

Web应用开发

大作业

|  |  |
| --- | --- |
| 姓 名： | 刘宇阳 |
| 院 （系）： | 计算机学院 |
| 班 级： | 软件1901 |
| 学 号： | 04193016 |
| 课程教师： | 张荣 |

**现需要完成一个适用于家电销售管理系统，能实现基本的登陆、登录用户管理、家电分类管理、家电信息管理、家电销售管理等，请完成如下方案的设计。**

**1.构架设计（20分）：**本系统基于B/S设计开发，分析比较主流的架构的区别，并说明选用哪个架构完成本系统；详细描述本系统在架构各层实现的内容。

**答：**

现在比较主流的架构都有MVC架构、MVVM架构以及MVP架构模式。

**MVC架构**是主流且应用较为广泛的软件架构模式，基础模式为Model、View和Controller， 这三层是紧密联系在一起的，但又是互相独立的，每一层内部的变化不影响其他层。每一层都对外提供接口（Interface），供上面一层调用。这样一来，软件就可以实现模块化，修改外观或者变更数据都不用修改其他层，大大方便了维护和升级。

**MVVM架构**的基础模式为Model、View和ViewModel，现在主要流行的前端框架都是用的是MVVM架构，比如Vue、Angular和React。它将Controller和ViewModel做数据绑定，把Controller的数据和逻辑处理部分从中抽离出来，用一个专门的对象去管理，这个对象就是ViewModel，是Model和Controller之间的一座桥梁。

**MVP架构**的基础模式为Model、View和Presenter，分别表示数据层、视图层、发布层，它是MVC架构的一种演变。作为一种新的模式，MVP与MVC有着一个重大的区别：在MVP中View并不直接使用Model，它们之间的通信是通过Presenter (MVC中的Controller)来进行的，所有的交互都发生在Presenter内部，而在MVC中View会直接从Model中读取数据而不是通过 Controller。

本家电销售管理系统使用的是MVC架构模式，将Model、View和Controller三层分别实现不同的功能。在Controller层中，接收前端发送的请求，比如搜索购买家电或登陆注册等功能，并判断请求内容并向Model层传递数据。Model层用来处理数据并返回最终结果，例如用户的电话、购买的时间、购买的家电名称等等。而View层是将Model层返回的数据在页面中做渲染处理，让使用者可以观察到数据。

**2.需求设计（30分）：**

2.1设计本系统包含的角色、总体用例图，并详细说明。（15分）

**答：**

管理员主要有登录系统、管理用户、管理家电、管理订单四个模块功能，其中“管理用户”可以修改用户信息和删除用户信息，“管理家电”中可以添加家电信息、搜索家电信息、修改家电信息和删除家电信息。“管理订单”中额可以搜索订单信息、修改订单信息和删除订单信息，具体用例图如图1所示。

图示

描述已自动生成

图1 管理员用例图

用户主要用登陆系统、查看家电和查看订单等三个模块功能，其中“查看家电”可以搜索家电和购买家电，“查看订单”中可以搜索订单和删除订单，用户用例图如下图2所示。

图示

描述已自动生成

图2 用户用例图

2.2通过时序图来描述从前端页面输入电器名搜索到控制器、Service层、DAO层，最后返回页面的过程，对各层调用方法和传递参数进行说明，并用文字对上述时序图进行描述。（15分）

**答：**

电器名搜索模块实现的主要原理，前端将表单中的家电名称homeapp\_name使用Ajax中post异步请求的方法传入给后端的HomeAppsServlet控制器层，Servlet中使用request. getParameter()方法获取到家电名称的值，并使用List<T>创建列表并调用HomeAppSrv().search(homeapp\_home)方法访问Service层，在Service层中，调用DAO层中的select(homeapp\_name)方法可以查询数据库中的数据，并在DAO层中将数据插入到homeAppLists列表中并返回给Service层，Service层收到后返回给Servlet层，在Servlet层中，使用Json将列表解析并转换为字符串返回给前端，前端使用JSON.parse(req.responseText)方法将字符串重新转换为JSON格式并使用动态创建标签的方法将数据渲染在页面中。

图片包含 图示

描述已自动生成

图3 电器搜索时序图

**3.数据库设计（20分）**：设计此系统数据库表结构，标明表名、字段名、字段类型、主外键等约束关系。

**答：**

homeapp数据库建表语句：<https://github.com/2193560021/HomeApp/blob/master/homeapp.sql>

表1管理员表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 字段类型 | 允许为空 | 说明 |
| man\_id | int(11) | NO | 管理员ID（PRI） |
| man\_name | varchar(100) | YES | 管理员姓名 |
| man\_tel | varchar(20) | YES | 管理员电话 |
| man\_pwd | text | YES | 管理员密码 |

