成 绩 评 定 表

学生姓名	宁高聪	班级学号	1503130115
专　业	网络工程	课程设计题目	教室管理系统
评 语	组长签字：		
成绩			
日期	2017 年 7 月 13 日		

课程设计任务书

学　院	信息学院	专 业	网络工程
学生姓名	宁高聪	班级学号	1503130115
课程设计题目	**教室管理系统**		

实践教学要求与任务：

1. 任务描述

使用 JAVA 作为编程语言，mysql 作为数据库，完成教室管理系统的设计与实现。教室管理系统包括教师信息的管理、教师信息管理、教室安排信息管理等。

2. 具体要求（每个组员根据各自的工作从下面的任务选择或新增相应的工作）

（1）教室信息，包括教室容纳人数、教室空闲时间、教室设备等；

（2）教师信息，包括教师姓名、教授课程、教师职陈、安排上课时间等；

（3）教室安排信息，包括何时空闲、空闲的开始时间、结束时间等。

（4）按照一定条件查询、统计

3. 本人任务

（1）教室信息，包括教室容纳人数、教室空闲时间、教室设备等；

（2）教师信息，包括教师姓名、教授课程、教师职陈、安排上课时间等；

（3）教室安排信息，包括何时空闲、空闲的开始时间、结束时间等。

（4）按照一定条件查询、统计

工作计划与进度安排：

19 周：布置课程设计任务，查阅资料，分组设计

20 周：实验室编写代码与调试，验收，答辩，编写课程设计报告。

指导教师：	专业负责人：	学院教学副院长：
谭小波、关世杰	周越	张文波
2017 年 7 月 3 日	2017 年 7 月 3 日	2017 年 7 月 3 日

摘要

从互联网出现到现在，网络已经成为人才进行流动的最主要渠道。在我国，随着网络技术的不断发展，网络招聘求职开始走入人们的视野，并越来越成为企业招聘人才，求职者应聘主要渠道之一。网络招聘在国内处于主流地位，正在突破传统招聘求职与互联网单一媒体的束缚，整合平面媒体和电视媒体，打造跨平台招聘服务的整合平台。本系统正是基于为招聘者营造一个好的交流平台的思想而设计开发的。网上招聘系统的设计首先应该设计完善招聘求职的基本功能，明确网站的需求，然后才能明确系统的数据库设计，并通过定义的功能逐步实现其实际网页和用例流程的开发。在完成系统后还要对系统进行详细的测试才能发布网站，以及对系统进行必要的维护工作，以便使网站功能与内容保持信息的及时性，使之能真正成为一个能解决实际问题的网上招聘求职系统。

关键词　B/S 架构；Hibernate；Struts2；Bootstrap

目录

1 问题描述

随着计算机技术、网络技术和信息技术的发展，现在办公系统更趋于系统化、学化和网络化。网络办公自动化系统是计算机技术和网络迅速发展的一个办公应用解决方案，它的主要目的是实现信息交流和信息共性，提供协同工作的手段，提高办公的效率，让人们从繁琐的有纸办公中解脱出来。现在许多的机关单位的人事管理水平还停留在纸介质的基础上，这样的机制已经不能适应时代的发展，因为它浪费了许多的人力和物力，在信息时代这种传统的管理方法必然被计算机为基础的信息管理所取代。本系统是对教室的使用情况进行管理，为用户提供了一套操作简单、使用可靠、界面友好、易于管理和使用的处理工具。本系统对教室使用情况进行统一处理，避免数据存取、数据处理的重复，提高工作效率，减少了系统数据处理的复杂性。本系统不仅使管理人员从繁重的工作中解脱出来，而且提高了教室管理的效率，提高了教室管理的科学性，方便了用户查询、管理人员进行管理。

2 需求分析

2.1 需求分析

本系统采用的是 B/S 模式，其主要的功能在服务器上执行。B/S 结构的客户端可以完成浏览、查询、数据输入等简单功能，绝大部分工作要由服务器承担，包括对数据的保存，如：数据存储、恢复，以及对系统失效的后果及恢复的处理方法等。有教师账号的教师可以通过浏览器访问查看教室的使用情况以及线上申请教室的使用权限等。

