**详细设计报告**



项目名称：表白墙网站开发

专业班级： 软件工程1903

小组成员：吴联想、王义博、郑航舰

指导教师： 杨枨老师

二零二一年十一月十四日

说明

l.《系统/子系统设计(结构设计)说明》(SSDD)描述了系统或子系统的系统级或子系统级设计与体系结构设计。SSDD可能还要用《接口设计说明》(IDD)和《数据库(顶层)设计说明》(DBDD)加以补充。

2.SSDD连同相关的IDD和DBDD是构成进一步系统实现的基础。贯穿本文的术语“系统”如果适用的话,也可解释为“子系统”。所形成的文档应冠名为“系统设计说明”或“子系统设计说明”。

**版本记录**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 编号 | 修订日期 | 版本/状态 | 修订人 | 备注 |
| 01 | 2021.11.14 | 0.1 | 郑航舰 | 初稿 |
| 02 | 2021.11.14 | 0.2 | 郑航舰 | 填补内容 |
| 03 | 2021.11.21 | 0.3 | 郑航舰 | 层次图、数据库、PDL、界面原型更新 |
| 04 | 2021.12.15 | 0.4 | 郑航舰 | 接口设计更新 |
| 05 | 2021.12.22 | 0.5 | 郑航舰 | 部分数据结构修改 |
| 06 |  |  |  |  |
| 07 |  |  |  |  |
| 08 |  |  |  |  |
| 09 |  |  |  |  |
| 10 |  |  |  |  |

**目录**

详细设计报告

目录

1.引言

1.1标识

1.2系统概述

1.3文档概述

1.4基线

2引用文件

3 系统级设计决策

4 系统体系结构设计

4.1系统总体设计

4.1.1概述

4.1.1.1功能描述

4.1.1.2运行环境

4.1.2设计思想

4.1.2.1系统构思

4.1.2.2关键技术与算法

4.1.2.3关键数据结构

4.1.3―基本处理流程

4.1.3.1系统流程图

4.1.3.2数据流程图

4.1.4系统体系结构

4.1.4.1系统配置项

4.1.4.2系统层次结构

4.1.4.3系统配置项设计

4.1.5功能需求与系统配置项的关系

4.1.6人工处理过程

4.2系统部件

4.3执行概念

4.4接口设计 4.4.1接口标识和图表 4.4.x（接口的项目唯一标识符)

5运行设计

5.1系统初始化

5.2运行控制

5.3运行结束

6系统出错处理设计

6.1出错信息

6.2补救措施

7系统维护设计

7.1检测点的设计

7.2检测专用模块的设计

8尚待解决的问题

9需求的可追踪性

10注解

附录

1 引言

1.1标识

|  |  |  |
| --- | --- | --- |
| 文件状态：  [ ] 草稿  [ ] 正式发布  [√] 正在修改 | 文件标识： | SE2021-G11-系统/子系统设计(结构设计)说明 |
| 当前版本： | 0.5 |
| 作者： | 吴联想、王义博、郑航舰 |
| 完成日期： | 2021-12-22 |

1.2系统概述

本校园表白墙是一款社交娱乐网站,是一个匿名表白交友社区，通过表白墙可以向喜欢的她或者他进行表白,这是一个大众化的表白信息公布平台,能够让彼此之间避免面对面表白失败的尴尬场面。

项目的投资方：郑航舰、王义博、吴联想；

需方：郑航舰、王义博、吴联想；

用户：城院在校大学生；

开发方：郑航舰、王义博、吴联想；

支持机构：无;

计划的运行现场：浙大城市学院。

1.3文档概述

该文档描述了“系统应该如何实现”这个问题，确定系统中每个程序是由哪些模块组成的，以及这些模块相互间的关系。无与其使用有关的保密性或私密性要求。

1.4基线

《可行性分析报告》0.4。

《软件需求规格说明》0.6。

《总体设计报告》0.2。

2引用文件

《可行性分析报告》0.4

《GB/T 8567-2006》

《软件工程导论（第6版）》

3 系统级设计决策

TBD

4系统体系结构设计

4.1系统总体设计

4.1.1概述

4.1.1.1功能描述



4.1.1.2运行环境

Windos10操作系统

4.1.2设计思想

4.1.2.1系统构思

该系统分为用户、管理员两个模块，用户包含个人 中心、注册、登录、反馈、举报等功能，管理员则对帖子、 用户进行管理，包括删除、冻结等管理方式，他们的操作 都会同步对数据库进行更新。

4.1.2.2关键技术与算法

/\*相关依赖导入\*/

Procedure 登录模块

用户进入登录界面

用户输入账号密码

IF 用户点击登录（@click=login）

methods：{/\*方法调用login函数\*/

login(){

request.post([url:”/api/user/login”).then(res=>{](url:”/api/user/login”).then(res=%3e%7b)/\*向后台发送读取的用户账号密码数据\*/

if(res.code==’0’){/\*后台数据匹配成功\*/

this.$massage({

type:”success”

message:”登录成功”

