课设题目--

设计报告

题目要求：

实践内容 实验室设备管理系统

建立一个实验室设备管理系统, 编写应用程序完成系统开发。

建立基本表：

设备表（设备号，设备名称，价格，配置，厂家名称，出厂日期，实验室名）

故障设备信息表（故障序号，故障名称，设备号，故障日期，故障信息，经手人）

维修表（维修号，设备号，故障序号，维修日期，维修内容，改变配置，维修金额，维修人，备注）

用户表（用户号，用户名，密码，用户类别）

要求实现以下功能：

（1）登录功能

系统有四类用户：一般用户、管理员、维修员、系统管理员

（2）设备基本信息管理

要求：设备基本信息查询，单个/批量设备添加，单个/批量设备信息修改，单个/批量设备信息删除

（3）故障设备信息管理

要求：故障查询，添加故障设备，修改故障设备信息，删除故障设备信息；维修信息查询，维修信息添加

（4）用户管理

要求：用户信息查询，用户注册，用户信息维护，用户权限修改

（5）报表信息管理

（6) 系统维护

1.概述

系统开发的背景

实验室往往有很多设备需要管理，而且设备的故障率往往很高，如果我们手工记录的话，往往会有很多的不便，而且实验室设备昂贵，如果手工记录失误的话，容易引起较大的损失，如果我们能开发出一款实验室设备管理系统，我们就能有秩序的方便的管理实验室的各种设备，这样做大大简化了实验室的运转步骤，提升了效率，让实验室人员有更多的时间用于在实验上。

系统开发的意义

实验室设备管理系统会帮助实验室管理人员十分清楚的了解到实验室设备的各种信息，大大简化了管理人员的操作步骤和工作强度，使设备维护变得简单清晰透明，任何人都可以得知设备的情况，专业人士可以对系统的信息进行修改和更新，所有的故障都记录在系统里长期保存，这对于实验室的管理人员来说十分重要。

系统开发工具

代码编辑工具：VScode

网页服务器：wampserver3.1.7(apache 2.4.37, mysql 5.7.24 PHP 7.2.14)

框架：thinkphp5.0框架

操作系统：Windows 10

需求分析：

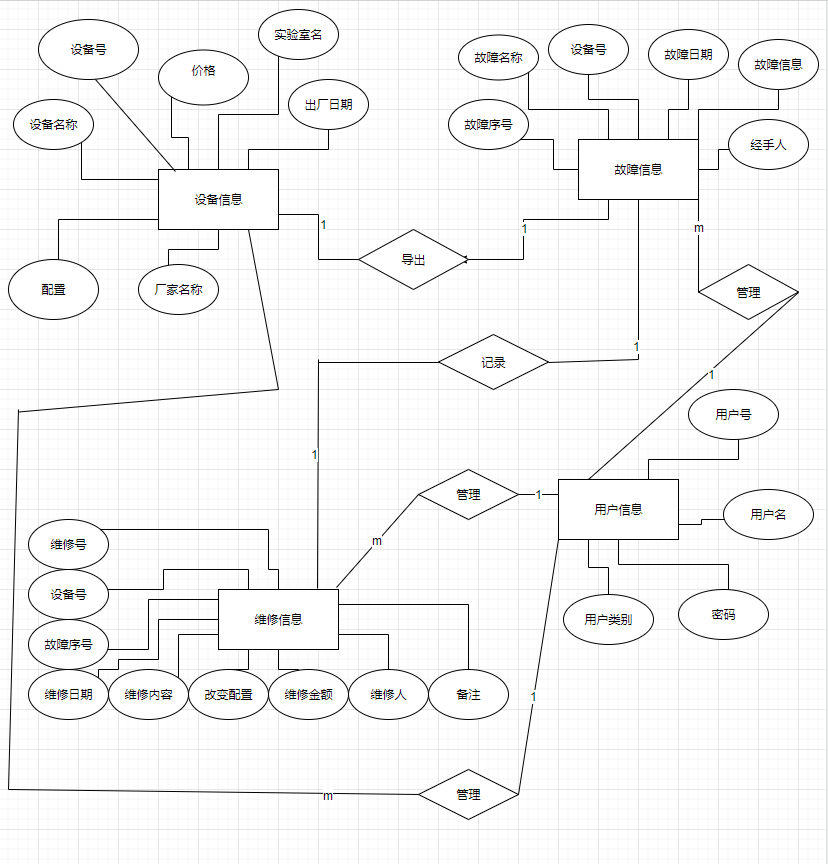
根据题目描述，这个系统有着四类用户，分别有着不同的权限，而且这些权限是平级关系，

