

**期末项目设计报告**

|  |  |  |  |
| --- | --- | --- | --- |
| 题 目 | 基于GitHub的实验管理平台的详细设计 | | |
| 课程 | 软件系统分析与设计技术 | | |
| 学 院 | 信息科学与工程学院 | | |
| 专 业 | 软件工程 | 年级 | 2016级 |
| 学生姓名 | 王宇 | 学号 | 201610414421 |
| 指导教师 | 赵卫东 | 职称 | 副教授 |

|  |  |  |  |
| --- | --- | --- | --- |
| **评分项** | **评分标准** | **满分** | **得分** |
| 文档整体 | 文档内容详实、规范，美观大方 | 20 |  |
| 用例图及规约 | 用例图完整，准确，能够完全体现需求 | 10 |  |
| 顺序图与活动图 | 顺序图能够完全描述用例的设计思路和业务流程，活动图能够描述部分算法的流程。 | 10 |  |
| 类图 | 类图能够完整，准确反映业务的需数据的组织结构 | 10 |  |
| 数据库 | 数据库表的设计来自于类图，合理 | 10 |  |
| 界面设计 | 界面设计美观，清晰，合理，能够完全反映用例图的设计思路 | 10 |  |
| API接口设计 | 为每个界面设计合理的API接口，以便进行前后端分离开发 | 15 |  |
| 内容一致性 | 用例图，类图、数据库、界面相互印证，相互依赖，环环相扣 | 15 |  |
| **得分合计** | | |  |

2019 年 5 月 1 日

目录

[1. 概述 5](#_Toc8013042)

[2. 系统总体结构 5](#_Toc8013043)

[3. 用例图设计 5](#_Toc8013044)

[3.1原码 5](#_Toc8013045)

[3.2用例图 6](#_Toc8013046)

[4. 类图设计 7](#_Toc8013047)

[4.1原码 7](#_Toc8013048)

[4.2 类图 8](#_Toc8013049)

[5. 数据库设计 9](#_Toc8013050)

[5.1 USERS表（用户表） 9](#_Toc8013051)

[5.2 TEACHERS表（老师表） 9](#_Toc8013052)

[5.3 STUDENTS表（学生表） 9](#_Toc8013053)

[5.4 GRADES表（学生实验成绩表） 10](#_Toc8013054)

[5.5 TESTS表（实验项目表） 10](#_Toc8013055)

[5.6 COURSES表（课程表） 10](#_Toc8013056)

[6. 用例及界面详细设计 11](#_Toc8013057)

[6.1“学生列表”用例 11](#_Toc8013058)

[6.1.1 用例规约 11](#_Toc8013059)

[6.1.2 业务流程（顺序图） 11](#_Toc8013060)

[6.1.3. 界面设计 12](#_Toc8013061)

[6.1.4 算法描述 14](#_Toc8013062)

[6.1.5 参照表 14](#_Toc8013063)

[6.2 评定成绩 14](#_Toc8013064)

[6.2.1 用例规约 14](#_Toc8013065)

[6.2.2 业务流程（顺序图） 15](#_Toc8013066)

[6.2.3 界面设计 15](#_Toc8013067)

[6.2.4 算法描述 18](#_Toc8013068)

[6.2.5 参照表 18](#_Toc8013069)

[6.3查看成绩 18](#_Toc8013070)

[6.3.1 用例规约 18](#_Toc8013071)

[6.3.2 业务流程（顺序图） 19](#_Toc8013072)

[6.3.3 界面设计 19](#_Toc8013073)

[6.3.4 算法描述 21](#_Toc8013074)

[6.3.5 参照表 21](#_Toc8013075)

[6.4修改密码 21](#_Toc8013076)

[6.4.1 用例规约 21](#_Toc8013077)

[6.4.2 业务流程 22](#_Toc8013078)

[6.4.3 界面设计 22](#_Toc8013079)

[6.4.4 算法描述 23](#_Toc8013080)

[6.4.5 参照表 23](#_Toc8013081)

[6.5 修改用户信息 23](#_Toc8013082)

[6.5.1 用例规约 23](#_Toc8013083)

[6.5.2 业务流程 23](#_Toc8013084)

[6.5.3 界面设计 24](#_Toc8013085)

[6.5.4 算法描述 26](#_Toc8013086)

[6.5.5 参照表 26](#_Toc8013087)

[6.6查看用户信息 26](#_Toc8013088)

[6.6.1用例规约 26](#_Toc8013089)

[6.6.2 业务流程 27](#_Toc8013090)

[6.6.3 界面设计 27](#_Toc8013091)

[6.6.4 算法描述 28](#_Toc8013092)

[6.6.5 参照表 28](#_Toc8013093)

[6.7登出 28](#_Toc8013094)

[6.7.1用例规约 28](#_Toc8013095)

[6.7.2 业务流程 29](#_Toc8013096)

[6.7.3 界面设计 29](#_Toc8013097)

[6.7.4 算法描述 30](#_Toc8013098)

[6.7.5 参照表 30](#_Toc8013099)

[6.8登录 30](#_Toc8013100)

[6.8.1用例规约 30](#_Toc8013101)

[6.8.2 业务流程 31](#_Toc8013102)

[6.8.3 界面设计 31](#_Toc8013103)

[6.8.4 算法描述 32](#_Toc8013104)

[6.8.5 参照表 33](#_Toc8013105)

[6.9选择课程 33](#_Toc8013106)

[6.9.1用例规约 33](#_Toc8013107)

[6.9.2 业务流程 34](#_Toc8013108)

[6.9.3 界面设计 35](#_Toc8013109)

[6.9.4 算法描述 37](#_Toc8013110)

[6.9.5 参照表 37](#_Toc8013111)

[6.10添加课程 37](#_Toc8013112)

[6.10.1用例规约 37](#_Toc8013113)

[6.10.2 业务流程 37](#_Toc8013114)

[6.10.3 界面设计 38](#_Toc8013115)

[6.10.4 算法描述 39](#_Toc8013116)

[6.10.5 参照表 39](#_Toc8013117)

[6.11 添加用户 39](#_Toc8013118)

[6.11.1 用例规约 39](#_Toc8013119)

[6.11.2 业务流程 40](#_Toc8013120)

[6.10.3 界面设计 40](#_Toc8013121)

