东北大学秦皇岛分校

Java 程序设计实践设计报告

基于 JavaWeb 的社区街道管理系统

学　　院	数学与统计学院
专　　业	数据科学与大数据技术
班级序号	200221
学　　号	202015140
姓　　名	周华
指导教师	王子健、陆杰
开始日期	2022 年 12 月 28 日
结束日期	2023 年 1 月 6 日

教师评阅意见书

一、态度 1．工作态度（□认真、□较好、□一般、□较差） 2．出勤情况（□无缺勤、□缺勤不超过1/3、□缺勤超过1/3） 3．上交时间（□按时、□迟交1天、□迟交1天以上）
二、格式规范 4．文字部分（□符合规范、□较符合规范、□一般、□不符合规范） 5．图表部分（□符合规范、□较符合规范、□一般、□不符合规范） 6．数学公式（□符合规范、□较符合规范、□一般、□不符合规范） 7．参考文献（□符合规范、□较符合规范、□一般、□不符合规范）
三、报告内容 8．任务量和可行性（□合理、□较为合理、□一般、□不合理） 9．报告结构（□合理、□较合理、□一般、□不合理） 10．文字叙述（□清晰流畅、□较为清晰、□一般、□不清晰） 11．图表准确性（□准确、□较准确、□一般、□不准确）
四、综合能力 12．综合运用知识能力（□很强、□较强、□一般、□较弱） 13．实践与动手能力（□很强、□较强、□一般、□较弱） 14．创新意识（□很强、□较强、□一般、□较弱）
综合评价：□优秀、□良好、□中等、□及格、□不及格

评阅教师签字： 日期：

I. 引言

1.1 问题背景和意义

街道社区是通往人民的“第一线”，是民生所向，作为城市基层社会管理的，街道社区的政务水平与人民的幸福感，安全感息息相关，直接影响了政府的公信力。随着中国经济社会的发展和工业化的浪潮，越来越多的街道社区被规划、建设，随着带来了社区街道的管理问题。

因此，社区街道管理是社会治理中及其重要的部分，一个好的社区街道管理系统能够帮助管理者快速了解社会民生，查询社会信息，帮助管理者和决策者提高决策速度与管理能力。

1.2 社区街道管理系统

本文通过 Java 语言及相关知识，实现了社区街道管理系统界面的设计与实现，社区街道信息管理系统包含登录注册功能，新用户可以选择注册功能注册新的账号，新注册的账号会被保存到 MySQL 中，登录成功之后即可对街道社区信息进行管理，系统中包含 5 张表，分别记录了**社区，街道，楼层，户主，家庭成员**的信息，所有的表均保存到 **MySQL** 中。

系统的界面包含**左中右三栏**，左栏可以选择要查看的表，中栏将显示左栏选择表的信息以及可进行的操作，右栏实现了左栏选择的操作功能。

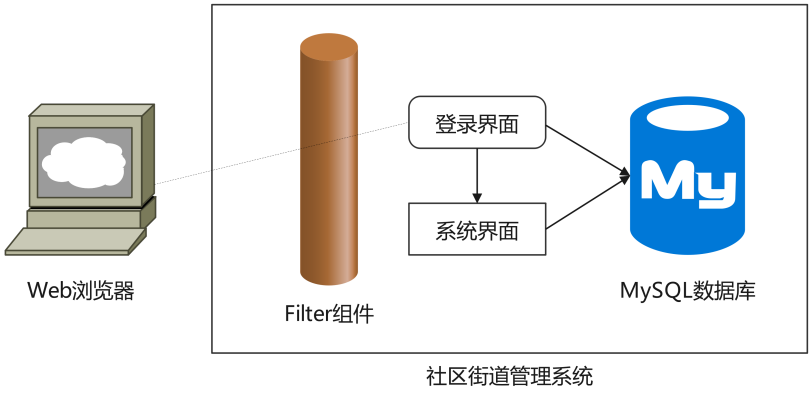

图 1.1 系统整体框架

系统的整体框架如图 1.1,Web 浏览器通过互联网访问系统，经过 Filter 组件时，如

果用户没有登录，必须先进行登录或者注册，否则不能进入系统。登录的用户信息和社区街道的信息均存储在 MySQL 数据库中。

II. 任务实现过程

2.1 前端设计

本文前端设计采用 Html 与 JSP 知识进行。登录界面，使用 Jsp 页面进行编写，包含登录与注册功能；系统界面分为左中右三栏，分别展示**社区街道管理导航栏，数据展示和可操作的选择，操作的数据**。

2.1.1 登录界面

登录界面包含登录界面，和注册界面。登录界面，使用 JSP 编写可供用户编写的数据表单，用户填写表单后，前端会将数据发送给后端，后端获取表单信息后，在 MySQL 数据库中查询数据是否存在，如果存在，登录成功。登录界面的前端页面如图 2.1.