2.2 系统功能结构

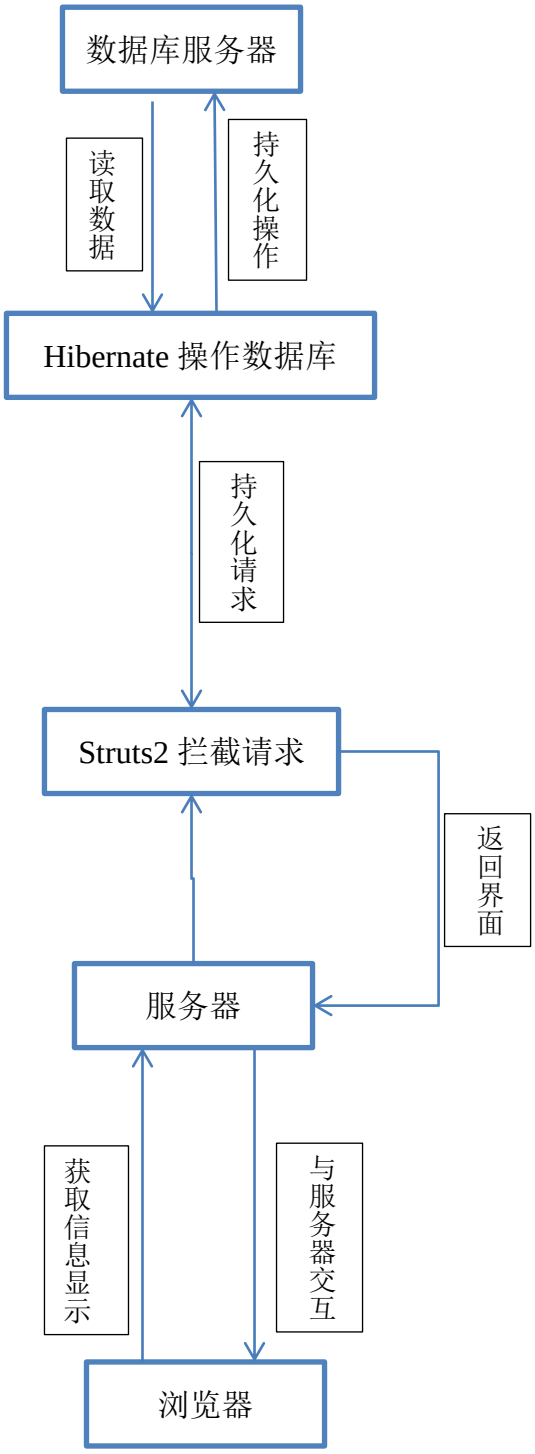

图 2.1 系统功能结构图

2.3 逻辑结构设计

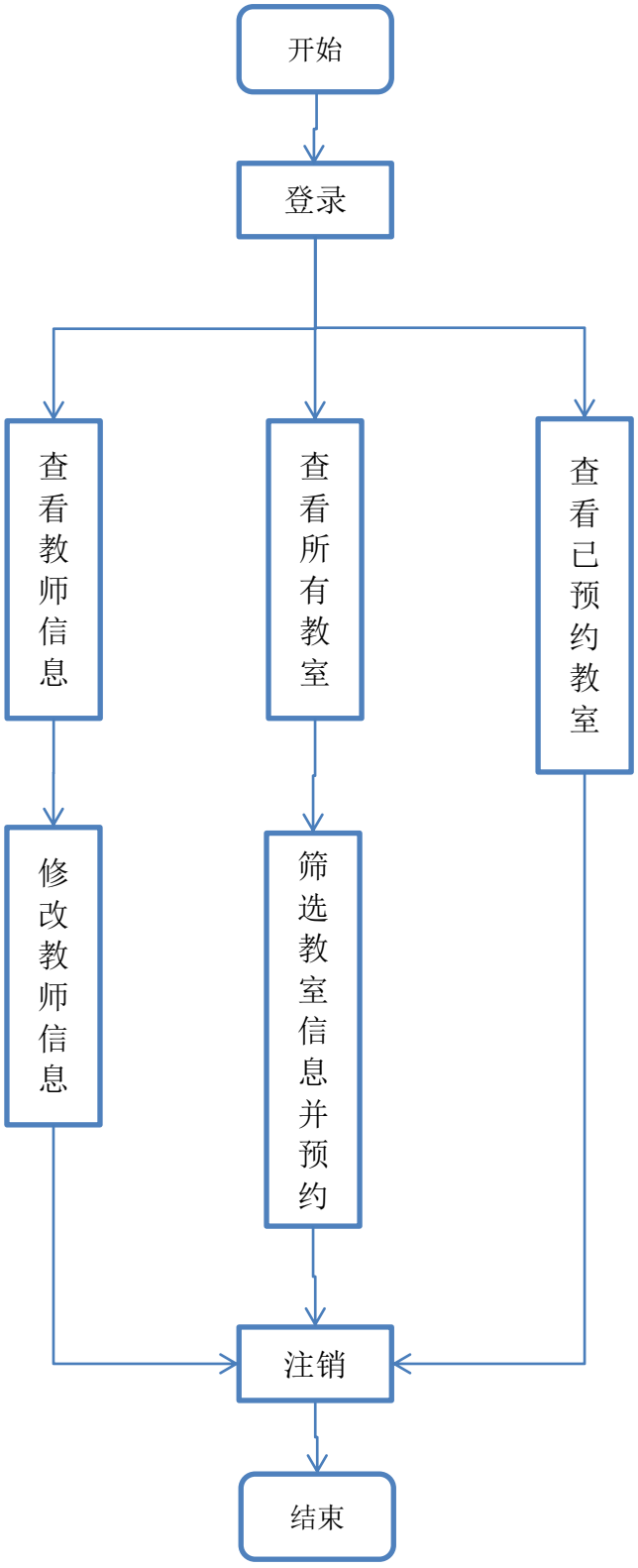

图 2.3.1 逻辑结构图

3 系统设计与实现

3.1 系统开发环境

硬件环境：ACER 笔记本一台

软件环境：MySQL Workbench + IntelliJ IDEA 2017

3.2 详细设计与实现

本程序采用 B/S 架构设计，使用数据、业务、视图分离的方式设计。

视图使用 HTML+JSP 设计，使用了 Twitter 提供的 bootstrap 样式库，并使用 javascript 实现了网页与用户的交互，以此设计了登录，菜单，表格等样式，为用户操作提供了便利。

业务层使用 Struts2+Hibernate 设计，Hibernate 实现持久化操作，提供数据的操作，Struts2 拦截用户浏览器发送的请求，进行页面重定向，并通过操作 Hibernate 为用户提供数据。

数据层采用 MySQL 数据库，使用 MySQL Workbench 设计了包括教室信息，教师信息，教室预约信息等表，以此保存用户信息。

4 主要源代码

Struts2 配置及部分实现代码

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE struts PUBLIC "-//Apache Software Foundation//DTD Struts Configuration 2.1//EN" "http://struts.apache.org/dtds/struts-2.1.dtd">
<struts>
    <constant name="struts.enable.DynamicMethodInvocation" value="true" />
    <constant name="struts.devMode" value="true" />
    <package name="default" namespace="/" extends="struts-default">
        <!-- Login -->
        <action name="login" class="com.ning.server.web.Login" method="login">
            <result name="success">index.jsp</result>
            <result name="error">login.jsp</result>
        </action>
        <action name="exit" class="com.ning.server.web.Login" method="exit">
            <result name="success">login.jsp</result>
        </action>
        <!--  Time -->
        <action name="freetime" class="com.ning.server.web.Time" method="freetime">
```

```
            <result name="success">freetime.jsp</result>
        </action>
        <!-- ClassroomManage -->
        <action name="showCInfo" class="com.ning.server.web.ClassroomManage"
method="showInfo">
            <result name="success">classroom-info.jsp</result>
        </action>
        <!-- TeacherManage -->
        <action name="updateTeacher" class="com.ning.server.web.TeacherManage"
method="update">
            <result name="success">user-info.jsp</result>