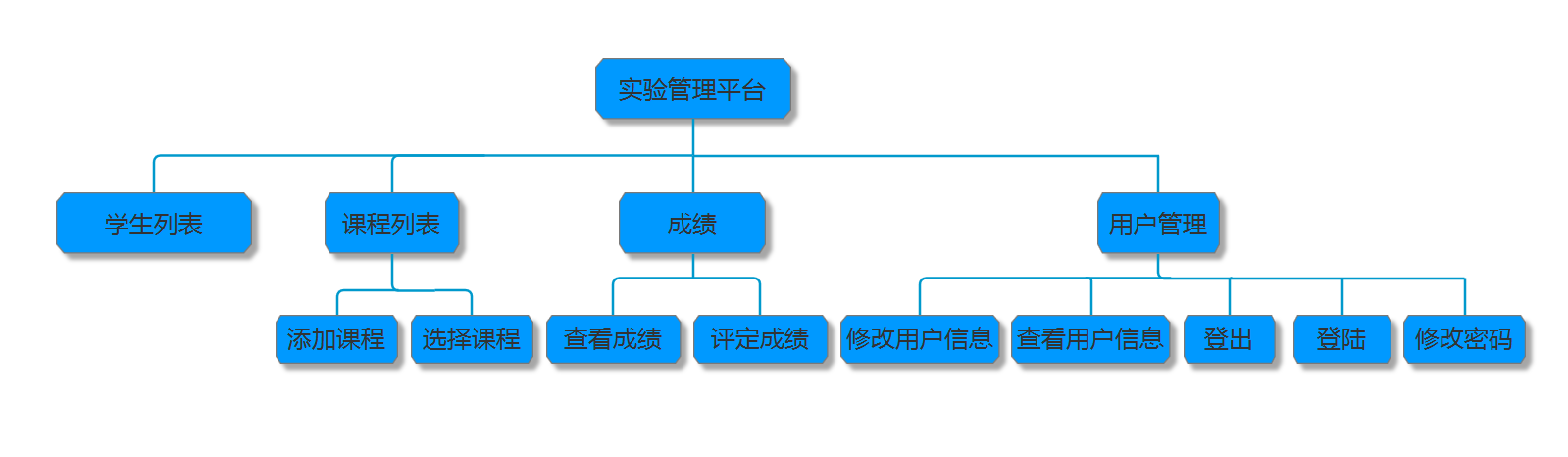
[6.10.4 算法描述 42](#_Toc8013122)

[6.10.5 参照表 42](#_Toc8013123)

# 概述

1. 基于GitHub的实验管理平台的作用是在线管理实验成绩的Web应用系统。学生和老师的实验内容均存放在GitHUB  
   页面上。
2. 学生的功能主要有：一是设置自己的GitHub用户名，二是查询自己的实验成绩，三是选择课程。学生的GitHub用户名是公开的，但成绩不公开。
3. 老师的功能主要有：一是批改每个学生的成绩，二是查看每个学生的成绩，三是挑选选择了自己所教课程的学生。
4. 老师和学生都能通过本系统的链接方便地跳转到学生的每个GitHUB实验目录，以便批改实验或者查看实验情况。
5. 实验成绩按数字分数计算，每项实验的满分为100分，最低为0分。  
   每个实验的实验成绩细分为多个评分项，每个评分项对应各自的评分标准。
6. 老师在批改实验的时候，对每个评分项进行评分并输入对应的文字评价，系统自动计算出所有评分项的成绩之和为该实验的总成绩。  
   系统自动计算每个学生的所有实验的平均分。

## 系统总体结构

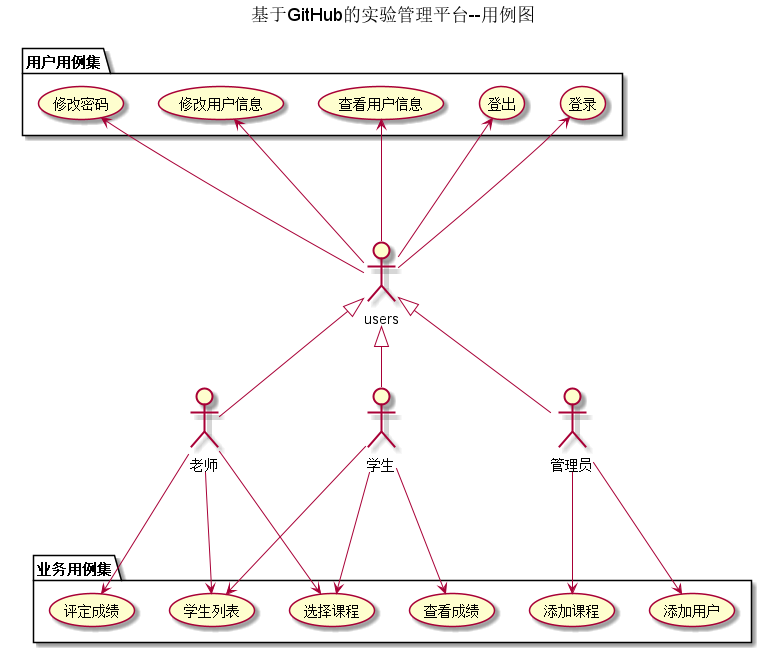


# 用例图设计

## 3.1原码

@startuml  
**title** 基于GitHub的实验管理平台--用例图  
**actor** 老师  
**actor** 学生  
**actor** 管理员  
**actor** users  
users <|-- 老师  
users <|-- 学生  
users <|-- 管理员  
**package** 用户用例集 {  
users --**up**-> (登录)  
users --**up**-> (登出)  
users --**up**-> (查看用户信息)  
users --**up**-> (修改用户信息)  
users --**up**-> (修改密码)  
}  
**package** 业务用例集 {  
老师 ---> (评定成绩)  
老师 ---> (学生列表)  
学生 ---> (学生列表)  
老师 ---> (选择课程)  
学生 ---> (选择课程)  
学生 ---> (查看成绩)  
管理员 ---> (添加课程)  
管理员 ---> (添加用户)  
}  
@enduml

