Testing Reports

Revision History:

|  |  |  |
| --- | --- | --- |
| Date | Author | Description |
| 2020.10.16 | Zhao Yan | Testing Cases |
| 2020.10.16 | Li Lingling | Testing Plan |
| 2020.10.16 | Xue Jiangying | Testing Plan |
| 2020.10.16 | Shi Kaiyue | Testing Plan |
| 2020.10.16 | Lu Jinghan | Testing Plan |
| 2020.10.23 | Zhao Yan | Sort out modifications and additions |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

目录

[1. Introduction 3](#_Toc7110)

[1.1. Intended Audience and Purpose 3](#_Toc177)

[1.2    How to use the document 3](#_Toc30487)

[2. Testing Cases 3](#_Toc30180)

[2.1. process the X-Ray photograph 3](#_Toc7985)

[2.2. do a cobb angle recognition 4](#_Toc3874)

[2.3. classify the case of scoliosisrecognition 5](#_Toc9461)

[2.4. migrate the model 6](#_Toc30457)

[3. Testing Plan 6](#_Toc14630)

[3.1 Intended Audience and Purpose 6](#_Toc12770)

[3.1.1 Intended Audience and Purpose 6](#_Toc4626)

[3.1.2 Background 7](#_Toc14792)

[3.1.3 Range 7](#_Toc13186)

[3.2 test requirements 7](#_Toc22194)

[3.3 test strategy 8](#_Toc9653)

[3.3.1 Function Test 8](#_Toc9900)

[3.3.2 Performance Evaluation 9](#_Toc2214)

[3.3.3 Strength Test 9](#_Toc2632)

[3.3.4 Configuration Test 10](#_Toc19487)

[3.4 Resources 11](#_Toc14864)

[3.4.1 role 11](#_Toc32462)

[3.4.2 system 12](#_Toc18964)

[3.5 Project milestone 12](#_Toc3117)

[3.6 Deliverables 13](#_Toc18639)

[3.7 Appendix A: Project tasks 13](#_Toc32278)

[3.8 Testing Results 15](#_Toc238)

[3.8.1 Unit testing 15](#_Toc9163)

[3.8.2 The average processing time of each picture after model training 15](#_Toc16052)

[3.8.3 Integration testing 16](#_Toc4715)

[The interface test 16](#_Toc23838)

## Introduction

## Intended Audience and Purpose

This document provides the testing method and results, corresponding to the requirement from the customer. It consists of 3 parts, the testing cases, the test plan, and the testing results.

## 1.2    How to use the document

You may refer to the content section for the structure of the document, in which Sec. Testing Cases collect the unit and module test information from each team; Sec. Testing Plan shows the steps and expected results of the integration test; Sec. Results describes the real world data out of the test, and the correspondence to the requirements.

## Testing Cases

Our group is the algorithm group.In this section, our team propose our testing cases on unit and module testing.

## process the X-Ray photograph

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Project/Software** | the algorithm part of the whole system | **The program version** | 1.0.0 | |
| **Function module** | process the X-Ray photograph | **Prepare the people** | xxx | |
| **Related use cases** | NULL | | | |
| **features** | Verify that the input image is formatted correctly | | | |
| **Test purposes** | Verify whether the input of legal information, allow legal picture information, prevent illegal picture information | | | |
| **Preset conditions** | NULL | **Description of special procedures** | NULL | |
| **The reference information** | Instructions on "processing the X-ray photograph" in the requirements notes | | | |
| **The test data** | a X-Ray image | | | |
| **steps** | **Operation description** | **data** | **Desired outcome** | **The actual results** |
| 1 | Input null, execute algorthm | Empty | The algorithm feeds back the empty information of the picture and terminates the algorithm in advance | feeds back the empty information of the picture |
| 2 | Input image information, perform algorithm | Wrong format picture | The algorithm feeds back the image format error information and terminates the algorithm in advance | feeds back the image format error information |
| 3 | Input image information, perform algorithm | Correct spine picture information | Execute the algorithm correctly and return the digitized image | return the digitized image |
| **The tester** |  | **The developer** |  | **Project Leader** |

