<项目名称>

测试报告

版本 <1.0>

修订历史记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **作者** |
| 2019/7/9 | 1.0 | 初始化 | 苑齐超 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

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

测试报告

# 简介

## 目的

本测试报告为FIT食物智能识别系统测试，目的在于总结测试阶段的测试以及分析测试结果，描述系统是否符合需求，并依据结果对该产品做出评价和建议。适用范围包括FIT系统的用户、测试人员、开发人员、项目管理其他质量人员和需要阅读本报告的老师助教等。

## 范围

本报告覆盖FIT项目的所有测试，项目架构文档、需求规约、开发进度等都会根据此测试结果，参照此报告进行调整。

## 定义、首字母缩写词和缩略语

FIT：项目组开发系统名称

## 参考资料

《FIT项目软件需求规约-迭代一》

《FIT项目测试用例-迭代一》

## 概述

测试包括前端的手工测试，后端的自动化测试单元测试。其中保证单元测试覆盖率达到100%。

# 测试概要

测试时间：2019/7/10

测试地点：软院楼3101

测试人员：FIT小组

测试方法：前端与前后端调通部分采用手工测试，后端为单元测试

测试内容：前端测试UI设计等，后端与前后端调通部分测试相关功能（登录，注册服务）的实现

# 测试环境

服务器信息：

1. 数据库服务器：

内存：8GB； VCPU数量：4；磁盘：80GB

1. FIT服务器：

内存：16GB；VCPU数量：8；磁盘：160GB

手机端信息：

手机品牌：华为

手机型号：VTR-AL00

Android版本：8.0.0

处理器：Hisilicon Kirin 960

运行内存：4GB

手机储存：可用空间 29.68GB，总容量：128GB

屏幕：1920\*1080

PC端信息：

处理器：Inter i7-4710MQ CPU @ 2.50GHz

RAM：8.00GB（7.89GB可用）

Windows版本：Window 10 家庭中文版

操作系统：Windows10 64位操作系统 基于x64的处理器

测试软件：Intellij IDEA 2019.1 x64

# 测试结果及分析

## 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** | **注册** | 4 | 100% | 0 | 0% |  |
| **登录** | 2 | 100% | 0 | 0% |  |
| **……** |  |  |  |  |  |
| **功能项小计** | 6 | 100% | 0 | 0% |  |
| **非功**  **能项** | **性能** | 0 | 0 | 0 | 0 |  |
| **可靠性** | 0 | 0 | 0 | 0 |  |
| **…** |  |  |  |  |  |
|  | **非功能项小计** | 0 | 0 | 0 | 0% |  |
|  | **总计** | 0 | 0 | 0 | 0% |  |

## 缺陷严重程度

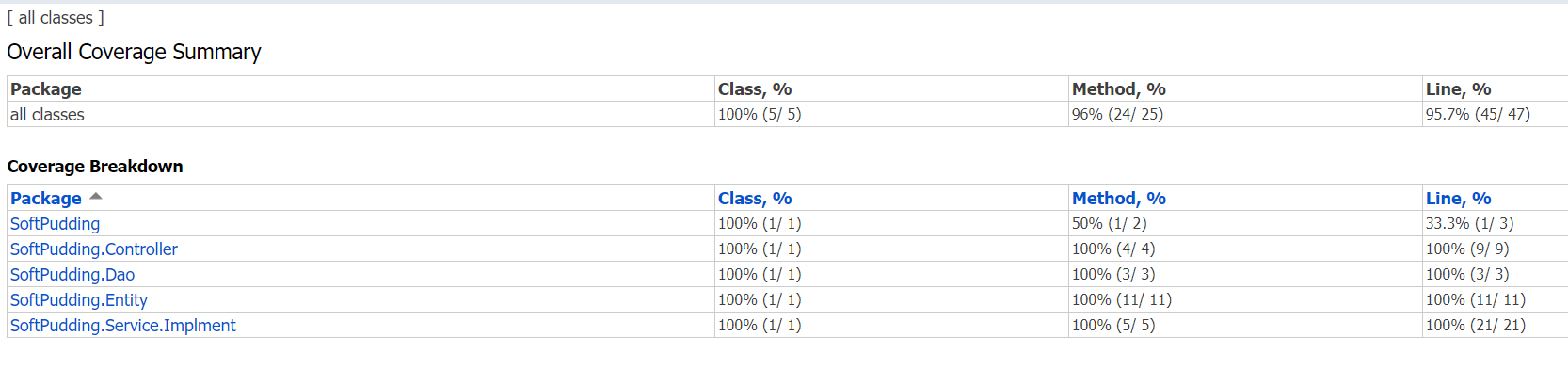
**表2 缺陷严重程度**

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

## 前端界面测试

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **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** | **注册** | 5 | 100% | 0 | 0% |  |
| **登录** | 6 | 100% | 0 | 0% |  |
| **……** |  |  |  |  |  |
| **功能项小计** | 11 | 100% | 0 | 0% |  |
| **非功**  **能项** | **性能** | 0 | 0 | 0 | 0 |  |
| **可靠性** | 0 | 0 | 0 | 0 |  |
| **…** |  |  |  |  |  |
|  | **非功能项小计** | 0 | 0 | 0 | 0% |  |
|  | **总计** | 0 | 0 | 0 | 0% |  |

**表3 单元测试覆盖率**



Tip：覆盖率未达到100%的原因是Application中，Springboot.run函数没有找到方法测试。

## 界面测试

# 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 |  |  |  |  |  |  |
| 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

*[简要阐明非功能缺陷汇总情况，并给出具体非功能缺陷列表。(根据所测系统的实际情况，可选择性地对系统进行必要的非功能性测试，包括系统的性能、可靠性、易用性、安全性、兼容性等。)请参考功能缺陷列表给出相关缺陷清单。]*

**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 |  |  | 性能 |  |  |  |
| 2 |  |  | 性能 |  |  |  |
| … |  |  | … |  |  |  |
| n |  |  | 可靠性 |  |  |  |

# 测试结论与建议

[描述测试过程中所遇到的问题，根据测试中所遇到的问题、对测试结果所进行的分析，给出对本次测试的结论、及可取的建议措施等。]

迭代一的测试结果有以下几点可以提升：

1. 前端测试较多为手工测试，缺少有关的前端测试框架的使用。可以在后续开发中尝试添加相关测试与总结
2. 由于第一次迭代要求功能较少，所以本次测试的功能性测试数量较少，也未能暴露出相关问题。这在以后的迭代计划中会随着所需功能测试增多而解决。
3. 单纯的后端压力测试采用的是POSTMAN串行测试，暂时缺乏并行测试，后续开发可考虑使用Jenkins工具进行并行测试。且得到的时间结果仅为后端运行时间，不足以代表软件整体性能。
4. 在前后端调通后，采用的测试方法主要是测试人员手工填写信息，进行服务的测试，后续应考虑使用系统的集成测试提高测试覆盖率和效率。