云计算概论课程设计报告

姓名：姚振杰

学号：16337285

1. 设计选题

基于共有云平台设计实现一个云服务。如基于AWS、Window Azure、阿里云等公有云平台开发云服务，具体功能可以自定。（**题目二**）

2.功能实现

2.1 需求分析

本次实验在公有云平台上搭建了一个问卷网站，旨在帮助云用户生成问卷以及统计问卷填写结果。（地址：http://129.211.94.198:8080/Questionnaire/prepareQuestionnaire.html）

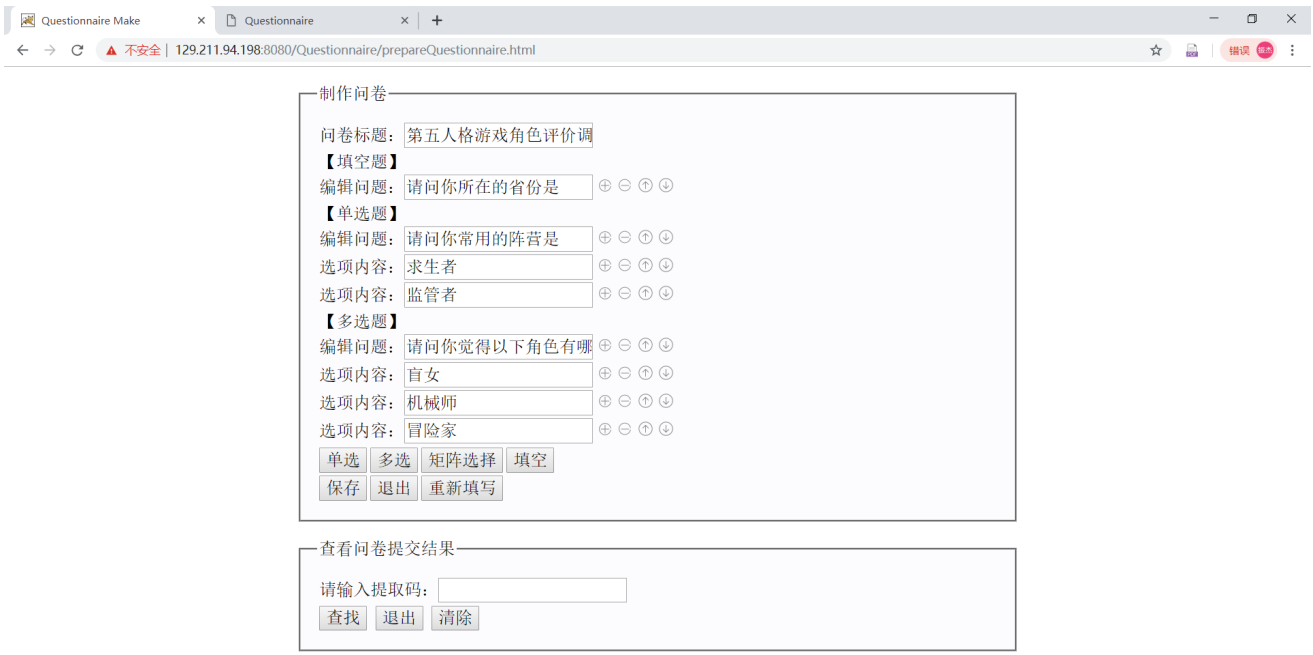
该问卷系统大致的使用流程是：用户可以先访问云服务器上的网页，编辑问卷的标题、问题以及选项，生成所需的问卷（网址以及提取码），访问问卷的网址即可填写问卷；最后输入提取码即可获取问卷填写的统计结果。

需求的细节具体如下所示：

- 可点击对应的按键从前往后插入单选、多选、矩阵选择和填空四种题型，题目和选项的内容可以填写在输入框内。（单选多选默认的选项个数是2，矩阵选择则默认是2行2列，后续操作可增减选项个数）
- 可点击加、减按键增加或删除问题和选项。（增加是往原题目的下方插入相同题型的题目）
- 可点击上、下按键上移或下移问题和选项。
- 生成问卷网址的同时可以看到问卷的预览以及提取码。（支持在预览界面填写问卷并提交）
- 填写提取码可以查看对应问卷填写的统计结果，可下载答案表格。