## 3.2用例图

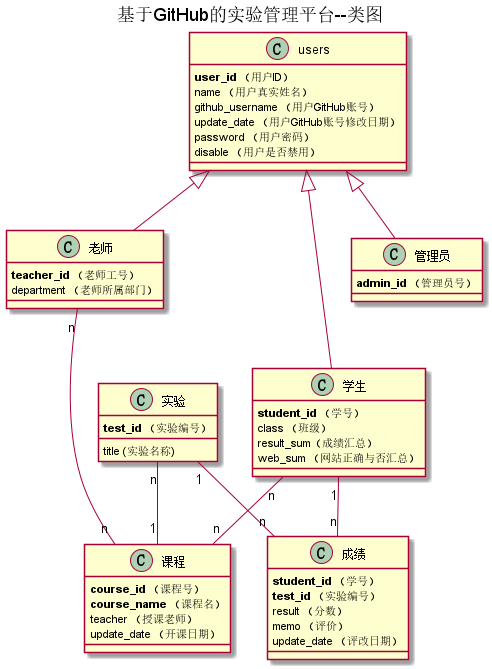


# 类图设计

## 4.1原码

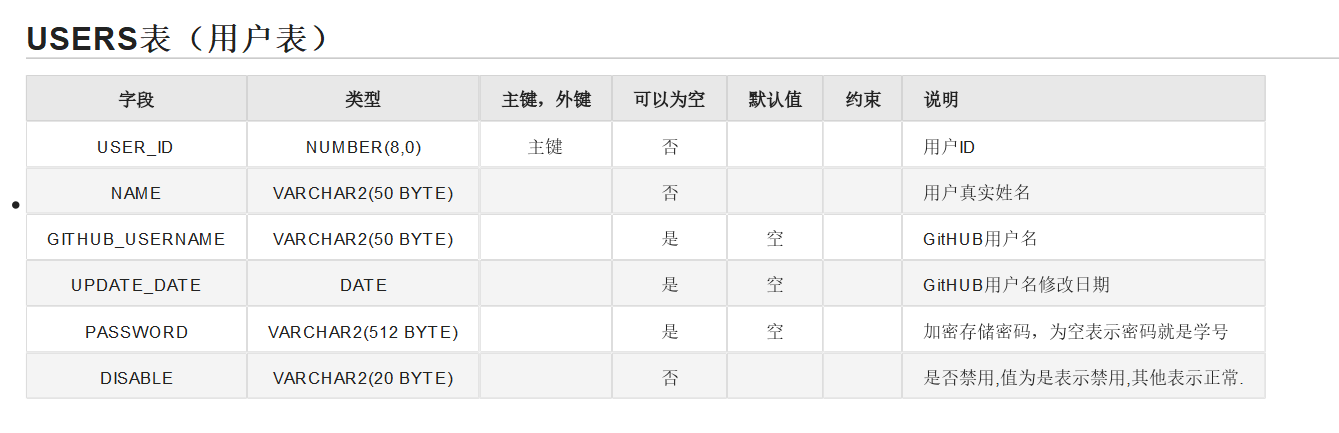
@startuml  
**title** 基于GitHub的实验管理平台--类图  
**class** users {  
 <b>user\_id</b> （用户ID）  
 name （用户真实姓名）  
 github\_username （用户GitHub账号）  
 update\_date （用户GitHub账号修改日期）  
 password （用户密码）  
 disable （用户是否禁用）  
}  
**class** 管理员{  
 <b>admin\_id</b> （管理员号）  
}  
**class** 老师{  
 <b>teacher\_id</b> （老师工号）  
 department （老师所属部门）  
}  
**class** 学生{  
 <b>student\_id</b> （学号）  
 **class** （班级）  
 result\_sum（成绩汇总）  
 web\_sum （网站正确与否汇总）  
}  
users <|-- 管理员  
users <|--- 学生  
users <|-- 老师  
**class** 成绩 {  
 <b>student\_id</b> （学号）  
 <b>test\_id</b> （实验编号）  
 result （分数）  
 memo （评价）  
 update\_date （评改日期）  
}  
**class** 课程{  
 <b>course\_id</b> （课程号）  
 <b>course\_name</b> （课程名）  
 teacher （授课老师）  
 update\_date （开课日期）  
}  
**class** 实验 {  
 <b>test\_id</b> （实验编号）  
 **title** (实验名称)  
}  
学生 "1" -- "n" 成绩  
实验 "1" -- "n" 成绩  
实验 "n" -- "1" 课程  
学生 "n" -- "n" 课程  
老师 "n" - "n" 课程  
@enduml

