<牙科医院管理系统>

系统测试报告

版本 <2.0>

修订历史记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **作者** |
| <18/6/2022> | <1.0> | <终版> | <第六组全员> |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

目录

1. 简介 4

1.1 目的 4

1.2 范围 4

1.3 定义、首字母缩写词和缩略语 4

1.4 参考资料 4

1.5 概述 4

2. 测试概要 4

3. 测试环境 4

4. 测试结果及分析 4

4.1 需求覆盖率及缺陷分布 4

4.2 缺陷严重程度 5

5. 缺陷清单 5

5.1 功能性缺陷 5

5.2 非功能性缺陷 6

6. 测试结论与建议 6

系统测试报告

# 简介

## 目的

对项目的完成度、功能的可靠度等等进行检验。帮助开发与运维人员更好的进行之后的维护。

## 范围

限第6小组完成的思源牙科医院管理系统

## 定义、首字母缩写词和缩略语

本系统：指第6小组设计的思源牙科医院管理系统。

## 参考资料

《如何用Junit测试》

《postman使用教程大全》

## 概述

本测试报告包括了测试概要、测试结果、缺陷分布以及结论与建议。

# 测试概要

测试时间：2022.6.15 - 2022.6.18

地点：宿舍

人员：丁佳阳，罗丹宇，游铭桢

方法：junit以及postman

内容：针对各功能进行功能测试、性能测试、压力测试、易用性测试、安全性测试以及界面测试等。

# 测试环境

使用IntelliJ IDEA，junit 搭配Postman进行测试。

电脑：联想Y7000P2020

CPU：Intel core i7-10875H CPU @2.30GHz

内存：16GB

硬盘：512GB

# 测试结果及分析

测试功能点数为4，测试用例数为38，缺陷数为1。

## 需求覆盖率及缺陷分布

**表1 需求覆盖率及缺陷分布**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **测试内容** | | **测试**  **用例数** | **需求**  **覆盖率** | **缺陷数** | **缺陷率** | **备注** |
| **功能项** | **登录** | 6 | 100% | 0 | 0 |  |
| **注册** | 4 | 100% | 0 | 0 |  |
| **挂号** | 10 | 100% | 0 | 0 |  |
| **设置诊疗流程** | 2 | 100% | 0 | 0 |  |
| **搜索医生** | 8 | 100% | 1 | 12.5% |  |
| **功能项小计** | 30 | 100% | 1 | 3.33% |  |
| **非功**  **能项** | **性能** | 5 | 100% | 0 | 0 |  |
| **安全性** | 3 | 100% | 0 | 0 |  |
| **非功能项小计** | *8* | *100%* | *0* | *0* |  |
|  | **总计** | *38* | *100%* | *1* | *2.63%* |  |

## 缺陷严重程度

**表2 缺陷严重程度**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **严重程度** | **致命** | **严重** | **一般** | **细微** | **总数** |
| **缺陷个数** | 0 | 0 | 1 | 0 | 1 |
| **占缺陷百分比** | 0 | 0 | 100% | 0 | 1 |

# 缺陷清单

## 功能性缺陷

**表3 功能性缺陷列表**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序**  **号** | **缺陷**  **编号** | **严重程度** | **功能模块** | **缺陷标题** | **缺陷描述** | **测试用例编号** |
| 1 | 1 | 一般 | 搜索医生 | 搜索重名医生 | 搜索重名医生时出现问题 | testcase-002 |
| 2 |  |  |  |  |  |  |
| 3 |  |  |  |  |  |  |

表中相关项说明：

1. 缺陷编号：为每个缺陷赋予一个唯一的编号，可以通过此编号对缺陷进行跟踪。例如：Bug001。
2. 缺陷严重程度：缺陷可以根据严重程度分为以下几种情况。
3. 致命（fatal）：致命的错误，测试执行直接导致系统死机、蓝屏、挂起、或是程序非法退出；系统的主要功能或需求没有实现。
4. 严重（critical）：严重错误，系统的次要功能点或需求点没有实现；数据丢失或损坏。执行软件主要功能的测试用例导致系统出错，程序无法正常继续执行；程序执行过于缓慢或是占用过大的系统资源。
5. 一般（major）：不太严重的错误，这样的缺陷虽然不影响系统的基本使用，但没有很好地实现功能，没有达到预期的效果。如次要功能丧失，界面错误，打印内容、格式错误，提示信息不太正确，或用户界面太差，简单的输入限制未放在前台进行控制，删除操作未给出提示，操作时间长等。
6. 细微（minor）：一些小问题，对功能几乎没有影响，产品及属性仍可使用。如软件的实际执行过程与需求有较小的差异；程序的提示信息描述容易使用户产生混淆；有个别错别字、 文字排列不整齐；辅助说明描述不清楚，显示格式不规范，长时间操作未给用户进度提示，提示窗口文字未采用行业术语，可输入区域和只读区域没有明显的区分标志，系统处理未优化等。
7. 功能模块：所测试并出现该缺陷的功能模块名称。
8. 缺陷标题：描述缺陷的标题。
9. 缺陷描述：对缺陷的详细描述。描述缺陷的重现步骤和问题所在，描述需清晰明了，使研发人员可以按照描述将缺陷重现出来。必要时可以放上相应的截图，以便于研发人员理解问题所在。

## 非功能性缺陷

未发现严重非功能性缺陷。

**表4 非功能性缺陷列表**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序**  **号** | **缺陷**  **编号** | **严重程度** | **测试类型** | **缺陷标题** | **缺陷描述** | **测试用例编号** |
| 1 | / | / | / | / | / | / |

# 测试结论与建议

测试出现的问题：

1. 测试用例不够优秀，例如在大量并发挂号请求测试时，由于使用的是Postman，其并发程度不是很高，因此测试都能通过，但在更高并发程度的情况下就有可能失败。
2. 考虑的不够全面。例如医生重名时搜索问题是后来才想到要测的。

建议：提前准备需要测试的角度，否则可能最后时间仓促准备的测试用例不周全；寻求他人帮助，从使用者的角度来看我们的系统并针对其设计测试用例。