}）

this.$router.push(“/”) //登录成功之后页面跳转到主页

}else{

this.$massage({

type:”error”

message：res.msg /\*返回登录错误的类型\*/

}）

}

})

}

}

ELSE

this.$router.push(“/login”)/\*返回到登录界面\*/

END IF

Procedure 用户注册模块

Import request from ‘@/utils/request’

用户填写相应数据

IF 用户点击注册（@click=submitForm(ruleform)）/\*\*/

  methods: {

    submitForm(formName) {/\*这是注册的函数，此处的formName中的数据应该是ruleForm,ruleForm中包含着数据库关于用户的字段，已经读取了用户输入的内容\*/

      this.$refs[formName].validate((valid) => {/\*校验，判断用户输入的数据是否满足条件\*/

        IF (valid) {

          this.loading = true/\*发出请求后一段时间不能再编辑，防止用户多次发出请求导致服务器报错\*/

          userRegister(this.ruleForm)/\*调用已经定义好的接口，将用户输入的表单字段发送到服务器 \*/

            .then((value) => {/\*value是服务器经过二次判断返回的json数据\*/

              const { code, message } = value

              if (code === 200) {/\*判断注册成功\*/

                this.$message({

                  message: '账号注册成功',

                  type: 'success'

                })

                setTimeout(() => {/\*跳转到登录界面\*/

                  this.loading = false

                  this.$router.push({ path: this.redirect || '/login' })

                }, 0.1 \* 1000)

              } else {

                this.$message.error('注册失败，' + message)

              }

            })

            .catch(() => {

              this.loading = false

            })

        } else {

          return false

        }

      })

    },

    resetForm(formName) {/\*清空表单\*/

      this.$refs[formName].resetFields()

    }

  }

ELSE

this.$router.push(“/register”)/\*返回到注册界面\*/

END IF

Procedure 表白墙管理

进入管理员界面

点击表白墙管理按钮

Select\* from Wall ORDER BY wall\_time//按照最新时间展示所有表白帖

DO WHILE

IF 选中帖子点击编辑→精选

wall\_state=”精华”

END IF

IF 选中帖子点击编辑→删除

wall\_state=”已删除”

END IF

END DO

返回管理员界面

Procedure 反馈管理

进入管理员界面

点击用户反馈按钮

DO WHILE

IF 点击举报

Select\* from Wall where wall\_report>=1 ORDER BY wall\_report desc//从高到低展示举报数大于等于1的所有帖子

选中帖子进行帖子的基本操作//语句同表白墙管理模块，修改wall\_state值

END IF

IF 点击问题

Select\* from Feedback order by feedback\_state desc,feedback\_date asc//按反馈未处理、反馈时间最新的顺序显示反馈信息

IF 点击某条反馈消息

feedback\_state=1//1代表信息已读

END IF

END IF

END DO

Procedure 用户管理

进入管理员界面

点击用户管理按钮

Select\* from **User ORDER BY** user\_jurisdiction ,user\_registertime//按照用户权限、注册时间的顺序展示所有用户

DO WHILE

IF 选中用户点击编辑

Select\* from **User where user\_id=””//展示此用户的信息以修改**

**IF 状态选择封禁**

user\_state=3//3代表封禁

END IF

**IF 密码填入\*\*\*\***

user\_pwd=”\*\*\*\*”

END IF

END IF

END DO

返回管理员界面

PROCEDURE 发布表白墙

BEGIN

点击发布按钮

打开编辑框

输入表白内容

DO WHILE 点击提交

var t=document.form1.str1.value;

var c=document.form2.str2.value;

IF c.length>100 or c.length == 0

alert("字符长度不能为0且必须保持在100以内");

ELSE

String sql = “insert into Wall(wall\_content, wall\_contenttitle, wall\_time, wall\_userid, wall\_state) values(c,t,getdate(),,正常)”;

alert("提交成功");

返回编辑界面；

END IF

END DO

PROCEDURE 修改表白墙

BEGIN

点击修改按钮

打开编辑框

输入修改的表白内容

DO WHILE 点击提交

var t=document.form1.str1.value;

var c=document.form2.str2.value;

IF c.length>100

alert("字符长度必须保持在100以内");

ELSE

String sql = “update Wall set wall\_content =c”;

sql = “update Wall set wall\_title = t”;

sql = “update Wall set wall\_time = getDate()”;

alert("提交成功");

返回编辑界面；

END IF

END DO

PROCEDURE 删除表白墙

BEGIN

点击删除按钮

String sql = “delete from Wall where wall\_id = 要删除的表白墙id”

alert("删除成功");

