

**信息系统设计实训报告**

**2018-2019-3学期**

**2019.7.1**

小组成员组成及成绩评定

|  |  |  |  |
| --- | --- | --- | --- |
| **小组总评成绩** | |  | |
| 姓名 | 学号 | 负责内容概况 | 成绩 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

撰写报告说明：

1. 信息系统设计实训是以项目为基础，以小组合作完成，因此，报告为小组报告，每小组只需要上交一份，但在每小组各个成员需全程参与，分工完成，报告中需要明确每位小组成员负责内容，作为成绩评定的依据。
2. 报告需按照指定格式完成，不得随意删减内容。报告正文使用宋体小四号字，1.5倍行距，一律用A4纸单面打印，页边距上、下、左、右均为2.5厘米。
3. 正文分章节撰写，第一级标题用“第1章”、“第2章”、“第3章”等连续编号，每章应另起一页，标题末尾不加标点(问号、叹号、省略号除外)，标题居中排列，下空一行接写第二级标题。从第二级标题开始，用阿拉伯数字连续编号，在不同层次的数字之间加一个下圆点相隔，最末数字后不加标点。如第二级标题为“1.1”、“2.1”、“3.1”等，第三级标题为“1.1.1”、“2.1.1”、“3.1.1”等，第四级标题为“1.1.1.1”、“2.1.1.1”、“3.1.1.1”等。正文中的标题一般不超过四级，标题层次要清晰，第二至第四级标题均单独占一行，且靠左端书写，第二级标题序数前不留空格，第三、四级标题序数前要空两个汉字位置。各级标题序数后均空一格接写标题。
4. 每幅图都应有图题，图题由图号和图名组成。图号按章编排，如“图2-4”表示第二章第4张插图，图号与图名之间空一格排写，图题居中置于图下，图中若有分图时，分图号用(a)、(b)等置于分图之下。每个表格应有自已的表题和表序，表题应写在表格上方正中，表序写在表题左方不加标点，空一格接写表题，表题末尾不加标点。表格应逐章编序，如“表2-2”表示第二章的第2张表。表序必须连续。表格允许下页接写，接写时表题省略，表头应重复书写，并在右上方写“续表××”。数字空缺的格内加“－－”字线（占2个数字），不允许为空；表中有附注时，写在表的下方，句末加标点。
5. 首页、成员组成页、撰写说明页均为一页，其余内容根据实际情况确定页数。

目录

[第一章 项目概况及要求 5](#_Toc12817985)

[1.1 项目背景 5](#_Toc12817986)

[第二章 系统需求分析 5](#_Toc12817987)

[2.1 用户功能要求 5](#_Toc12817988)

[2.2 系统性能要求 6](#_Toc12817989)

[2.3 输入输出要求 6](#_Toc12817990)

[2.4 用户其他要求 6](#_Toc12817991)

[第三章 系统分析 7](#_Toc12817992)

[3.1 组织结构及业务流程分析 7](#_Toc12817993)

[3.2 用例图分析(含用例描述和活动图） 9](#_Toc12817994)

[3.3.实体类图分析 10](#_Toc12817995)

[第四章 系统设计 11](#_Toc12817996)

[4.1 总体设计( 功能树设计） 11](#_Toc12817997)

[4.2 顺序图设计 12](#_Toc12817998)

[4.3 类图设计 13](#_Toc12817999)

[4.4 编码设计（编码规则） 13](#_Toc12818000)

[4.5 输入输出设计 13](#_Toc12818001)

[4.6 模块算法设计 14](#_Toc12818002)

[第五章 系统测试与部署 14](#_Toc12818003)

[5.1 系统架构选择（或应用程序结构设计） 14](#_Toc12818004)

[5.2 系统部分代码示例 16](#_Toc12818005)

[5.3 系统界面实现（贴界面实际图） 16](#_Toc12818006)

[5.4 系统测试 17](#_Toc12818007)

[5.5 系统实施应用 17](#_Toc12818008)

[第六章 系统测试与部署 17](#_Toc12818009)

[第七章 其他说明 17](#_Toc12818010)

[第八章 反思日志 17](#_Toc12818011)

# 第一章 项目概况及要求

## 1.1 项目背景

本项目来自企业实际某油田公司存在如下问题：

一是固定资产具有数量大、种类多、价值高、使用周期长、使用地点分散等特点，管理难度大。二是很多单位目前仍然依赖手工记账的管理方式，由于管理单据众多、盘点工作繁重，需占用大量的人力物力，而且固定资产的历史操作和资产统计工作异常困难，导致资产流失和资产重复购置，使单位成本大幅增加。三是存在账、卡、物不相符合，难于满足现代管理的需要，由于缺乏有效的资产实物的日常管理手段，即使单位花大力气进行了资产清查，没多久，账实不符的情况又会重新出现，因此，必须有一套有效的管理手段对实物进行管理。四是固定资产缺乏中间跟踪管理，没有固定资产的历史记录，如安装、移动、调拨、报废、维修等。

# 第二章 系统需求分析

## 2.1 用户功能要求

根据企业的实际运行状况，可以将系统的功能需求分为四个部分，分别是资产管理功能、资产信息统计分析功能、基本信息 管理、其他功能。资产信息管理包括资产的上报、审核、记录等。资产信息统计分析功能指资产信息的统计和应用功能。基本信息 管理包括资产种类的管理、用户信息的管理等。其他功能包括系 统信息库、系统结构管理等。

## 2.2 系统性能要求

企业的资产管理系统需要一定的日常维护，并为用户提供信息的共享和查询平台，但在登录使用权限上必须做好管理和限制。要登录企业资产管理的平台，用户必须首先到企业的相关负责人处进行登记，在完成身份信息的核对后，再给予用户一定的账号用以登录系统。在操作权限上也应该进行划分。

## 2.3 输入输出要求

键盘输入、以web页面输出

## 2.4 用户其他要求

企业资产管理系统的管理人员拥有对系统内容进行更改、填充、删除的权限，而登录用户只拥有浏览信息的权限，而无法更改系统内部的信息。其次要做好系统数据的备份。资产管理系统中的数据都具有 重要的价值，一旦丢失将会影响企业的正常运行。为了防止系统 出现故障导致信息的丢失，必须对数据做好备份工作

