 发表收班费帖
POST /classMoney
Params {
  classMoneys : {
    classId         : String,   // 班级 ID
    date            : Date,     // 截止时间
    payedMembers    : [Number], // 已经缴费的人的 ID
    unpayedMembers  : [Number], // 尚未缴费的人的 ID
    authorId        : Number    // 发帖人 ID
  }
```

```
}
Response {
  code : '0'
}

Response Excetion : {code : '', msg : ''}
code     msg
3004A   数据库查询错误
3005 获取评优等生帖
GET /goodStudent
Params {
  classId : String // 哪个班
}
Response
{
    code : '0',
    data : {
      goodStudents : [{
        classId     : String,   // 班级 ID
        title       : String,   // 奖项名称
        stuId       : Number,   // 获奖人 ID
        bond        : Number,   // 奖金
```

```
        authorId    : Number    // 发帖人 ID
      }]
    }
}
```

```
Response Excetion : {code : '', msg : ''}
code     msg
3005A   数据库查询错误
3006 发表评优等生帖
POST /goodStudent
Params {
  goodStudents  : [{
    classId     : String,   // 班级 ID
    title       : String,   // 奖项名称
    stuId       : Number,   // 获奖人 ID
    bond        : Number,   // 奖金
    authorId    : Number    // 发帖人 ID
  }]
}
Response {
  code : '0'
}
```

```
Response Excetion : {code : '', msg : ''}
```

code	msg
3004A	数据库查询错误

关于其他发布/获取全部帖子接口

考勤结果帖接口和回复帖接口结构与其他帖子的类似，不再赘述

三、成果展示

（给出程序的运行效果、同时说明这些效果是如何解决对应的实验要求、设计说明中提出的问题的；没有给出说明对应关系的截图将无法获得有效得分）

1.注册和登录部分

a) 注册

b) 登录

c）登录成功后自动跳转

左下角显示了统计结果，然后左上角是各种帖子 目前是普通学生只能发表回复帖，其他帖子只能看但是没有发布用的表单

2.管理员部分

切到/manager 下

可以看到主界面，

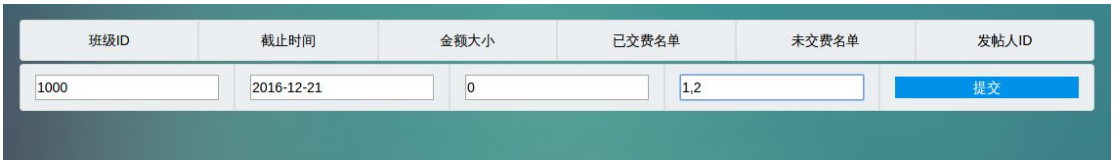
输入密码 maple，打开管理面板

可以看到刚刚注册的学生是 0 号（系统自动分配，从 0 开始自增），他是一个普通学生，我们现在把他改成班长

再次登录可以看到能发开班会帖子了 证明权限已经改变

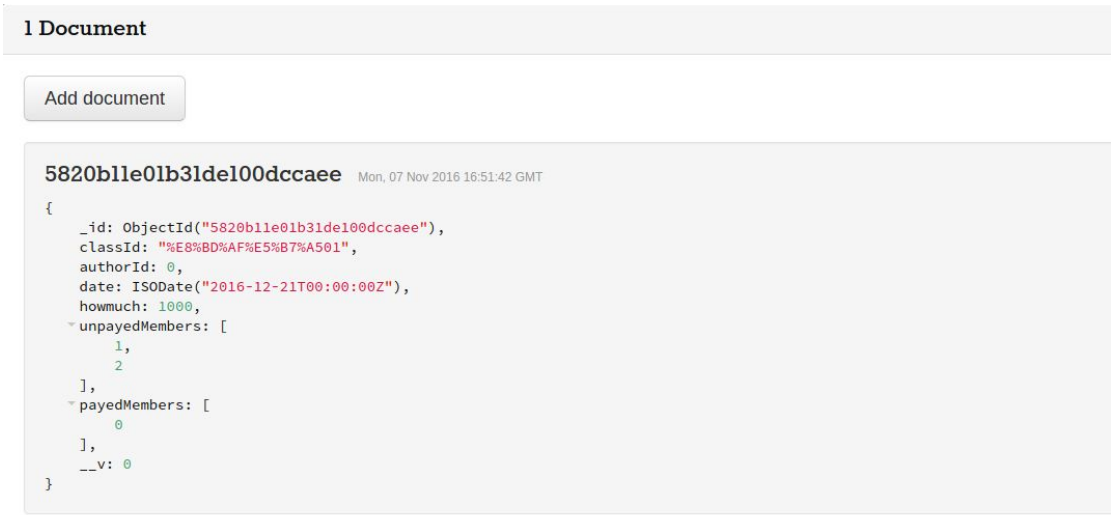
3.发帖相关

我们尝试发收班费帖

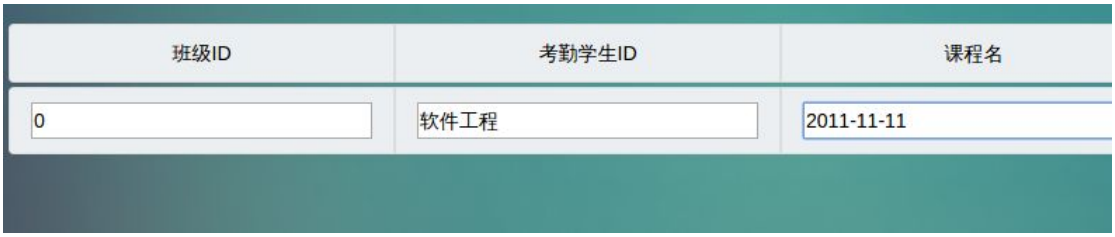


发帖结果

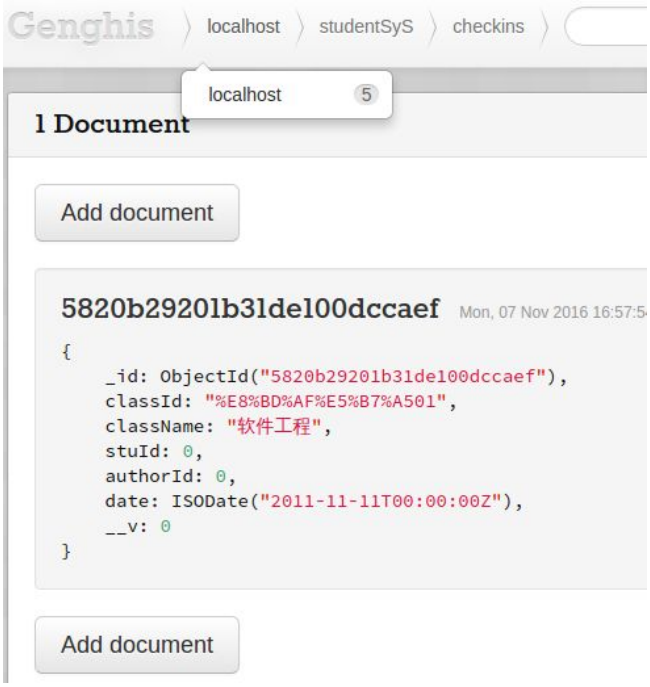
此时数据库结果



我们尝试发考勤帖



结果



4.统计相关

再次登录，我们看到了统计结果

四、实验总结

还可以考虑下响应式设计、部署到云服务器

指导教师批阅意见：

成绩评定：

指导教师签字：

年　　月　　日

备注：

注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。

2、教师批改学生实验报告时间应在学生提交实验报告时间后 10 日内。