PROCEDURE 发布评论

BEGIN

点击评论按钮

打开编辑框

输入评论内容

DO WHILE 点击提交

var s=document.form.str.value;

IF s.length>100

alert("字符长度必须保持在100以内");

ELSE

String sql = “insert into Reply(wall\_id, reply\_content, reply\_userid, reply\_time,reply\_state) values(当前表白墙id,s,回复用户id,getdate(),正常)”;

alert("回复成功");

返回评论界面；

END IF

END DO

PROCEDURE 删除评论

BEGIN

点击删除按钮

String sql = “delete from Reply where reply\_id = 要删除的评论id”

alert("删除成功");

PROCEDURE 收藏

BEGIN

Int f = 0;

DO WHILE 点击收藏按钮

IF f==0

String sql = “insert into Collection(collection\_wallid, collection\_time) values(当前表白墙id,getdate())”;

alert("收藏成功");

f=1；

收藏图标变红；

ELSE

String sql = “delete from Collection while wall\_id = 当前表白墙id and user\_id = 当前用户id)”;

f = 0;

收藏图标变白；

END IF

END DO

PROCEDURE 删除收藏

BEGIN

点击删除按钮

String sql = “delete from Collection where collection\_id = 要删除的收藏id”

alert("删除成功");

PROCEDURE 点赞

BEGIN

Int f = 0;

DO WHILE 点击点赞按钮

IF f==0

String sql = “updata Wall set wall\_good = wall\_good + 1 where wall\_id = 当前表白墙id”;

f=1；

点赞图标变红；

ELSE

String sql = “updata Wall set wall\_good = wall\_good – 1 where wall\_id = 当前表白墙id”;

f = 0;

点赞图标变白；

END IF

END DO

PROCEDURE 排序

BEGIN

int sort = 0;(0:时间排序，1:点赞数排序，2:评论数排序，3:收藏数排序)

DO WHILE 点击排序按钮

IF sort==0

String sql = “select \* from wall order by wall\_time”;

ELSE IF sort==1

String sql = “select \* from wall order by wall\_good”;

ELSE IF sort==2

String sql = “select \* from wall order by wall\_reply”;

ELSE

String sql = “select \* from wall order by wall\_collection”;

END IF

显示到网页

END DO

4.1.2.3关键数据结构

1.数据表

| **表名** | **描述** |
| --- | --- |
| User | 用户表 |
| Wall | 用户表白墙表 |
| Reply | 表白墙回复表 |
| Feedback | 用户反馈表 |
| Collection | 用户收藏表 |
| Choose | 用户认领表 |

2.数据表字段

**User表**

user\_id(主键，用户id，Variable characters(10));

user\_name(用户昵称，Variable characters(10));

user\_pwd(用户密码，Variable characters(10));

user\_registertime(注册时间，Date)

user\_state(用户状态，int)

user\_sex(用户性别，int)

user\_jurisdiction(用户权限，Integer)

user\_signature(个性签名，Variable characters(100))

**Wall表**

wall\_id(主键，表白墙id，Integer);

wall\_content(表白墙正文，Variable characters(140));

wall\_contenttitle(表白墙正文标题，Variable characters(10));

wall\_time(表白墙发表时间，Date)

wall\_userid(发表该表白墙的用户，Variable characters(10))

wall\_state(表白墙状态，Variable characters(10))

wall\_good(表白墙点赞数，Integer)

wall\_collection(表白墙收藏数，Integer)

wall\_talk(表白墙热度，Integer)

wall\_report(表白墙举报数，Integer)

wall\_to(被表白对象，Variable characters(10))

**Reply表**

reply\_id(主键，回复编号，Integer)

wall\_id(在谁的帖子下回复，Integer)

reply\_content(回复正文，Variable characters（20）)

reply\_userid (回复的用户id，Variable characters(10))

reply\_time(回复时间，Date)

reply\_state(回复状态，Integer)

**Feedback表**

feedback\_id(主键，反馈id号，Integer)

feedback\_title(反馈标题，Variable characters(10))

feedback\_content(反馈正文，Variable characters(140))

feedback\_userid(反馈人，Variable characters(10))

feedback\_date(反馈时间，date)

feedback\_state(反馈状态，integer)

**Collection表**

collection\_id(主键，收藏编号，Integer)

collection\_userid (收藏的用户id，Variable characters(10))

collection\_wallid(收藏的帖子id，Integer)

collection\_time(收藏时间，Date)

**Choose表**

`choose\_id` int(11) NOT NULL AUTO\_INCREMENT COMMENT '认领id'

`choose\_wallid` int(11) NOT NULL COMMENT '表白墙id'

`choose\_beuserid` varchar(10) NOT NULL COMMENT '被认领人'

