目录

**[1.1需求分析 1](#_Toc8536)**

[1.2 新增检查组 1](#_Toc6733)

[1.2.1 完善页面 1](#_Toc11454)

[1.2.2提交请求 4](#_Toc2350)

[1.2.3 后台代码 5](#_Toc21894)

[1.3 检查组分页 9](#_Toc13494)

[1.3.1 完善页面 9](#_Toc20211)

[1.3.2 后台代码 10](#_Toc2948)

[1.4 编辑检查组 12](#_Toc23525)

[1.4.1 完善页面 12](#_Toc8372)

[1.4.2 后台代码 13](#_Toc6109)

# 1.1需求分析

检查组其实就是多个检查项的集合，例如有一个检查组为“一般检查”，这个检查组可以包 括多个检查项：身高、体重、收缩压、舒张压等。所以在添加检查组时需要选择这个检 查组包括的检查项。

检查组对应的实体类为CheckGroup，对应的数据表为t\_checkgroup。检查组和检查项 为多对多关系，所以需要中间表t\_checkgroup\_checkitem进行关联

## 1.2 新增检查组

### 1.2.1 完善页面

检查组管理页面对应的是checkgroup.html页面，根据产品设计的原型已经完成了页面基 本结构的编写，现在需要完善页面动态效果。

#### 1.2.1.1 弹出新增窗口

页面中已经提供了新增窗口，只是出于隐藏状态。只需要将控制展示状态的属性 dialogFormVisible改为true即可显示出新增窗口。点击新建按钮时绑定的方法为 handleCreate，所以在handleCreate方法中修改dialogFormVisible属性的值为true即 可。同时为了增加用户体验度，需要每次点击新建按钮时清空表单输入项。

由于新增检查组时还需要选择此检查组包含的检查项，所以新增检查组窗口分为两部分 信息：基本信息和检查项信息，如下图：



新建按钮绑定单击事件，对应的处理函数为handleCreate



handleCreate函数首先需要将新增窗口显示出来，同时还需要将表单重置(便于下一次打开新增窗口时，原有的新增的内容需要被清空。)



handleCreate函数还需要将新增窗口中的检查项信息加载出来，便于用户勾选。

#### 1.2.1.2 动态展示检查项列表

现在虽然已经完成了新增窗口的弹出，但是在检查项信息标签页中需要动态展示所有的 检查项信息列表数据，并且可以进行勾选。具体操作步骤如下。

通过分析页面我们发现，检查项信息的显示其实是放在一个table中的。然后使用循环遍历的方式动态加载检查项信息。





后台代码：

CheckItemController:

|  |
| --- |
| *//查询所有检查项信息* @RequestMapping(**"/findAll"**) **public** Result findAll(){  **try**{  List<CheckItem> list = **checkItemService**.findAll();  **return new** Result(**true**, MessageConstant.***QUERY\_CHECKITEM\_SUCCESS***,list);  }**catch** (Exception e){  e.printStackTrace();  *//服务调用失败* **return new** Result(**false**, MessageConstant.***QUERY\_CHECKITEM\_FAIL***);  } } |

CheckItemService:

|  |
| --- |
| List<CheckItem> findAll(); |

CheckItemServiceImpl:

|  |
| --- |
| @Override **public** List<CheckItem> findAll() {  **return checkItemDao**.findAll(); } |

CheckItemDao:

|  |
| --- |
| List<CheckItem> findAll(); |

CheckItemDao.xml:

|  |
| --- |
| <**select id="findAll" resultType="com.qf.pojo.CheckItem"**>  select *\** from t\_checkitem </**select**> |

启动测试：



### 1.2.2提交请求

当用户点击新增窗口中的确定按钮时发送ajax请求将数据提交到后台进行数据库操作。提 交到后台的数据分为两部分：检查组基本信息（对应的模型数据为formData）和检查项 id数组（对应的模型数据为checkitemIds）。

为确定按钮绑定单击事件，对应的处理函数为handleAdd。



#### 1.2.2.1 完善handleAdd方法

通过发送ajax请求，将新增数据提交给后台处理。



### 1.2.3 后台代码

#### 1.2.3.1 controller

在health\_backend工程中创建CheckGroupController。

|  |
| --- |
| */\*\*  \* 检查组管理  \*/* @RestController @RequestMapping(**"/checkgroup"**) **public class** CheckGroupController {   @Reference  **private** CheckGroupService **checkGroupService**;   *//新增检查组* @RequestMapping(**"/add"**)  **public** Result add(@RequestBody CheckGroup checkGroup, Integer[] checkitemIds){  **try**{  **checkGroupService**.add(checkGroup,checkitemIds);  }**catch** (Exception e){  e.printStackTrace();  **return new** Result(**false**, MessageConstant.***ADD\_CHECKGROUP\_FAIL***);*//新增失败* }  **return new** Result(**true**,MessageConstant.***ADD\_CHECKGROUP\_SUCCESS***);*//新增成功* } } |