图 2.1 登录界面

注册页面，使用 JSP 编写可供用户编写的数据表单，用户填写表单后，前端会将数据发送给后端，后端获取表单信息后，在 MySQL 数据库中查询数据是否存在，如果存在，注册失败，如果不存在，将数据写入 MySQL 数据库中。

2.1.2 系统界面

系统界面分为左中右三栏，分别展示社区街道管理导航栏，数据展示和可操作的选择，操作的数据。设计出的系统界面如图 2.2。

在系统界面中，使用 **Frameset** 设置系统的界面为左中右三栏，占据页面的大小分别为 20%，40%，40%, 分别显示 **left.jsp，CommunityPresent，community_add** 页面。

left.jsp 页面是社区街道的五张表的管理导航，放置了五个管理系统的超链接，超链接在获取页面时会跳转到对应系统的管理界面，并且在跳转时会将页面显示在当前页面的中栏，从而实现刷新中栏的作用。

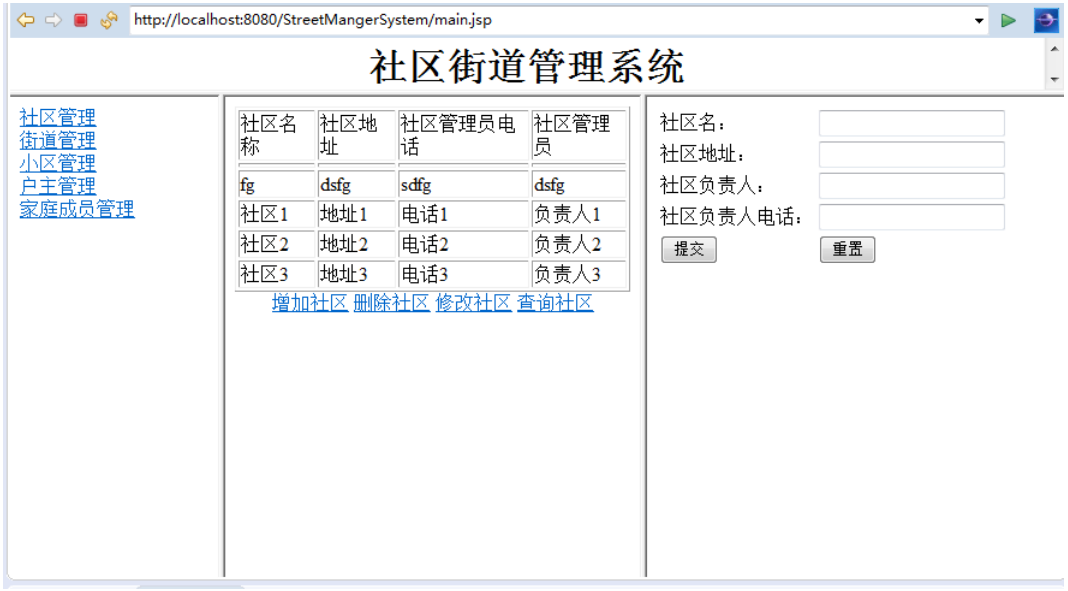

图 2.2 系统界面

中栏的页面是社区街道的五张表的管理界面，会显示左栏选择的系统管理界面，中栏界面会向后端发送请求，向数据库获取表信息，将数据库中信息展示在中栏界面，并且提供数据表的操作功能超链接，超链接在获取页面时会跳转到对应功能的设置界面，并且在跳转时会将页面显示在当前页面的右栏，从而实现刷新右栏的作用。中栏的超链接代码为：

```
<center>
<a href="right/community/community_add.jsp" target="right">增加社区</a>
<a href="right/community/community_delete.jsp" target="right">删除社区</a>
<a href="right/community/community_update.jsp" target="right">修改社区</a>
<a href="right/community/community_query.jsp" target="right">查询社区</a>
</center>
```

右栏的界面是中栏选择的功能的设置页面，使用 JSP 编写可供用户编写的数据表单，用户填写表单后，前端会将数据发送给后端，后端根据相应的功能连接数据库进行相应的操作，从而实现数据库的增删该查操作。

subsection 数据库设计 社区街道管理系统使用 MySQL 数据存储数据，安装 MySQl 设置数据库的访问端口为 **3306**，开启服务，建立一个数据库为 **streetmanagersystem**，用于存储社区街道管理系统的数据。**Java** 使用 **JDBC** 的方式即可访问数据库，并对数据库进行操作。