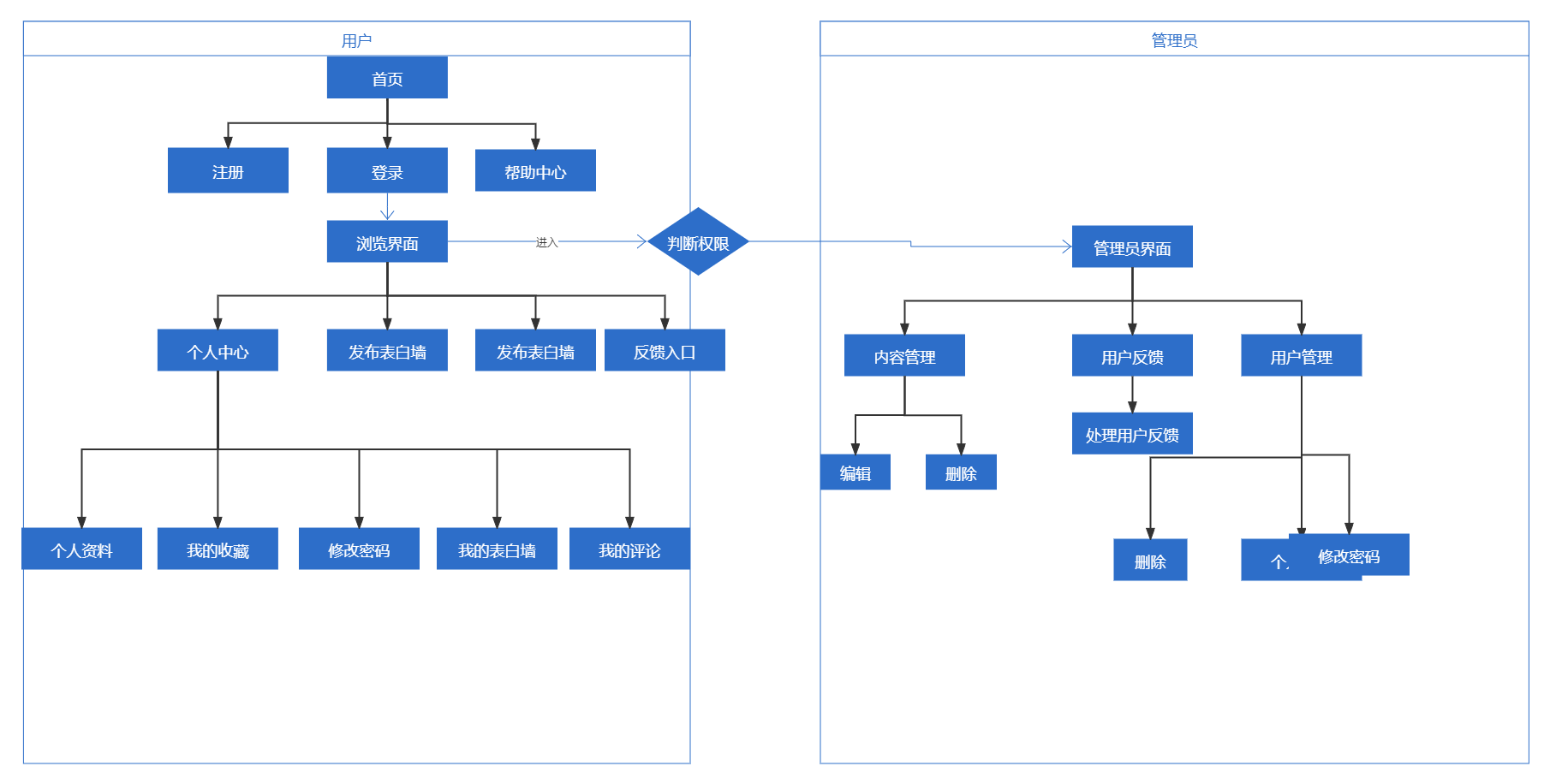
`choose\_userid` varchar(10) NOT NULL COMMENT '认领人',