2.2 功能展示

2.2.1 制作问卷

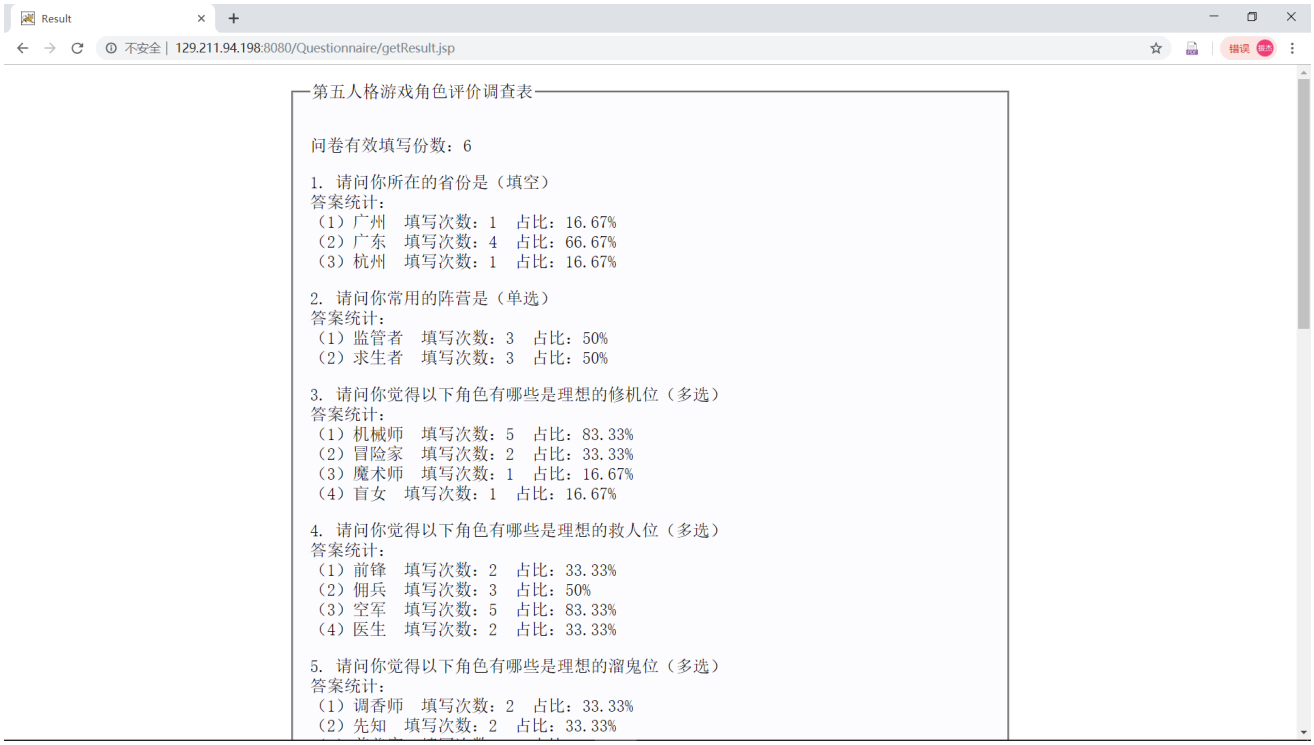

Questionnaire Make
Questionnaire
不安全 | 129.211.94.198:8080/Questionnaire/prepareQuestionnaire.html
制作问卷
问卷标题：第五人格游戏角色评价调
【填空题】
编辑问题：请问你所在的省份是
【单选题】
编辑问题：请问你常用的阵营是
选项内容：求生者
选项内容：监管者
【多选题】
编辑问题：请问你觉得以下角色有哪
选项内容：盲女
选项内容：机械师
选项内容：冒险家
单选 多选 矩阵选择 填空
保存 退出 重新填写
查看问卷提交结果
请输入提取码：
查找 退出 清除