## 类图

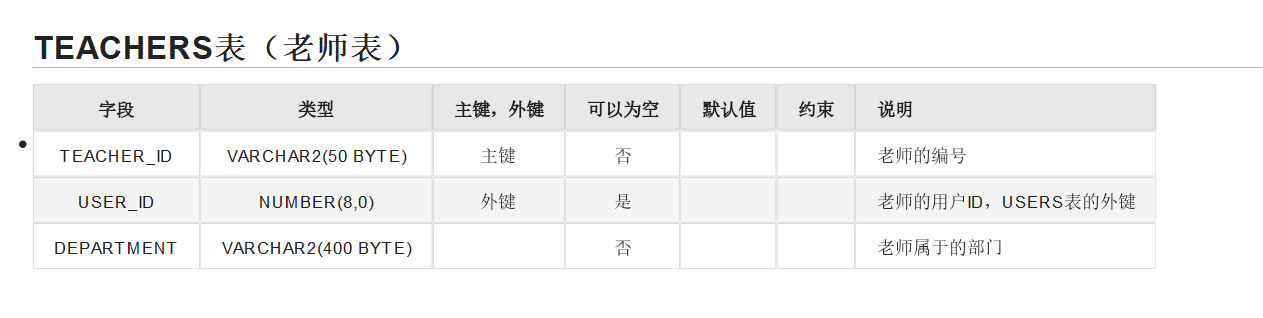


# 数据库设计

## 5.1 USERS表（用户表）



## TEACHERS表（老师表）



## STUDENTS表（学生表）



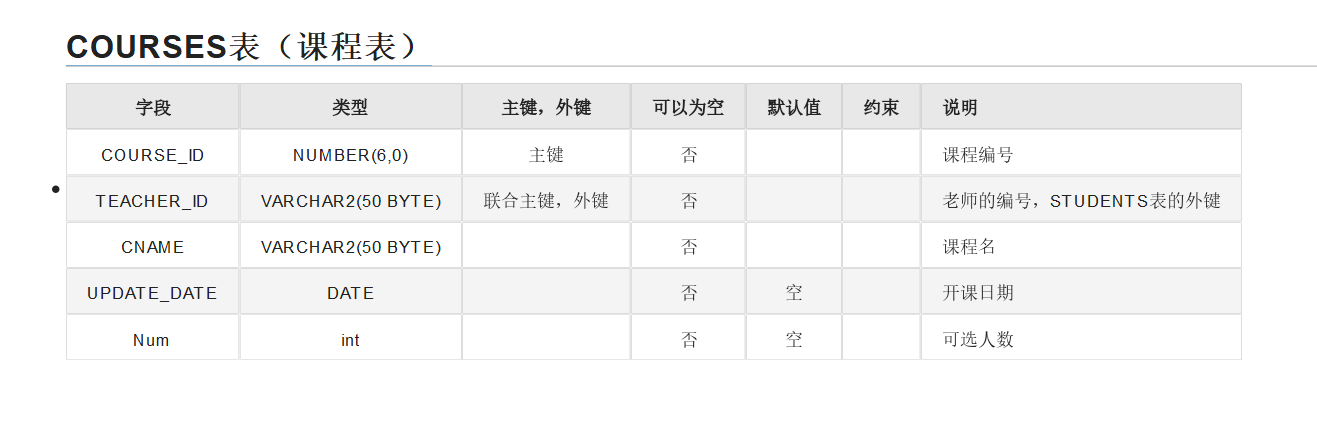
## GRADES表（学生实验成绩表）



## TESTS表（实验项目表）



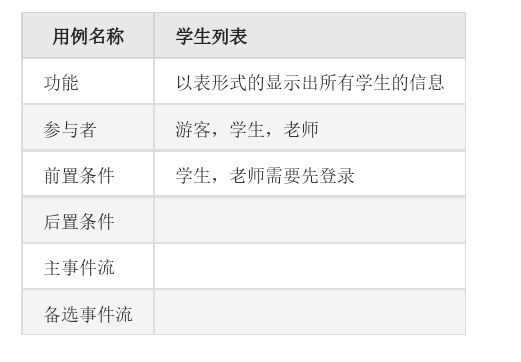
## COURSES表（课程表）



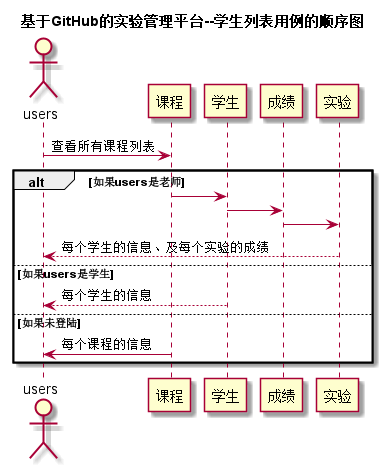
# 用例及界面详细设计

## 6.1“学生列表”用例

6.1.1 用例规约



6.1.2 业务流程（顺序图）



原码：

@startuml  
**title** 基于GitHub的实验管理平台--学生列表用例的顺序图  
**actor** users  
users -> 课程: 查看所有课程列表  
**alt** 如果users是老师  
 课程 -> 学生  
 学生 -> 成绩  
 成绩 -> 实验  
 实验 --> users:每个学生的信息、及每个实验的成绩  
**else** 如果users是学生  
 学生 --> users:每个学生的信息  
**else** 如果未登陆  
 课程 -> users:每个课程的信息  
**end**  
@enduml

6.1.3. 界面设计



**接口调用：**

**# 接口：getStudents   
-** 权限：  
 学生/访客：不能看到RESULT\_SUM，WEB\_SUM  
 老师：可以看到RESULT\_SUM，WEB\_SUM。  
**-** 功能：  
 返回所有学生的列表。  
**-** API请求地址：  
 接口基本地址/v1/api/getStudents  
**-** 请求方式 ：  
 GET  
**-** 请求参数说明:  
 无  
**-** 返回实例：  
 {  
 "status": true,  
 "info": null,  
 "total": 121,  
 "data": [  
 {"WEB\_SUM": "Y,Y,Y,Y,Y,N",  
 "RESULT\_SUM": "83.75,90,80,80,85,N",  
 "GITHUB\_USERNAME": "Chinajuedui",  
 "STUDENT\_ID": "201510315203",  
 "CLASS": "软件(本)15-1",  
 "NAME": "陈松华",  
 "UPDATE\_DATE": "2018-04-02 13:48:01"},  
 {  
 ...其他学生  
 }  
 ]  
 }  
  
**-** 返回参数说明：  


6.1.4 算法描述

* WEB\_SUM解析为列表
  + WEB\_SUM是接口getStudents的返回值中的一部分，形如："WEB\_SUM": "Y,Y,Y,Y,Y,N"。需要在前端进行解析。
  + 作用是判断GitHub网址是否正确，用逗号分开，Y代表正确，N代表不正确。
  + 第1位代表总的GitHUB地址是否正确，第2位表示第1次实验的地址，第3位表示第2位实验地址，依此类推。
  + 比如：“Y,Y,Y,Y,Y,N”表示第5次实验地址不正确，其他地址正确。
  + 算法是使用使用字符串分离方法将其分开。
* RESULT\_SUM解析为列表
  + RESULT\_SUM是接口getStudents的返回值中的一部分，形如："RESULT\_SUM": "81.25,70,80,85,90,N"。需要在前端进行解析。
  + 成绩汇总（来自GRADES表），以逗号分开，第一个成绩是平均成绩,后面是每次实验的成绩，N表示未批改，平均分只计算已批改的。
  + 比如：“81.25,70,80,85,90,N”表示一共批改了4次，第5次未批改，4次的成绩分别是81.25,70,80,85,90,N，4次的平均分是81.25
  + 算法是使用使用字符串分离方法将其分开。

