基于区块链的电子病历共享信息系统

第二迭代测试评估报告

版本<1.0>

修订历史记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **作者** |
| 2018/11/4 | 0.1 | 增加简介和前端页面测试报告 | 蒋海波 |
| 2018/11/4 | 0.2 | 增加java后台接口测试报告 | 张宇燊 |
| 2018/11/4 | 0.3 | 增加获取数据地址的智能合约测试报告 | 朱良 |
| 2018/11/4 | 1.0 | 整理文档 | 蒋海波 |

目录

1. 简介 2

1.1 目的 2

1.2 范围 2

1.3 定义、首字母缩写词和缩略语 2

1.4 参考资料 2

1.5 概述 2

2. 测试结果摘要 2

3. 基于需求的测试覆盖 2

4. 基于代码的测试覆盖 2

5. 建议措施 2

6. 图 2

# 简介

## 目的

本次测试旨在测试基于区块链的电子病历共享信息系统第二迭代实现的功能。

## 范围

本次测试范围为三个方面：前端的基础页面，java后台获取数据库数据的接口，获取数据地址的智能合约。

## 定义、首字母缩写词和缩略语

# 前端页面测试

## 测试用例

|  |  |  |
| --- | --- | --- |
| 用例 | 步骤 | 预期 |
| 1 | 打开页面，输入身份证号，点击查询 | 页面正常显示，能正常输入身份证号，点击查询按钮后得到该病人的所有病历 |
| 2 | 打开页面，输入病名，点击查询 | 页面正常显示，能正常输入病名，点击查询按钮后得到该病名的所有病历 |
| 3 | 打开页面，输入身份证和病名，点击查询 | 页面正常显示，能正常输入身份证号和病名，点击查询按钮后得到该病人该病名的病历 |
| 4 | 打开页面，直接点击查询 | 页面提示请输入参数 |

## 测试执行

|  |  |  |  |
| --- | --- | --- | --- |
| 测试序号 | 执行用例 | 通过 | Bug原因 |
| 1 | 1，2，3，4 | 无 | 没有做mock |
| 2 | 1，2，3，4 | 1，2，3 | 没有正确使用antd的message |
| 3 | 1，2，3，4 | 1，2，3，4 |  |

## 测试结果

最终前端页面功能均已实现并通过测试。其中发现bug数2个，修复完成bug数2个，未修复bug数0个。

## 图



图2-1 用例1结果



图2-1 用例2结果



图2-3 用例3结果



图2-4 用例4结果

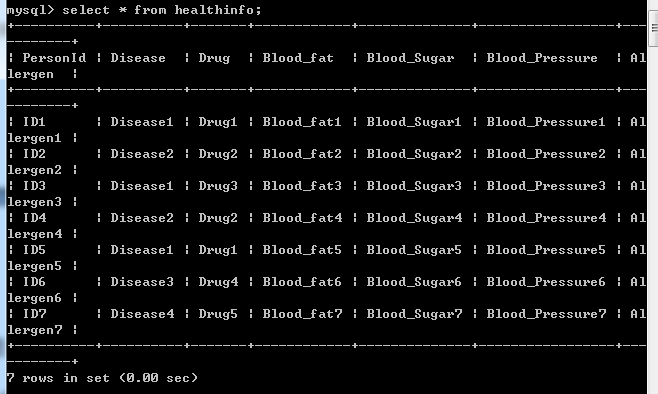
# 后台接口测试

# 3.1 测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| 用例 | 目的 | 步骤 | 预期 |
| 1 | 测试后台getHealthInfo接口 | 在url处输入localhost:8080/BH/list | 返回数据库中healthinfo表中所有数据 |
| 2 | 测试后台getByID接口 | 在url处输入localhost:8080/BH/ getByID?personid=ID6 | 返回数据库中healthinfo表中personid为ID6的数据 |
| 3 | 测试后台getByDisease接口 | 在url处输入localhost:8080/BH/getByDisease?disease=disease1 | 返回数据库中healthinfo表中disease为disease1的数据 |