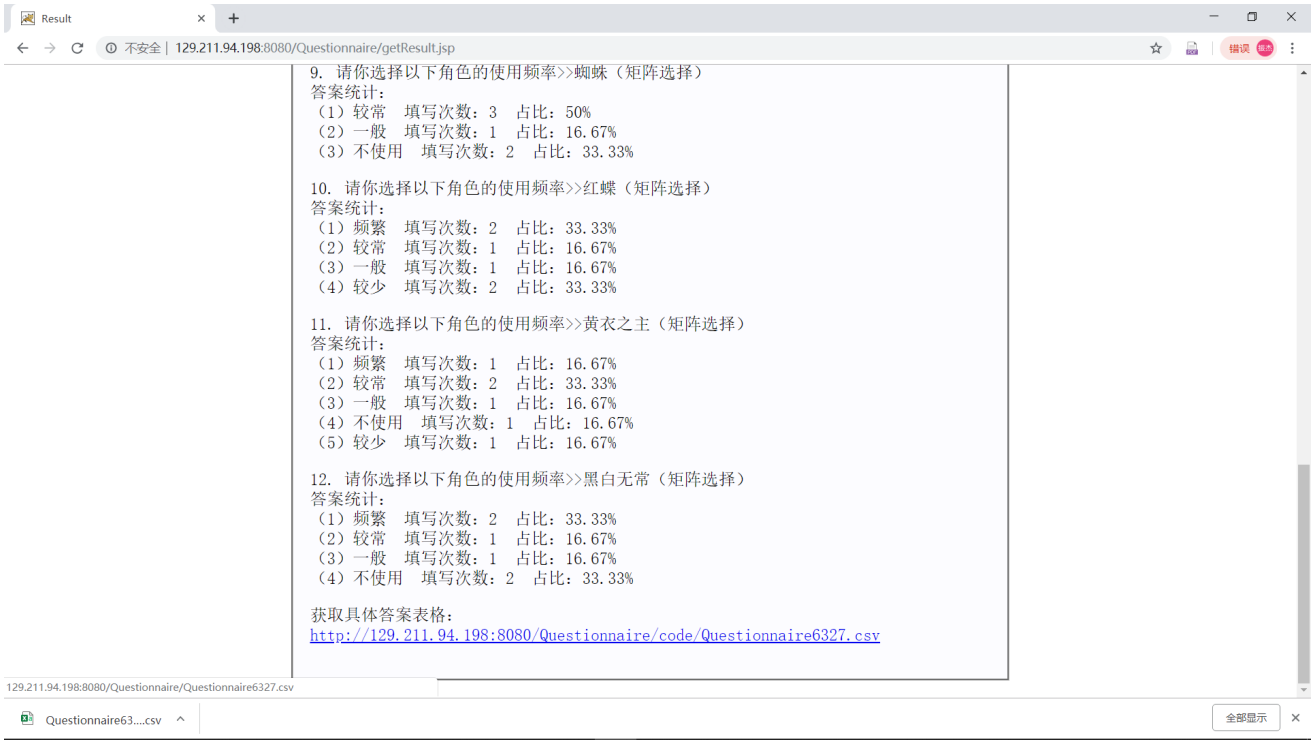
2.2.2 生成问卷


Result
不安全 | 129.211.94.198:8080/Questionnaire/makeQuestionnaire.jsp
预览如下所示：
第五人格游戏角色评价调查表
1. 请问你所在的省份是
2. 请问你常用的阵营是
求生者
监管者
3. 请问你觉得以下角色有哪些是理想的修机位
盲女
机械师
冒险家
魔术师
4. 请问你觉得以下角色有哪些是理想的救人位
前锋
空军
佣兵
医生
5. 请问你觉得以下角色有哪些是理想的溜鬼位
慈善家
魔术师
调香师
先知
6. 请你选择以下角色的使用频率
频繁 较常 一般 较少 不使用
厂长
鹿头

2.2.3 填写问卷

2.2.4 查看问卷填写结果





3. 实验环境

- 云服务器：腾讯云（1核2G，1M带宽系统盘50G）
- 操作系统：Windows Server 2012R2
- 编程语言：以html、JavaScript 和 Java（JSP技术）为主
- 环境配置：jdk1.8.0、tomcat-8.5.41 和 SQL Server2017
- 工具使用：VS Code、SQL Server Management Studio

4. 步骤流程

4.1 制作问卷

该模块即为用户访问的主页面，主要用于制作问卷，代码见于prepareQuestionnaire.html。

4.1.1 静态结构

基本的静态表单、表格、按键、输入框等元素可以使用html来实现，然后通过CSS来调整布局显示。

题目表格的基本结构代码如下所示（以单选题为例）：

```
<table id = "RadioButton_1">
    <thead>
        <tr><td>【单选题】</td></tr>
    </thead>
    <tbody>
        <tr id="RadioButton_1.Question">
            <td>编辑题目：<input name="RadioButton_1.Question"></td>
        </tr>
        <tr id="RadioButton_1.Content_1">
            <td>选项内容：<input name="RadioButton_1.Content_1"></td>
        </tr>
        <tr id="RadioButton_1.Content_2">
            <td>选项内容：<input name="RadioButton_1.Content_2"></td>
        </tr>
    </tbody>
</table>
```