采用 JDBC 的模式，使用 Java 在数据库创建六张表，部分代码如下：

```
Class.forName("com.mysql.jdbc.Driver");
Connection conn = null;

conn =DriverManager.getConnection("jdbc:mysql://localhost:3306/
streetmanagersystem?characterEncoding=utf-8","root","");

Statement stmt = conn.createStatement();

String sql5 = "create table street(
streetName varchar(20) not null,
streetManager varchar(20) not null,
streetManagerPhone varchar(20) not null,
primary key(streetName))";

String sql6 = "insert into street values('街道1','负责人1','电话1')";

stmt.execute(sql5);stmt.execute(sql6);

conn.close();
```

创建的 6 张表分别为 **user，community，street，house，household，houshouldmember**，分别存储**用户信息，社区信息，街道信息，小区信息，业主信息，家庭成员信息**。创建表之后，向表中插入了一些基本的信息。

2.2 MVC 框架

2.2.1 框架介绍

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。典型的 MVC 框架如图 2.3。

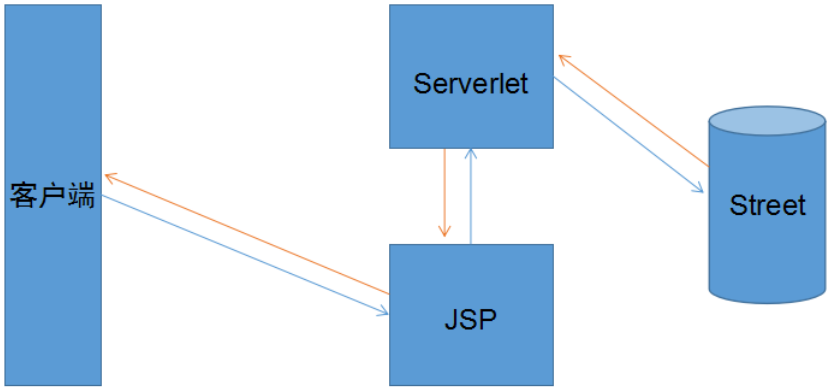

图 2.3 MVC 模式

在 **MVC** 模式中，客户端向 **JSP** 页面请求，**JSP** 页面为客户端显示可视化页面，并且可以将客户端提供的表单交给 **Serverlet** 处理，**Serverlet** 根据表单内容对数据库进行相关的操作，并将结果返回给 **JSP**，**JSP** 将结果可视化返回给客户端，在此模型中，JSP 是 **View**，**Serverlet** 是 **Controller**，数据库中的表是 **Model**。

2.2.2 *Model*

为了 Serverlet 能够方便的对数据库进行操作以及进行数据的传递，需要为数据库中的 6 表建立对应的实体类，实体类是 Java 里的一种数据结构，包含了对象的属性，并且每个属性都包含标准的 **get** 方法和 **set** 方法，本文建立的实体类如图 2.4。

图 2.4 实体类

分别代表了**用户，社区，街道，楼层，户主，家庭成员**的信息，所有的表在 MySQl

数据库中均有一一映射的表。user 实体类的代码如下：

```
public class Users{
private String username;
private String password;

public String getUsername() {return username;}
public void setUsername(String account) {this.username = account;}

public String getPassword() {return password;}
public void setPassword(String password) {this.password = password;}
}
```

2.2.3 *View*

JSP 向客户端提供提交表单功能，以及向客户端展示数据内容，提交表单的一个代码如下：

```
</html>
<body>
<form action="../../CommunityDelete" method="post">
<table>
<tr>
<td>社区名：</td>
<td><input type="text" name="community_name"></td>
</tr>
<tr>
<td><input type="submit" value="提交"></td>
<td><input type="reset" value="重置"></td>
</tr>

</table>
</form>
</body>
</html>
```

当用户在客户端提交表单后，**JSP** 页面会将表单数据转发给对应的 **Serverlet**，**Serverlet** 进行相应处理后，如果有必要，会将数据进一步转发给 **JSP** 页面，**JSP** 页面将数据以表格的形式展示给用户，展示的一个示例代码如下：

```
<html>
<body>
<% List<Street> list = (List<Street>)request.getAttribute("comlist"); %>
<table border="1">
<tr>
<td>街道名称</td>
<td>街道管理员</td>
<td>街道管理员电话</td>
</tr>

<% for(Street com:list){ %>

<tr>
<td><%=com.getStreetname()%></td>
<td><%=com.getStreetmanager()%></td>
<td><%=com.getStreetphone()%></td>
</tr>

<% } %>
</table>
</body>
</html>
```

展示的页面是 **Street** 数据的展示页面，**JSP** 页面向 **Serverlet** 请求数据，**Serverlet** 处理后将数据返回给 **JSP**，**JSP** 将街道的数据展示到客户端。