对企业的资产管理系统来说，系统的安全性是十分关键的。企业的资产管理系统处理的都是企业生产运行过程中产生的重要资金数据，涉及到了企业的决策制定和计划执行等重要内部机密，一旦发生泄露将会对企业造成严重的经济损失，因此必须确保企 业资产管理系统的安全性。

# 第三章 系统分析

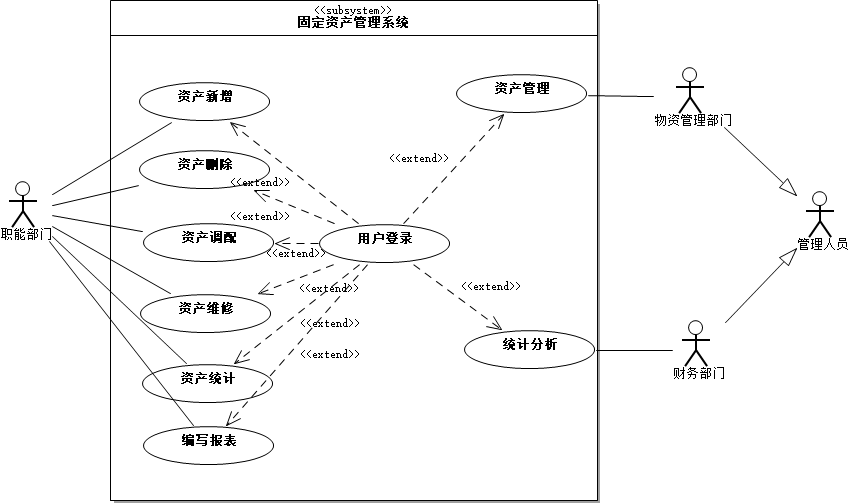
## 3.1 组织结构及业务流程分析

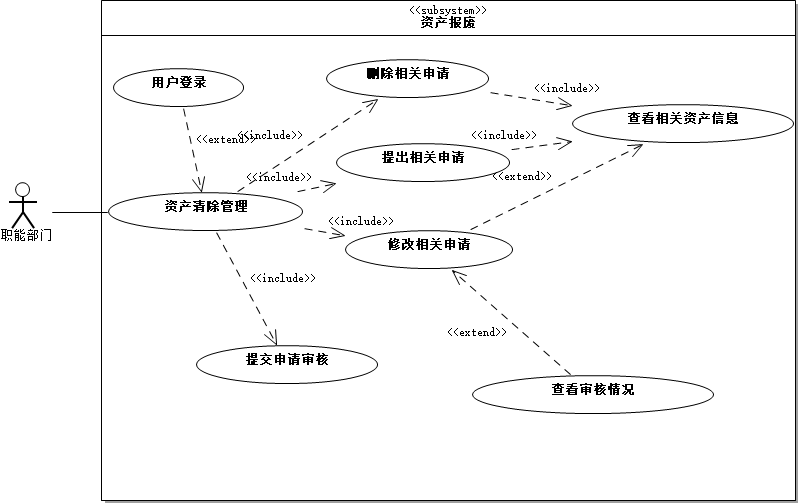
企业结构图

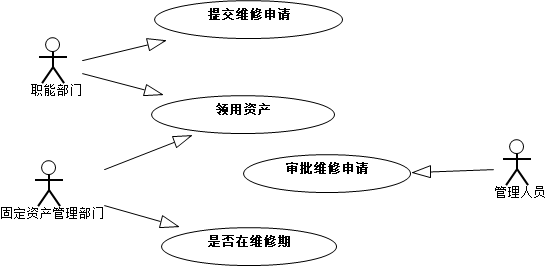




## 3.2 用例图分析(含用例描述和活动图）

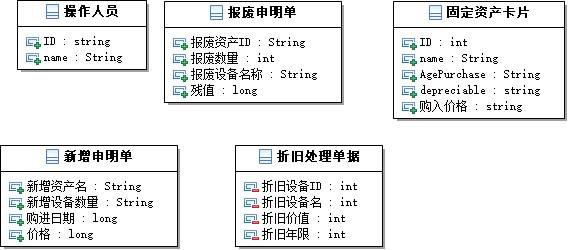


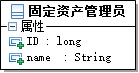
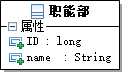
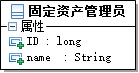