        </action>
    </package>
</struts>
```

```
package com.ning.server.web;

import com.google.gson.Gson;
import com.ning.DAO.TeacherEntity;
import com.ning.DO.ClassroomOperation;
import com.ning.DO.TeacherOperation;
import com.opensymphony.xwork2.ActionSupport;
import org.apache.struts2.ServletActionContext;
import org.hibernate.Session;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;

/**
 * Created by ning on 2017/7/4.
 */
public class Login extends ActionSupport {

    ClassroomOperation co = new ClassroomOperation();
    TeacherOperation to = new TeacherOperation();
    private String name;
    private String password;
    public String getName() {
        return name;
```

```
}
public void setName(String name) {
    this.name = name;
}
public String getPassword() {
    return password;
}
public void setPassword(String password) {
    this.password = password;
}
@Override
public String execute() throws Exception {
    return "Hello world";
}
public void test1() {
    try {
        String json = ready();
        System.out.println(json);

        String result = co.selectById(1).toJSON();
        response.getWriter().append(result);
        System.out.println(result);
        System.out.println("t");
        System.out.println("exit0");
    } catch (IOException e) {
        e.printStackTrace();
    }
    return ;
}
Gson gson = new Gson();
private TeacherEntity teacher;
private HttpServletRequest request;
private HttpServletResponse response;
private InputStream is;
private BufferedReader bufferedReader;
private HttpSession session;
public String ready() throws IOException {
    response = ServletActionContext.getResponse();
    request = ServletActionContext.getRequest();
    session = request.getSession();
    response.setContentType("textml;charset=utf-8");
    response.setCharacterEncoding("UTF-8");
    request.setCharacterEncoding("UTF-8");
    is = request.getInputStream();
```