#### 1.2.3.2 服务接口

在health\_interface工程中创建CheckGroupService接口

|  |
| --- |
| **public interface** CheckGroupService {  **void** add(CheckGroup checkGroup, Integer[] checkitemIds); } |

#### 1.2.3.3服务实现类

在health\_provider工程中创建CheckGroupServiceImpl实现类

|  |
| --- |
| */\*\*  \* 检查组服务  \*/* @Service(interfaceClass = CheckGroupService.**class**) @Transactional **public class** CheckGroupServiceImpl **implements** CheckGroupService {  @Autowired  **private** CheckGroupDao **checkGroupDao**;  *//新增检查组，同时需要让检查组关联检查项* **public void** add(CheckGroup checkGroup, Integer[] checkitemIds) {  *//新增检查组，操作t\_checkgroup表* **checkGroupDao**.add(checkGroup);  *//设置检查组和检查项的多对多的关联关系，操作t\_checkgroup\_checkitem表* Integer checkGroupId = checkGroup.getId();  **this**.setCheckGroupAndCheckItem(checkGroupId,checkitemIds);  }   *//建立检查组和检查项多对多关系* **public void** setCheckGroupAndCheckItem(Integer checkGroupId,Integer[] checkitemIds){  **if**(checkitemIds != **null** && checkitemIds.**length** > 0){  **for** (Integer checkitemId : checkitemIds) {  Map<String,Integer> map = **new** HashMap<>();  map.put(**"checkgroupId"**,checkGroupId);  map.put(**"checkitemId"**,checkitemId);  **checkGroupDao**.setCheckGroupAndCheckItem(map);  }  }  } } |

#### 1.2.3.4 Dao接口

创建CheckGroupDao接口

|  |
| --- |
| **public interface** CheckGroupDao {  **void** add(CheckGroup checkGroup);   **void** setCheckGroupAndCheckItem(Map<String, Integer> map); } |

#### 1.2.3.5 Mapper映射文件

创建CheckGroupDao.xml映射文件

|  |
| --- |
| *<?***xml version="1.0" encoding="UTF-8"** *?>* **<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"  "http://mybatis.org/dtd/mybatis-3-mapper.dtd" *>*** <**mapper namespace="com.qf.dao.CheckGroupDao"**>  *<!--插入检查组数据-->* <**insert id="add" parameterType="com.qf.pojo.CheckGroup"**>  *<!--通过mybatis框架提供的selectKey标签获得自增产生的ID值-->* <**selectKey resultType="java.lang.Integer" order="AFTER" keyProperty="id"**>  select LAST\_INSERT\_ID()  </**selectKey**>  insert into t\_checkgroup(code,name,sex,helpCode,remark,attention)  values  (#{code},#{name},#{sex},#{helpCode},#{remark},#{attention})  </**insert**>   *<!--设置检查组和检查项多对多关系-->* <**insert id="setCheckGroupAndCheckItem" parameterType="map"**>  insert into t\_checkgroup\_checkitem(checkgroup\_id,checkitem\_id)  values  (#{checkgroupId},#{checkitemId})  </**insert**> </**mapper**> |

#### 1.2.3.6 启动测试



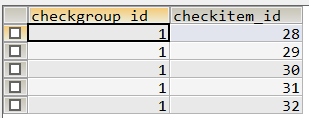


查看数据库：

检查组表(t\_checkgroup):



中间表(t\_checkgroup\_checkitem):



## 1.3 检查组分页

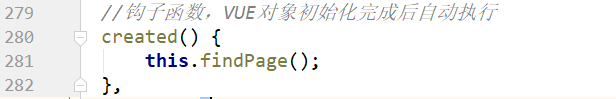
### 1.3.1 完善页面

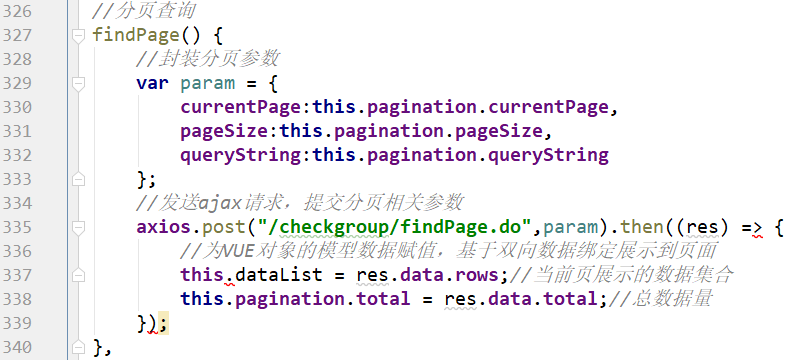
#### 1.3.1.1 定义相关分页模型数据



#### 1.3.1.2 定义分页方法

在页面中提供了findPage方法用于分页查询，为了能够在checkgroup.html页面加载后 直接可以展示分页数据，可以在VUE提供的钩子函数created中调用findPage方法。