## do a cobb angle recognition

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Project/Software** | the algorithm part of the whole system | **The program version** | 1.0.0 | |
| **Function module** | do a cobb angle recognition | **Prepare the people** | xxx | |
| **Related use cases** | NULL | | | |
| **features** | Verify that the pictures and content | | | |
| **Test purposes** | Make sure that the image is digitized and input into the digital image format. At the same time, verify whether the image content is a normal spine image and identify the normal image | | | |
| **Preset conditions** | a digital X-Ray image is available | **Description of special procedures** | NULL | |
| **The reference information** | Instructions on do a cobb angle recognition" in the requirements notes | | | |
| **The test data** | a digital X-Ray image | | | |
| **steps** | **Operation description** | **data** | **Desired outcome** | **The actual results** |
| 1 | Input null, eixecute algorthm | Empty | The algorithm feeds back the empty information of the picture and terminates the algorithm in advance | feeds back the empty information |
| 2 | Input image information, perform algorithm | Other animal spine | The algorithm feeds back the image format error information and terminates the algorithm in advance | feeds back the image format error information |
| 3 | Input image information, perform algorithm | Landscape photo information | The algorithm feeds back the error message of the picture, and can't detect the human bone, so the algorithm | feeds back the error message |
| 4 | Input image information, perform algorithm | digital X-Ray image | Execute the algorithm correctly and return the cobb angle and marked bones from the X-Ray image | return the cobb angle and marked bones from the X-Ray image |
| **The tester** |  | **The developer** |  | **Project Leader** |

## classify the case of scoliosisrecognition

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Project/Software** | the algorithm part of the whole system | **The program version** | 1.0.0 | |
| **Function module** | classify the case of scoliosisrecognition | **Prepare the people** | xxx | |
| **Related use cases** | NULL | | | |
| **features** | Make sure the input Angle is correct | | | |
| **Test purposes** | According to the Angle classification, guarantees the input and the result reliability | | | |
| **Preset conditions** | results of cobb angles is available | **Description of special procedures** | NULL | |
| **The reference information** | Instructions on ”classify the case of scoliosis" in the requirements notes | | | |
| **The test data** | cobb angles | | | |
| **steps** | **Operation description** | **data** | **Desired outcome** | **The actual results** |
| 1 | Input null, eixecute algorthm | Empty | The algorithm feeds back the empty information of the picture and terminates the algorithm in advance | feeds back the empty information |
| 2 | Input angles, perform algorithm | A negative number | The feedback Angle of the algorithm is incorrect, so it ends in advance | feedback Angle of the algorithm is incorrect |
| 3 | Input angles, perform algorithm | Greater than 360 degrees | The feedback Angle of the algorithm is incorrect, so it ends in advance | feedback Angle of the algorithm is incorrect |
| 4 | Input angles, perform algorithm | Input random character | The feedback Angle of the algorithm is incorrect, so it ends in advance | feedback Angle of the algorithm is incorrect |
| 5 | Input angles, perform algorithm | Different angles of correct format | Execute the algorithm correctly and return classification of scolios | return classification of scolios |
| **The tester** |  | **The developer** |  | **Project Leader** |

## migrate the model

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Project/Software** | the algorithm part of the whole system | **The program version** | 1.0.0 | |
| **Function module** | migrate the model | **Prepare the people** | xxx | |
| **Related use cases** | NULL | | | |
| **features** |  | | | |
| **Test purposes** | Verify multiple times to ensure the reliability of the migration system | | | |
| **Preset conditions** | The algorithm module is working and a network model is available | **Description of special procedures** | NULL | |
| **The reference information** | Instructions on ”migrate the model" in the requirements notes | | | |
| **The test data** | NULL | | | |
| **steps** | **Operation description** | **data** | **Desired outcome** | **The actual results** |
| 1 | eixecute algorthm | Empty | extracts the model of this case and output to a file |  |
| **The tester** |  | **The developer** |  | **Project Leader** |

## Testing Plan

Here comes the complete testing plan for integration, referring to the workflows in the system design document.

## Intended Audience and Purpose

## 3.1.1 Intended Audience and Purpose

This document provides the testing method and results, corresponding to the requirement from the server. it helps us to test our completed task and process through feasible means, and obtain practical test results to ensure the quality of our algorithm.

Testing reports of “Algorithm\_Luckydogs” are aimed to achieve targets as follows:

* Determine existing api information corresponding to server and sub software components that should be tested
* List the recommended test requirements
* Recommend and explain the test strategies that can be used
* Determine the required resources and estimate the test workload
* List the deliverable elements of the test project

## Background

In order to better support the reliability and feasibility of the entire scoliosis detection application service, this module aims to provide accurate and effective algorithms to meet the technical requirements of the server, mainly including the following functions and processes:

X-ray image processing -----> Train cone recognition model ------>

------> Calculate cobb angles ------> Classify scoliosis type

|| ||

Output Output

## Range

Two phases will be carried out on our algorithm module: Unit Testing and System Testing. Unit testing deals with functional quality issues, that is, performing functional testing, while system testing deals with feasibility and performance issues, that is, performing performance testing.