```
            bufferedReader = new BufferedReader(new InputStreamReader(is, "utf-8"));
            StringBuffer sb = new StringBuffer();
            String line = null;
            while ((line = bufferedReader.readLine()) != null) {
                sb.append(line);
            }
            System.out.println(sb.toString());
            return sb.toString();
        }
        public String login() throws IOException {
            ready();
            teacher = to.verify(name, password);
            if (teacher != null) {
                System.out.println(teacher.toJSON());
                if ("SUCCESS".equals(teacher.getPassword())) {
                    //保存 teacher 的 id 和 name
                    session.setAttribute("teacher_id", teacher.getId());
                    session.setAttribute("name", teacher.getName());
                    System.out.println(session.getAttribute("name"));
                    return SUCCESS;
                } else if("FAILED".equals(teacher.getPassword())) {
                    session.setAttribute("error", "用户名或密码错误！");
                }
            } else {
                session.setAttribute("error", "用户不存在！");
            }

            return ERROR;
        }
        public String exit() throws IOException {
            ready();
            System.out.println("exit:name" + session.getAttribute("name"));
            System.out.println("exit:teacher_id" +
    session.getAttribute("teacher_id"));
            session.removeAttribute("name");
            session.removeAttribute("teacher_id");
            return SUCCESS;
        }
        public String selectById() throws IOException {
            ready();
            return ERROR;
        }
    }
```

Hibernate Bean 及操作类

```
<?xml version='1.0' encoding='utf-8'?>
<!DOCTYPE hibernate-configuration PUBLIC
        "-//Hibernate/Hibernate Configuration DTD//EN"
        "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
    <session-factory>

        <property
name="connection.url">jdbc:mysql://123.207.244.139:3306/db_classroom</property>
        <property
name="connection.driver_class">com.mysql.jdbc.Driver</property>

        <property name="connection.username">root</property>
        <property name="connection.password">*********</property>
        <property name="show_sql">false</property>
        <property name="format_sql">true</property>
        <mapping class="com.ning.DAO.ClassroomEntity"/>
        <mapping class="com.ning.DAO.HireEntity"/>
        <mapping class="com.ning.DAO.TeacherEntity"/>
        <mapping class="com.ning.DAO.TimeEntity"/>
        <mapping class="com.ning.DAO.DayEntity"/>
    </session-factory>
</hibernate-configuration>
```

```
package com.ning.DAO;

import javax.persistence.*;
import java.util.HashSet;
import java.util.Set;

/**
 * Created by ning on 2017/7/11.
 */
@Entity
@Table(name = "classroom", schema = "db_classroom", catalog = "")
public class ClassroomEntity implements IEntity {
    private int id;
    private String number;
    private int capacity;
    private String extra;

    private transient Set<TeacherEntity> teachers = new HashSet<>();
```

```
    @Id
    @Column(name = "id")
    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    @Basic
    @Column(name = "number")
    public String getNumber() {
        return number;
    }
    public void setNumber(String number) {
        this.number = number;
    }
    @Basic
    @Column(name = "capacity")
    public int getCapacity() {
        return capacity;
    }
    public void setCapacity(int capacity) {
        this.capacity = capacity;
    }
    @Basic
    @Column(name = "extra")
    public String getExtra() {
        return extra;
    }
    public void setExtra(String extra) {
        this.extra = extra;
    }
    @ManyToMany(mappedBy="classrooms",cascade=CascadeType.ALL, fetch =
FetchType.EAGER)
    public Set<TeacherEntity> getTeachers() {
        return teachers;
    }
    public void setTeachers(Set<TeacherEntity> teachers) {
        this.teachers = teachers;
    }
    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
```

```
        if (o == null || getClass() != o.getClass()) return false;
        ClassroomEntity that = (ClassroomEntity) o;
        if (id != that.id) return false;
        if (capacity != that.capacity) return false;
        if (number != null ? !number.equals(that.number) : that.number != null)
return false;
        if (extra != null ? !extra.equals(that.extra) : that.extra != null) return
false;
        return true;
    }
    @Override
    public int hashCode() {
        int result = id;
        result = 31 * result + (number != null ? number.hashCode() : 0);
        result = 31 * result + capacity;
        result = 31 * result + (extra != null ? extra.hashCode() : 0);
        return result;
    }
}

package com.ning.DO;

import com.ning.DAO.ClassroomEntity;
import com.ning.DAO.DayEntity;
import com.ning.DAO.TeacherEntity;
import org.hibernate.query.Query;

import java.util.ArrayList;
import java.util.List;

