# Test plan

| | | | | |
|---|---|---|---|---|
| Document identification: | | | Current version: | 1.0 |
| current state: | rough draft | | date of issue: | |
| | publish | ✓ | | |

| Revise the history | | | | | |
|---|---|---|---|---|---|
| date | edition | author | revise content | Review number | Change control number |
| 2023/05/25 | 1.0 | Si-min wang | | | |
| | | | | | |

| | | | |
|---|---|---|---|
| maker: | Wang Simin | date: | 2023/05/25 |
| auditor: | Qu Ruyun | date: | 2023/05/26 |
| approver: | Xu Mengmeng | date: | 2023/05/27 |

# catalogue

# 1 Introduction

## 1.1 Writing purpose

The objectives of this test plan are as follows:

1. Check whether the mobile phone software system meets the function / performance requirements of En chantedMovie software specification, En chantedMovie UI Spec and En chantedMovie product description.

2. The testers in the test group shall start the test work after the project starts, such as writing the software system test plan, the software system test cases (including the function and performance of the mobile phone software, the stress test, etc.), and the construction of the software test environment, etc. According to the functional and performance requirements defined in the En chantedMovie software requirements specification, the En chantedMovie software system test case is written by the En chantedMovie UI Spec, En chantedMovie Product feature description.

3. In the actual operation (use) environment, test the software system according to the software system test plan and the software system test case passed by the review, and form the software system test record and test log.

4. Organize and evaluate the software system test record, TestLog and other relevant information based on the result data of the test record, and form the software system test report (weekly report, milestone report, summary report).

## 1.2 Project background

The procedures of manual ticketing are complicated and inefficient, which brings a lot of inconvenience to the management personnel with a strong sense of time. The cinema lacks a set of perfect ticketing system software. In order to facilitate the management of ticket sales, it is necessary to develop a system that can conduct ticketing in the cinema. And on this basis, you can watch some movies that have been shown and are not temporarily sold on the software, so that users do not need to download another software to watch movies and movies.

With the continuous application and improvement of computer technology, the computer has been deep into every corner of social life. And the use of manual ticketing method, not only low efficiency, easy to make mistakes, cumbersome procedures, but also cost a lot of manpower. In order to meet the efficient management of ticket sales, ticket booking, ticket refund, on the premise that the staff have certain computer operation ability, this system software is specially compiled to improve the management efficiency of the cinema and users' viewing experience.

The relationship between this system and the other systems is as follows:

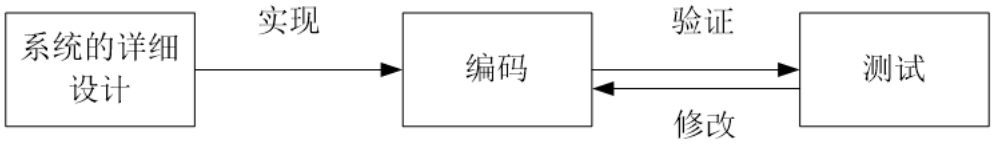


Figure 1 System relationship

## 1.3 Range

This test mainly adopts the method of black box test, which is mainly aimed at the functional test module of the system. For the performance test, load test, safety test and other aspects of the test, the corresponding test will be given according to the time and progress.

## 1.4 Definition

black-box testing:

The black box test, also known as the function test, is used to test whether each function can be used properly. In the test, the program is regarded as a black box that cannot be opened. In the absence of the internal structure and characteristics of the program, it only checks whether the program function is normally used in accordance with the requirement specification, and whether the program can properly receive the input data and produce the correct output information. The black box test focuses on the external structure of the program, excluding the internal logical structure, mainly for the software interface and software functions.

The black box test is conducted from the user's point of view to the corresponding relationship between the input data and the output data. Obviously, if there is a problem with the external feature itself or the specification is wrong, the black box test method will not be found.

white box testing:

White box test is also known as structural test or logical drive test, it is according to the structural test procedure of the program, through the test to detect whether the internal action of the product in accordance with the provisions of the design specification, whether each channel in the test procedure can work correctly according to the predetermined requirements. This method regards the test object as an open box. When the tester design or select test cases to test all the information about the internal logical structure of the program. By checking the status of the program at different points, they determine whether the actual state is consistent with the expected state.