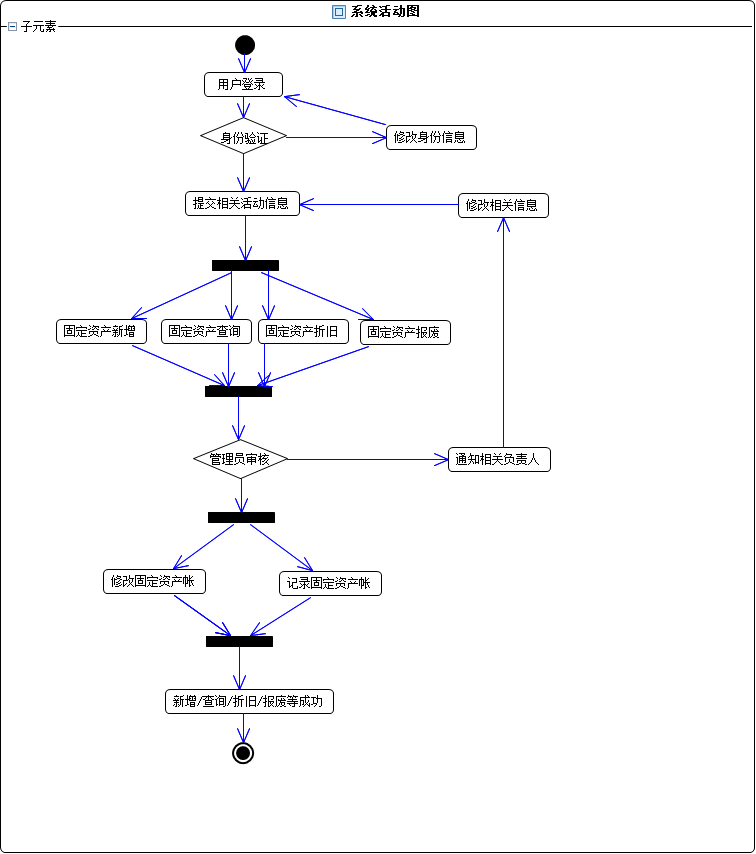
# 第四章 系统设计

## 3.3.实体类图分析

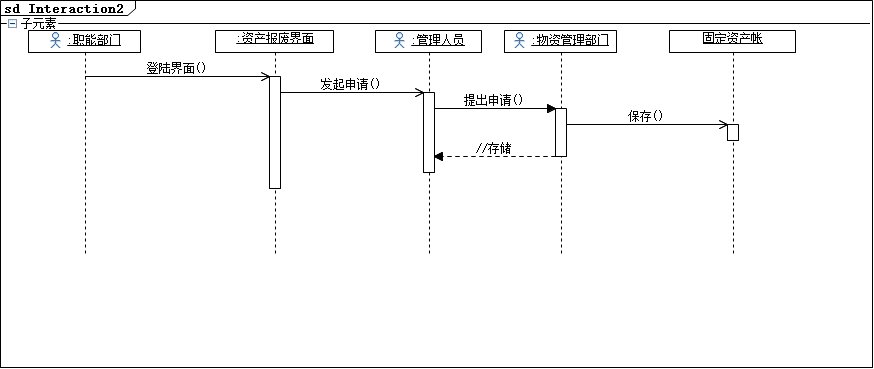


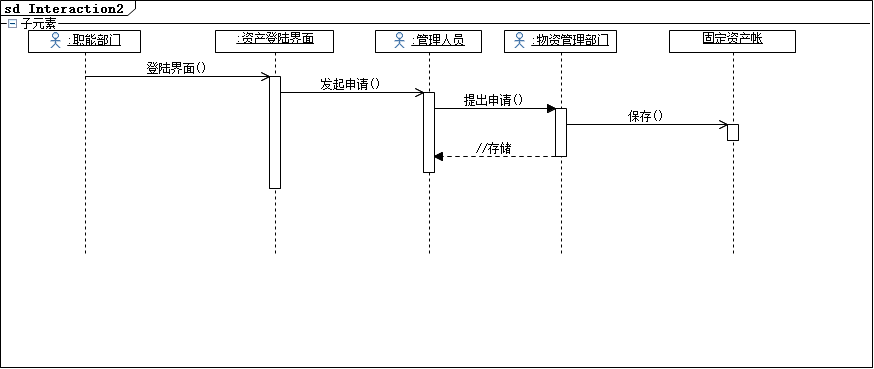
  

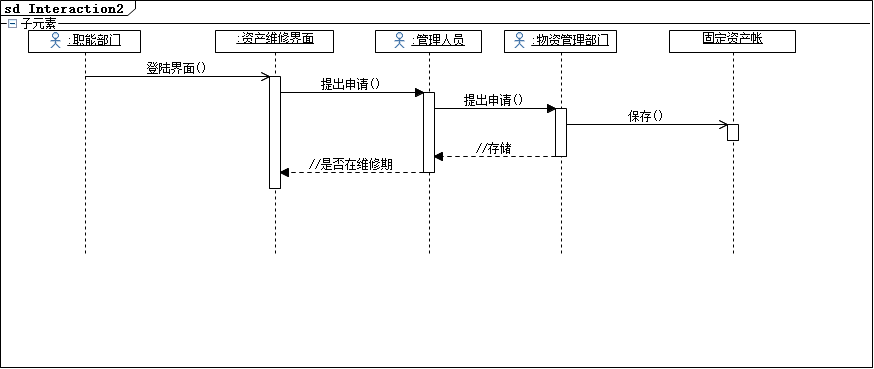
## 4.1 总体设计( 功能树设计）



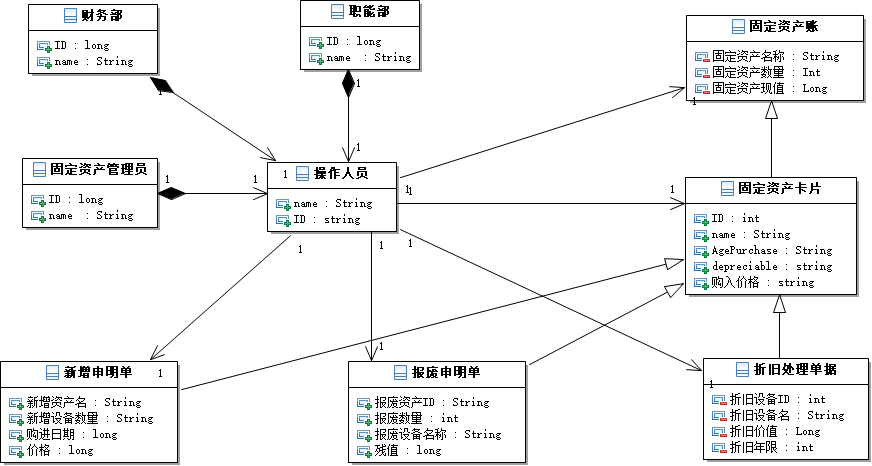
## 4.2 顺序图设计







## 4.3 类图设计

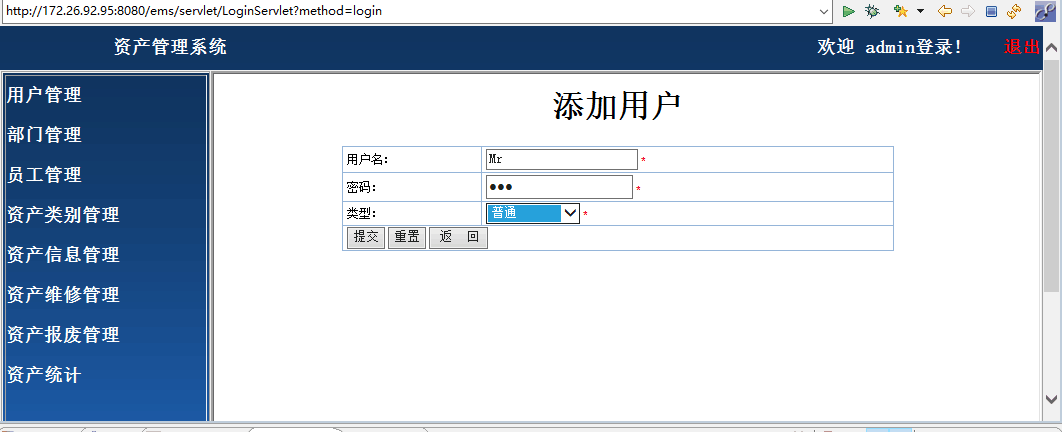


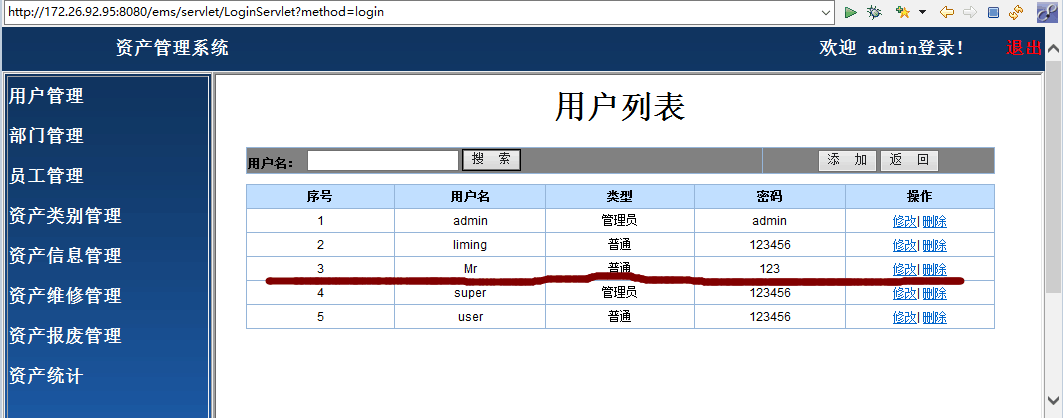
## 4.4 编码设计（编码规则）

编码是在固定资产管理中是用于唯一标识固定资产信息的数据，良好的编码能够在现实的操作中加快检索查询速度，更能提高查询的精度与效率，是一个良好的管理软件所不可或缺的部分，本系统的编码规则如F:

1. 固定资产信息表:采用10位编码，以ZC开始，如ZC10000001；
2. 报废固定资产表:采用10位编码，以BF开始，如BF10000001；
3. 维修固定资产表:采用10位编码，以WX开始，如WX0000001

## 4.5 输入输出设计





# 第五章 系统测试与部署

## 5.1 系统架构选择（或应用程序结构设计）

分为表现层、服务层、持久层

系统架构图



服务层

持久层

表现层

## 5.2 系统部分代码示例

1.用户登录

package com.bean;

public class Users {

int uid;

String uname;

String upassword;

String utype;

public int getUid() {

return uid;

}

public void setUid(int uid) {

this.uid = uid;

}

public String getUname() {

return uname;

}

public void setUname(String uname) {

this.uname = uname;

}

public String getUpassword() {

return upassword;

}

public void setUpassword(String upassword) {

this.upassword = upassword;

}

public String getUtype() {

return utype;

}

public void setUtype(String utype) {

this.utype = utype;

}

}

1. 页码导航

package com.bean;

public class PagingModel {

private int CurrentP; //当前页码

private int AllP; //总页数

private int AllR; //总记录数

private int PerR; //每页显示记录数

private String pageLink; //分页导航栏信息

private String pageInfo; //分页状态显示信息

public PagingModel(){

CurrentP=1;

AllP=1;