每一行除了提示文字和输入框以外，还具有加（增加）、减（删除）、上（上移）、下（下移）四个按键（以单选题的选项行为例）：

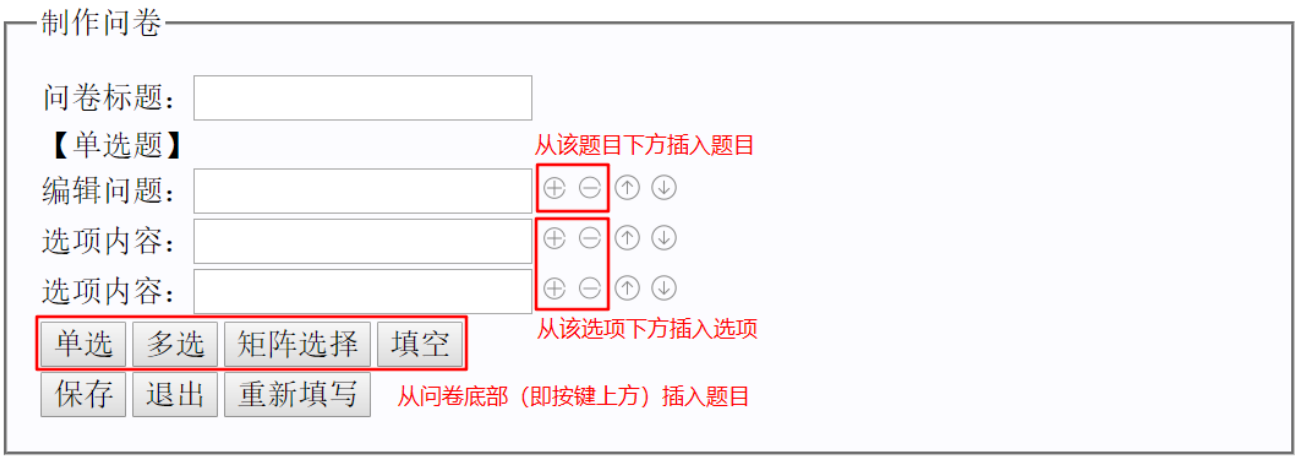
```
"<tr id="RadioButton_1.Content_1">
    <td>选项内容：<input name="RadioButton_1.Content_1"></td>
    <td><img class="plus" src="images\\plus.png"
             onClick="insert('RadioButton', 'content', 'RadioButton_1.Content_1')"
             onmouseover="over('RadioButton_1.Content_1', 'plus')"
             onmouseout="out('RadioButton_1.Content_1', 'plus')"></td>
    <td><img class="minus" src="images\\minus.png"
             onClick="deleteById('RadioButton_1.Content_1')"
             onmouseover="over('RadioButton_1.Content_1', 'minus')"
             onmouseout="out('RadioButton_1.Content_1', 'minus')"></td>
    <td><img class="up" src="images\\up.png"
             onClick="moveUp('RadioButton_1.Content_1')"
             onmouseover="over('RadioButton_1.Content_1', 'up')"
             onmouseout="out('RadioButton_1.Content_1', 'up')"></td>
    <td><img class="down" src="images\\down.png"
             onClick="moveDown('RadioButton_1.Content_1')"
             onmouseover="over('RadioButton_1.Content_1', 'down')"
             onmouseout="out('RadioButton_1.Content_1', 'down')"></td>
</tr>
```

但实际上静态实现是远远不够的，增删和移动等更新操作需要使用JavaScript脚本来动态实现，而且上述html代码主要是在insert函数中调用insertAdjacentHTML函数中实现的，通过点击按键来触发。

最后点击【保存】按键以POST方式将表单填写内容（名值对）提交到makeQuestionnaire.jsp。

4.1.2 增加操作

增加操作可以分为增加选项和增加题目（注：题目=问题+选项）两种：增加选项可以通过点击选项右边的加号按键向下插入选项；增加题目可以通过点击问卷底部的按键从问卷的底部插入问题，或者点击问题右边的加号按键向下插入题目。



根据以上不同的增加操作，具体的函数实现也会有所不同。insert函数的定义如下：

```
/**
 * 增加题目或选项
 * @paras
 * type表示增加的题型(RadioButton, CheckBox, RadioMatrix, BlankTofillIn)
 * click表示点击的按键(Content表示选项的加号按键, Question表示题目的加号按键, Button表示底部的插入按键)
 * id表示选项(tr元素)或题目(table元素)或插入按键(table元素)的id
 */
function insert(type, click, id)
```

- 增加选项

如果是增加选项，即直接在该选项的下方插入选项，可以先根据id获取该节点元素，然后调用insertAdjacentHTML方法插入新的选项（第一个参数是"afterend"表示直接在该节点后方插入）。注意生成新的选项的id需要先遍历所有选项获取最大的id，然后在此基础上加一即可获得新的id。