一种是普通用户，他只有查看设备的权限，没有任何的更新修改权限，第二种是维修员，维修员可以查看两个表，故障设备信息表和维修信息表，故障设备修好之后可以删除数据，也可以修改数据，所以维修员可以对故障设备表进行增删查改，但是维修信息表是作为备案信息，所以维修员也只有增加和查询的权限，没有修改的权限。第三种用户为管理员，我这里理解为用户管理员，用户管理员可以对用户表进行直接的增删查改操作，超级管理员即对以上所有内容都有着最高的权限。

数据库设计：

E—R图

：



功能分析：

方便各类用户查找修改删除自己用户所对应的表数据。

拥有严谨的验证器结构，不合法的数据无法写入数据库，并给出错误提示。

安全分析：

在管理系统中，数据的安全性十分重要，所以所有在数据库里的用户密码都经过了md5加密，而且有着严格的数据验证，如果输入了不和法的数据，是不会进入数据库，而且一些简单的SQL注入也是没有办法直接登录系统的。

逻辑结构设计：

设备表（设备号，设备名称，价格，配置，厂家名称，出厂日期，实验室名）

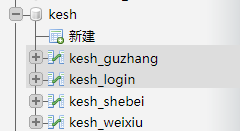
故障设备信息表（故障序号，故障名称，设备号，故障日期，故障信息，经手人）

维修表（维修号，设备号，故障序号，维修日期，维修内容，改变配置，维修金额，维修人，备注）

用户表（用户号，用户名，密码，用户类别）

红色代表主键，其中设备号是故障设备信息表和维修表的外键。

概览：









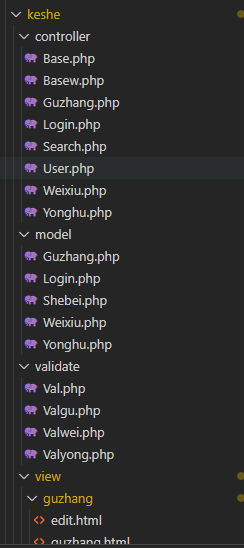


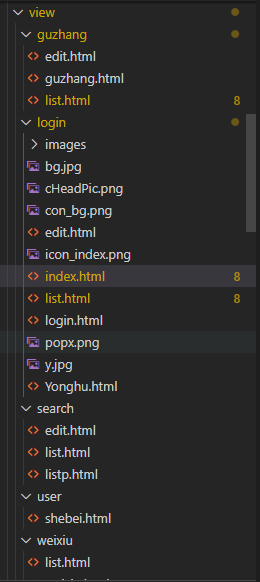
这是四张表的概览。

源代码目录简介：

目录采用MVC结构

MVC全名是Model View Controller，是模型(model)－视图(view)－控制器(controller)的缩写，一种软件设计典范，用一种业务逻辑、数据、界面显示分离的方法组织代码，将业务逻辑聚集到一个部件里面，在改进和个性化定制界面及用户交互的同时，不需要重新编写业务逻辑。MVC被独特的发展起来用于映射传统的输入、处理和输出功能在一个逻辑的图形化用户界面的结构中。





在thinkphp框架下，有

Controller：控制器接受用户的输入并调用模型和视图去完成用户的需求，所以当单击Web页面中的超链接和发送HTML表单时，控制器本身不输出任何东西和做任何处理。它只是接收请求并决定调用哪个模型构件去处理请求，然后再确定用哪个视图来显示返回的数据。在本课设里，我们使用的模块为keshe模块，模块的控制器即在controller文件夹下。

Model：与数据库有关的操作都需要用到model，模型表示企业数据和业务规则。在MVC的三个部件中，模型拥有最多的处理任务。被模型返回的数据是中立的，就是说模型与数据格式无关，这样一个模型能为多个视图提供数据，由于应用于模型的代码只需写一次就可以被多个视图重用，所以减少了代码的重复性。

View：视图是用户看到并与之交互的界面。对老式的Web应用程序来说，视图就是由HTML元素组成的界面，在新式的Web应用程序中，HTML依旧在视图中扮演着重要的角色。