AllR=0;

PerR=3;

pageLink="";

pageInfo="";

}

/\*\* 设置每页显示记录数 \*/

public void setPerR(int PerR){

this.PerR=PerR;

}

/\*\* 设置总记录数 \*/

public void setAllR(int AllR){

this.AllR=AllR;

}

/\*\* 计算总页数 \*/

public void setAllP(){

AllP=(AllR%PerR==0)?(AllR/PerR):(AllR/PerR+1);

}

/\*\* 设置当前页码 \*/

public void setCurrentP(String currentP) {

if(CurrentP<1)

CurrentP=1;

if(CurrentP>AllP)

CurrentP=AllP;

CurrentP = Integer.parseInt(currentP);

}

/\*\* 设置分页状态显示信息 \*/

public void setPageInfo(){

if(AllP>1){

pageInfo="<table border='0' cellpadding='3' ><tr><td style='width:100%;'>";

pageInfo+="每页显示："+PerR+"/"+AllR+" 条记录！";

pageInfo+="当前页："+CurrentP+"/"+AllP+" 页！";

pageInfo+="</td></tr></table>";

}

}

/\*\* 设置分页导航栏信息 \*/

public void setPageLink(String gowhich){

if(gowhich==null)

gowhich="";

if(gowhich.indexOf("?")>=0)

gowhich+="&";

else

gowhich+="?";

if(AllP>1){

pageLink="<table border='0' cellpadding='3' ><tr><td style='width:100%;'>";

if(CurrentP>1){

pageLink+="<a href='"+gowhich+"showpage=1'>首页</a>&nbsp;";

pageLink+="<a href='"+gowhich+"showpage="+(CurrentP-1)+"'>上一页</a>&nbsp;";

}

if(CurrentP<AllP){

pageLink+="<a href='"+gowhich+"showpage="+(CurrentP+1)+"'>下一页</a>&nbsp;";

pageLink+="<a href='"+gowhich+"showpage="+AllP+"'>尾页</a>";

}

pageLink+="</td></tr></table>";

}

}

/\*\* 返回每页记录数 \*/

public int getPerR(){

return PerR;

}

/\*\* 返回总记录数 \*/

public int getAllR() {

return AllR;

}

/\*\* 返回总页数 \*/

public int getAllP() {

return AllP;

}

/\*\* 返回当前页码 \*/

public int getCurrentP() {

return CurrentP;