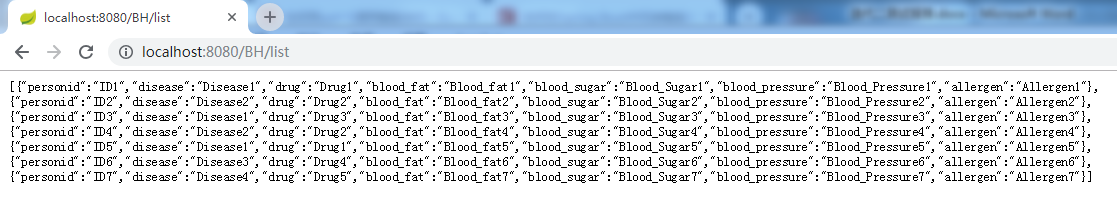
# 3.2测试结果

数据库中数据：



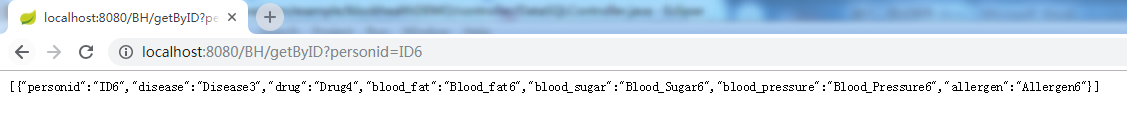
用例一：

页面返回结果：

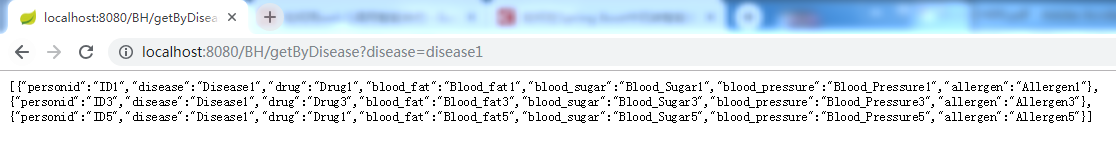


用例二：

页面返回结果：



用例三：



# 3.3测试结果

接口正常工作；目前医疗数据尚未获得，所以上述接口在获得数据后会进行一定的修改。

# 智能合约测试

## 测试结果摘要

本轮测试本别针对基于需求的测试覆盖和基于代码的测试覆盖两方面展开测试工作。

### 基于需求的测试覆盖

|  |  |  |
| --- | --- | --- |
|  | 需求详情 | 测试结果 |
| 功能性 | 接口一：向前端返回所有医疗机构名称。 | 测试通过 |
| 接口二：输入医院名称，输出医疗机构的IP地址。 | 测试通过 |
| 接口三：输入疾病名称，输出与疾病关联的医疗机构名称。 | 测试通过 |
| 接口四：医疗机构信息载入智能合约。输入医疗机构信息，更新所有信息（医疗机构信息、疾病信息、医疗机构信息表、疾病信息表和所有邻接关系）。 | 测试通过 |
| 非功能性 | 易用性：接口便于前端开发人员使用。 | 测试通过 |
| 可靠性：平均可用时间≥94.00%；MTBF≥150天。 | **测试未通过** |
| 性能：平均占用内存≈0.5 GB，占用内存≤8.0 GB。 | **测试未通过** |
| 可支持性：易于扩展接口，内部方法易于维护和调用。 | 测试通过 |

### 基于代码的测试覆盖

|  |  |  |
| --- | --- | --- |
| 测试种类 | 测试方法 | 测试结果 |
| 路径覆盖测试 | DD路径测试 | 通过 |

## 基于需求的测试覆盖

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 需求详情 | 实现方法 | 测试过程 | 结果 |
| 功能性 | 接口一：向前端返回所有医疗机构名称。 | getAllHpstName() | Output:  “h1,h2,” | 功能实现 |
| 接口二：输入医院名称，输出医疗机构的IP地址。 | getHpstIp( string \_Name ) | Input:(“h1”),(“h2”)  Output:  “10.168.0.3”, “12.128.0.2” | 功能实现 |
| 接口三：输入疾病名称，输出与疾病关联的医疗机构名称。 | getHpstFromDss( string \_DssName ) | Input:(“d1”),(“d2”),(“d3”)  Output:  “h1,”, “h1,h2,”, “h2,” | 功能实现 |
| 接口四：医疗机构信息载入智能合约。输入医疗机构信息，更新所有信息（医疗机构信息、疾病信息、医疗机构信息表、疾病信息表和所有邻接关系）。 | loadHsptInfo( string \_Name , string \_Ip , string \_DssList ) | Input:  (“h1”, “10.168.0.3”, “d1,d2,”),  (“h2”, “12.128.0.2”, “d2,d3,”)  Check:  AllHsptList[0],AllHsptList[1],  AllDssList[0],AllDssList[1],  AllDssList[2]  Output:  h1,h2,d1,d2,d3 | 功能实现 |
| 非功能性 | 易用性：接口便于前端开发人员使用。 | 无 | 检查接口的注释是否详细。 | 接口使用说明详细。测试通过。 |
| 可靠性：平均可用时间≥94.00%；MTBF≥150天。 | 无 | 同时开启20个浏览器，运行RemixIDE，每分钟调用所有接口各三次，持续1小时不关闭，记录测试结果。 | 接口运行良好，但受性能限制，17个浏览器在一小时内出现重载情况。**测试未通过。** |
| 性能：平均占用内存≈0.5 GB，占用内存≤8.0 GB。 | 无 | 在5台内存为16GB的设备上，运行RemixIDE各一次，多次调用接口四，向智能合约中载入1000条医院的信息。 | 载入3条医院信息后出现内存过大警告。5台设备全部出现重载情况。内存占用均超过8GB。**测试未通过。** |

## 基于代码的测试覆盖

基于上述需求覆盖测试，通过路径覆盖测试，发现接口四中：

AllDssList[getDssNum(retTmp)].HsptList=strConcat(AllDssList[getDssNum(retTmp)].HsptList,\_Name); AllDssList[getDssNum(retTmp)].HsptList=strConcat(AllDssList[getDssNum(retTmp)].HsptList,",");

在循环中占用了大量的内存。

## 建议措施

针对内存需求过大导致性能不佳的情况，需要进一步优化代码的信息存储机制。

# 总结

本次测试结果为项目第二迭代所需完成的功能均已实现并基本通过测试，部分未通过测试的内容将会在后续迭代中修改和优化。