Validate：用来验证输入内容的正确性的验证器，在添加或者修改信息时，我们输入的信息有可能不匹配数据库的内容，

重要代码解释：

如设备表的控制器：

<?php

namespace app\keshe\controller;

use app\keshe\controller\Base;

use app\keshe\model\Shebei as UserModel;

use app\keshe\validate\Val as UserValidate;

class Search extends Base

{

    public function list()

    {

        $data = UserModel::paginate(8);

        $page = $data->render();

        $this->assign('data',$data);

        $this->assign('page',$page);

        return $this->fetch();

    }

    public function listp()

    {

        $data = UserModel::paginate(8);

        $page = $data->render();

        $this->assign('data',$data);

        $this->assign('page',$page);

        return $this->fetch();

    }

    public function edit()

    {

        $shebeihao = input('get.shebeihao');

        $data = UserModel::get($shebeihao);

        $this->assign('data',$data);

        return $this->fetch();

    }

    public function update()

    {

        $data = input('post.');

        $shebeihao = input('post.shebeihao');

        $val = new UserValidate();

        if (!$val->check($data)){

            $this->error($val->getError());

            exit;

        }

        $user = new UserModel();

        $ret = $user->allowField(true)->save($data,['shebeihao'=>$shebeihao]);

        if($ret){

            $this->success('修改设备信息成功','Search/list');

        }

        else{

            $this->error('修改设备信息失败');

        }

    }

    public function delete()

    {

        $shebeihao = input('get.shebeihao');

        $ret = UserModel::destroy($shebeihao);

        if($ret){

            $this->success('删除设备成功','Search/list');

        }

        else{

            $this->error('删除设备失败');