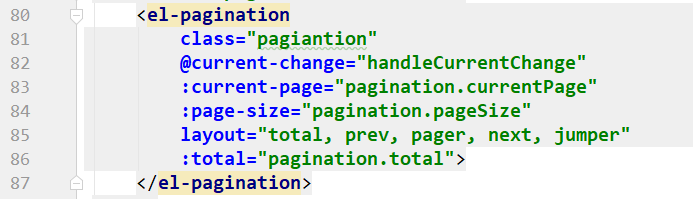
#### 1.3.1.3完善分页方法执行时机

除了在created钩子函数中调用findPage方法查询分页数据之外，当用户点击查询按钮或 者点击分页条中的页码时也需要调用findPage方法重新发起查询请求。

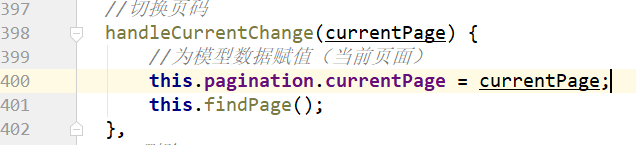
为查询按钮绑定单击事件，调用findPage方法。



为分页条组件绑定current-change事件，此事件是分页条组件自己定义的事件，当页码 改变时触发，对应的处理函数为handleCurrentChange



定义handleCurrentChange方法：



### 1.3.2 后台代码

#### 1.3.2.1 controller

在CheckGroupController中增加分页查询方法

|  |
| --- |
| *//分页查询* @RequestMapping(**"/findPage"**) **public** PageResult findPage(@RequestBody QueryPageBean queryPageBean){  **return checkGroupService**.pageQuery(queryPageBean); } |

#### 1.3.2.2 服务接口

在CheckGroupService服务接口中扩展分页查询方法

|  |
| --- |
| **public interface** CheckGroupService {  **void** add(CheckGroup checkGroup, Integer[] checkitemIds);    **PageResult pageQuery(QueryPageBean queryPageBean);** } |

#### 1.3.2.3服务实现类

在CheckGroupServiceImpl服务实现类中实现分页查询方法，基于Mybatis分页助手插件 实现分页。

|  |
| --- |
| *//分页查询* @Override **public** PageResult pageQuery(QueryPageBean queryPageBean) {  PageHelper.*startPage*(queryPageBean.getCurrentPage(),queryPageBean.getPageSize());  List<CheckGroup> checkGroupList = **checkGroupDao**.findPage(queryPageBean.getQueryString());  PageInfo<CheckGroup> pageInfo = **new** PageInfo<>(checkGroupList);  **return new** PageResult(pageInfo.getTotal(),pageInfo.getList()); } |

#### 1.3.2.4 Dao接口

在CheckGroupDao接口中扩展分页查询方法。

|  |
| --- |
| **public interface** CheckGroupDao {  **void** add(CheckGroup checkGroup);  **void** setCheckGroupAndCheckItem(Map<String, Integer> map);   **Page<CheckGroup> findByCondition(String queryString)**; } |

#### 1.3.2.5 Mapper映射文件

在CheckGroupDao.xml文件中增加SQL定义

|  |
| --- |
| *<!--根据条件进行查询-->* <**select id="findByCondition" parameterType="string" resultType="com.qf.pojo.CheckGroup"**>  select *\** from t\_checkgroup  <**if test="value != null and value != '' and value.length > 0"**>  where code = #{value} or name = #{value} or helpCode = #{value}  </**if**> </**select**> |