实现代码如下所示（具体插入的HTML代码可以参考4.1.1，题目选项id替换为contentId即可，由于边幅过大此处不再赘述，仅用contentHTML省略代替）：

```
var tableId = btn.parentNode.parentNode.id;
var max = 0; // 找出最大的选项id，用于生成新的id(+1)
for(var i = 0; i < btn.parentNode.childNodes.length; i++){
    var index = parseInt(btn.parentNode.childNodes[i].id.split(".Content_")[1]);
    if(max < index)
        max = index;
}
var contentId = tableId + ".Content_" + (++max);
btn.insertAdjacentHTML("afterend", contentHTML);    // 向后插入表格（点击选项加号键）
```

- 增加题目

如果是通过点击选项右边加号按键增加表格，即直接在该题目下方插入题目，实现方法和以上增加选项的方法类似；如果是通过点击问卷底部按键增加表格，即在按键上方插入题目，也可以先根据id获取该节点元素，然后调用insertAdajecentHTML方法插入新的题目（第一个参数是"beforebegin"表示直接在该节点前方插入）。题目id的获取需要使用一个全局变量count来计数。另外，为了减少硬编码可以先插入题目（**table**元素）以及问题（**tr**元素），然后再对题目的子节点元素（**tbody**元素）循环调用insertAdajecentHTML方法插入新的选项（第一个参数是"beforeend"表示直接在该节点的最后一个子节点的后方插入）。

实现代码如下所示（具体插入的HTML代码可以参考4.1.1，题目、问题和选项id分别替换为tableId, questionId和contentId即可，此处同不再赘述，仅用tableHTML和contentHTML省略代替）：

```
var tableId = "RadioButton_" + count;
var questionId = tableId + ".Question";
if(click == "button")
    btn.insertAdjacentHTML("beforebegin", tableHTML);  // 向前插入表格（点击题目插入键）
else if(click == "question")
    btn.insertAdjacentHTML("afterend", tableHTML);     // 向后插入表格（点击题目加号键）
var table = document.getElementById(tableId);
for(var i = 1; i <= 2; i++){
    var contentId = tableId + ".Content_" + i;
    table.children[1].insertAdjacentHTML("beforeend", contentHTML);
}
count++;
```

4.1.3 删除操作

删除操作其实很简单，直接根据id获得当前节点元素，然后使用其父元素节点调用函数removeChild即可把该节点删除。

```
// 删除题目或选项
function deleteById(id){
   var node = document.getElementById(id);
   node.parentNode.removeChild(node);
}
```

4.1.4 上/下移操作

上移操作可以使用其父元素节点调用函数insertBefore将该节点插入到前一个节点的前一个位置；下移操作则是将该节点插入到后第二个节点的前一个位置，如果后一个节点已经是父节点的最后一个子节点了，则使用其父节点调用函数appendChild即可插入到最后一个子节点的后面。当然，不管是上移还是下移都要考虑到不做任何操作的边界条件。

```
// 上移题目或选项
function moveUp(id){
   // 注意：父节点的第一个节点是question(当前节点为content)或title(当前节点为table)
   var node = document.getElementById(id);
   var prenode = node.previousElementSibling;
   if(prenode.previousElementSibling != null){
      node.parentNode.insertBefore(node, prenode);
   }
}
// 下移题目或选项
function moveDown(id){
   // 注意：父节点的最后一个子节点是button(当前节点为table)
   var node = document.getElementById(id);
   var next = node.nextElementSibling;
   if(next != null){
      var nextext = next.nextElementSibling;
      if(nextext != null)
         node.parentNode.insertBefore(node, nextext);
      else if(next.id != "insert_button")
         node.parentNode.appendChild(node);
   }
}
```

4.1.5 鼠标悬停

用于问题和选项后面的按键是自定义图标，使用的是image元素，为了更加凸显按键显示的效果，可以在鼠标悬停和鼠标离开的时候使用不同颜色的原图片。鼠标悬停触发的机制是onmouseover="over(contentId, type)"；鼠标离开触发的机制是onmouseout="out(contentId, type)"。

over函数与out函数的实现如下所示：

```
// 鼠标悬停（按键变成红色）
function over(id, type){
   if(type == "plus")
      document.getElementById(id).childNodes[1].childNodes[0].src =
"images\\plus_over.png";
   else if(type == "minus")
      document.getElementById(id).childNodes[2].childNodes[0].src =
"images\\minus_over.png";
   else if(type == "up")
      document.getElementById(id).childNodes[3].childNodes[0].src =
"images\\up_over.png";
   else if(type == "down")
      document.getElementById(id).childNodes[4].childNodes[0].src =
"images\\down_over.png";
```

```
}
// 鼠标移走（按键恢复灰色）
function out(id, type){
   if(type == "plus")
      document.getElementById(id).childNodes[1].childNodes[0].src = "images\\plus.png";
   else if(type == "minus")
      document.getElementById(id).childNodes[2].childNodes[0].src ="images\\minus.png";
   else if(type == "up")
      document.getElementById(id).childNodes[3].childNodes[0].src = "images\\up.png";
   else if(type == "down")
      document.getElementById(id).childNodes[4].childNodes[0].src = "images\\down.png";
}
```