****6.1.5 参照表****

**STUDENTS**

**GRADES**

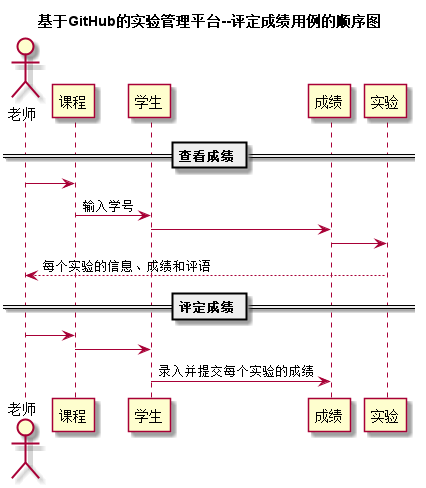
**TESTS**

## 评定成绩

* + 1. 用例规约



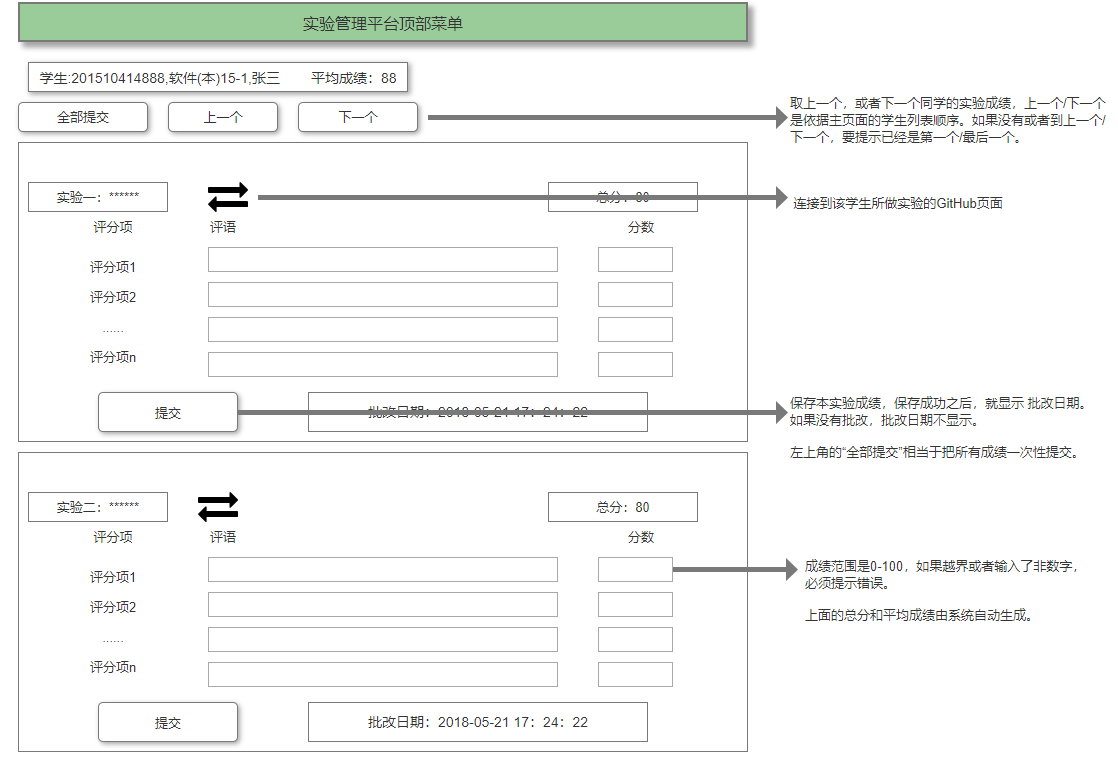
* + 1. 业务流程（顺序图）



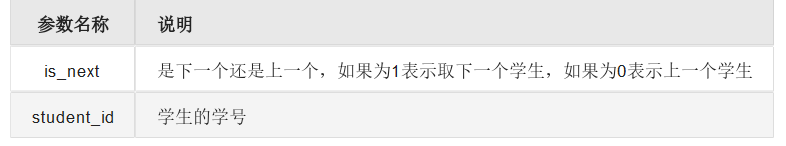
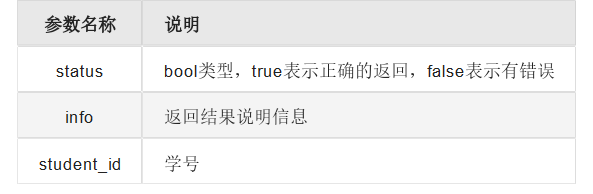
原码：

@startuml  
**title** 基于GitHub的实验管理平台--评定成绩用例的顺序图  
**actor** 老师  
== 查看成绩 ==  
老师 -> 课程  
课程 -> 学生 : 输入学号  
学生 -> 成绩  
成绩 -> 实验  
实验 --> 老师:每个实验的信息、成绩和评语  
== 评定成绩 ==  
老师 -> 课程  
课程 -> 学生  
学生 -> 成绩 : 录入并提交每个实验的成绩  
@enduml

* + 1. 界面设计



接口调用：

接口1： getNextPrevStudent  **-** 功能：  
 返回一个学生的上一个或者下一个学生的学号。  
**-** 权限：   
 老师：只有老师可以调用该API。  
**-** API请求地址：   
 接口基本地址/v1/api/getNextPrevStudent/<is\_next>/<student\_id>  
**-** 请求方式 ：  
 GET  
**-** 请求参数说明:   
  **-** 返回实例：  
 {   
 "status": true,  
 "info": null,   
 "student\_id": "201510315203"  
 }  
**-** 返回参数说明：   


接口2：getOneStudentResults  **-** 功能：  
 返回一个学生的所有实验成绩和实验评价。  
**-** 权限：  
 学生：只能查看自己的成绩，即接口参数student\_id必须等于登录学生的student\_id  
 老师：可以查看所有学生的成绩。  
**-** API请求地址：   
 接口基本地址/v1/api/getOneStudentResults/<student\_id>  
**-** 请求方式 ：

GET **-** 返回实例：   
 {   
 "status": true,  
 "info": null,   
 "student\_id": "201510315203",   
 "github\_username": "chinajuedui",   
 "class": "软件(本)15-1",   
 "name": "陈松华",   
 "total": 6,  
 "avgresult":90.5,   
 "data": [  
 {  
 "test\_id":1,  
 "web\_exists": true,   
 "result": 91,   
 "memo":"本实验做得好",  
 "update\_date": "2018-04-02 13:48:01"  
 },   
 {  
 ...其他实验  
 }  
 ]   
 }  
**-** 返回参数说明：   


* + 1. 算法描述

无

6.2.5 参照表

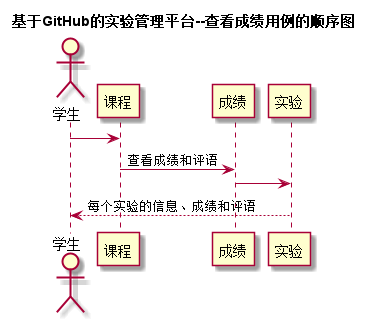
* STUDENTS
* GRADES
* TESTS

## 6.3查看成绩

6.3.1 用例规约



6.3.2 业务流程（顺序图）



原码：

@startuml  
**title** 基于GitHub的实验管理平台--查看成绩用例的顺序图  
**actor** 学生  
学生 -> 课程  
课程 -> 成绩: 查看成绩和评语  
成绩 -> 实验  
实验 --> 学生:每个实验的信息、成绩和评语  
@enduml

6.3.3 界面设计



接口调用：

接口：getOneStudentResults  **-** 功能：  
 返回一个学生的所有实验成绩和实验评价。  
**-** 权限：  
 学生：只能查看自己的成绩，即接口参数student\_id必须等于登录学生的student\_id  
 老师：可以查看所有学生的成绩。  
**-** API请求地址：   
 接口基本地址/v1/api/getOneStudentResults/<student\_id>  
**-** 请求方式 ：

GET **-** 返回实例：   
 {   
 "status": true,  
 "info": null,   
 "student\_id": "201510315203",   
 "github\_username": "chinajuedui",   
 "class": "软件(本)15-1",   
 "name": "陈松华",   
 "total": 6,  
 "avgresult":90.5,   
 "data": [  
 {  
 "test\_id":1,  
 "web\_exists": true,   
 "result": 91,   
 "memo":"本实验做得好",  
 "update\_date": "2018-04-02 13:48:01"  
 },   
 {  
 ...其他实验  
 }  
 ]   
 }  
**-** 返回参数说明：   


6.3.4 算法描述

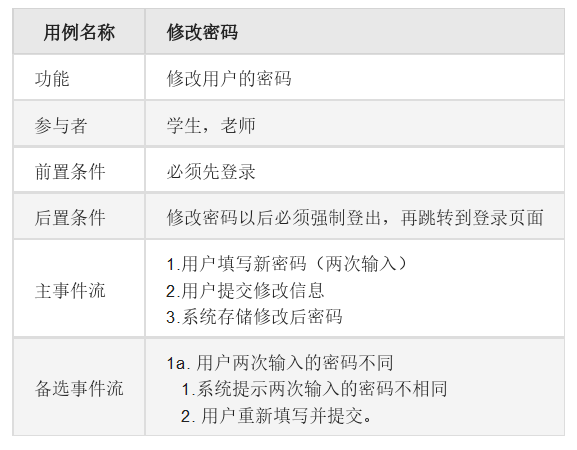
无

6.3.5 参照表

* STUDENTS
* GRADES
* TESTS

## 6.4修改密码

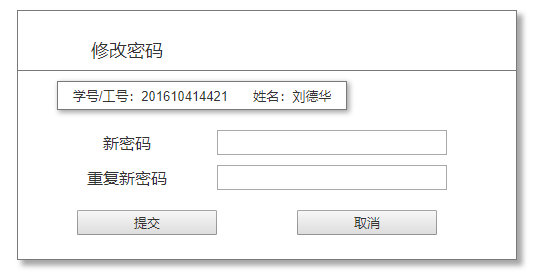
6.4.1 用例规约



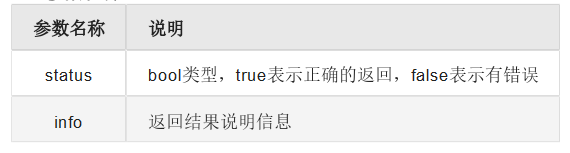
6.4.2 业务流程

无

6.4.3 界面设计



接口调用：

接口：setPassword  
- 功能：  
 修改（设置）用户的密码。  
- 权限：  
 学生/老师：修改自己的密码，必须先登录。   
- API请求地址：   
 接口基本地址/v1/api/setPassword  
- 请求方式 ：  
 POST  
- 请求实例：   
 {  
 "user\_id":1234,  
 "password":AF#W@#AAAASDF  
 }  
- 请求参数说明:   
  