}

/\*\* 返回分页状态显示信息 \*/

public String getPageInfo(){

return pageInfo;

}

/\*\* 返回分页导航栏信息 \*/

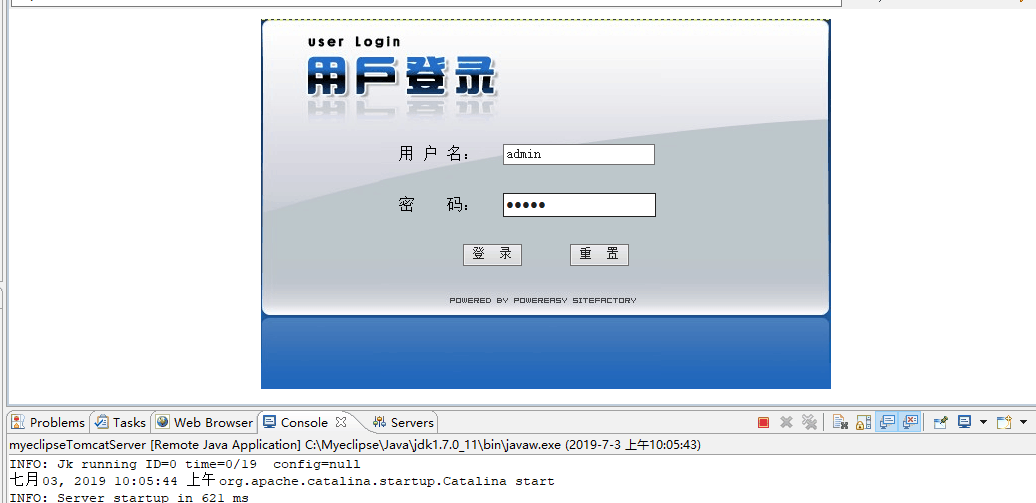
public String getPageLink(){

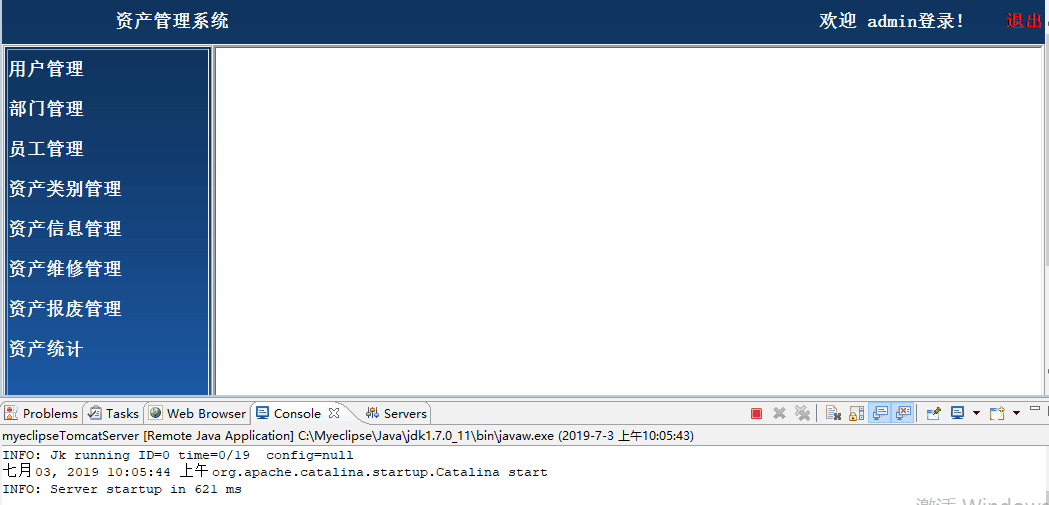
return pageLink;

