**Test plan of Smart campus small assistant**

**Table of Contents**

Table of Contents

[1 General overview 2](#_Toc136433617)

[1.1 Project Background 2](#_Toc136433618)

[1.2 Test Purpose 2](#_Toc136433619)

[1.3 Test Scope 2](#_Toc136433620)

[1.4 Testing the Process 3](#_Toc136433621)

[1.4.1 Testing procedure 3](#_Toc136433622)

[1.4.2 nullTest phase Division 3](#_Toc136433623)

[1.4.3 nullTest Implementation Process 4](#_Toc136433624)

[1.4.4 nullReview of Test Methods 4](#_Toc136433625)

[1.4.2 Test process 4](#_Toc136433626)

[2 Test the strategy 5](#_Toc136433627)

[2.1 Functions of the test 5](#_Toc136433628)

[1.1.1 Main content of the test 5](#_Toc136433629)

[1.1.1 Test feature point list 6](#_Toc136433630)

[2.3 Test Methods 8](#_Toc136433631)

[2.4 Test Environment 8](#_Toc136433632)

[2.5 Testing Tools 8](#_Toc136433633)

[3. Schedule 8](#_Toc136433634)

[3.1 Test task division 9](#_Toc136433635)

[3.2 Schedule 10](#_Toc136433636)

[4. Risk 10](#_Toc136433637)

[1.2 Risk Analysis 10](#_Toc136433638)

[5. Test cases 11](#_Toc136433639)

[5.1 Test Case Design 11](#_Toc136433640)

[5.2 Specific test cases 11](#_Toc136433641)

1 In general

## Project background

In campus life, problems such as waiting in the cafeteria, occupying seats in the library, and having no free space in the basketball court often bother us and affect our time arrangement.By obtaining the number of people in these venues in real time, we can arrange our time plan more rationally.

Through the wechat mini program, we can query the real-time number of people in crowded places on campus in real time, and decide the appropriate venues to go to according to the number of people.

## 1.2 Test purpose

Smart Campus small assistant wechat mini program is a wechat mini program used to count the number of people in school libraries, playgrounds and canteens on mobile devices.The test aims to evaluate its functionality and robustness to ensure that it can correctly complete pedestrian traffic statistics in a variety of scenarios and provide a user-friendly experience.

## Scope of testing

This test plan will test the following aspects:

1. Image input function
2. Whether the number of people detected is accurate
3. Algorithm accuracy
4. Robustness
5. Performance

## 1.4 Testing Progress

### 1.4.1 Testing process



### Test phase division

In this project, we divided the whole testing process into several testing phases, and only after reaching one testing phase can we transfer to the next phase to control the whole process.

We divided the entire testing process into the following stages:

|  |  |
| --- | --- |
| Testing phase | **Completion Criteria** |
| System training: | 1. Complete training for all systems to be tested in this project 2. The tester has used all the systems/modules under test and understood the specific functions of the system under test |
| Test requirements: | 1. All specific test areas have been identified 2. Test requirements have been developed 3. All test requirements are approved by the customer |
| Test design: | 1. Test cases have covered all test requirements 2. The test case design is complete |
| Test execution: | 1. All test cases are executed 2. Defects found have defect records 3. Test procedures have test records |
| Analysis of results: | 1. Complete the test analysis report |

### Test implementation process

In this project, a tester is responsible for testing different sub-functions. The implementation process is as follows:

1. 4. Prepare the required environment for testing
2. Prepare the data required for the test
3. Execute test cases according to the system operation structure
4. Document the test process and any defects found
5. Reporting defects

### A review of test methods

Tests in this project include:

1. Functional test: test whether there are defects in each function
2. Performance test: Test the performance data of the system under a certain environment
3. When the tester performs the test, it should strictly follow the content of the test case to perform the test work.
4. The tester records the test execution in the test execution log document.
5. The tester records any problems found during the test into the defect record.
6. Test organization

This chapter mainly describes the structure and responsibilities of the test team, the functional division of test participants, and their contact information

2. Test strategy

## 2.1 Test function points

| **Test items** | **Importance** | **Passing criteria** |
| --- | --- | --- |
| Check the number of users | high | Test the accuracy, sensitivity and response speed of the population detection algorithm to ensure that the algorithm can normally detect the number of people information and update the population data in real time. |
| Data storage and processing | high | Test the stability and reliability of data storage and processing to ensure that population data can be stored and processed accurately and that data can be acquired and updated in real time. |
| Smart recommendation | Medium | Test the accuracy of the intelligent recommendation algorithm and the relevancy of the recommendation results to ensure that the recommended products, activities and services are highly attractive to users. |

## 2.2 Testing Methods

The test will be conducted using the method of automated testing.The automated test evaluates the user interface and interactive functions of the Smart Campus small Assistant wechat mini program as well as the functions and performance of the smart Campus small assistant wechat mini program.

The automated test will use tools such as Appium.The test cases will be written according to the function points and requirements of the wechat mini program of the Smart Campus small Assistant, and will be run on actual devices.