The function and performance of the following subsystems will be tested:

|  |  |
| --- | --- |
| Function | 1. X-Ray image formatting and processing 2. Cobb angles calculation 3. Scoliosis classification |
| Performence | 1. Image Resolution 2. Model Accuracy: How are the testing results of cobb angles compared with the doctor’s judgement? 3. Model Migration Ability: How does our model work when recognizing other bones? |

## test requirements

## The following list lists the projects (use cases, functional requirements, non functional requirements) that have been identified as test objectives. The list describes the objects to be tested.

**Functional testing**

* the algorithm process and identify the X-Ray photograph.
* return the cobb angle when the server wants to do a cobb angle recognition.
* classify the case of scoliosis when the server wants to classify the case of scoliosis.
* the algorithm module extracts the model of this case and output to a file when the server wants to migrate the model.

**performance evaluation**

* Verify whether the system response has timed out and whether the operation after the timeout is correct.

**Business cycle testing**

None

**User interface testing**

* Whether the input parameters and return values of the interface are legal
* Test interface stability and error rate

**Load Testing**

None

**Configuration test**

Test whether it is suitable for the server platform

**Access test**

Verify that the algorithm can be called and run normally after use

## test strategy

## 3.3.1 Function Test

Functional test should be able to trace directly to use cases or test requirements of business functions and business rules. The goal of this test is to verify that the data is correctly received, processed, and retrieved, and that business rules are properly implemented. This aspect of the test is based on black box testing, testers only use the UI interface presented by the user evaluation, through the interactive output to verify whether the internal function is perfect.

|  |  |
| --- | --- |
| **Test Goal** | Make sure that the functions of the test object are working properly,include processing the X-Ray photograph,cobb angle recognition,classifying the case of scoliosis. |
| **Method** | Use valid and invalid data to perform the corresponding functions of each use case, and check whether the running status is reasonable and correct according to different situations   * After inputting valid data,the operation should be correct. * After inputting invalid data, it should have some tips and carry on with the following operations. * Each business should be applied correctly. |
| **Standard** | * All the functions should be implemented correctly. * The defect has been solved. |
| **Tip** |  |

## 3.3.2 Performance Evaluation

Performance evaluation measures response time, transaction rate and other time related requirements. The goal is to verify that all performance requirements have been met. In this system, the most important is the response time, which has a direct impact on the efficiency and effect of the whole program.

|  |  |
| --- | --- |
| **Test Goal** | Ensure that the test object can complete the expected workload in the effective time during the execution. |
| **Method** | * Use the test process developed for functional testing. * The test should be run on one computer and repeated on multiple clients. |
| **Standard** | * Complete the corresponding tasks within the expected time. * There are correct prompt and follow-up processing for the time exceeding the response time. |
| **Tips** |  |

## 3.3.3 Strength Test

Strength testing is a kind of performance test. The purpose of implementing and executing this kind of test is to find out the errors caused by insufficient resources or resource contention. In this system, although each user can only access their own part of the data, but because of the transfer processing from user to user, there are many users will operate on the same data or object at the same time.

|  |  |
| --- | --- |
| **Test Goal** | Verify that the test target can run normally without any error when multiple users operate on the same data object at the same time. |
| **Method** | * Use the tests specified for performance evaluation. * Test on the same computer, do the same operation to see the program execution. |
| **Standard** | All tests have been performed, and there are no failures during the execution, or the failures are not within the given conditions. |
| **Tips** | Simultaneous access to the same data account by multiple customers should be synchronized |

## 3.3.4 Configuration Test

Configuration test verifies the performance of the test object in different software and hardware configurations. In most production environments, the specific hardware specifications for client workstations, network connections, and database servers vary. Client workstations may have different software installed - for example, applications, drivers, etc. - and at any time, many different combinations of software may be running, occupying different resources. In order to ensure the stable operation of the system in the given environment, the following tests are needed

|  |  |
| --- | --- |
| **Test Goal** | Verify that the test object can operate properly in the required hardware and software configurations |
| **Method** | * Use functional test scripts. * Before the test starts, open various programs that are not related to the test, and then close them. * Execute the selected transaction to simulate the interaction with test object and fee test object during execution. |
| **Standard** | For various combinations of test objects and non test objects, all transactions can complete normally without failure. |
| **Tips** |  |

## 3.4 Resources

This section lists the resources recommended for testing algorithms and their main responsibilities, knowledge, or skills.