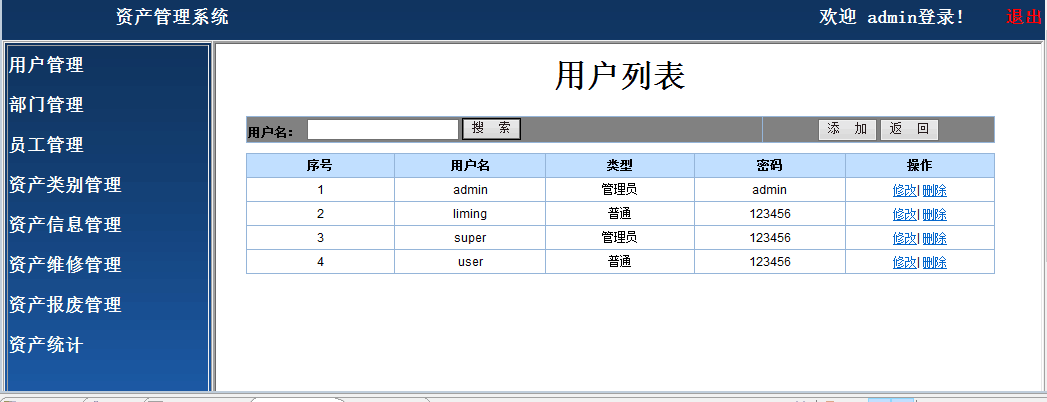
}

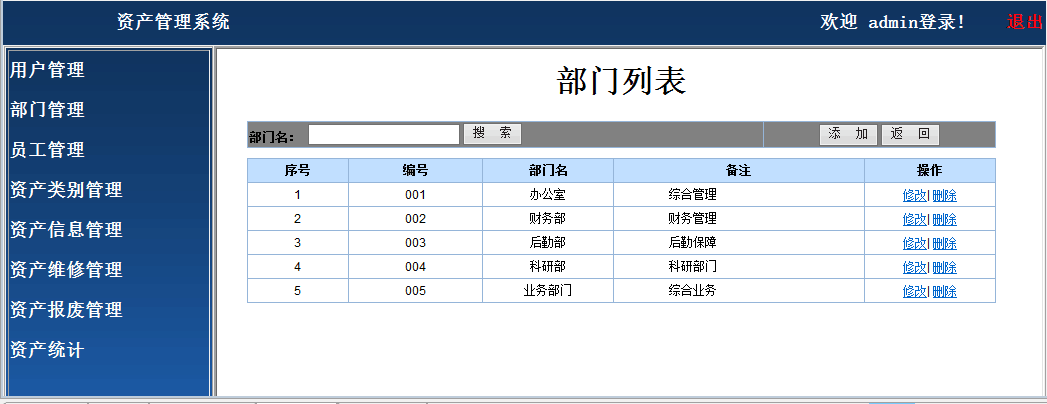
}

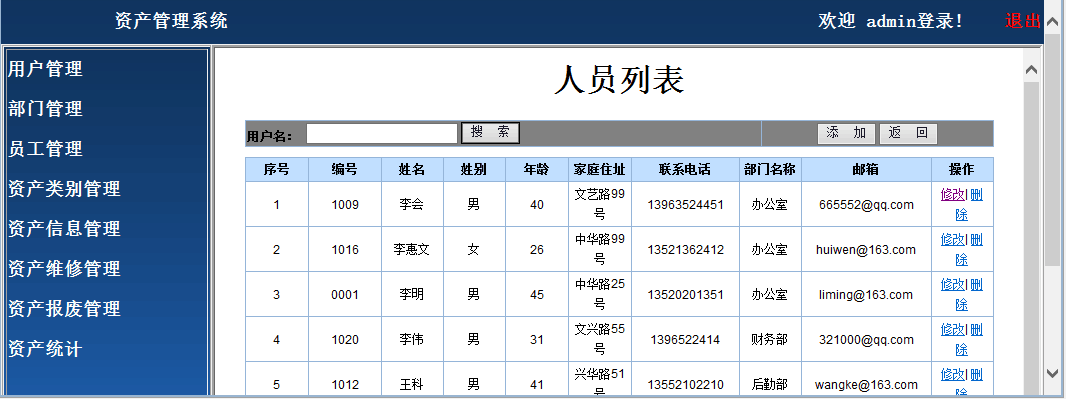
## 5.3 系统界面实现（贴界面实际图）

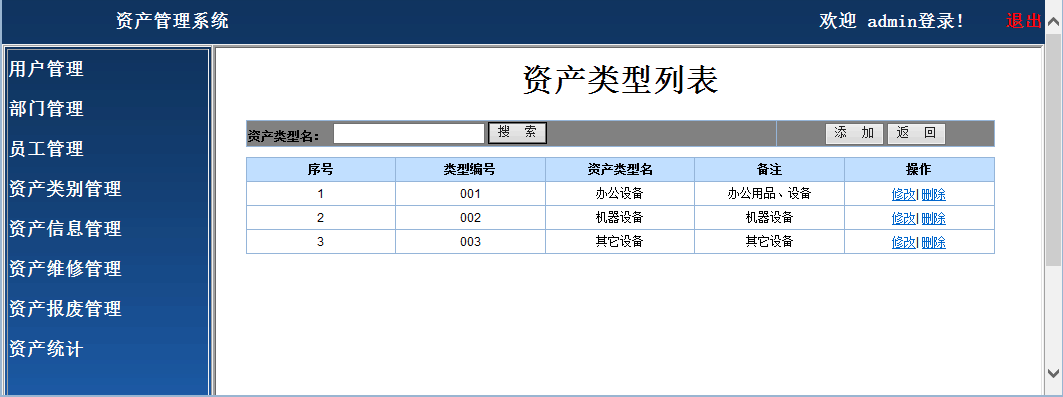


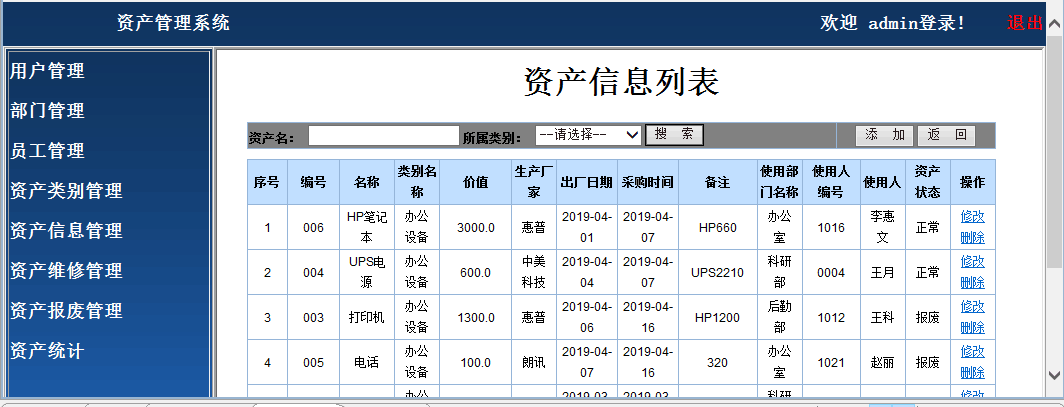


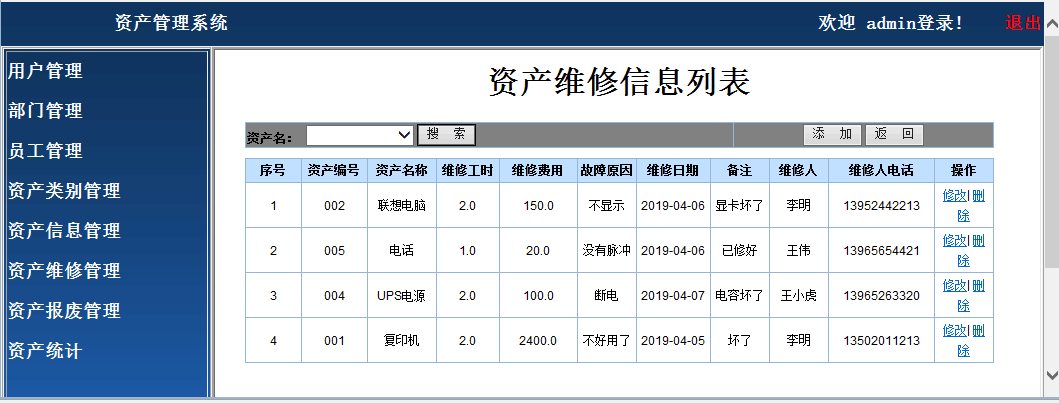


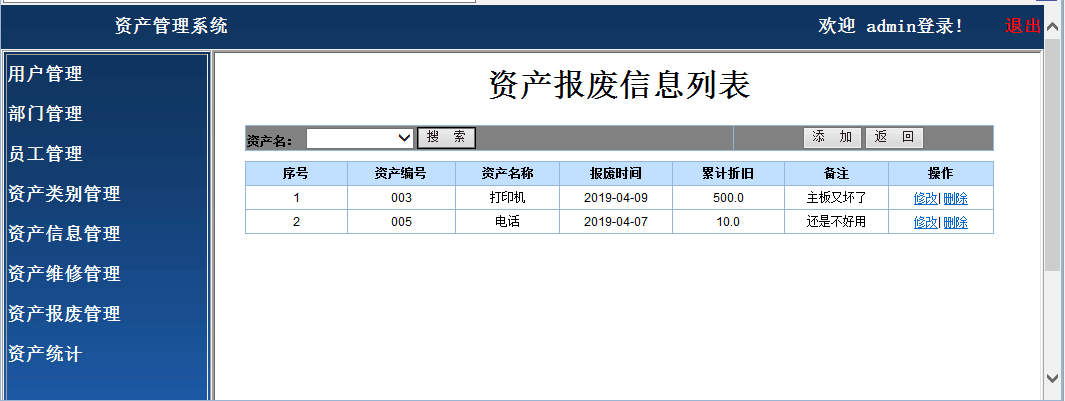


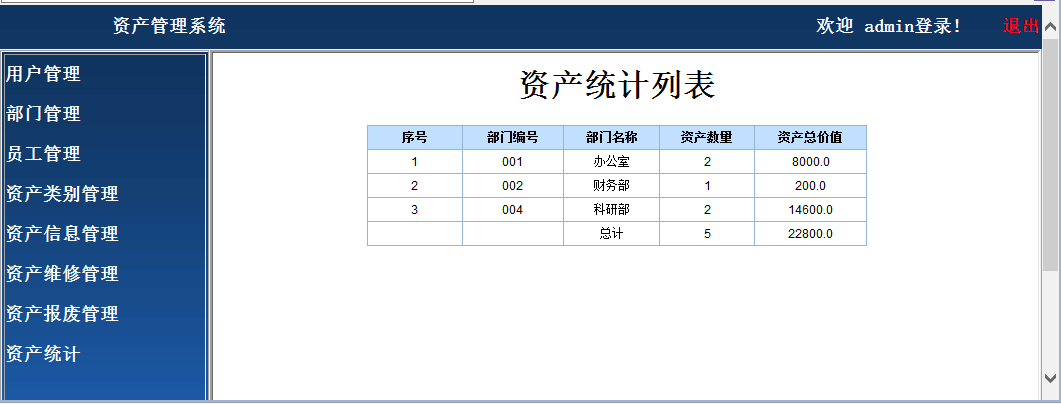






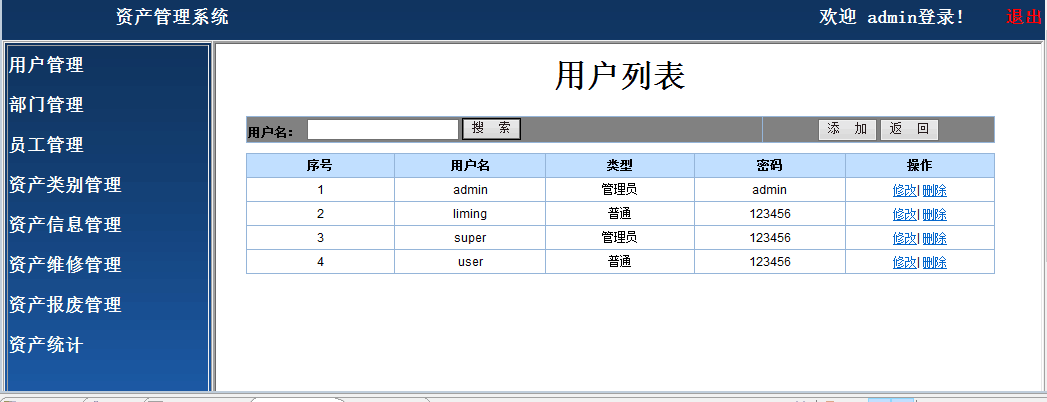


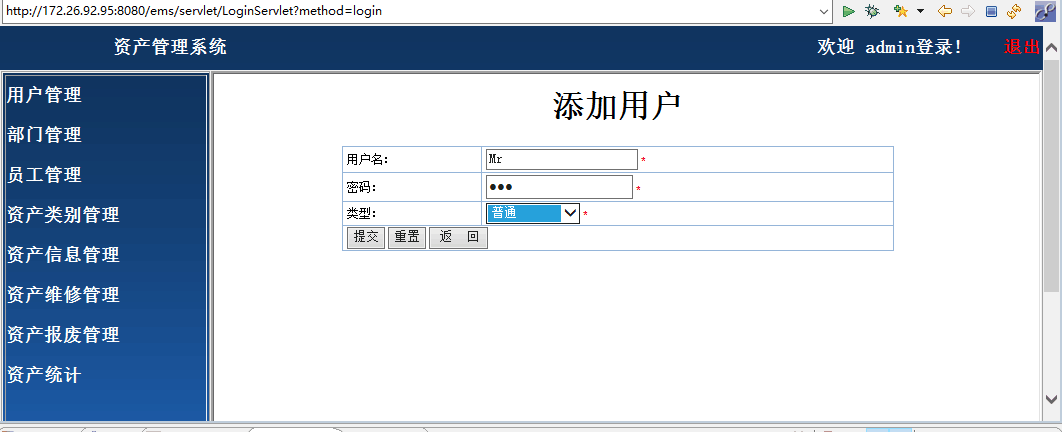


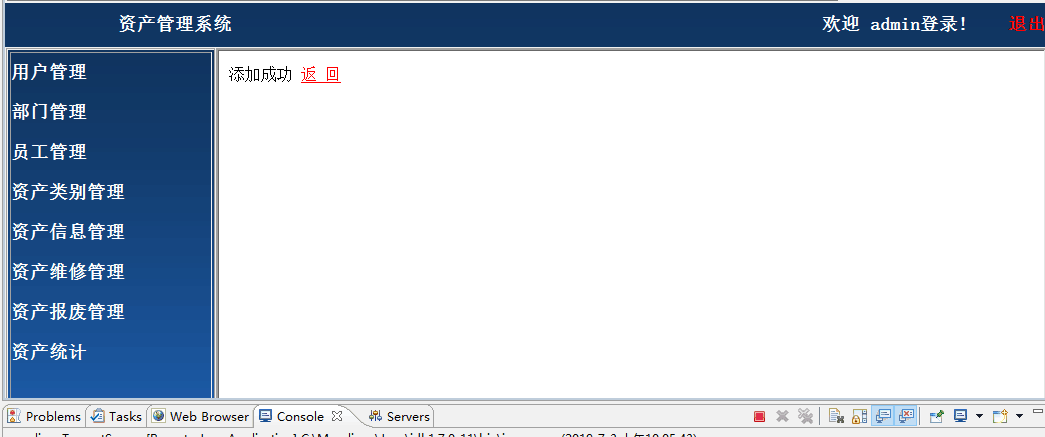


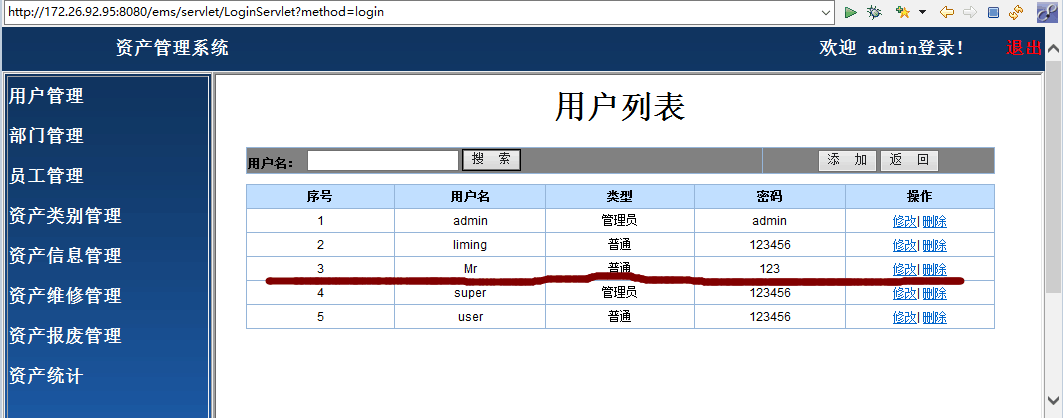
## 5.4 系统测试

1.用户管理