/**
 * Created by ning on 2017/7/4.
 */
public class ClassroomOperation implements IOperation<ClassroomEntity> {
    Operation<ClassroomEntity> o = new Operation<>();

    @Override
    public boolean delete(int id) {
        return o.delete(id, (session, integer) -> {
            ClassroomEntity c = new ClassroomEntity();
            c.setId(id);
            session.delete(c);
            return true;
        });
```

```
    }

    @Override
    public ClassroomEntity selectById(int id) {
        return o.selectById(id, (session, integer) -> {
            String hql = "from ClassroomEntity where id = :id";
            Query<ClassroomEntity> query = session.createQuery(hql);
            query.setParameter("id", id);
            ClassroomEntity c = query.uniqueResult();
            return c;
        });
    }

    public List<ClassroomEntity> selectAll() {
        return o.selectAll((session, classroomEntity) -> {
            String hql = "from ClassroomEntity";
            Query<ClassroomEntity> query = session.createQuery(hql,
ClassroomEntity.class);
            return query.list();
        });
    }

    public List<Integer> selectHireDayById(int id) {
        ClassroomEntity c = selectById(id);
        ArrayList<Integer> teacherIds = new ArrayList<>();
        ArrayList<Integer> dayIds = new ArrayList<>();
        if (c == null || c.getTeachers().size() == 0) {
            return null;
        }
        for (TeacherEntity t : c.getTeachers()) {
            teacherIds.add(t.getId());
        }
        HireOperation ho = new HireOperation();
        for (int teacher_id : teacherIds) {
            dayIds.add(ho.selectByForeignKey(id, teacher_id));
        }
        return dayIds;
    }
}

package com.ning.DO;

import org.hibernate.Session;
```

```java
/**
 * Created by ning on 2017/7/4.
 */
public interface DoInDO<T, V> {
    /**
     *
     * @param session
     * @param t
     * @return
     * @throws Exception catch by IIOperation
     */
    public V dosomething(Session session, T t) throws Exception;
}
```

```java
package com.ning.DO;

import java.util.List;
/**
 * Created by ning on 2017/7/4.
 */
public interface IIOperation<T> {
    /**
     *
     * @return true/false
     */
    public boolean insert(T t, DoInDO<T, Boolean> doInDO);
    public boolean delete(int id, DoInDO<Integer, Boolean> doInDO);
    public boolean update(T t, DoInDO<T, Boolean> doInDO);
    public T selectById(int id, DoInDO<Integer, T> doInDO);
    public List<T> selectAll(DoInDO<T, List<T>> doInDO);
}
```

```java
package com.ning.factory;

import org.hibernate.HibernateException;
import org.hibernate.Metamodel;
import org.hibernate.SQLQuery;
import org.hibernate.query.Query;
import org.hibernate.Session;
import org.hibernate.cfg.Configuration;

import javax.persistence.metamodel.EntityType;

/**
```

```
 * Created by ning on 2017/7/3.
 */
public class MySessionFactory {
    private static final org.hibernate.SessionFactory ourSessionFactory;
    static {
        try {
            Configuration configuration = new Configuration();
            configuration.configure();