2.2.4 *Controller*

Serverlet 的功能主要是处理 JSP 提交的表单以及对数据进行操作，将结果返回给 JSP 页面。Serverlet 处理 JSP 删除社区信息请求，并对数据库进行操作的一个示例代码如下：

```
protected void doGet(HttpServletRequest request, HttpServletResponse response){
String communityname = request.getParameter("community_name");
Connection conn = null;

Class.forName("com.mysql.jdbc.Driver");
conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/
streetmanagersystem?useUnicode=true&characterEncoding=utf-8","root","");

java.sql.PreparedStatement pstmt = conn.prepareStatement(
"delete from community where communityName = ?");

pstmt.setString(1, communityname);
pstmt.execute();

conn.close();
request.getRequestDispatcher("right/community/community_delete.jsp")
.forward(request, response);
}
```

在此代码中，**Serverlet** 首先会获取 **JSP** 提供的 **community_name** 信息，然后连接数据库，将数据从数据库中删除，最后返回提交表单的 **JSP** 页面。**Serverlet** 有时需要将数据转发给 **JSP** 页面以便展示，示例代码如下：

```
protected void doGet(HttpServletRequest request, HttpServletResponse response){
Connection conn = null;
List <Street> comlist = new ArrayList<Street>();

Class.forName("com.mysql.jdbc.Driver");

conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/
streetmanagersystem?useUnicode=true&characterEncoding=utf-8","root","");

java.sql.PreparedStatement pstmt = conn.prepareStatement("select * from street");
java.sql.ResultSet rs = pstmt.executeQuery();
```

```
while(rs.next()){
Street street = new Street();
street.setStreetname(rs.getString("streetName"));
street.setStreetmanager(rs.getString("streetManager"));
street.setStreetphone(rs.getString("streetManagerPhone"));
comlist.add(street);}
conn.close();

request.setAttribute("comlist", comlist);
request.getRequestDispatcher("center/street.jsp")
.forward(request, response);
}
```

此代码是展示街道信息的 **Sererlet**，首先会创建 **Street** 实体类的 **comlist** 以便操作，连接数据库查询所有的数据，并将数据保存到 **comlist** 中，最后将 **comlist** 转发给 **JSP**，**JSP** 页面将数据以表格形式展示给用户。

III. 系统运行结果

系统运行的界面如图 2.2, 其中，一个查询街道的运行结果如 3.1。

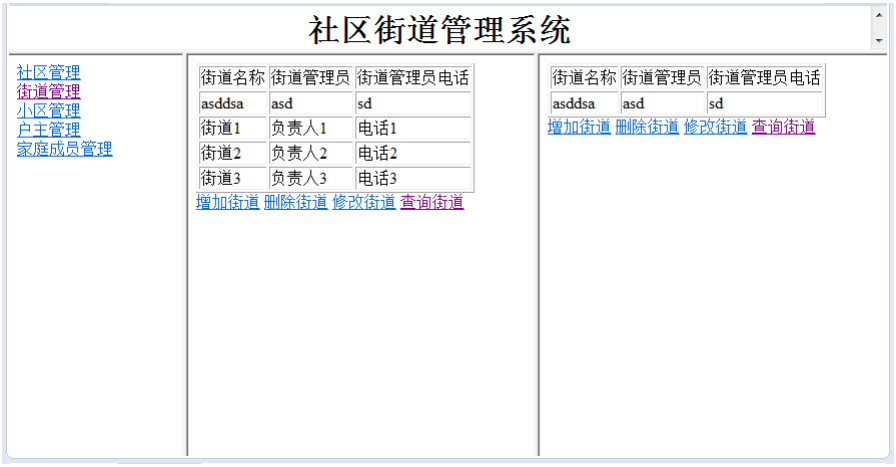

图 3.1 查询街道名为 **asddsa** 的街道信息

在使用系统之前，需要先进行注册才能使用，在使用过程中，在对数据进行增删改查之后，需要点击左侧的导航栏，才能显示最新的数据。

IV. 结论与改进

本系统前端界面简介大方，在一个界面进行系统的所有操作，让人一目了然；后端分工明确，Serverlet 负责处理数据的请求，对数据库的增删改查操作，JSP 负责表单的处理，数据的可视化等与图形化界面相关的操作，数据库用于存储数据。

通过本课程设计，学习到了网站前端和后端开发的简单流程，对网站设计的流程有了认识，为日后学习打下了更坚实的基础，在开发过程中，由于时间紧迫和笔者知识有限，开发出的系统较简单，功能较简单；在进行增删改查之后，需要手动点击导航栏才能查看最新的数据；并且在写类的过程中没有用到 java 反射机制，导致很多代码冗余，编写的 java 文件也变多。

参考文献

[1] 戴劲松, 王海江. 基于 GIS 的街道信息管理系统设计与实现 [J]. 城市勘测,2018(04):29-32.