## 1.5 Reference materials

[1] Software Engineering, Zhang Haipeng, People's Posts and Telecommunications Press.

[2] Liu Bing, software Engineering Practice Course, Mechanical Engineering publication.

[3] En chantedMovie System Requirements Analysis Description.

[4] The Feasibility Analysis Manual of the En chantedMovie System.

[5] En chantedMovie System Summary and Detailed Design Specification.

# 2 Test progress

The test schedule time table is as follows:

| Test activities | Plan the start date | Actual start date | deadline |
|---|---|---|---|
| Make a test plan | 2023/05/04 | 2023/05/04 | 2023/05/05 |
| Design test | 2023/05/05 | 2023/05/06 | 2023/05/10 |
| interface testing | 2023/05/09 | 2023/05/10 | 2023/05/19 |
| ST | 2023/05/18 | 2023/05/20 | 2023/05/21 |
| performance testing | 2023/05/20 | 2023/05/21 | 2023/05/22 |
| The test was evaluated | 2023/05/23 | 2023/05/23 | 2023/05/24 |
| product release | 2023/05/25 | 2023/05/25 | 2023/05/25 |

# 3 Test resources

## 3.1 Human resources

### 3.1.1 Personnel arrangement

| role | personnel | Specific responsibilities or notes |
|---|---|---|
| Test manager | Si-min wang | Conduct management and supervision.<br>Responsibilities: Provide technical guidance, obtain appropriate resources, and provide management reports |
| Test designer | Si-min wang | Determine test cases, prioritize test cases, and implement test cases.<br>Responsibilities: Generate the test plan, generate the test models, and evaluate the effectiveness of the test work |
| test controler | Si-min wang | Execute the test.<br>Responsibilities: perform tests, record results, recover from errors, and record change requests |
| Test the system administrator | Qu Ruyun | Ensure that the test environment and assets are managed and maintained.<br>Responsibilities: manage the test system, grant and manage role access to the test system |
| data base administrator | Qu Ruyun | Ensure that the test data (database) environment and assets are managed and maintained.<br>Responsibilities: Manage the test data (database) |

| | | |
|---|---|---|
| data base practitioner | Qu Ruyun | Identifies and defines the actions, attributes, and associations for a test class.<br>Responsibilities: Identify and define test classes, determine and define test packages |
| Database implementer | Xu Mengmeng | Implement test classes and test packages and test them in unit units.<br>Responsibilities: Create the test classes and test packages implemented in the test model |

## 3.1.2 Training

| S N | Training content | participant | Training lecturer |
|---|---|---|---|
| 1 | The Software Test Specification | Software test group personnel | Si-min wang |
| 2 | 《software testing plan》 | Software test group personnel | Qu Ruyun |
| 3 | The Software Test Report, | Software test group personnel | Si-min wang |

## 3.1.3 Division of responsibilities

**The Project Test Leader:**

Responsible for tracking and managing the testing of the project and ensuring that the product quality of the project without serious errors. Responsible for writing En chantedMovie test plan, En chantedMovie test case, writing En chantedMovie test case, En chantedMovie test record, software test report, capturing software test Log (as required), verifying Bug on Clearquest on each officially released release version, and close after confirmation. For the intermediate version or pretest version (when preparing for draw) version, organize and combine the test results with the results submitted by the test engineer, and submit them to the project manager after confirmation. Communicate effectively with the project manager, test manager and test engineer, and coordinate with the relevant staff. Be responsible for testing the project, and cooperate when the project manager.

**Quality Testing Engineer:**

Responsible for performing, performing tests, and recording test results. Cooperate with the test work of the project test leader, perform the test according to the En chantedMovie test plan, En chantedMovie test case and En chantedMovie test case, and submit the test recorded results and test report directly to the project test leader. For the officially released release version test, also executed according to the XX system test case, the specific part of the test is arranged by the project test leader. The discovered Bug shall be submitted to the Git hub after confirmation by the project test leader, and the test record results shall be submitted directly to the project test leader. When working overtime as required by the project manager, try to cooperate.