- 返回实例：   
 {   
 "status": true,  
 "info": null,   
 }  
- 返回参数说明：   


6.4.4 算法描述

无

6.4.5 参照表

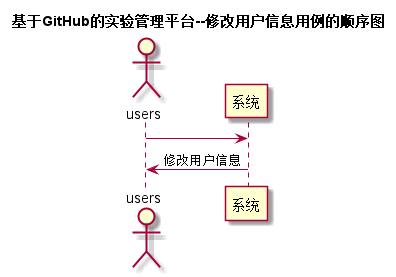
* USERS

## 6.5 修改用户信息

6.5.1 用例规约



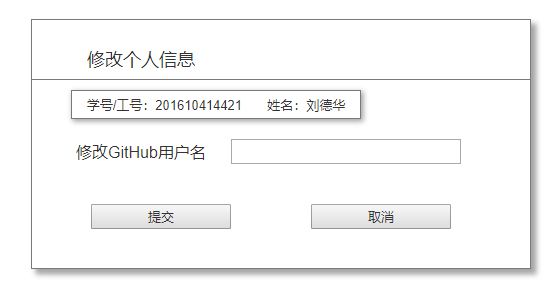
6.5.2 业务流程



原码：

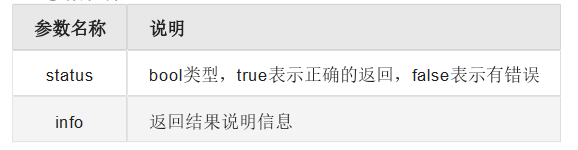
@startuml  
**title** 基于GitHub的实验管理平台--修改用户信息用例的顺序图  
**actor** users  
users -> 系统  
系统 -> users: 修改用户信息  
@enduml

6.5.3 界面设计



接口调用：

接口：getUserInfo **-** 功能：  
 查看用户的所有信息。  
**-** 权限：  
 学生/老师：查看自己的信息，必须先登录，不能查看其他用户的信息。   
**-** API请求地址：   
 接口基本地址/v1/api/getUserInfo/<user\_id>  
**-** 请求方式 ：  
 GET  
**-** 请求参数说明:   
 **-** 返回实例：   
 {   
 "status": true,  
 "info": null,  
 "ID":"2101201281",   
 "name":"张三",  
 "class\_dept":"软件工程1班"  
 "github\_username":"ABCD",  
 "type":"学生"   
 }  
**-** 返回参数说明：   


接口：setUserInfo  **-** 功能：  
 修改用户的GitHub用户名。  
**-** 权限：  
 学生/老师：修改自己的密码，必须先登录。   
**-** API请求地址：   
 接口基本地址/v1/api/setUserInfo  
**-** 请求方式 ：  
 POST  
**-** 请求实例：   
 {  
 "id":"21048329823",  
 "github\_username":"ABCDE",   
 }  
   
**-** 请求参数说明:   
 **-** 返回实例：   
 {   
 "status": true,  
 "info": null  
 }  
**-** 返回参数说明：   


6.5.4 算法描述

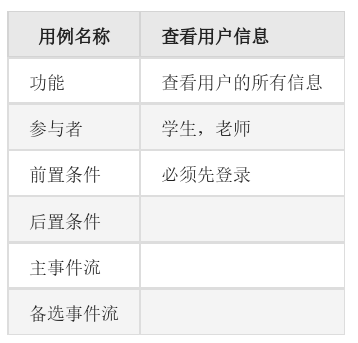
无

6.5.5 参照表

* USERS

## 6.6查看用户信息

6.6.1用例规约



6.6.2 业务流程

无

6.6.3 界面设计



接口调用：

接口：getUserInfo **-** 功能：  
 查看用户的所有信息。  
**-** 权限：  
 学生/老师：查看自己的信息，必须先登录，不能查看其他用户的信息。   
**-** API请求地址：   
 接口基本地址/v1/api/getUserInfo/<user\_id>  
**-** 请求方式 ：  
 GET  
**-** 请求参数说明:   
 **-** 返回实例：   
 {   
 "status": true,  
 "info": null,  
 "ID":"2101201281",   
 "name":"张三",  
 "class\_dept":"软件工程1班"  
 "github\_username":"ABCD",  
 "type":"学生"   
 }  
**-** 返回参数说明：   


6.6.4 算法描述

无

6.6.5 参照表

* STUDENTS
* TEACHERS
* USERS

## 6.7登出

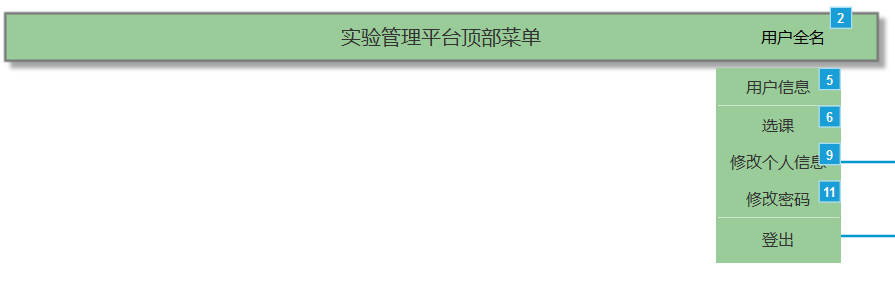
6.7.1用例规约



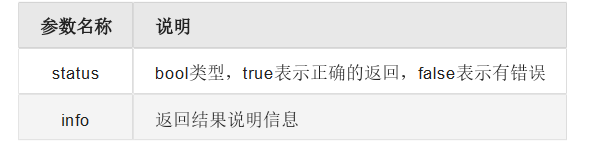
6.7.2 业务流程

无

6.7.3 界面设计



接口调用：

接口：logout   
- 功能：  
 已经登录的用户登出平台。  
- 权限：  
 学生/老师已经登录。   
- API请求地址：   
 接口基本地址/v1/api/logout  
- 请求方式 ：  
 POST  
- 请求实例：   
 {  
 "user\_id":324111  
 }  
- 请求参数说明:   
  