[2] https://www.jianshu.com/p/ff6de219f988.MVC、MVP、MVVM 模式的概念与区别

[3] 殷人昆, 数据结构：用面向对象方法与 C++ 语言描述, 北京：清华大学出版社，2007.6

V. 附录

代码清单 1 Street 实体类

```
//生成一个实体类street，字段有街道名，街道负责人，街道负责人电话，主键街道名

package datastruct;

public class Street {
private String streetname;
private String streetmanager;
private String streetphone;

//get方法
public String getStreetname() {
return streetname;
}

//set方法
public void setStreetname(String streetname) {
this.streetname = streetname;
}

//get方法
public String getStreetmanager() {
return streetmanager;
}

//set方法
public void setStreetmanager(String streetmanager) {
this.streetmanager = streetmanager;
}

//get方法
```

```
public String getStreetphone() {
return streetphone;
}

//set方法
public void setStreetphone(String streetphone) {
this.streetphone = streetphone;
}
}
```

代码清单 2 社区插入 CommunityInsert 类

```
package community;

import java.io.IOException;
import java.sql.Connection;
import java.sql.DriverManager;
import java.util.ArrayList;
import java.util.List;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import datastruct.Community;

/**
* Servlet implementation class CommunityInsert
*/
public class CommunityInsert extends HttpServlet {
private static final long serialVersionUID = 1L;

/**
```

```
* @see HttpServlet#HttpServlet()
*/
public CommunityInsert() {
super();
// TODO Auto-generated constructor stub
}

/**
* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
*/
protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
request.setCharacterEncoding("utf-8");
response.setContentType("text/html;charset=utf-8");

String communityname = request.getParameter("community_name");
String communityaddress = request.getParameter("community_address");
String communitymanager = request.getParameter("communitymanager");
String communityphone = request.getParameter("communityphone");

Connection conn = null;
//User类的list集合

//尝试导入驱动，建立数据库连接，建立连接的密码为空，建立user表
try {
Class.forName("com.mysql.jdbc.Driver");
conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/
streetmanagersystem?useUnicode=true&characterEncoding=utf-8","root","");

//定义prepat
java.sql.PreparedStatement pstmt = conn.prepareStatement("insert into community
    values(?,?,?,?)");
//执行查询
pstmt.setString(1, communityname);
```

```
pstmt.setString(2, communityaddress);
pstmt.setString(3, communitymanager);
pstmt.setString(4, communityphone);
pstmt.execute();
//关闭连接
pstmt.close();
conn.close();
//将list集合存入request中

request.getRequestDispatcher("right/community/community_add.jsp").forward(request,
    response);

} catch (Exception e) {
// TODO Auto-generated catch block
e.printStackTrace();
}
}

/**
* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse
    response)
*/
protected void doPost(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
// TODO Auto-generated method stub
doGet(request, response);
}
}
```

代码清单 3 Filter 类

```
package filter;

import java.io.IOException;
```

```
import javax.servlet.FilterChain;
import javax.servlet.FilterConfig;
import javax.servlet.ServletException;
import javax.servlet.ServletRequest;
import javax.servlet.ServletResponse;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
* Servlet Filter implementation class Filter
*/
public class Filter implements javax.servlet.Filter {

/**
* Default constructor.
*/
public Filter() {
// TODO Auto-generated constructor stub
}

/**
* @see Filter#destroy()
*/
public void destroy() {
// TODO Auto-generated method stub
}

/**
* @see Filter#doFilter(ServletRequest, ServletResponse, FilterChain)
*/
public void doFilter(ServletRequest request, ServletResponse response,
    FilterChain chain) throws IOException, ServletException {

HttpServletRequest req = (HttpServletRequest) request;
```

```
req.setCharacterEncoding("utf-8");
req.setCharacterEncoding("utf-8");

String url = req.getRequestURI();

if(url.equals("/StreetMangerSystem/Login") ||
    url.equals("/StreetMangerSystem/login.jsp") ||
    url.equals("/StreetMangerSystem/register.jsp") ||
    url.equals("/StreetMangerSystem/Register") ){
chain.doFilter(request, response);
}

else{
Object user = req.getSession().getAttribute("user");

if(user != null){
chain.doFilter(request, response);
}

else{
HttpServletResponse resp = (HttpServletResponse) response;
resp.sendRedirect("login.jsp");
}
}
}

/**
* @see Filter#init(FilterConfig)
*/
public void init(FilterConfig fConfig) throws ServletException {
// TODO Auto-generated method stub
}
}
```