#### 1.3.2.6 测试

为了便于观察效果，向检查组表(t\_checkgroup)导入一些数据：

|  |
| --- |
| INSERT INTO `t\_checkgroup` VALUES ('5', '0001', '一般检查', 'YBJC', '0', '一般检查', '无');  INSERT INTO `t\_checkgroup` VALUES ('6', '0002', '视力色觉', 'SLSJ', '0', '视力色觉', NULL);  INSERT INTO `t\_checkgroup` VALUES ('7', '0003', '血常规', 'XCG', '0', '血常规', NULL);  INSERT INTO `t\_checkgroup` VALUES ('8', '0004', '尿常规', 'NCG', '0', '尿常规', NULL);  INSERT INTO `t\_checkgroup` VALUES ('9', '0005', '肝功三项', 'GGSX', '0', '肝功三项', NULL);  INSERT INTO `t\_checkgroup` VALUES ('10', '0006', '肾功三项', 'NGSX', '0', '肾功三项', NULL);  INSERT INTO `t\_checkgroup` VALUES ('11', '0007', '血脂四项', 'XZSX', '0', '血脂四项', NULL);  INSERT INTO `t\_checkgroup` VALUES ('12', '0008', '心肌酶三项', 'XJMSX', '0', '心肌酶三项', NULL);  INSERT INTO `t\_checkgroup` VALUES ('13', '0009', '甲功三项', 'JGSX', '0', '甲功三项', NULL);  INSERT INTO `t\_checkgroup` VALUES ('14', '0010', '子宫附件彩超', 'ZGFJCC', '2', '子宫附件彩超', NULL);  INSERT INTO `t\_checkgroup` VALUES ('15', '0011', '胆红素三项', 'DHSSX', '0', '胆红素三项', NULL); |



## 1.4 编辑检查组

### 1.4.1 完善页面

用户点击编辑按钮时，需要弹出编辑窗口并且将当前记录的数据进行回显，用户修改完 成后点击确定按钮将修改后的数据提交到后台进行数据库操作。此处进行数据回显的时 候，除了需要检查组基本信息的回显之外，还需要回显当前检查组包含的检查项（以复 选框勾选的形式回显）。

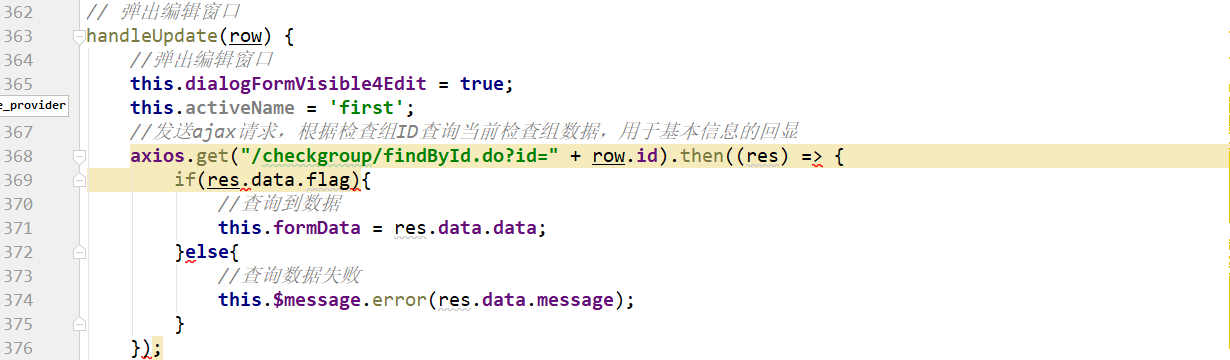
#### 1.4.1.1 绑定单击事件

需要为编辑按钮绑定单击事件，并且将当前行数据作为参数传递给处理函数



#### 1.4.1.2 弹出编辑窗口回显数据

当前页面的编辑窗口已经提供好了，默认处于隐藏状态。在handleUpdate方法中需要将 编辑窗口展示出来，并且需要发送多个ajax请求分别查询当前检查组数据、所有检查项数 据、当前检查组包含的检查项id用于基本数据回显。