- 返回实例：   
 {   
 "status": true,  
 "info": null,   
 }  
- 返回参数说明：   
   
 

6.7.4 算法描述

无

6.7.5 参照表

* USERS

## 6.8登录

6.8.1用例规约



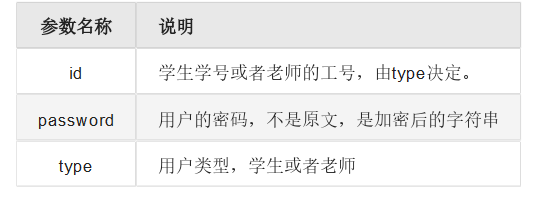
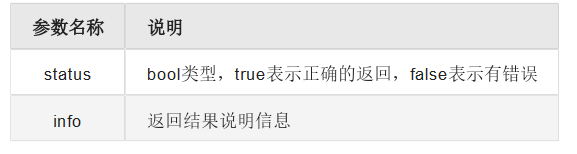
6.8.2 业务流程

无

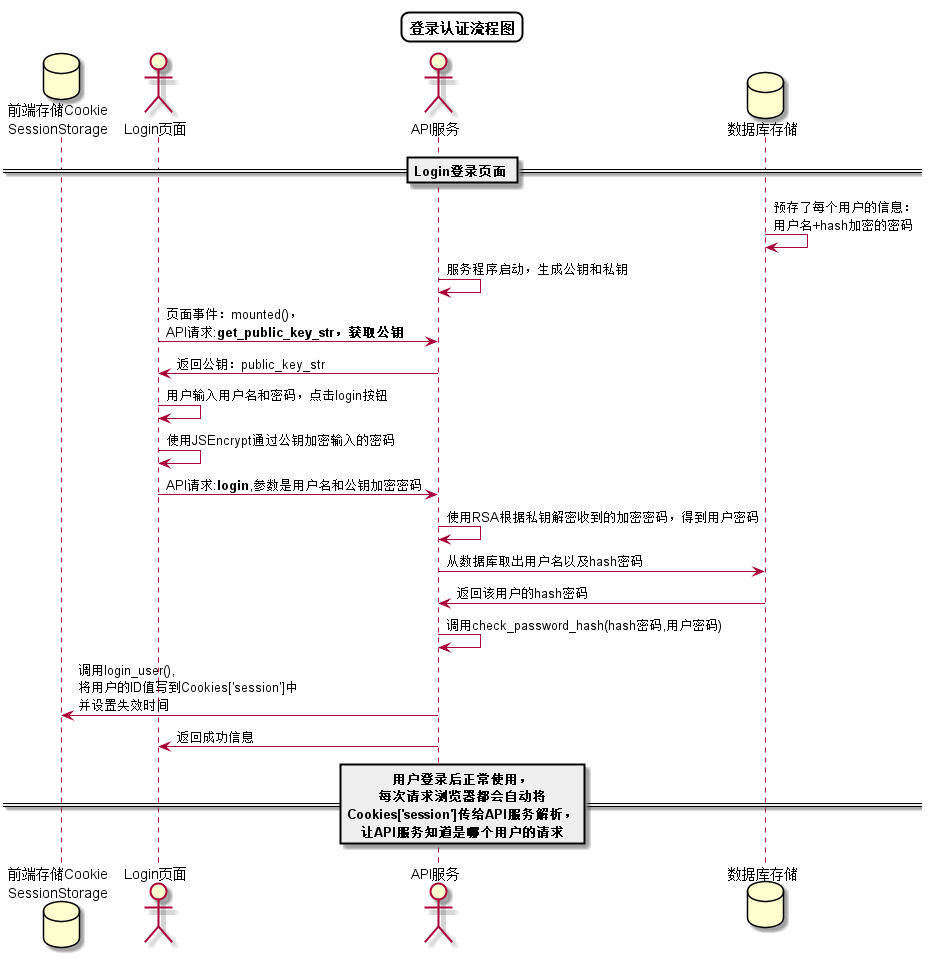
6.8.3 界面设计



接口调用：

接口：login [返回](../README.md)  
用例： [登录](../用例/登录用例.md)   
- 功能：  
 登录到平台。  
- 权限：  
 访客。   
- API请求地址：   
 接口基本地址/v1/api/login  
- 请求方式 ：  
 POST  
- 请求实例：   
 {  
 "id":"21048329823",  
 "password":"ABCDE",  
 "type":"学生"  
 }  
- 请求参数说明:   
  
- 返回实例：   
 {   
 "status": true,  
 "info": null,   
 }  
- 返回参数说明：   


6.8.4 算法描述



原码：

@startuml  
**skinparam** titleBorderRoundCorner 15  
**skinparam** titleBorderThickness 2  
**skinparam** titleBorderColor #Black  
**skinparam** titleFontSize 24  
Title 登录认证流程图  
**database** "前端存储Cookie\nSessionStorage" **as** 前端存储  
**actor** Login页面  
**actor** API服务  
**database** 数据库存储  
== Login登录页面 ==  
数据库存储->数据库存储:预存了每个用户的信息：\n用户名+hash加密的密码  
API服务->API服务:服务程序启动，生成公钥和私钥  
Login页面->API服务:页面事件：mounted()，\nAPI请求:<b>get\_public\_key\_str<b>，获取公钥  
API服务->Login页面:返回公钥：public\_key\_str  
Login页面->Login页面:用户输入用户名和密码，点击login按钮  
Login页面->Login页面:使用JSEncrypt通过公钥加密输入的密码  
Login页面->API服务:API请求:<b>login</b>,参数是用户名和公钥加密密码  
API服务->API服务:使用RSA根据私钥解密收到的加密密码，得到用户密码  
API服务->数据库存储:从数据库取出用户名以及hash密码  
数据库存储->API服务:返回该用户的hash密码  
API服务->API服务:调用check\_password\_hash(hash密码,用户密码)  
API服务->前端存储:调用login\_user(),\n将用户的ID值写到Cookies['session']中\n并设置失效时间  
API服务->Login页面:返回成功信息  
== 用户登录后正常使用，\n每次请求浏览器都会自动将\nCookies['session']传给API服务解析，\n让API服务知道是哪个用户的请求==  
@enduml

6.8.5 参照表

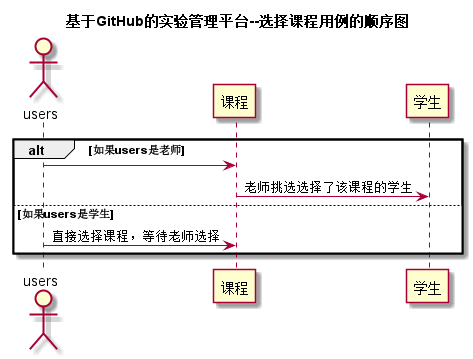
* USERS

## 6.9选择课程

6.9.1用例规约



6.9.2 业务流程



原码：

@startuml  
**title** 基于GitHub的实验管理平台--选择课程用例的顺序图  
**actor** users  
**alt** 如果users是老师  
 users -> 课程  
 课程 -> 学生:老师挑选选择了该课程的学生  
**else** 如果users是学生  
 users -> 课程:直接选择课程，等待老师选择  
**end**  
@enduml