        }

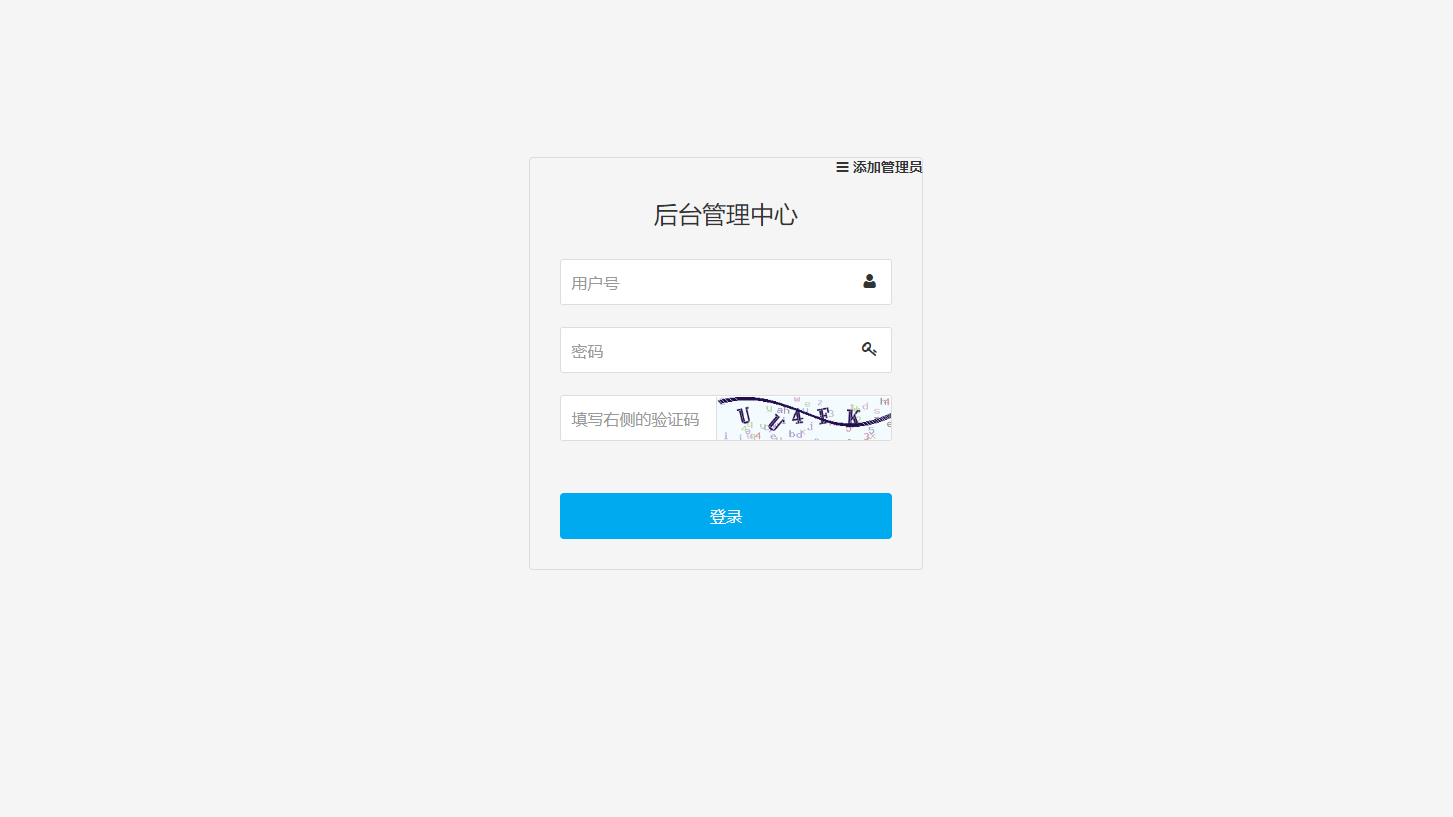
    }

}

在这段代码中，开头首先声明这是一个PHP文件，然后引用了各种类和模型，最重要的是class中的内容，即方法。这段代码中包含了5种方法，即展示，不带修改删除按钮的展示，修改，更新和删除五种方法，这五种方法既独立又相互联系，如edit和update方法在修改操作中都会用到，在方法中 我们会调用到view模板文件，用来提交表单信息。