`choose\_state` varchar(11) NOT NULL COMMENT '认领状态，0未认领1认领',

`choose\_time` datetime NOT NULL COMMENT '认领时间',

4.1.3―基本处理流程

4.1.3.1系统流程图



4.1.3.2数据流程图

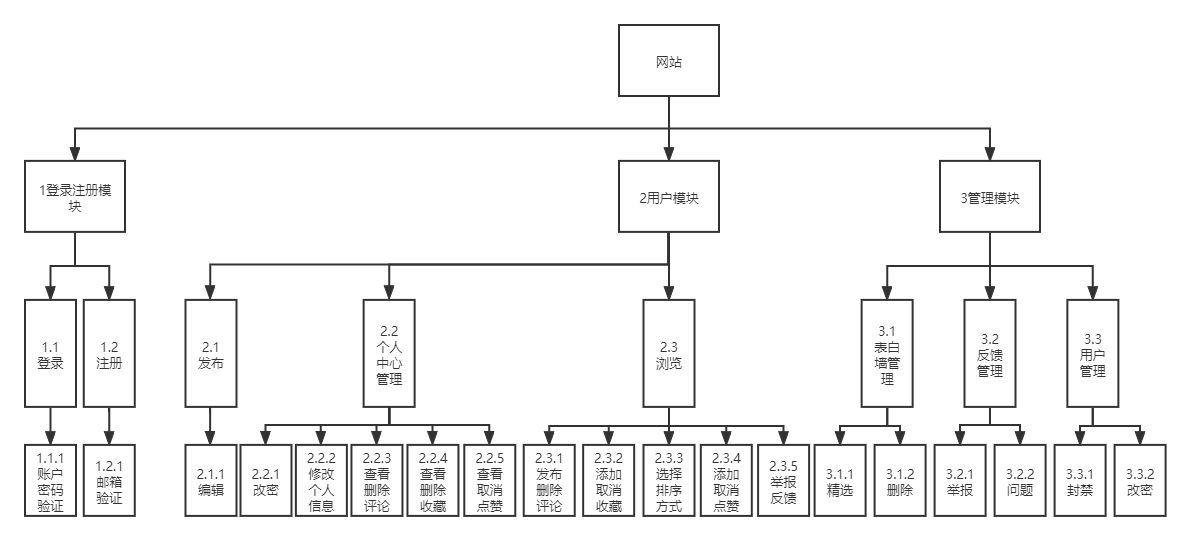


4.1.4系统体系结构

4.1.4.1系统配置项

TBD

4.1.4.2系统层次结构



4.1.4.3系统配置项设计

TBD

4.1.5功能需求与系统配置项的关系

TBD

4.1.6人工处理过程

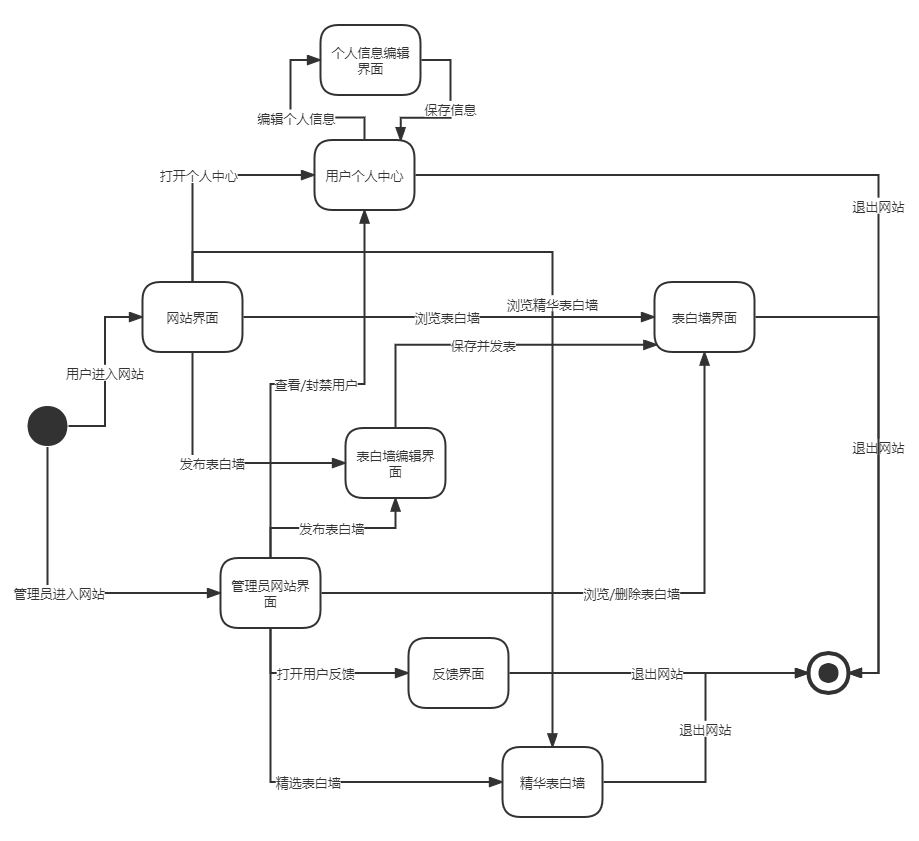
管理员权限赋予（后台手动操作）

4.2系统部件

网站、服务器、数据库

4.3执行概念





4.4接口设计

用户操作UserController:

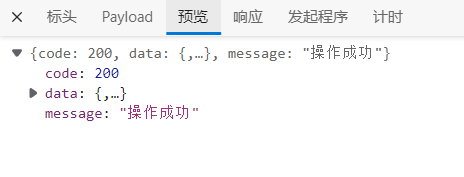
4.4.1分页查询载入用户信息

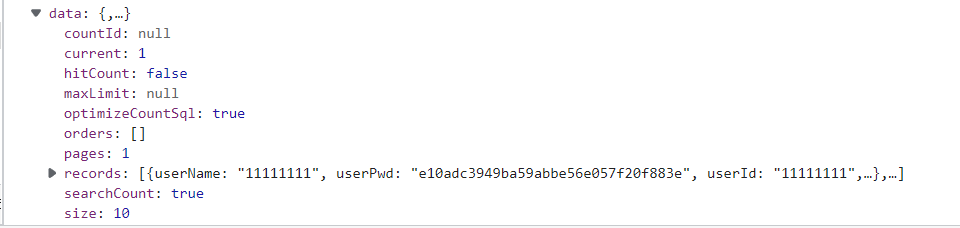
请求方式: GET

<http://localhost:8000/user/findPage>

| **请求参数** | **是否必须** | **说明** |
| --- | --- | --- |
| pageNum | 是 | 当前页数：默认是1 |
| pageSize | 是 | 当前展示数据条数：默认是10 |
| search | 是 | 输入用户id查找用户：默认空（全查找） |

正确时返回JSON数据包：





4.4.2更新用户数据

请求方式：PUT（需要管理员权限，没有会报401错误）

<http://localhost:8000/user/update>

请求一个json对象，含有数据如下：

| **请求参数** | **是否必须** | **说明** |
| --- | --- | --- |
| userId | 是 | 用户编号 |
| userJurisdiction | 是 | 用户权限 |
| userName | 是 | 用户姓名 |
| userPwd | 是 | 用户密码 |
| userRegistertime | 是 | 用户注册时间 |
| userSex | 是 | 用户性别 |
| userSignature | 否 | 用户个性签名 |
| userState | 是 | 用户状态 |

正确时返回json数据包：



4.4.3重置用户密码为123456

请求方式：PUT

<http://localhost:8000/user/reset>

请求一个json对象，含有数据如下：

| **请求参数** | **是否必须** | **说明** |
| --- | --- | --- |
| userId | 是 | 用户编号 |
| userJurisdiction | 是 | 用户权限 |
| userName | 是 | 用户姓名 |
| userPwd | 是 | 用户密码 |
| userRegistertime | 是 | 用户注册时间 |
| userSex | 是 | 用户性别 |
| userSignature | 否 | 用户个性签名 |
| userState | 是 | 用户状态 |

正确时返回json数据包：



4.4.4删除用户信息

请求方式：delete

<http://localhost:8000/user/>delete/{userid}

| **请求参数** | **是否必须** | **说明** |
| --- | --- | --- |
| userId | 是 | 要删除的用户的id |

正确时返回JSON数据包

{

"code": 200,

"data": null,

"message": "操作成功"

}

表白墙操作WallController

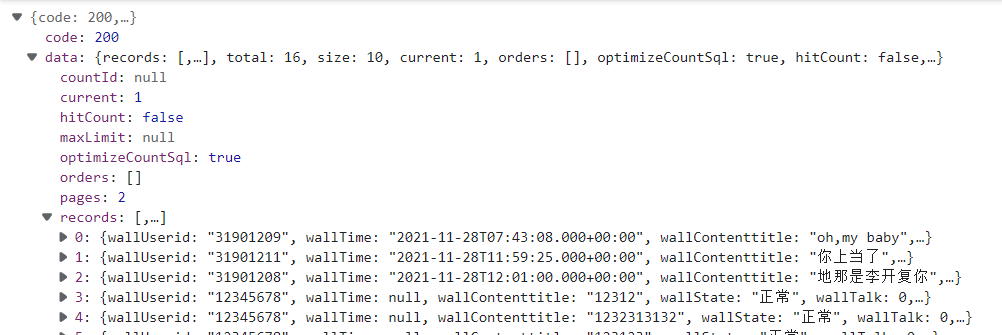
4.4.5分页查询载入表白墙信息

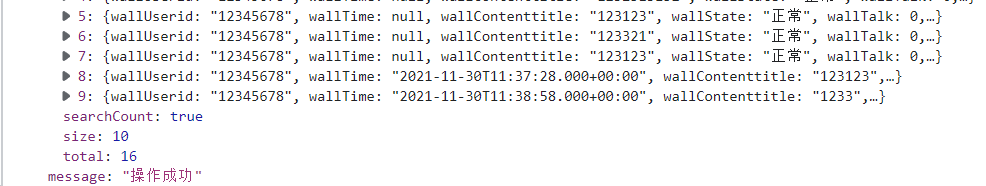
请求方式: GET

<http://localhost:8000/wall/findPage>

| **请求参数** | **是否必须** | **说明** |
| --- | --- | --- |
| pageNum | 是 | 当前页数：默认是1 |
| pageSize | 是 | 当前展示数据条数：默认是10 |
| search | 是 | 输入表白墙标题查找表白墙 |