6.9.3 界面设计







接口调用：

接口：selectCourses  
- 功能：  
 返回课程列表   
- 权限：  
 学生：可以选择所有课程  
 老师：只有老师只能挑选已经选择了该课程的学生。  
- API请求地址：   
 接口基本地址/v1/api/selectCourses/<course\_id>/<student\_id>  
- 请求方式 ：  
 GET  
- 请求参数说明:   
   
- 返回实例：   
 {   
 "info": ture,   
 "course\_id": "00001"   
 "student\_id": "201510315203"  
 }  
- 返回参数说明：   
 

6.9.4 算法描述

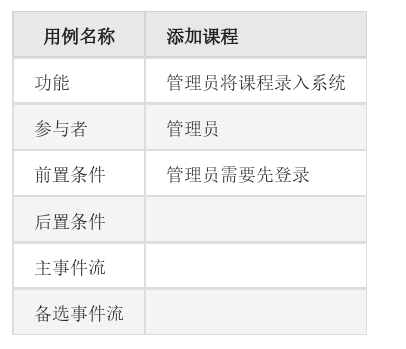
无

6.9.5 参照表

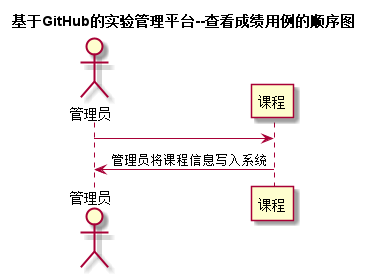
* STUDENTS
* TEACHERS
* COURSES

## 6.10添加课程

6.10.1用例规约



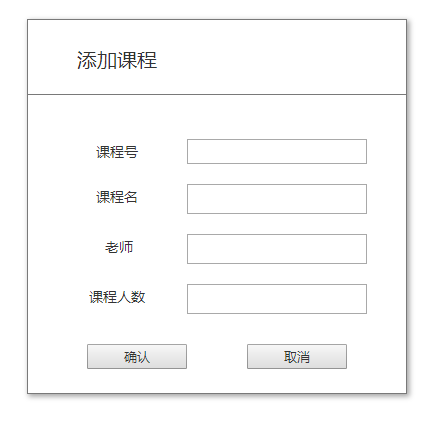
6.10.2 业务流程



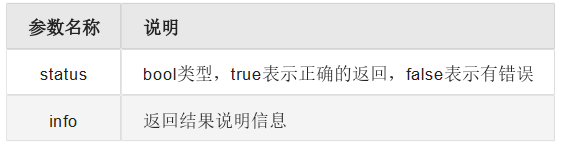
原码：

@startuml  
**title** 基于GitHub的实验管理平台--添加课程用例的顺序图  
**actor** 管理员  
管理员 -> 课程  
课程 -> 管理员: 管理员将课程信息写入系统  
@enduml

6.10.3 界面设计



接口调用：

接口：addCourse  **-** 功能：  
 将课程信息添加进数据库。  
**-** 权限：  
 管理员。   
**-** API请求地址：   
 接口基本地址/v1/api/addCourse  
**-** 请求方式 ：  
 POST  
  
**-** 请求实例：   
 {  
 "course\_id":"000001",  
 "Cname":"计算机网络",  
 "teacher\_id":"21231356651"  
 }   
**-** 请求参数说明:   
 **-** 返回实例：   
 {   
 "status": true,  
 "info": null,   
 }  
**-** 返回参数说明：   
 

6.10.4 算法描述

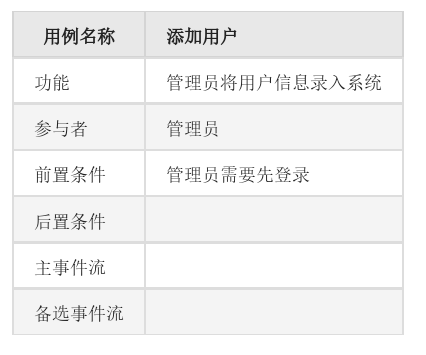
无

6.10.5 参照表

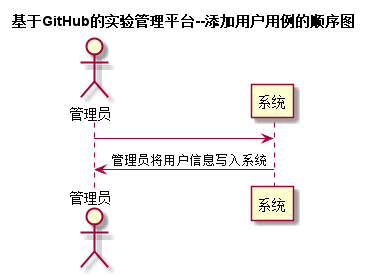
* COURSES
* TEACHERS

## 6.11 添加用户

6.11.1 用例规约



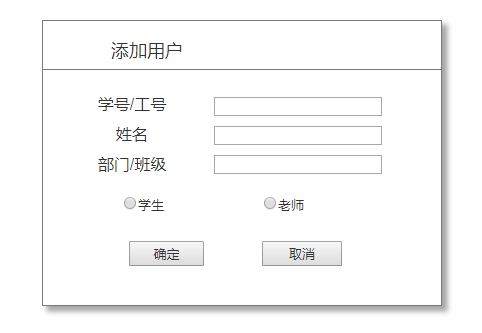
6.11.2 业务流程



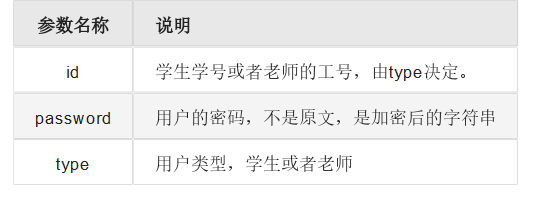
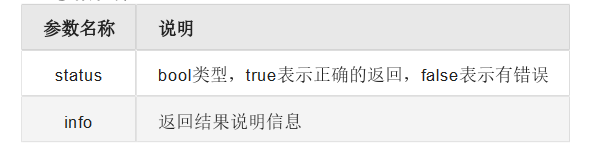
原码：

@startuml  
**title** 基于GitHub的实验管理平台--添加用户用例的顺序图  
**actor** 管理员  
管理员 -> 系统  
系统 -> 管理员: 管理员将用户信息写入系统  
@enduml

6.10.3 界面设计



接口调用：

接口：addUsers   
- 功能：  
 将用户信息添加进数据库。  
- 权限：  
 管理员。   
- API请求地址：   
 接口基本地址/v1/api/addUsers  
- 请求方式 ：  
 POST  
- 请求实例：   
 {  
 "id":"21048329823",  
 "password":"ABCDE",  
 "type":"学生"  
 }  
- 请求参数说明:   
  
- 返回实例：   
 {   
 "status": true,  
 "info": null,   
 }  
- 返回参数说明：   


6.10.4 算法描述

无

6.10.5 参照表

* USERS