表2用户表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 字段类型 | 允许为空 | 说明 |
| cus\_id | int(11) | NO | 管理员ID（PRI） |
| cus\_name | varchar(100) | YES | 管理员姓名 |
| cus\_tel | varchar(20) | YES | 管理员电话 |
| cus\_pwd | text | YES | 管理员密码 |

表3家电表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 字段类型 | 允许为空 | 说明 |
| homeapp\_id | int(11) | NO | 家电ID（PRI） |
| homeapp\_name | varchar(100) | YES | 家电名称 |
| homeapp\_img | varchar(50) | YES | 家电图片 |
| homeapp\_kind | varchar(50) | YES | 家电种类 |
| homeapp\_intro | text | YES | 家电简介 |
| homeapp\_price | varchar(50) | YES | 家电价格 |

表4订单表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 字段类型 | 允许为空 | 说明 |
| order\_id | int(11) | NO | 订单ID（PRI） |
| order\_homeapp\_name | varchar(100) | YES | 家电名称 |
| order\_owner\_tel | varchar(50) | YES | 顾客电话 |
| order\_homeapp\_img | text | YES | 家电图片 |
| order\_price | varchar(50) | YES | 售出价格 |

**4.编程实现（30分）：**

4.1前端页面：完成搜索家电展示页面功能，在文本框中输入家电名，点击“搜索”按钮发起请求；将服务端返回的家电详细信息显示出来。（15分）

**答：**

|  |
| --- |
| **html:**  <div>  <input type="text" id="homeapp\_name">  <input type="button" id="homeapp\_search" value="搜索" onclick="search()">  <hr>  <img src="" id="homeapp\_img" >  <h1 id="homeapp\_name"></h1>  <h3 id="homeapp\_kind"></h3>  <h4 id="homeapp\_price"></h4>  <p id="homeapp\_intro"></p>  </div> |
| **JavaScript:**  function search(){  var url = "HomeAppsServlet";  if (window.XMLHttpRequest)  req = new XMLHttpRequest();  else if (window.ActiveXObject)  req = new ActiveXObject("Microsoft.XMLHTTP");  if (req) {  req.open("post", url, true);  req.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");  req.onreadystatechange = searchComplete;  req.send("type=searchByName&homeapp\_name="+document.getElementById("homeapp\_name").value);  }  }  function searchComplete() {  if (req.readyState == 4 && req.status == 200) {  var json = JSON.parse(req.responseText);  document.getElementById("homeapp\_img").setAttribute("src",json[0].homeapp\_img);  document.getElementById("homeapp\_name").innerHTML = json[0].homeapp\_name  document.getElementById("homeapp\_kind").innerHTML = json[0].homeapp\_kind  document.getElementById("homeapp\_price").innerHTML = json[0].homeapp\_price  document.getElementById("homeapp\_intro").innerHTML = json[0].homeapp\_intro  }  } |

前端部分运行结果如下图4所示

图形用户界面, 应用程序

描述已自动生成

图4 搜索-前端部分页面

4.2控制器端：获取前端提交过来的家电名称，调用如下API获取该家电具体信息，并将获取到的家电信息返回到前端。（15分）

|  |  |
| --- | --- |
| 类名 | HAInfoDAO |
| 方法名 | public HAInfo getHAInfoByName(String name) |

**答：**

|  |
| --- |
| **HomeAppsServlet:**  @WebServlet("/HomeAppsServlet ")  public class HomeAppsServlet extends HttpServlet{  protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException  {  doPost(request, response);  }  protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException  {  String type=request.getParameter("type");  if(type.equalsIgnoreCase("searchByName"))  searchByName(request, response);  …  }  private void searchByName(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException  {  response.setCharacterEncoding("UTF-8");  PrintWriter out=response.getWriter();  String homeapp\_name=request.getParameter("homeapp\_name");  HAInfo result=null;  result=new HAInfoDAO().getHAInfoByName(homeapp\_name);  String jsonStr="";  try  {  JSONArray array=new JSONArray();  JSONObject json;  for(HAInfo s : result)  {  json=new JSONObject();  json.put("homeapp\_id", s.getHomeapp\_id());  json.put("homeapp\_name", s.getHomeapp\_name());  json.put("homeapp\_img", s.getHomeapp\_img());  json.put("homeapp\_kind", s.getHomeapp\_kind());  json.put("homeapp\_intro", s.getHomeapp\_intro());  json.put("homeapp\_price", s.getHomeapp\_price());  array.put(json);  }  jsonStr=array.toString();  }  catch(JSONException e)  {  e.printStackTrace();  }  finally  {  out.println(jsonStr);  out.flush();  out.close();  }  } |

后端通过接收并返回请求到前端后，前端渲染信息到页面上，如下图5所示。

图形用户界面, 文本

描述已自动生成

图5 搜索结果