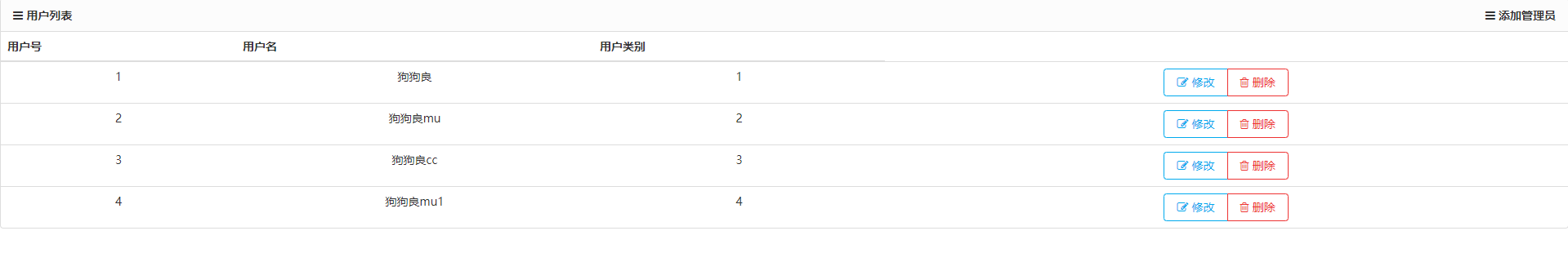
系统功能演示：

用户登录：



在用户登录界面，我们输入用户号和密码还有验证码，系统后台会自动检测出用户的类别，并跳转到相应的界面：

用户管理员：（跳转到用户管理界面）



在这个界面可以清楚的看到所有的用户，并且有修改和删除的权限，比如我们修改4号用户的用户名为BUCT123：

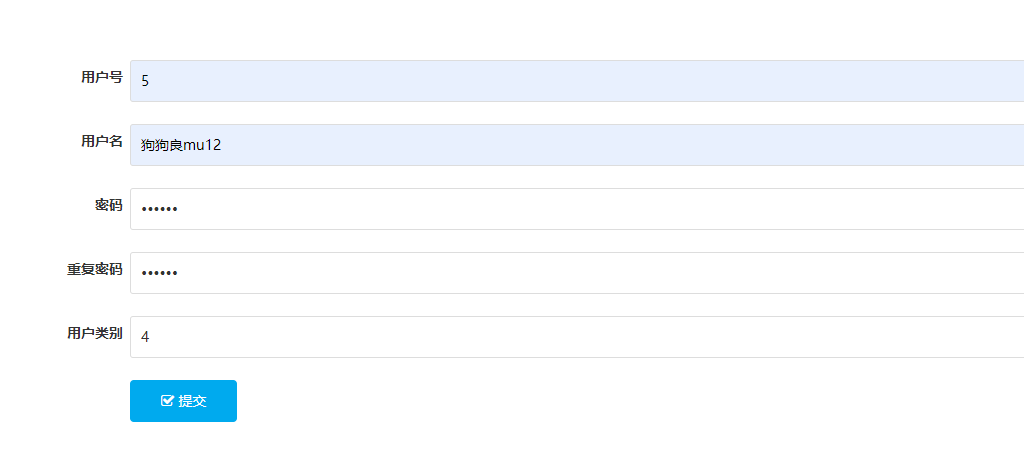
先点击修改

，

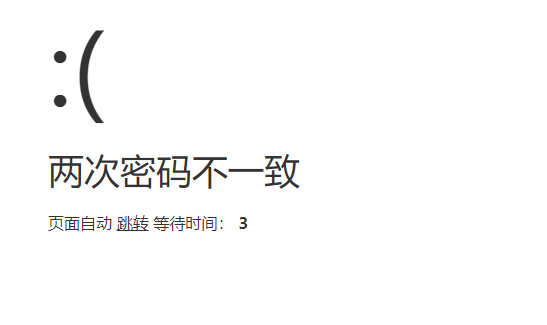
修改用户名，然后提交，我们就跳转回到刚才的页面。信息已经被修改：



界面右上角还有添加用户的按钮，在这里我们可以添加新用户：



这里的输入是有验证器验证的，如果信息为空，或者密码与重复密码不相同，则会弹出提示：





而且刚才输入的用户号用户名内容不会丢失，在输入正确的用户信息后，我们就能添加一条用户信息：

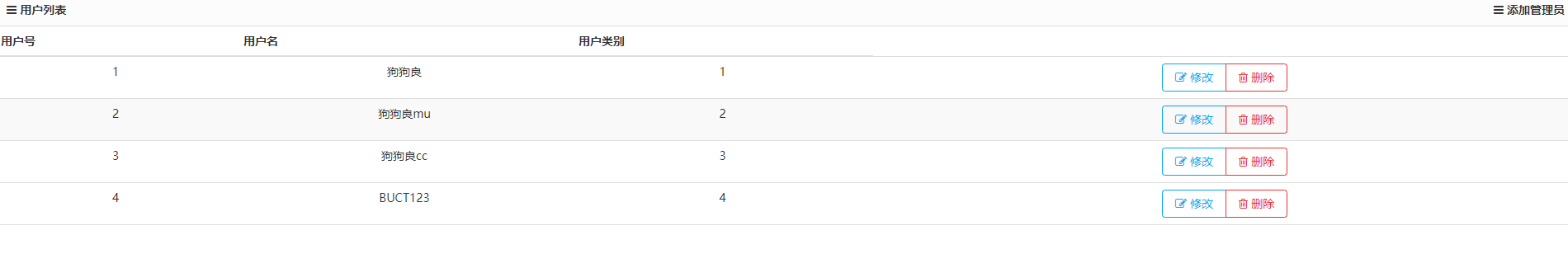




如果我们想删除一条信息，只要点击删除按钮即可：

如我们删除刚插入的信息：





我们现在是用户管理员模式，意味着我们只能访问用户界面，但是如果我们知道其他表的路由，是否可以直接访问呢？



答案是不行，因为每个用户的用户号在访问页面前都会有一个专门的控制器去检测这个用户的用户类别，除非是超级管理员，否则是不能随便访问的。

当然，如果你没有登录，那更不可以进入系统了：



接下来是维修员：



维修员登录后会跳转到维修信息表，题目要求维修信息表只有添加功能，这里便没有给出修改删除的接口。右上角有添加功能和进入故障设备信息表功能，我们先进入故障设备信息表，

