**公共课室管理系统**

**软件设计文档**

**Version x.x**

|  |  |  |
| --- | --- | --- |
| 文件状态：  [√] 草稿  [ ] 正式发布  [ ] 正在修改 | 文件标识： |  |
| 当前版本： | 1.1 |
| 作 者： 者： | 陈梓浩 梁喻源 李宇捷 |
| 完成日期： | 2016.7.13 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **修订历史** | | | | |
| **修订日期** | **修订人** | **版本号** | **审核人** | **修订说明** |
| 2016.7.11 | 陈梓浩 | V1.0 | 李宇捷 | 撰写软件设计文档模板 |
| 2016.7.13 | 梁喻源 | V1.1 |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

目录

[1. 引言 4](#_Toc456130175)

[1.1. 编写目的 4](#_Toc456130176)

[1.2. 项目背景 4](#_Toc456130177)

[1.3. 相关定义 4](#_Toc456130178)

[1.4. 参考文献 4](#_Toc456130179)

[2. 总体设计 4](#_Toc456130180)

[2.1. 系统目标概要 4](#_Toc456130181)

[2.2. 系统方案（用例）概要 4](#_Toc456130182)

[2.3. 系统功能 4](#_Toc456130183)

[2.4. 系统性能 4](#_Toc456130184)

[2.5. 子系统设计 4](#_Toc456130185)

[2.6. 项目使用到的软件技术 4](#_Toc456130186)

[2.7. 技术选型理由 4](#_Toc456130187)

[3. 程序系统的结构 4](#_Toc456130188)

[3.1. 模块划分 4](#_Toc456130189)

[3.2. 模块之间的关系 5](#_Toc456130190)

[3.3. 模块对应的软件技术 5](#_Toc456130191)

[4. 运行环境 5](#_Toc456130192)

[5. 外部因素 5](#_Toc456130193)

[6. 架构设计 5](#_Toc456130194)

[6.1. 系统体系结构 5](#_Toc456130195)

[6.2. 类图 5](#_Toc456130196)

[6.3. 模块视图 5](#_Toc456130197)

[6.4. 用户界面 5](#_Toc456130198)

[7. 测试计划 5](#_Toc456130199)

[8. 研发风险 5](#_Toc456130200)

[8.1. 技术风险 5](#_Toc456130201)

[8.2. 团队管理风险 5](#_Toc456130202)

# 引言

## 编写目的

经过我们的讨论，发现一个面向高校学生的公共课室管理系统（Public Classroom Management System以下简称PCM）对于高校学生的学习活动将有很大效率的提高，也将减少同学们在学校内各部门之间来回奔波的繁琐。于是我们决定编写一个PCM系统。

## 项目背景

该项目为软件综合设计课程的课程项目，由李宇捷、林勇洁、林育新、陈梓浩、梁喻源共同开发。

## 相关定义

|  |  |  |
| --- | --- | --- |
| **术语** | **英语名词** | **定义** |
| 公共课室管理系统 | Public Classroom Management (PCM) System | 本次的开发任务，能够进行课室申请、占用状态查询、在线审批的系统。 |
| 普通用户 | User | 自行注册并且能够查询课室、申请课室的用户，但此类用户不能进行审批。 |
| 管理员 | Admin | 由注册员注册、能够查看申请记录并且审批申请的用户。 |
| 申请记录 | Record | 指普通用户对使用公共课室的申请，与用户相关联。 |
| 课室占用 | Occupation | 指课室在该时间段以被使用，也即已有申请已被审批通过了。 |
| 申请队列 | Applicant Queue | 申请某课室、某时间段使用的申请者队列，其中越早申请者，其优先级越高。 |
| 审批记录 | Approval Record | 指由管理员对普通用户提交的申请进行审批。有两种可能的结果：通过与拒绝。 |

## 参考文献

# 总体设计

## 系统目标概要

PCM系统目标允许用户通过移动移动终端浏览器访问该系统。对于用户, PCM系统具有用户注册登录、课室申请、课室查询、申请状态查询等功能。对于管理员用户，PCM系统管理员能够查看过去及未来所有课室的空闲、待审核和占用状态，并且能够审核待审核的申请。

## 系统方案（用例）概要



## 系统功能

用户注册:用户可以通过邮箱注册成为PCM用户。

用户登录：用户通过邮箱账号以及密码登录PCM用户，首次登录会要求用户完善个人用户信息。

找回密码：用户如果忘记密码可以通过绑定的邮箱验证身份并重置密码。

课室查询：PCM系统允许用户查看当前及未来一段时间所有公共课室的占用状态，登录后的用户可以通过输入查询时间段等一系列条件来查询课室的占用情况。

课室申请：用户通过PCM系统可以申请特定时间段的特定的公共课室 (如不指定则随机选择一个可用的公共教室) 的使用，如果该时间段的该课室已被其他用户申请，则提醒用户该非空申请状态，若用户确定申请则增加一条申请记录，如果未被其余用户申请，则直接增加一条记录。除此之外，用户还能够取消其未来的申请。

审核申请：管理员能够查看过去及未来所有课室的空闲、待审核和占用状态，并且能够审核待审核的申请。所谓审核，即通过或者拒绝用户的申请。一旦一条申请记录的状态改变，系统即向相关用户的绑定邮箱发送一封提醒邮件，以告知用户其申请是否被允许通过。对于某个时间段、某个课室的申请队列长度大于1，则优先通过较早的申请同时拒绝队列中其他申请。