#### 1.4.1.3 发送请求

在编辑窗口中修改完成后，点击确定按钮需要提交请求，所以需要为确定按钮绑定事件 并提供处理函数handleEdit。





### 1.4.2 后台代码

#### 1.4.2.1 controller

在CheckGroupController中增加方法

|  |
| --- |
| *//根据ID查询检查组* @RequestMapping(**"/findById"**) **public** Result findById(Integer id){  **try**{  CheckGroup checkGroup = **checkGroupService**.findById(id);  **return new** Result(**true**,MessageConstant.***QUERY\_CHECKGROUP\_SUCCESS***,checkGroup);*//查询成功* }**catch** (Exception e){  e.printStackTrace();  **return new** Result(**false**, MessageConstant.***QUERY\_CHECKGROUP\_FAIL***);*//查询失败* } }  *//根据检查组ID查询检查组包含的多个检查项ID* @RequestMapping(**"/findCheckItemIdsByCheckGroupId"**) **public** Result findCheckItemIdsByCheckGroupId(Integer id){  **try**{  List<Integer> checkitemIds = **checkGroupService**.findCheckItemIdsByCheckGroupId(id);  **return new** Result(**true**,MessageConstant.***QUERY\_CHECKITEM\_SUCCESS***,checkitemIds);*//查询成功* }**catch** (Exception e){  e.printStackTrace();  **return new** Result(**false**, MessageConstant.***QUERY\_CHECKITEM\_FAIL***);*//查询失败* } }  *//编辑检查组* @RequestMapping(**"/edit"**) **public** Result edit(@RequestBody CheckGroup checkGroup,Integer[] checkitemIds){  **try**{  **checkGroupService**.edit(checkGroup,checkitemIds);  }**catch** (Exception e){  e.printStackTrace();  **return new** Result(**false**, MessageConstant.***EDIT\_CHECKGROUP\_FAIL***);*//新增失败* }  **return new** Result(**true**,MessageConstant.***EDIT\_CHECKGROUP\_SUCCESS***);*//新增成功* } |

#### 1.4.2.2 服务接口

在CheckGroupService服务接口中扩展方法。

|  |
| --- |
| **public interface** CheckGroupService {  **void** add(CheckGroup checkGroup, Integer[] checkitemIds);   PageResult pageQuery(QueryPageBean queryPageBean);    **CheckGroup findById(Integer id);   List<Integer> findCheckItemIdsByCheckGroupId(Integer id);   void edit(CheckGroup checkGroup, Integer[] checkitemIds);** } |

#### 1.4.2.3 服务实现类

在CheckGroupServiceImpl实现类中实现编辑方法

|  |
| --- |
| @Override **public** CheckGroup findById(Integer id) {  **return checkGroupDao**.findById(id); }  @Override **public** List<Integer> findCheckItemIdsByCheckGroupId(Integer id) {  **return checkGroupDao**.findCheckItemIdsByCheckGroupId(id); }  @Override **public void** edit(CheckGroup checkGroup, Integer[] checkitemIds) {  *//修改检查组基本信息，操作检查组t\_checkgroup表* **checkGroupDao**.edit(checkGroup);  *//清理当前检查组关联的检查项，操作中间关系表t\_checkgroup\_checkitem表* **checkGroupDao**.deleteAssocication(checkGroup.getId());  *//重新建立当前检查组和检查项的关联关系* Integer checkGroupId = checkGroup.getId();  **this**.setCheckGroupAndCheckItem(checkGroupId,checkitemIds); } |

#### 1.4.2.4 Dao接口

在CheckGroupDao接口中扩展方法

|  |
| --- |
| **public interface** CheckGroupDao {  **void** add(CheckGroup checkGroup);  **void** setCheckGroupAndCheckItem(Map<String, Integer> map);  Page<CheckGroup> findByCondition(String queryString);   **CheckGroup findById(Integer id);  List<Integer> findCheckItemIdsByCheckGroupId(Integer id);  void edit(CheckGroup checkGroup);  void deleteAssocication(Integer id); }** |

#### 1.4.2.5 Mapper映射文件

在CheckGroupDao.xml中扩展SQL语句

|  |
| --- |
| *<!--根据ID查询-->* <**select id="findById" parameterType="int" resultType="com.qf.pojo.CheckGroup"**>  select *\** from t\_checkgroup where id = #{id} </**select**>  *<!--根据检查组ID查询关联的多个检查项ID，查询中间关系表-->* <**select id="findCheckItemIdsByCheckGroupId" parameterType="int" resultType="int"**>  select checkitem\_id from t\_checkgroup\_checkitem where checkgroup\_id = #{id} </**select**>  *<!--根据ID动态修改检查组相关字段-->* <**update id="edit" parameterType="com.qf.pojo.CheckGroup"**>  update t\_checkgroup  <**set**>  <**if test="name != null"**>  name = #{name},  </**if**>  <**if test="sex != null"**>  sex = #{sex},  </**if**>  <**if test="code != null"**>  code = #{code},  </**if**>  <**if test="helpCode != null"**>  helpCode = #{helpCode},  </**if**>  <**if test="attention != null"**>  attention = #{attention},  </**if**>  <**if test="remark != null"**>  remark = #{remark},  </**if**>  </**set**>  where id = #{id} </**update**>  *<!--根据检查组ID清理关联的检查项信息，操作的表t\_checkgroup\_checkitem-->* <**delete id="deleteAssocication" parameterType="int"**>  delete from t\_checkgroup\_checkitem where checkgroup\_id = #{id} </**delete**> |