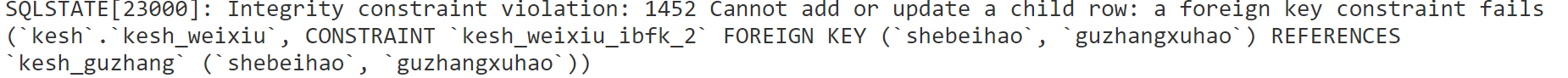


这里我们能够添加删除修改，和用户表的功能基本一致，这里就不在演示。

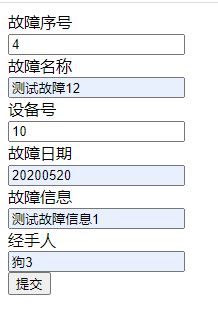
我们返回维修信息表，插入一条数据：



但是这条数据并不能插入成功，因为维修表中的设备号和故障序号要与故障表内的一致，否则是不能插入的。



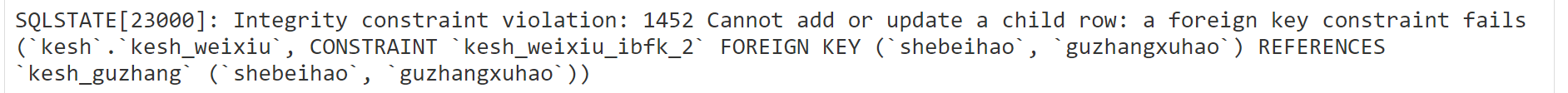
所以我们先去故障表内插入一条故障序号为4，设备号为10的信息

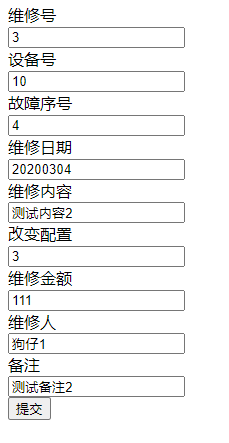


同理，故障设备表的信息，如设备号，也必须是设备表内有的内容，否则无法插入。

然后我们再去维修表内插入内容：

这里也会验证故障序号设备号的匹配性，如果故障表里的10号设备是4号故障，那么维修表里如果不是这样的话，还是会报错：



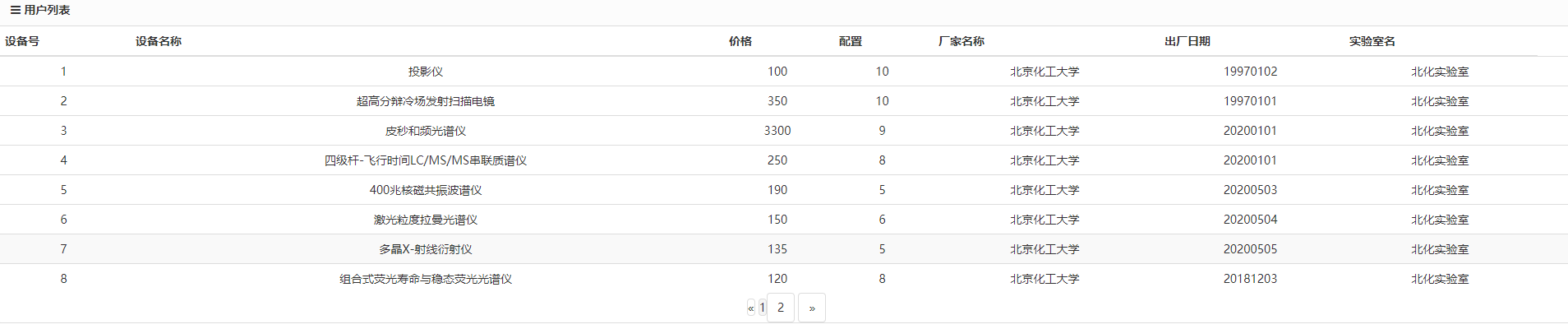


这样我们就插入成功了：





接下来是最简单的普通用户，普通用户只有浏览设备表的权限，他只能去查阅，但是不能修改删除。

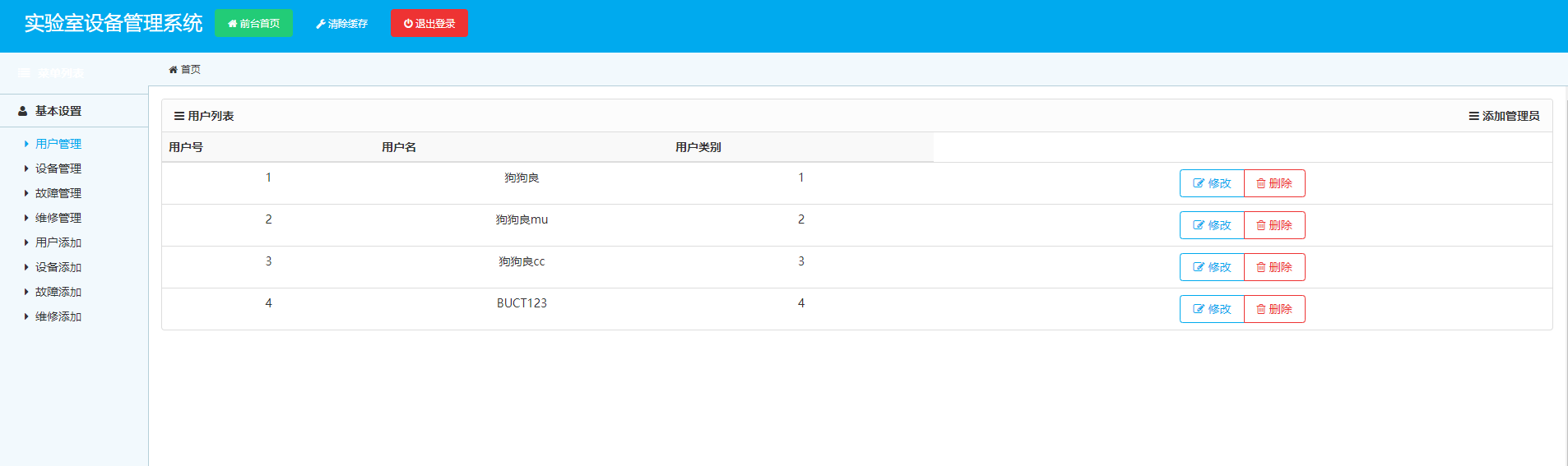


可以看到是没有修改删除的接口的，而且因为设备众多，我们用分页的方法使页面看起来更简洁一些。

当然他也是不能通过路由访问其他页面的：



接下来就是权限最大的超级管理员，超级管理员可以访问所有的界面，拥有修改一切信息的权力，用户类别是4，为了简洁清晰，我使用了一个网页的模板，使得其他网页能够直接套用在一个网页下，方便访问：



左侧是所有的界面：

如设备管理界面：



注意这里的设备管理是有修改删除功能的，方式和开始的用户管理一样。

还有设备添加：

、