## 3.4.1 role

The following table lists the various assumptions made in terms of staffing for this project.

|  |  |  |
| --- | --- | --- |
| **human resources** | | |
| **role** | **Minimum resources recommended** | **Specific responsibilities and notes** |
| Test Manager | 1 | Carry out management supervision.  responsibilities ：   * Provide technical guidance * Access to appropriate resources * Provide management reports |
| Test Designer | 1 | Determine the test cases, determine the priority of test cases and implement the test cases.  responsibilities ：   * Generate test plan * Generate test model * Evaluate the effectiveness of testing |
| Tester | 1 | Perform the test.  responsibilities ：   * Perform the test * Record the results * Recover from errors |
| Designer | 1 | Define and define the operations, properties, and associations of the test class.  responsibilities ：   * Identify and define test classes * Identify and define test packages |
| Implementer | 1 | Implement test classes and test packages, and unit test.  responsibilities ：   * Create test classes and test packages implemented in the test model |

## 3.4.2 system

The following table lists the system resources required for the test project.

|  |  |
| --- | --- |
| **System resources** | |
| **resources** | **Name/type** |
| Client test PC  Including special configuration requirements | TBD |
|  |
| Test development PC | TBD |

## 3.5 Project milestone

Testing of the algorithm model should include testing activities for the tests described in the sections above.   Separate project milestones should be identified for these tests to inform the status and results of the project.

|  |  |  |  |
| --- | --- | --- | --- |
| **Milestone task** | **The workload** | **Start date** | **End date** |
| Develop a test plan | 2 | 11/10/2020 | 12/10/2020 |
| Design test | 3 | 13/10/2020 | 15/10/2020 |
| The implementation of the test | 4 | 23/10/2020 | 26/10/2020 |
| Perform the test | 3 | 26/10/2020 | 28/10/2020 |
| Evaluate the tests | 2 | 29/10/2020 | 30/10/2020 |

## 3.6 Deliverables

**Test plan:**

Written in Microsoft Word, including the overall design of the test. By the project manager.

**Test model:**

For each test to be executed, a test results form is created. This includes the name or ID of the test, the relevant use case or supplementary specification of the test, the test date, the tester ID, the required test prerequisites, and the test results.

**Test log:**

Microsoft Word will be used to record and report test results.

**Defect report:**

The project home page on the Web is used to document defects.

## 3.7 Appendix A: Project tasks

Here are some test-related tasks:

Develop a test plan:

* Identify test requirements
* To assess risk
* Develop a test strategy
* Identify test resources
* Create schedule
* Generate test plans

Design test:

* Prepare workload analysis documents
* Identify and describe test cases
* Determine the test process and establish the structure of the test process
* Review and evaluate test coverage

The implementation of the test

* Record or programmatically create test scripts
* Identify test specific functions in the design and implementation model
* Create external data sets

Perform the test

* Execute the test procedure
* Evaluate the execution of the tests
* Resume suspended tests
* To verify the results
* Unexpected findings
* Record the defect

Evaluate the tests

* Evaluate test case coverage
* Assess code coverage
* Analyze the defect
* Determine if test completion and success criteria have been met

## 3.8 Testing Results

The results of the integration are listed here and you may find the correspondence to the requirements in the requirement analysist document.

## 3.8.1 **Unit testing**

The unit test results are in the third page of the test case.

## 3.8.2 **The average processing time of each picture after model training**

|  |  |  |
| --- | --- | --- |
| Number of images processed | Total processing time | Average processing time |
| 30 | 365.13(s) | 12.17(s) |
| 30 | 391.20(s) | 13.01(s) |
| 30 | 410.43(s) | 14.01(s) |
| 40 | 522.87(s) | 13.07(s) |
| 40 | 485.22(s) | 12.13(s) |
| 40 | 488.12(s) | 12.20(s) |
| 50 | 711.56(s) | 14.23(s) |
| 50 | 650.78(s) | 13.02(s) |
| 50 | 668.23(s) | 13.36(s) |
| Average processing time | | 13.04(s) |

## 3.8.3 **Integration testing**

## **The interface test**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case** | **Input** | **Excepted Output** | **The Real Result** |
| Receive the image from the server | PNG type image | Picture format is correct | Picture format is correct |
| Return result to server | Null | cobb = (cobb\_value, int(ry[i]), int(rx[i]), int(ry[i+1]), int(rx[i+1]), p0, p1),lenkeRes | cobb = (cobb\_value, int(ry[i]), int(rx[i]), int(ry[i+1]), int(rx[i+1]), p0, p1),lenkeRes |

**Concurrency test**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case** | **Module** | **Result** | **Corresponding Requirement** |
| Enter the test picture and get the results | algorithm | Mean corresponding time 13.04(s) | When more than one user requests an algorithm service simultaneously |

**Accuracy test**

|  |  |  |  |
| --- | --- | --- | --- |
| **Number of test images** | **Maximum error(degree)** | **Minimum error(degree)** | **Mean error**  **(degree)** |
| 20 | 10 | 2 | 7.5 |
| 30 | 11.5 | 2 | 7.3 |
| 40 | 12 | 2 | 8 |
| 50 | 12 | 2 | 7.6 |