代码清单 4 登录类 Login.java

```
package login;

import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import datastruct.Users;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;

/**
* Servlet implementation class Login
*/
public class Login extends HttpServlet {
private static final long serialVersionUID = 1L;

/**
* @see HttpServlet#HttpServlet()
*/
public Login() {
super();
// TODO Auto-generated constructor stub
}
```

```
/**
* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
*/
protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
// TODO Auto-generated method stub
request.setCharacterEncoding("utf-8");
response.setCharacterEncoding("utf-8");
response.setContentType("text/html;charset=utf-8");

String username = request.getParameter("username");
String password = request.getParameter("password");

//建立数据库连接
Connection conn = null;
//User类的list集合
List<Users> list = new ArrayList<Users>();

//尝试导入驱动，建立数据库连接，建立连接的密码为空，建立user表
try {
Class.forName("com.mysql.jdbc.Driver");
conn =
    DriverManager.getConnection("jdbc:mysql://localhost:3306/streetmanagersystem
?useUnicode=true&characterEncoding=utf-8","root","");
PreparedStatement stmt = conn.prepareStatement("select * from user where userName
    = ? and userPassword = ?");
stmt.setString(1, username);
stmt.setString(2, password);
ResultSet rs = stmt.executeQuery();
//如果查询到了，就将用户名和密码存入list集合中
while(rs.next()){
Users user = new Users();
user.setUsername(rs.getString("userName"));
user.setPassword(rs.getString("userPassword"));
```

```
list.add(user);
}

//如果list集合中有数据，就跳转到主页面，否则就跳转到登录页面
if(list.size() > 0){
request.getSession().setAttribute("user", username);
response.sendRedirect("main.jsp");
}else{
response.sendRedirect("login.jsp");
}

//关闭结果集
rs.close();

//关闭连接
stmt.close();
conn.close();
} catch (ClassNotFoundException e) {
// TODO Auto-generated catch block
e.printStackTrace();
} catch (SQLException e) {
// TODO Auto-generated catch block
e.printStackTrace();}
}

/**
* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse
    response)
*/
protected void doPost(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
// TODO Auto-generated method stub
doGet(request, response);
}
```

```
}
```

代码清单 5 MySQL 客户端类 Mysql.java

```
package mysql;


import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.Statement;

public class Mysql {
//主函数
public static void main(String[] args) {
try {
Class.forName("com.mysql.jdbc.Driver");
} catch (ClassNotFoundException e1) {
e1.printStackTrace();
}
Connection conn = null;
try{
conn =
    DriverManager.getConnection("jdbc:mysql://localhost:3306/streetmanagersystem?
useUnicode=true&characterEncoding=utf-8","root","");
Statement stmt = conn.createStatement();
//新建一个user表,表中有账号和密码,账号和密码都是字符串,长度为20,不可为空,账号为主键
String sql = "create table user(userName varchar(20) not null,userPassword
    varchar(20) not null,primary key(userName))";

//新建一个community表，字段有社区名，社区地址，社区负责人，社区负责人电话，主键社区名
String sql1 = "create table community(communityName varchar(20) not
    null,communityAddress varchar(20) not null,communityManager varchar(20) not
    null,communityManagerPhone varchar(20) not null,primary key(communityName))";
//向表中插入三条真实的数据,编码为utf-8
String sql2 = "insert into community values('社区1','地址1','负责人1','电话1')";
```

```
String sql3 = "insert into community values('社区2','地址2','负责人2','电话2')";
String sql4 = "insert into community values('社区3','地址3','负责人3','电话3')";

//新建一个street表，字段有街道名，街道负责人，街道负责人电话，主键街道名
String sql5 = "create table street(streetName varchar(20) not null,streetManager
    varchar(20) not null,streetManagerPhone varchar(20) not null,primary
    key(streetName))";
//向表中插入三条真实的数据
String sql6 = "insert into street values('街道1','负责人1','电话1')";
String sql7 = "insert into street values('街道2','负责人2','电话2')";
String sql8 = "insert into street values('街道3','负责人3','电话3')";

//新建一个house表，字段有房屋编号，房屋地址，房屋面积，房屋类型，房屋所属社区，房屋所属街道，
//主键为房屋编号
String sql9 = "create table house(houseNumber varchar(20) not null,houseAddress
    varchar(20) not null,houseArea varchar(20) not null,houseType varchar(20) not
    null,houseCommunity varchar(20) not null,houseStreet varchar(20) not
    null,primary key(houseNumber))";
//向表中插入三条真实的数据
String sql10 = "insert into house
    values('房屋1','地址1','面积1','类型1','社区1','街道1')";
String sql11 = "insert into house
    values('房屋2','地址2','面积2','类型2','社区2','街道2')";
String sql12 = "insert into house
    values('房屋3','地址3','面积3','类型3','社区3','街道3')";

//新建一个household表，字段有户主姓名，户主电话，户主身份证号，户主所属房屋编号，
//主键为户主身份证号
String sql13= "create table household(householdName varchar(20) not
    null,householdPhone varchar(20) not null,householdID varchar(20) not
    null,householdHouseNumber varchar(20) not null,primary key(householdID))";
//向表中插入三条真实的数据
String sql14 = "insert into household values('户主1','电话1','身份证1','房屋1')";
String sql15 = "insert into household values('户主2','电话2','身份证2','房屋2')";
```

```
String sql16 = "insert into household values('户主3','电话3','身份证3','房屋3')";

//新建一个householdMember表，字段有成员姓名，成员电话，成员身份证号，成员所属户主身份证号，
//主键为成员身份证号
String sql17= "create table householdMember(memberName varchar(20) not
    null,memberPhone varchar(20) not null,memberID varchar(20) not
    null,memberHouseholdID varchar(20) not null,primary key(memberID))";
//向表中插入三条真实的数据
String sql18 = "insert into householdMember
    values('成员1','电话1','身份证1','户主1')";
String sql19 = "insert into householdMember
    values('成员2','电话2','身份证2','户主2')";
String sql20 = "insert into householdMember
    values('成员3','电话3','身份证3','户主3')";

//执行所有sql语句
try {
stmt.execute(sql1);stmt.execute(sql5);stmt.execute(sql9);
stmt.execute(sql13);stmt.execute(sql17);}
catch(Exception e) {}
try {
stmt.execute(sql2);stmt.execute(sql3);stmt.execute(sql4);
stmt.execute(sql6);stmt.execute(sql7);stmt.execute(sql8);
stmt.execute(sql10);stmt.execute(sql11);stmt.execute(sql12);
stmt.execute(sql14);stmt.execute(sql15);stmt.execute(sql16);
stmt.execute(sql18);stmt.execute(sql19);stmt.execute(sql20);}
catch(Exception e) {}

stmt.close();
conn.close();
}
catch(Exception e){
e.printStackTrace();}
}}
```