## 2.3 Test environment

### 2.3.1 Hardware Test Environment

|  |  |
| --- | --- |
| **CPU** | 12th Gen Intel(R) Core(TM) i5-12500H 2.50GHz |
| **Operating System** | Windows 11 |
| **Memory** | 16GB |
| **Hard drive** | 512GB |
| **Video card** | NVIDIA GeForce  RTE 2050 |

### 2.3.2 Software Test Environment

|  |  |
| --- | --- |
| **Software requirements** | **Purpose** |
| Pycharm | Write back-end code that identifies the number of people algorithm |
| Wechat developer tools | Design the front-end wechat mini program interface of intelligent campus small assistant |

## 2.4 Testing Tools

|  |  |
| --- | --- |
| **Testing Tools** | **Uses** |
| Appium | Automatic test of UI for wechat mini program |
| PyCharm | Write test scripts using PyCharm |

1. Schedule

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Personnel** | **Test tasks** | **Workdays** | **Start time** | **End time** |
| Chen Xinyi, Tang Ah Qin, Zhang Hongwei, Chan Mei Han | 1. Determine test objectives 2. Determine test scope 3. Determine testing resources and schedule | 1 | May 26, 2023 | May 26, 2023 |
| Tang Yachen | Writing a test plan | 1 | May 26, 2023 | May 26, 2023 |
| Chen Xinyi | Review various test documents written by testers | 1 | May 26, 2023 | May 26, 2023 |
| Chen Xinyi, Tang Yaqin, Zhang Hongwei and Chan Mei Han | Identifying test points | 1 | May 26, 2023 | May 26, 2023 |
| Tang Yachen | Design test cases | 1 | May 27, 2023 | 27 May 2023 |
| Tang Yachen | Test environment setup and data preparation | 1 | May 28, 2023 | May 28, 2023 |
| Tang Yachen | Designing test scripts | 1 | May 29, 2023 | 29 May 2023 |
| Tang Yachen | Perform functional and performance tests | 1 | May 30, 2023 | May 30, 2023 |
| Tang Yachen | Writing test reports | 1 | May 30, 2023 | May 30, 2023 |

1. Risk analysis

|  |  |  |
| --- | --- | --- |
| **Project risk** | **Risk analysis** | **Avoidance methods** |
| **Failed tests** | If the submitted test version is developed and the smoke test is not passed, the test version must be called back and resubmitted, which may cause a delay in the schedule and affect the subsequent test work arrangement. | Development should take the time to test itself before submitting a test version, or schedule for unit and integration tests if they are not being performed.If the due version is called back, in order not to affect the progress of the overall plan, developers need to arrange for additional staff or overtime. |
| **Delays in development schedule** | If the development cannot be completed within the time specified in the plan, it will cause the delay in the arrangement of subsequent test work, thus causing the risk of the execution of the test plan. | Please ask the project manager to strictly control the progress during the project. If there is any risk of delay, please inform the tester immediately and negotiate to solve the problem.If the plan cannot be completed on time, the tester shall apply for modification of the test plan or apply for overtime work and inform the relevant leader. |
| **Difficult-to-fix permissions cause test cases to stall** | If bugs that are difficult to be fixed are found in the test version during the test execution, the function of the tested module will be blocked and the test schedule will be affected. | If such a problem occurs, the developer needs to fully cooperate with the test and modify the problem in time.If not fully fixed, provide the test with an interface to test the blocked module or a way to bypass it. |

# 5. Test cases

## 5.1 Test case design

The test cases of this test are organized and written by the tester in accordance with the system hierarchy according to the customer's introduction to the system and their own understanding of the system after system training.

The writing of this system case uses the analysis method commonly used in black box testing to design use cases;

1. For each test case, the test designer should specify the input (or operation), the expected output (or result);
2. For each test case, there must be a detailed description of the test steps;
3. All test cases of this test design must be saved in a standardized document;
4. During the whole test process, the test cases can be changed according to the actual situation of the project;
5. The preparation of test data in test cases, under the guidance of customers and business personnel and the assistance of developers.
6. Arrange the execution of use cases according to the operational structure of the system;

## Specific test cases

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test case No** | **Function points** | **Preconditions** | **Performing steps** | **Expected results** | **Purpose of the use case** | **Degree of importance** | **Execute use case test results** |
| 1 | Number of people tested | The user goes to the applet page | Enter the playground test image on an overcast day | The number of people included in the test image should be the same as the number returned by the applet | Test the accuracy of the number recognition algorithm | high | Through |
| 2 | Number of people tested | The user goes to the applet page | Enter the playground test image on an night day | The number of people included in the test image should be the same as the number returned by the applet | Test the accuracy of the number recognition algorithm | high | Through |
| 3 | Smart Recommendations | The user goes to the applet page | Click on [recommendations] to view recommendations | Return to the venue with the least foot traffic currently in the recommendation | Test if the applet can recommend places based on foot traffic | Medium | Don't pass |
| 4 | Smart Recommendations | The user goes to the applet page | Click on [recommendations] to view recommendations | Return the venue with the most historical foot traffic in the recommendation | Test whether the applet can recommend people to their favorite places based on historical data | Medium | Don't pass |