            ourSessionFactory = configuration.buildSessionFactory();
        } catch (Throwable ex) {
            throw new ExceptionInInitializerError(ex);
        }
    }
    public static Session getSession() throws HibernateException {
        return ourSessionFactory.openSession();
    }
    public static void main(final String[] args) throws Exception {
        final Session session = getSession();
        try {
            System.out.println("querying all the managed entities...");
            final Metamodel metamodel =
session.getSessionFactory().getMetamodel();
            for (EntityType<?> entityType : metamodel.getEntities()) {
                final String entityName = entityType.getName();
                final Query query = session.createQuery("from " + entityName);
                System.out.println("executing: " + query.getQueryString());
                for (Object o : query.list()) {
                    System.out.println("  " + o);
                }
            }
        } finally {
            session.close();
        }
    }
}
```

JSP 界面代码

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%><%@page import="com.ning.server.web.Login"%>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
```

```
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>沈阳理工大学教室管理系统——请登录</title>
    <!-- BOOTSTRAP STYLES-->
    <link href="assets/css/bootstrap.css" rel="stylesheet" />
    <!-- FONTAWESOME STYLES-->
    <link href="assets/css/font-awesome.css" rel="stylesheet" />
    <!-- GOOGLE FONTS-->
    <link href='http://fonts.googleapis.com/css?family=Open+Sans'
rel='stylesheet' type='text/css' />
    <% if(session.getAttribute("name") != null) {
        System.out.println("login:name:" + session.getAttribute("name"));
    %>
    <script  type="text/javascript">
        window.self.location = "index.jsp"
    </script >
    <% } %>
</head>
<body style="background-color: #E2E2E2;">

<%
    System.out.println("error" + session.getAttribute("error"));
%>

<div class="container">
    <div class="row text-center " style="padding-top:100px;">
        <div class="col-md-12">
            <%--<img src="assets/img/logo-book.png" />--%>
        </div>
    </div>
    <div class="row ">
        <div class="col-md-4 col-md-offset-4 col-sm-6 col-sm-offset-3 col-xs-10
col-xs-offset-1">
            <div class="panel-body">
                <form role="form" action="login">
                <br>
                    <h5>
                        <%
                            if (session.getAttribute("error") != null) {
                                out.print(session.getAttribute("error"));
                            }
                        %>
                    </h5>
```

```
                    <form action="login">
                        <div class="form-group input-group">
                            <span class="input-group-addon"><i class="fa
fa-tag"></i></span>
                            <input id="name" type="text" class="form-control"
placeholder="用户名 " name="name" value="ning"/>
                        </div>
                        <div class="form-group input-group">
                            <span class="input-group-addon"><i class="fa
fa-lock"  ></i></span>
                            <input id="password" type="password"
class="form-control"  placeholder="密码"  name="password" value="123"/>
                        </div>
                        <div class="form-group">
                            <input type="submit" class="btn btn-primary " value="
登录" />
                        </div>
                    </form>
                    <br>
                    上帝?   <a href="index.jsp" >点击此处</a>转到<a
href="index.jsp">首页</a>
                </form>
            </div>

        </div>
    </div>
</div>

</body>
</html>
```