4.2 生成问卷

该模块主要根据prepareQuestionnaire.html提交的名值对，使用相应的input元素将题目显示出来，并使用文件输出流将同样的代码写到文件Questionnaire.html中，并将该问卷的网址显示给用户以及生成相应提取码。其代码见于makeQuestionnaire.jsp。

4.2.1 生成提取码

由于问卷网站的用户不止一个，为了区分不同的问卷，我们需要使用不同的提取码，问卷相关的文件和数据库表都是使用"Questionnaire+提取码"来命名。另外用户提取问卷填写结果时也需要使用到提取码。

生成提取码使用0~10000的随机整数（即可以生成10000份不同的问卷），生成过的提取码存储在数据库的Manager表中，以保证不同问卷的提取码不同。

```
// 生成提取码
Random rand = new Random();
int id = rand.nextInt(10000);   // 在0~10000中选取随机值
String sql = "select * from Manager where id = " + id;
ResultSet rs = stmt.executeQuery(sql);
while(rs.next()){
    id = rand.nextInt(10000);
    rs = stmt.executeQuery(sql);
}
```

4.2.2 问卷的数据存储

每一份问卷可以以表的形式存储在数据库中，以问卷文件名为表名，问卷的问题为列名，这样的话每一份问卷的填写结果都可以依序插入到表中。但存在一个问题是，我们不能够直接将问卷的问题作为列名，因为如果问题中存在诸如"/"等sql语言中的运算符，可能就会导致不能正常建表。所以我们需要构建一个问题到列名的映射关系，第一个问题使用Question1作为列名，以此类推。具体的问题和标题内容可以存储在一个txt文件中。

连接数据库的代码实现如下所示：

```
// 连接数据库
String DBDRIVER="com.microsoft.sqlserver.jdbc.SQLServerDriver";
String DBURL="jdbc:sqlserver://127.0.0.1:1433;databaseName=questionnaire";
String DBUSER="sa";
String PASSWORD="Yunjisuan2019";
Class.forName(DBDRIVER);
Connection conn = DriverManager.getConnection(DBURL, DBUSER, PASSWORD);
Statement stmt = conn.createStatement();
stmt.excute(sql);   // sql为String类型的sql语句，如果参数是查询语句调用excuteQuery函数可以获得返回的结果集
```

写入文件的代码如下所示：

```
PrintWriter fout = null;
out.println("<div>预览如下所示：</div>");
try {
    fout = new PrintWriter(new BufferedWriter(new OutputStreamWriter(new FileOutputStream(file),"utf-8")));
} catch (Exception e) {
    e.printStackTrace();
}
fout.write(str);    // 将字符串str写入到文件中
fout.close();       // 必须要关闭PrintWriter对象才能完成写操作
```

4.2.3 预览问卷和生成问卷

预览问卷是在用户提交编辑好的问卷之后浏览器跳转看到的，主要是使用JSP技术将系统输出的语句转换成html语句，然后将html语句返回到浏览器中。生成问卷则是通过文件流输出，将相同的html语句写入到html文件中。

以一个单选题为例：

```
out.print("<table><tr><td><b>");
out.print(size + ". " + question);
out.print("</b></td></tr></table>");
fout.write("<table>\r\n<tr>\r\n<td>\r\n<b>");
fout.write(size + ". " + question);
fout.write("<b>\r\n</td>\r\n</tr>\r\n</table>\r\n");
out.print("<table>");
fout.write("<table>\r\n");
for(String content: contents){
    out.print("<tr><td>");
    out.print("<input name=\"RadioButton\1" + question + "\" value=\"" + content + "\" type=\"radio\" >" + content);
    out.print("</td></tr>");
    fout.write("<tr>\r\n<td>");
    fout.write("<input name=\"RadioButton\1" + question + "\" value=\"" + content + "\" type=\"radio\" >" + content);
    fout.write("</td>\r\n</tr>\r\n");
}
```