# 系统测试

## 6.1测试的步骤

## 与开发过程类似，测试过程也必须分步骤进行，每个步骤在逻辑上是前一个步骤的继续。大型软件系统通常由若干个子系统组成，每个子系统又由若千个模块组成。因此，大型软件系统的测试基本上由下述几个步骤组成:

1. 模块测试在这个测试步骤中所发现的往往是编码和详细设计的错误。
2. 系统测试在这个测试步骤中发现的往往是软件设计中的错误，也可能发现需求说明中的错误。
3. 验收测试在这个测试步骤中发现的往往是系统需求说明书中的错误。

## 6.2测试的主要内容

为了保证测试的质量，将测试过程分成几个阶段，即:代码审查、单元测试、集成测试、确认测试和系统测试。

(1) 单元测试

单元测试集中在检查软件设计的最小单位一模块上,通过测试发现实现该模块的实际功能与定义该模块的功能说明不符合的情况，以及编码的错误。

(2)集成测试

集成测试是将模块按照设计要求组装起来同时进行测试，主要目标是发现与接口有关的问题。如一个模块与另一个模块可能有由于疏忽的问题而造成有害影响;把子功能组合起来可能不产生预期的主功能;个别看起来是可以接受的误差可能积累到不能接受的程度;全程数据结构可能有错误等。

(3)确认测试

确认测试的目的是向未来的用户表明系统能够像预定要求那样工作。经集成测试后，已经按照设计把所有的模块组装成一个完整的软件系统，接口错误也已经基本排除了，接着就应该进一步验证软件的有效性，这就是确认测试的任务，即软件的功能和性能如同用户所合理期待的那样。

(4)系统测试

软件开发完成以后，最终还要与系统中其他部分配套运行，进行系统测试。包括恢复测试、安全测试、强度测试和性能测试等。单独对系统的测试主要从以下几入手:

1. 功能测试:测试是否满足开发要求，是否提供设计所描述的功能，是否用户的需求都得到满足。功能测试是系统测试最常用和必须的测试，通常还会以正式的软件说明书为测试标准。
2. 强度测试及性能测试:测试系统能力最高实际限度，即软件在一些超负荷情况下功能实现的情况。
3. 安全测试:验证安装在系统内的保护机构确实能够对系统进行保护，使之不受各种非常的干扰。针对本系统主要是对权限系统的测试和对无效数据、错数据、和非法数据干扰的能力的测试。

经过上述的测试过程对软件进行测试后，软件基本满足开发的要求，测试宣告结束。

# 其他说明

## 7.1系统的维护

## 本项目的运行主要包括以下内容:

## 启动Tomcat应用服务器后，打开Internet Explorer，在URL地址中输入http://localhost:8080/index.jsp,即可运行所写的JSP页。也可在MyEclipse环境下将工程属性中的Server改为Tomcat,按照上面的部署过程部署后，直接运行JSP页面即可。

本项目的维护主要包括以下内容:

1. 权限维护：管理员需要对用户的用户名、密码和权限进行维护，给每名员工分配享用的用户名和密码，确定操作级别。
2. 数据库维护：在日常运行中要不断对数据库管理及其表进行修改，对无用的表及表中的信息进行删除处理。这样确保系统能够正常，快速地运行。

# 第八章 反思日志