代码清单 6 街道显示类 StreetPresent.java

```
package present;

import java.io.IOException;
import java.sql.Connection;
import java.sql.DriverManager;
import java.util.ArrayList;
import java.util.List;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import datastruct.Street;

/**
* Servlet implementation class StreetPresent
*/
public class StreetPresent extends HttpServlet {
private static final long serialVersionUID = 1L;

/**
* @see HttpServlet#HttpServlet()
*/
public StreetPresent() {
super();
// TODO Auto-generated constructor stub
}

/**
* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
*/
protected void doGet(HttpServletRequest request, HttpServletResponse response)
```

```
    throws ServletException, IOException {
Connection conn = null;
//User类的list集合
List <Street> comlist = new ArrayList<Street>();

//尝试导入驱动，建立数据库连接，建立连接的密码为空，建立user表
try {
Class.forName("com.mysql.jdbc.Driver");
conn =
    DriverManager.getConnection("jdbc:mysql://localhost:3306/streetmanagersystem?
useUnicode=true&characterEncoding=utf-8","root","");

//定义prepat
java.sql.PreparedStatement pstmt = conn.prepareStatement("select * from street");
//执行查询
java.sql.ResultSet rs = pstmt.executeQuery();
//如果查询到了，就将存入list集合中
while(rs.next()){
Street street = new Street();
street.setStreetname(rs.getString("streetName"));
street.setStreetmanager(rs.getString("streetManager"));
street.setStreetphone(rs.getString("streetManagerPhone"));
comlist.add(street);}
//关闭连接
rs.close();
pstmt.close();
conn.close();
//将list集合存入request中
request.setAttribute("comlist", comlist);
//转发到Street.jsp页面
request.getRequestDispatcher("center/street.jsp").forward(request, response);
} catch (Exception e) {
// TODO Auto-generated catch block
e.printStackTrace();
```

```
}
}

/**
* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse
    response)
*/
protected void doPost(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
// TODO Auto-generated method stub
doGet(request, response);
}
}
```

代码清单 7 注册类 Register.java

```
package register;

import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import datastruct.Users;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;
```

```
/**
* Servlet implementation class Register
*/
public class Register extends HttpServlet {
private static final long serialVersionUID = 1L;

/**
* @see HttpServlet#HttpServlet()
*/
public Register() {
super();
// TODO Auto-generated constructor stub
}

/**
* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
*/
protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
request.setCharacterEncoding("utf-8");
response.setCharacterEncoding("utf-8");
response.setContentType("text/html;charset=utf-8");

//获取表单中的用户名和密码
String username = request.getParameter("username");
String password = request.getParameter("password");

//建立数据库连接
Connection conn = null;
//User类的list集合
List<Users> list = new ArrayList<Users>();

//尝试导入驱动，建立数据库连接，建立连接的密码为空，建立user表
```

```
try {
Class.forName("com.mysql.jdbc.Driver");
conn =
    DriverManager.getConnection("jdbc:mysql://localhost:3306/streetmanagersystem?
useUnicode=true&characterEncoding=utf-8","root","");

PreparedStatement stmt = conn.prepareStatement("select * from user where userName
    = ? and userPassword = ?");
stmt.setString(1, username);
stmt.setString(2, password);
ResultSet rs = stmt.executeQuery();

//如果查询到了，就将用户名和密码存入list集合中
while(rs.next()){
Users user = new Users();
user.setUsername(rs.getString("username"));
user.setPassword(rs.getString("password"));
list.add(user);
}
//如果list集合中无数据，就将数据写入到User表中，并且跳转到登录页面
if(list.size()==0){

PreparedStatement stmt1 = conn.prepareStatement("insert into user values(?,?)");
stmt1.setString(1, username);
stmt1.setString(2, password);
stmt1.execute();
response.sendRedirect("login.jsp");

}else{
response.sendRedirect("register.jsp");
}

} catch (ClassNotFoundException e) {
e.printStackTrace();
```

```
}
catch (SQLException e) {
e.printStackTrace();
}
}

/**
* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse
    response)
*/
protected void doPost(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
// TODO Auto-generated method stub
doGet(request, response);
}

}
```