4.2.4 显示问卷链接和提取码

使用a元素来来实现超链接，点击可以跳转到实现好的问卷。代码实现很简单，如下所示：

```
out.println("<a href=\"http://129.211.94.198:8080/Questionnaire/" + filename +
".html\">");
out.println("http://129.211.94.198:8080/Questionnaire/code/" + filename + ".html</a>");
```

实现的效果如下所示：

问卷链接： http://129.211.94.198:8080/Questionnaire/code/Questionnaire3130.html

请务必记住问卷结果提取码：3130

4.3 分析问卷填写结果

该模块主要是用于将问卷提交的内容插入到数据库中，需要注意的是多选题是以数组的形式存储。代码见于analyseQuestionnaire.jsp。

```
int counter = 1;    // 计数器
String sql = "insert into " + filename + " values";
Enumeration<String> enums = request.getParameterNames();
while(enums.hasMoreElements()){
    String name=(String)enums.nextElement();
    if(name.split("\1")[0].equals("CheckBox")){ // '\1'作为题目类型和题目内容的一个分隔符号
        String[] groups = request.getParameterValues(name);
        out.println(name.split("\1")[1] + ": " + Arrays.toString(groups) + "<br/>");
        sql = sql + (counter++ == 1 ? "(" : ",") + "'" + Arrays.toString(groups) + "'";
    }
    else if(name.split("\1").length > 1){
        out.println(name.split("\1")[1] + ": " + request.getParameter(name) + "<br/>");
        sql = sql + (counter++ == 1 ? "(" : ",") + "'" + request.getParameter(name) +
"'";
    }
}
sql = sql + ");";
stmt.execute(sql);
```

在云服务器中可以使用SQL Server Management Studio查询表的数据（问卷的填写内容）：

```
select * from Questionnaire6327;
```

161 %

结果 消息

	id	Question1	Question2	Question3	Question4	Question5	Question6	Question7	Question8	Question9	Question10	Question11	Question12
1	1	广东	求生者	[盲女]	[佣兵]	[慈善家]	较少	不使用	一般	较常	频繁	不使用	不使用
2	2	广东	求生者	[机械师]	[空军，佣兵]	[慈善家，魔术师]	较常	一般	较少	较常	一般	较常	不使用
3	3	广东	监管者	[机械师，冒险家]	[空军，医生]	[魔术师]	一般	较常	较常	一般	较少	较常	频繁
4	4	杭州	监管者	[机械师]	[空军]	[魔术师，调香师]	频繁	不使用	一般	不使用	频繁	频繁	较常
5	5	广州	求生者	[机械师，冒险家]	[前锋，空军]	[先知]	较常	较少	一般	不使用	较常	较少	一般
6	6	广东	监管者	[机械师，魔术师]	[前锋，空军，佣兵，医生]	[魔术师，调香师，先知]	较常	一般	一般	较常	较少	一般	频繁

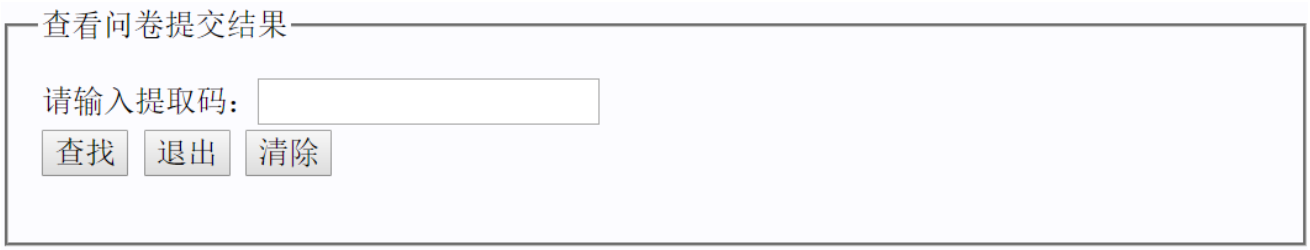
4.4 获取问卷填写结果

4.4.1 提取码填写入口

提交提取码的入口可见于prepareQuestionnaire.html，makeQuestionnaire.jsp，点击查找按键将填写的提取码提交到getResult.jsp中。代码实现如下所示：

```
<fieldset>
   <legend>查看问卷提交结果</legend>
   <form action="http://129.211.94.198:8080/Questionnaire/getResult.jsp" method="post">
      <table id="BlankToFillIn">
         <thead>
            <tr><td>请输入提取码：<input name="id"></td></tr>
         </thead>
         <tbody>
            <tr>
               <td>
                  <input name="search" type="submit" value="查找">
                  <input name="exit" type="button" value="退出"
onclick="window.close();document.write('<n>')">
                  <button name="reset" type="reset">清除</button>
               </td>
            </tr>
         </tbody>
      </table>
   </form>
</fieldset>
```