正确时返回JSON数据包：





4.4.6更改表白墙状态

请求方式: PUT

<http://localhost:8000/user/update>

请求一个json对象，包含数据如下：

| **请求参数** | **是否必须** | **说明** |
| --- | --- | --- |
| wallCollection | 是 | 表白墙收藏数 |
| wallContent | 是 | 表白墙正文 |
| wallContenttitle | 是 | 表白墙标题 |
| wallGood | 是 | 表白墙点赞数 |
| wallId | 是 | 表白墙id |
| wallReport | 是 | 表白墙举报数 |
| wallState | 是 | 表白墙状态 |
| walllTalk | 是 | 表白墙评论数 |
| wallTime | 是 | 表白墙发布时间 |
| wallTo | 是 | 被表白对象 |
| wallUserid | 是 | 表白墙发布者id |

正确时返回json数据包

{

"code": 200,

"data": null,

"message": "操作成功"

}

4.4.7删除表白墙

请求方式：delete

<http://localhost:8000/wall/>delete/{wallid}

| **请求参数** | **是否必须** | **说明** |
| --- | --- | --- |
| wallId | 是 | 要删除的表白墙id |

正确时返回JSON数据包

{

"code": 200,

"data": null,

"message": "操作成功"

}

4.4.8新增表白墙

请求方式：post

<http://localhost:8000/wall/>insertWall

| **请求参数** | **是否必须** | **说明** |
| --- | --- | --- |
| wallI | 是 | 整个表白墙 |

正确时返回JSON数据包

{

"code": 200,

"data": null,

"message": "操作成功"

}

4.4.9收藏表白墙

请求方式：post

[http://localhost:8000/collection/](http://localhost:8000/wall/)insertCollection

| **请求参数** | **是否必须** | **说明** |
| --- | --- | --- |
| collection | 是 | 要新增的整个收藏实体 |

正确时返回JSON数据包

{

"code": 200,

"data": null,

"message": "操作成功"

}

4.4.10认领表白墙

请求方式：post

[http://localhost:8000/choose/](http://localhost:8000/wall/)insertChoose

| **请求参数** | **是否必须** | **说明** |
| --- | --- | --- |
| choose | 是 | 要新增的整个认领实体 |

正确时返回JSON数据包

{

"code": 200,

"data": null,

"message": "操作成功"

}

反馈操作FeedbackController

4.4.11分页查询载入表白墙信息

请求方式: GET

<http://localhost:8000/feedback/findPage>

| **请求参数** | **是否必须** | **说明** |
| --- | --- | --- |
| pageNum | 是 | 当前页数：默认是1 |
| pageSize | 是 | 当前展示数据条数：默认是10 |
| search | 是 | 输入表白墙标题查找表白墙 |

正确时返回JSON数据包：



4.4.12更改反馈受理状态

请求方式: PUT

<http://localhost:8000/feedback/update>

请求一个json对象，包含数据如下：

| **请求参数** | **是否必须** | **说明** |
| --- | --- | --- |
| feedbackId | 是 | 反馈id |
| feedbackContent | 是 | 反馈正文 |
| feedbackState | 是 | 反馈状态 |
| feedbackTime | 是 | 反馈时间 |
| feedbackTitle | 是 | 反馈标题 |
| feedbackUserid | 是 | 反馈人id |

正确时返回json数据包

{

"code": 200,

"data": null,

"message": "操作成功"

}

4.4.13删除反馈

请求方式：delete

<http://localhost:8000/wall/>delete/{feedbackid}

| **请求参数** | **是否必须** | **说明** |
| --- | --- | --- |
| feedbackId | 是 | 要删除的反馈id |

正确时返回JSON数据包

{

"code": 200,

"data": null,

"message": "操作成功"

}

4.4.14新增反馈

请求方式：post

[http://localhost:8000/feedback/](http://localhost:8000/wall/)insertFeedback

| **请求参数** | **是否必须** | **说明** |
| --- | --- | --- |
| feedback | 是 | 要新增的整个feedback实体 |

正确时返回JSON数据包

{

"code": 200,

"data": null,

"message": "操作成功"

}

注册

请求方式：Post

http://localhost:8000/user/register

| **请求参数** | **是否必须** | **说明** |
| --- | --- | --- |
| userId | 是 | 用户学号 |
| userName | 是 | 用户昵称 |
| userPwd | 是 | 用户密码 |
| userRegistertime | 是 | 用户注册时间 |
| userSex | 是 | 用户性别 |