代码清单 8 street.jsp

```
<%@ page language="java" contentType="text/html; charset=utf-8"
pageEncoding="utf-8" import="java.util.*,datastruct.Street"%>



<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta charset=utf-8">
<title>Insert title here</title>
</head>
<body>

<%
```

```
List<Street> list = (List<Street>)request.getAttribute("comlist");
%>

<table border="1">
<tr>
<td>街道名称</td>
<td>街道管理员</td>
<td>街道管理员电话</td>
</tr>

<%
for(Street com:list){
%>

<tr>
<td><%=com.getStreetname()%></td>
<td><%=com.getStreetmanager()%></td>
<td><%=com.getStreetphone()%></td>
</tr>

<%
}
%>
</table>
<!-- 插入4个超链接，内容为增删改查，目标为right-->
<a href="right/street/addStreet.jsp" target="right">增加街道</a>
<a href="right/street/deleteStreet.jsp" target="right">删除街道</a>
<a href="right/street/updateStreet.jsp" target="right">修改街道</a>
<a href="right/street/queryStreet.jsp" target="right">查询街道</a>

</body>
</html>
```

代码清单 9 deleteStreet.jsp

```
<%@ page language="java" contentType="text/html; charset=utf-8"
pageEncoding="utf-8"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>Insert title here</title>
</head>
<body>

<!-- 生成表单给servlet, 表单字段包括街道名-->
<form action="../../StreetDelete" method="post">
<table>
<tr>
<td>街道名: </td>
<td><input type="text" name="streetName" /></td>
</tr>
<tr>
<td><input type="submit" value="提交" /></td>
<td><input type="reset" value="重置" /></td>
</tr>
</table>
</form>
</body>
</html>
```

代码清单 10 left.jsp

```
<%@ page language="java" contentType="text/html; charset=utf-8"
pageEncoding="utf-8"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">
```

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>Insert title here</title>
</head>
<body>
<a href="CommunityPresent" target="center">社区管理</a>
<br>
<a href="StreetPresent" target="center">街道管理</a>
<br>
<a href="HousePresent" target="center">小区管理</a>
<br>
<a href="HouseholdPresent" target="center">户主管理</a>
<br>
<a href="HouseholdmemberPresent" target="center">家庭成员管理</a>
<br>
</body>
</html>
```

代码清单 11 login.jsp

```
<%@ page language="java" contentType="text/html; charset=utf-8"
pageEncoding="utf-8"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>登录界面</title>
</head>
<body>
<form action="Login" method="post">
用户名: <input type="text" name="username" /><br />
密   码: <input type="password" name="password" /><br />
```

```
<input type="submit" value="登录" />
</form>

<a href="register.jsp">注册</a>

</body>
</html>
```

代码清单 12 main.jsp

```
<!-- 生成jsp模板 -->
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">

<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Insert title here</title>
</head>

<!-- 生成frameset页面, 分左中右三栏 -->
<frameset rows="10%,90%">
<frame src="title.jsp"/>
<frameset cols="20%,40%,40%">
<frame src="left.jsp" name="left" scrolling="auto" noresize="noresize" />
<frame src="CommunityPresent" name="center" scrolling="auto" noresize="noresize"
    />
<frame src="right/community/community_add.jsp" name="right" scrolling="auto"
    noresize="noresize" />
</frameset>
</frameset>
</html>
```

代码清单 13 register.jsp

```
<%@ page language="java" contentType="text/html; charset=utf-8"
pageEncoding="utf-8"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>登录界面</title>
</head>
<body>
<form action="Login" method="post">
用户名: <input type="text" name="username" /><br />
密   码: <input type="password" name="password" /><br />
<input type="submit" value="登录" />
</form>
<a href="register.jsp">注册</a>
</body>
</html>
```