申请状态查询：用户可以查看申请课室的状态，及时了解申请处于哪一过程以及申请结果。

## 系统性能

## 子系统设计

PCM 系统包括三个子系统：用户管理子系统、课室申请子系统和申请审批子系统。这三个子系统向主系统提供接口，由主系统调用这三个子系统的接口，以实现其功能。其关系如下图所示。

## 项目使用到的软件技术

项目技术

前端 vue.js

一个精简的 MVVM。从技术角度讲，Vue.js 专注于 MVVM 模型的 ViewModel 层。它通过双向数据绑定把 View 层和 Model 层连接了起来。实际的 DOM 封装和输出格式都被抽象为了Directives 和 Filters。Vue.js和其他库相比是一个小而美的库，作者的主要目的是通过一个尽量简单的 API 产生可反映的数据绑定和可组合的视图组件，

使用vue.js 的原因是因为

1. vue.js 官方文档清晰，比angular 更容易上手

2. 通过异步批处理的方式来进行dom 更新

3. 低耦合，通过组件化的方式，构建应用

4. 对各个模块更为友好，可以通过npm bower 等方式进行安装，不像angular 强迫各种语法设定，适用场景很多

前端ui 框架 Vux

与vue.js 完美锲合 一脉相承 简化开发

后端 express

express 作为一个web framework ，既不是最底层的，又不是最抽象的，但是应用广泛，又有典型架构，发展时间较久，官方文档完整，api 完整，自带Router 和路由规则等，很适合作为初学的目标

数据库 mysql

根据需求，各表之间关系较大，并且数据量不大，关系数据结构较为简单，小组讨论最终使用mysql 更为方便

后端构建工具 gulp

后端代码构建工具以及实现自动化的构建工具，开发过程中，不必要过多关注过多重复性操作，实现文件的压缩，并且打包至相对应的目录

## 技术选型理由

# 程序系统的结构

## 模块划分

## 模块之间的关系

## 模块对应的软件技术

# 运行环境

各主流现代浏览器。

# 外部因素

# 架构设计

## 系统体系结构

本系统基于B/S 架构，使用 MVC (Model-View-Controller) 的架构模式，分别为视图层、控制层及实体层。

* 视图层

视图层系统提供的用户交互界面，负责展现实体的状态及获取用户的请求与信息。本系统视图层的模块包括：用户管理模块（登录／注册）、用户个人首页、课室申请界面模块、后台管理员查看信息模块、课室状态查看模块等。

* 控制层

控制层是实现系统业务逻辑的核心，负责对业务逻辑代码进行抽象和封装。控制层接收用户的请求和信息、调用实体层的数据接口、执行相应的逻辑操作以及将结果返回给视图层。本系统控制层的模块包括：用户管理模块、课室申请及审批、课室状态查询、课室匹配模块等。

* 实体层

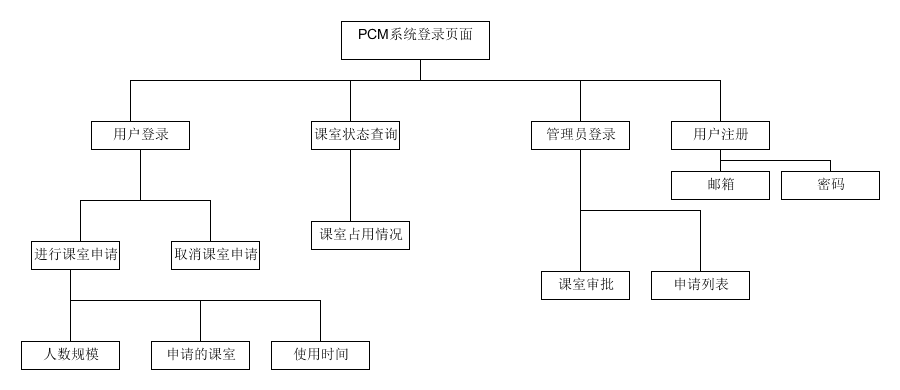
实体层负责系统的实体对象的数据访问，数据访问对象提供了控制层访问实体层的接口方法，实现对数据库的增删改查的操作。本系统实体层包括：普通用户、管理员用户、课室、申请记录。

## 类图

系统总类图:



## 模块视图



## 用户界面

初始页面：



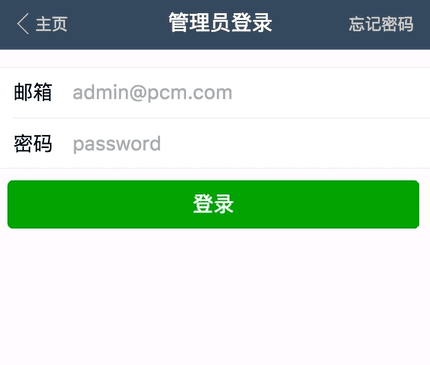
用户注册页面：



学生用户界面：



管理员登录界面：



管理员用户界面：



# 测试计划

## 测试目标

模块内部逻辑正确，模块间功能完备，交互准确，系统整体运行无差错

## 测试设计

与迭代周期相吻合，每次迭代之后进行下述的三层相吻合

第一层：单元测试，在模块内部主要逻辑处进行测试，保证无差错

第二层：模块测试，主要设计各种各样的测试输入，测试模块是否达到预期的逻辑与功能

第三层：集成测试，模块间拼接后，要进行功能性2测试，保证模块间逻辑与交互正确，符合预期效果

第一层测试以白盒测试为主，侧重逻辑

第二、三层测试以黑盒测试为主，侧重功能

# 研发风险

## 技术风险

## 团队管理风险