登录：

请求方式：Post

<http://localhost:8000/user/login>

| **请求参数** | **是否必须** | **说明** |
| --- | --- | --- |
| userId | 是 | 用户学号 |
| userPwd | 是 | 用户密码 |
| rememberMe | 是 | 记住密码 |

个人中心：

4.4.15个人资料：

请求方式：Get

<http://localhost:8000/user/info>

| **请求参数** | **是否必须** | **说明** |
| --- | --- | --- |
| userName | 是 | 用户昵称 |
| userSex | 是 | 性别 |
| userState | 是 | 用户状态 |
| userRegistertime | 是 | 注册日期 |
| userId+@stu.zucc.edu.cn | 是 | 用户邮箱 |

4.4.16我的表白墙

请求方式：Get

http://localhost:8000/user/+ userId

| **请求参数** | **是否必须** | **说明** |
| --- | --- | --- |
| pageNum | 是 | 当前页数：默认是1 |
| pageSize | 是 | 当前展示数据条数：默认是10 |
| userId | 是 | 当前用户id |

4.4.17我的评论

请求方式：Get

http://localhost:8000/user/reply/+ userId,

| **请求参数** | **是否必须** | **说明** |
| --- | --- | --- |
| pageNum | 是 | 当前页数：默认是1 |
| pageSize | 是 | 当前展示数据条数：默认是10 |
| userId | 是 | 当前用户id |

4.4.18我的收藏

请求方式：Get

http://localhost:8000/user/coll/+ userId,

| **请求参数** | **是否必须** | **说明** |
| --- | --- | --- |
| pageNum | 是 | 当前页数：默认是1 |
| pageSize | 是 | 当前展示数据条数：默认是10 |
| userId | 是 | 当前用户id |

4.4.19我的认领

请求方式：Get

http://localhost:8000/user/choose/+ userId,

| **请求参数** | **是否必须** | **说明** |
| --- | --- | --- |
| pageNum | 是 | 当前页数：默认是1 |
| pageSize | 是 | 当前展示数据条数：默认是10 |
| userId | 是 | 当前用户id |

4.4.20认领我的

请求方式：Get

http://localhost:8000/user/bechoose/+ userId,

| **请求参数** | **是否必须** | **说明** |
| --- | --- | --- |
| pageNum | 是 | 当前页数：默认是1 |
| pageSize | 是 | 当前展示数据条数：默认是10 |
| userId | 是 | 当前用户id |

5运行设计

5.1系统初始化

输入网址，打开表白墙网站

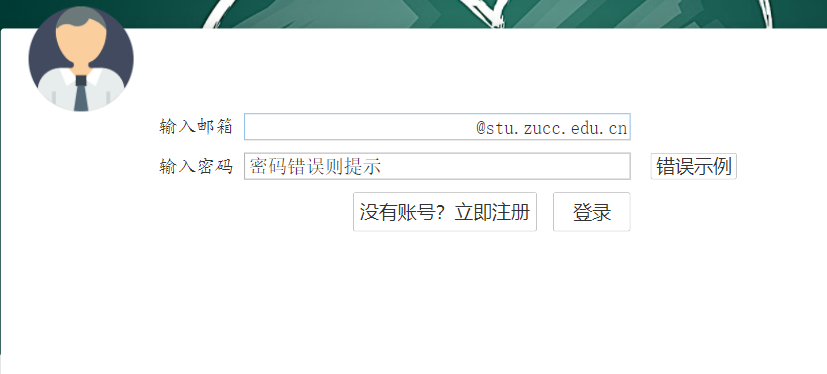
5.2运行控制

注册、登录、表白：

点击注册按钮进入注册界面：



点击登录按钮进入登录界面：



点击发布表白墙按钮进入发布界面：



点击十大热帖按钮进入浏览十大热帖：



点击反馈入口按钮进入反馈界面：



详情见界面原型：<https://app.mockplus.cn/rp/editor/DQNswQhMCKn/89-iH8IrqWBq>

<https://app.mockplus.cn/rp/editor/xR4n4qVyZF08>

5.3运行结束

点击注销登录，退出网页

6系统出错处理设计

6.1出错信息

系统将有以下出错信息和警告信息：

1)注册界面：邮箱、密码格式错误 提示：邮箱/密码格式错误

2）登录界面：密码输入错误 提示：密码错误

3）编辑界面：字数超出限制 提示：字数超出限制，请将字数控制在xx之内

4）浏览界面：收藏/评论/点赞出现错误 提示：收藏/评论/点赞错误，请稍后重试

等等,详情见界面原型演示<https://app.mockplus.cn/rp/editor/DQNswQhMCKn/89-iH8IrqWBq>

6.2补救措施

TBD

7系统维护设计

7.1检测点的设计

TBD

7.2检测专用模块的设计

TBD

8尚待解决的问题

TBD

9需求的可追踪性

TBD

10 注解

附录