等等所有的功能。上面还有三个按钮，最主要的是退出登录，退出登录后，我们会返回到登录界面，然后取消我们的登录界面，这样我们就不能通过路由访问了：





总结与反思：

在做课设的过程中，我从零开始学习了thinkphp 的技术，thinkphp是国内公司使用的非常多的一款框架，我觉得课设并不是终点，我学习thinkphp也一定不会因为课设结束而停止，在学习了数据库原理之后，我也对系统开发较以往有了更加清晰的理解和认识，对于数据库的理解也不在局限于仅仅是储存数据的一张表，他还有很多约束关系，运算方法，这些都只有自己在亲身设计数据库时才能体会到，这次课设让我了解了一般网页开发的流程，万事开头难，这次的课设做完之后，我一定会继续研究这样的开发技术，锻炼自己的能力，让自己更上一层楼。

这次完成课设的过程中出现了大大小小非常多的错误，有些错误排查一天都难以发现问题，有的是路径问题，有的是环境变量问题，有的是函数用法问题，再加上我是从零开始学习，遇到的错误会更多，解决错误的难度也会增大，所以在开发的过程中养成一颗平常心是非常重要的，遇到错误不能自暴自弃，而是要运用各种办法去解决它，比如让我印象最深刻的一个问题：如何实现跨控制器的跳转？这个问题在课设的最终阶段遇到，主要是用于超级管理员页面的跳转，当时使用了好多种办法都没有解决，突然想到，用$this->success（）这个办法，这样就巧妙的避开了一些复杂的参数设置，直接用最简单的办法进行跳转，所以碰到问题时的思路也要灵活，做课设或者开发的时，错误是不可避免的，所以解决错误的能力也是很重要的。

参考资料：

https://www.bilibili.com/video/BV1Vx411V7fE 千锋教育ThinkPHP5.0框架全方位解读与实战

https://www.bilibili.com/video/BV17J411K7Zd 李炎恢ThinkPHP5.1PHP框架十天精品课堂

thinkphp5.0官方文档