5 结论

5.1 运行结果

图 5.1.1 登录界面

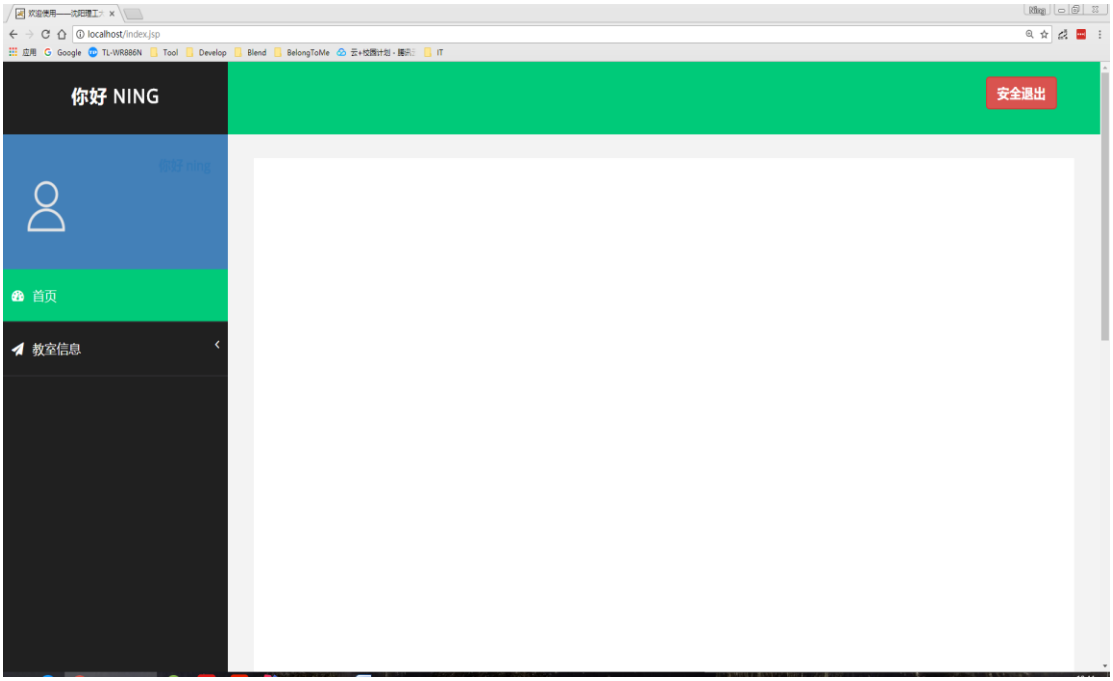

图 5.1.2 首页

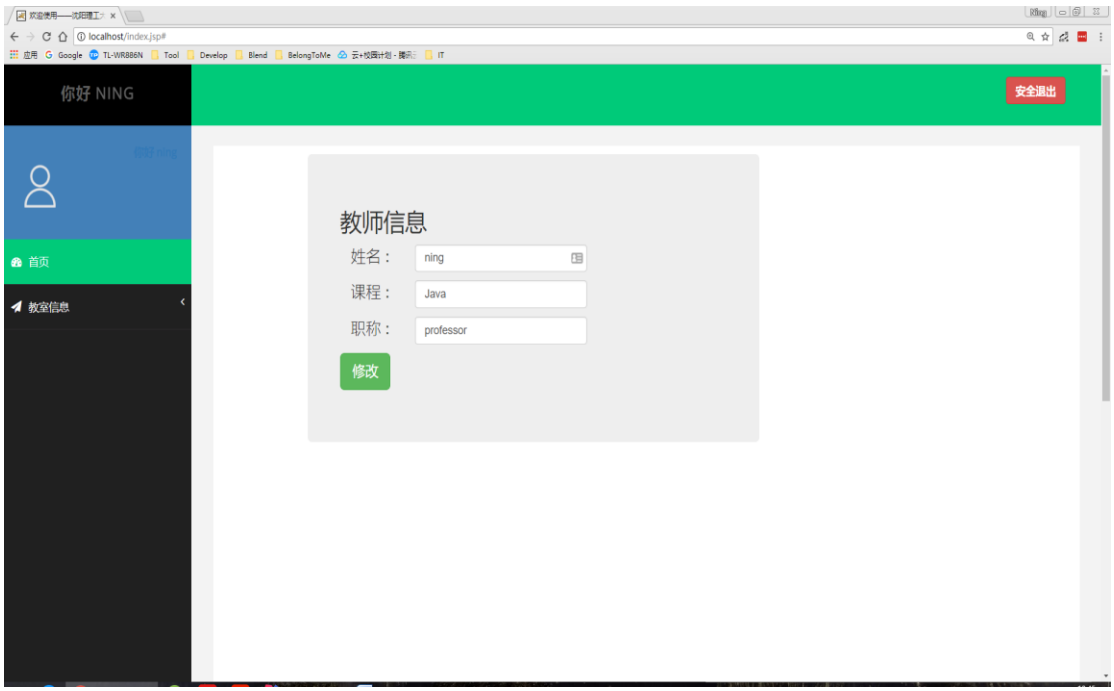

图 5.1.3 教师信息查看及修改界面

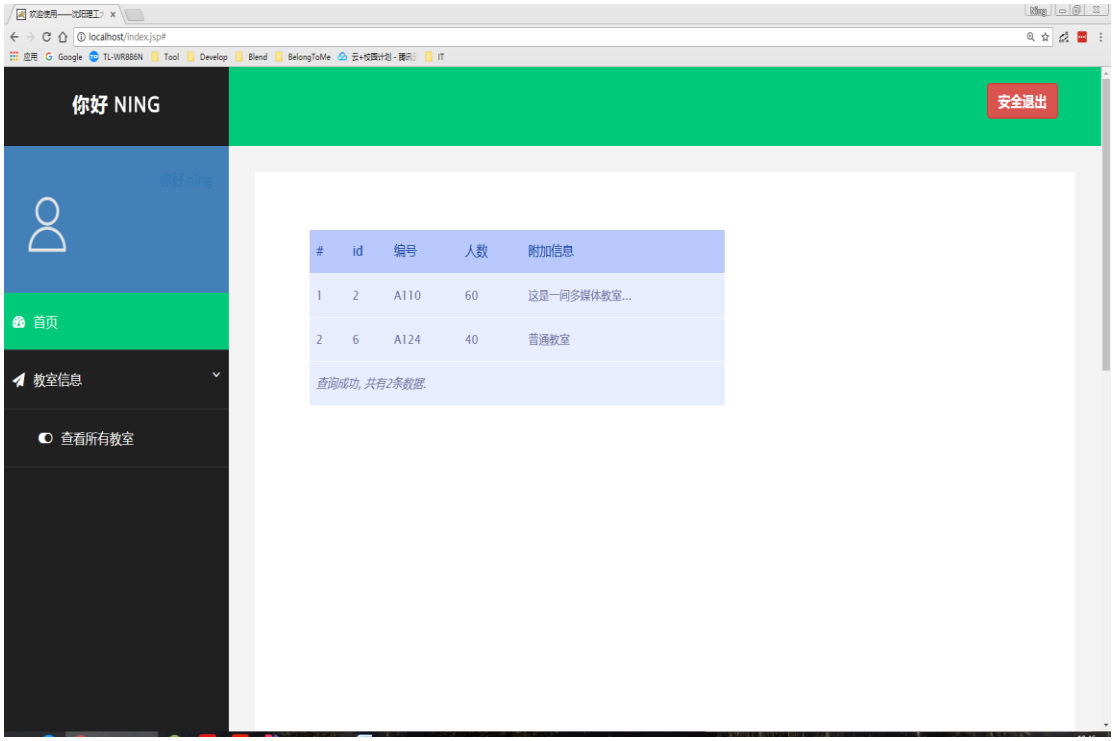

图 5.1.4 教室信息界面

图 5.1.5 教室详细信息界面

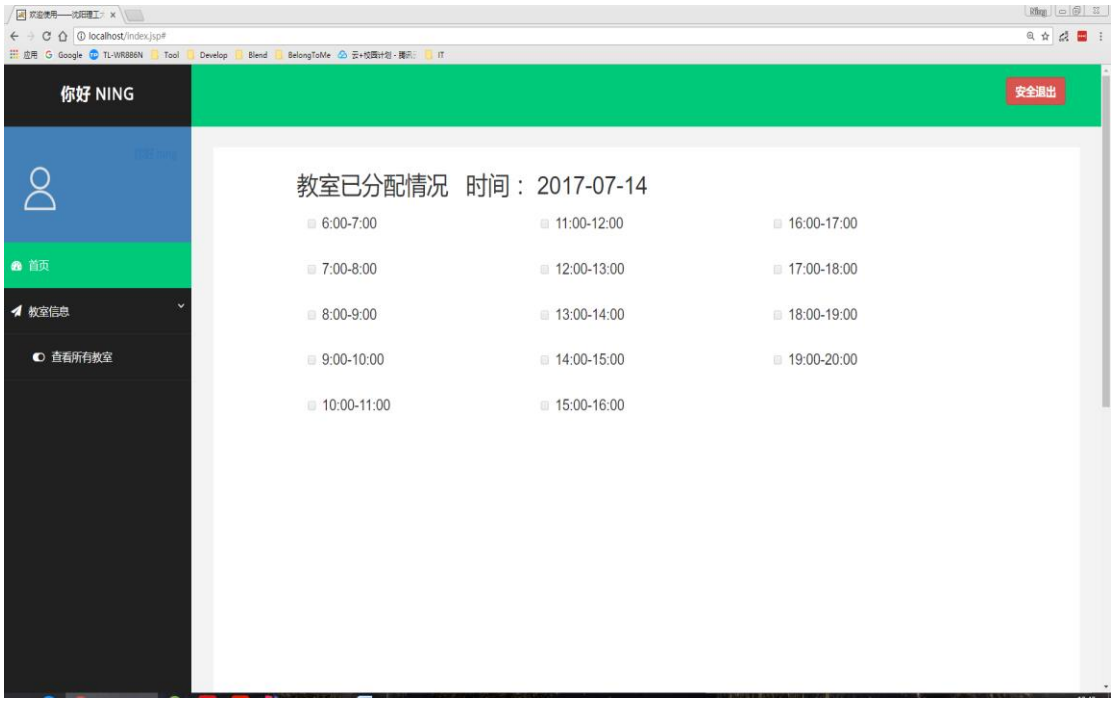

图 5.1.6 教室分配信息

5.2 结论

本次课程设计，让我获益匪浅，以前在学 java 和数据库时，虽然上课也好好听老师讲，但是在课后自己没有亲自动手实践过，现在有许多知识都忘记了。因此，在这次课程设计中，有许多 java 的知识我都忘记了，许多 java 函数的功能我都不会了。所以，在课程设计中我经常遇到许多问题，这次课程设计中我遇到问题，皆可以问老师或者其他会的同学。当然，许多时候我们会不停地翻 sql 与 java EE。通过这次数据库开发的课程设计，我更加深入的了解数据库这门学科，更使自己有了继续探索的兴趣。于个人而言，在程序设计的过程中，我深感“认真严谨”这个词的重要性，一点点小的马虎，便会导致整个程序不能正常运行。在今后的学习中，我定将“认真严谨”时刻作为自的谨言。与此同时，我们小组成员的互帮互助，让我体会到了团结的力量，而更让人难以忘怀的是在热烈讨论问题时，那激情横溢的场面。总之，此次课程设计在我的学生生涯中启上了至关重要的作用。

6 参考文献.

[1] 霍斯特曼，科内尔. Java 核心技术 卷一 基础知识[M]. 北京: 机械工业出版社，2013.11

[2] Ben Forta. MySQL 必知必会[M]. 北京: 人民邮电出版社，2009.1.

[3] 唐汉明. 深入浅出 MySQL[M]. 北京: 人民邮电出版社，2014.1