实现的效果如下所示：



4.4.2 分析填写结果

- 首先从问卷对应的txt文件中读出问卷的标题和问题内容，然后从数据库查询得到各个问题的填写结果。
- 然后将填写结果显示在浏览器的网页上，并统计答案出现的占比。
- 最后还将所有答案依次写进csv文件中，可以点击链接进行下载。

具体的实现原理和4.2.2, 4.2.3, 4.2.4中的叙述相近，在此不再赘述。实现的效果如下所示：

第五人格游戏角色评价调查表

问卷有效填写份数：6

1. 请问你所在的省份是（填空）
答案统计：
（1）广州　填写次数：1　占比：16.67%
（2）广东　填写次数：4　占比：66.67%
（3）杭州　填写次数：1　占比：16.67%

2. 请问你常用的阵营是（单选）
答案统计：
（1）监管者　填写次数：3　占比：50%
（2）求生者　填写次数：3　占比：50%

12. 请你选择以下角色的使用频率>>黑白无常（矩阵选择）
答案统计：
（1）频繁　填写次数：2　占比：33.33%
（2）较常　填写次数：1　占比：16.67%
（3）一般　填写次数：1　占比：16.67%
（4）不使用　填写次数：2　占比：33.33%

获取具体答案表格：
http://129.211.94.198:8080/Questionnaire/code/Questionnaire6327.csv

5. 项目总结

5.1 云平台环境部署

在这一步中，腾讯云运营商作为云提供者，交付的是IaaS，网站搭建者作为云用户在此基础上部署所需的环境，进一步实现并运行问卷网站的程序。

5.1.1 Java开发与运行环境

jdk和jre的下载安装直接访问Oracle官网就可以了，虽然操作简单但是有一定的耗时。安装完成后设置好相应的环境变量即可，在命令行能够获取到Java和Javac的版本即可说明配置成功。

```
C:\Users\Administrator>java -version
java version "1.8.0_211"
Java(TM) SE Runtime Environment (build 1.8.0_211-b12)
Java HotSpot(TM) Client VM (build 25.211-b12, mixed mode)

C:\Users\Administrator>javac -version
javac 1.8.0_211
```

5.1.2 tomcat服务器安装

tomcat服务器的下载安装可以直接访问Apache Tomcat官网，安装完成后在bin文件夹下运行startup.bat脚本文件即可启动tomcat服务器（注：前提条件是要先配置好Java环境）。

登录http://localhost:8080可以访问到webapps/ROOT下，所以项目在此目录下搭建即可。外网只需主机名localhost改成该服务器的外网ip(129.211.94.198)即可登录到该网站。

5.1.3 数据库安装以及驱动搭建

SQL Server数据库的下载安装访问Microsoft官网即可，安装完成后可以进一步安装SQL Server Manager Studio来集成管理数据库，方便与数据库的创建、查询和更新。

第一次登陆数据库先使用Windows身份认证，然后启用sa登录名并更改密码，然后可以在ODBC数据源管理程序（在管理工具可以找到）添加以SQL Server为驱动程序的系统DSN。

然后打开SQL Server配置管理器，找到SQL Server网络配置下的MSSQLS的协议，启用TCP/IP协议，重新启动SQL Server服务。

最后在代码实现上即可连接上数据库：

```
String DBDRIVER="com.microsoft.sqlserver.jdbc.SQLServerDriver";
String DBURL="jdbc:sqlserver://127.0.0.1:1433;databaseName=questionnaire";
String DBUSER="sa";
String PASSWORD="Yunjisuan2019";
Class.forName(DBDRIVER);
Connection conn = DriverManager.getConnection(DBURL, DBUSER, PASSWORD); // 连接数据库
```

至此，云平台的环境部署基本完备，可以开始编写代码实现网站后台的逻辑运转了。

5.2 问卷网站服务

在这一情形下，问卷网站运营方作为云提供者，向云用户提供制作问卷以及提取问卷结果的云服务，以SaaS的形式进行交付。云用户只需访问http://129.211.94.198:8080/Questionnaire/prepareQuestionnaire.html，按照2.2.1的展示即可使用这一云服务。

不过问卷网站提供的服务尚有许多需要改善的地方，比如可以增加新的题型，例如打分题、排序题等等。另外，安全性也亟需提升，任何用户都可以通过提取码获取问卷填写结果，这显然是不符合保密性需求的，应该通过创建账户和登录以及数据库设置登录名权限来解决这一安全性问题。