## 3.2 Test environment

software environment:

The system is tested using black-box testing methods in the environments of Windows 10 and macOS Ventura, utilizing automated testing tools.

hardware environment:

| system | processor | internal storage |
|---|---|---|
| Windows 10 Home Chinese version | 2.80 GHz | 16.0 GB |
| Windows10 Home Chinese version | 2.90GHz | 8.00GB |
| macOS Ventura 13.3.1 | Apple M1Max | 32GB |

## 3.3 Test tools

Django Provide the unit test.

This class and the Python standard library of the unittest. TestCase Similarly, it only expands the following functions:

A client property is provided, and this client is an instance of the Client. Think of Client as a function library that initiates HTTP requests (similar to requests), so that we can easily use this class to test view functions.

Create the database automatically before running the test, and automatically destroy the database after the test run. We certainly do not want the automatically generated test data to affect the real data.

django Unit testing applied include:

Test whether the model, model's method returns the expected data and operates on the database correctly.

Test the form, if the data verification logic meets the expectations.

Test whether the view has returned the expected response for a specific type of request.

Some other auxiliary methods or classes, etc.

# 4. Test strategy

## 4.1 Integrity testing of the data and of the database

| | |
|---|---|
| test object | Ensure that the database installation is successful, access process is not errors, query, add, modify and other database operations can run normally, data will not be damaged |
| technology | Call individual database access methods and processes and populwith valid and invalid data (or requests for data).<br>Check the database to ensure that the data is populated as expected and that all data additions have occurred normally; or check the returned data, to ensure that the correct data is retrieved for valid reasons. |
| completion criteria | The database was successfully installed, the database operation was normal, and the data will not be damaged |
| Test priorities and priorities | The installation of the database and the operation of the database can be normal, because the data directly affects the progress of almost all operations, so its priority is high |
| Special matters to consider | Small or minimal databases (limited number of records) should be used to provide greater visibility of all unacceptable events. |

## 4.2 Interface test

### 4.2.1 User module test

#### 4.2.1.1 Registration function

Plan the form

| Field name | description |
|---|---|
| Test items | Registration function |
| designer | Si-min wang |
| Test environment requirements | You can connect to the server normally<br>Software: IE 7.0 browser and above |
| test method | Unit testing was performed using an automated test framework |
| Enter the instructions | Send a POST request with the following parameters: |

| | |
|---|---|
| | User _ name: The user name of the registered user<br>User _ password: The password of the registered user |
| Output standard | Verify that the HTTP response status code is 200<br>Verify that the response meets expectations |

test case

| number | test case | import | expected result |
|---|---|---|---|
| 1 | Test registration was successful | user_name: "TestUser"<br>user_password: "TestPassword123" | The response status code is 200<br>The response content is "register success"<br>Check that a user object with the appropriate user name and password is created in the database |
| 2 | Test the repeat user name | User _ name: "TestUser" (duplicate with existing users)<br>user_password: "TestPassword123" | The response status code is 200<br>The response content is "User name is already exist" |

### 4.2.1.2 Login function

Plan the form

| Field name | description |
|---|---|
| Test items | Login function |
| designer | Si-min wang |
| Test environment requirements | You can connect to the server normally<br>Software: IE 7.0 browser and above |
| test method | Unit testing was performed using an automated test framework |
| Enter the instructions | Send a POST request with the following parameters:<br>User _ name: The user name of the test user<br>User _ password: Test the user's password<br>User _ loginState: Test the login status of the user |
| Output standard: | Verify that the HTTP response status code is |

| | |
|---|---|
| | 200<br>Verify that the response content contains the expected prompt message |

test case

| number | test case | import | expected result |
|---|---|---|---|
| 1 | The test login was successful | Enter the correct user name, password, and login status. | The response status code is 200, and the response content contains the "login success" prompt message |
| 2 | Test password error | Enter the correct user name, but the incorrect password and login status | The response status code is 200, and the response content contains the "User Password Error" prompt message. |
| 3 | Test user does not exist | Enter a non-existing user name, arbitrary password, and login status | The response status code is 200, and the response content contains the "User Account does not exist" prompt message. |

### 4.2.1.3 Exit function

Plan the form

| Field name | description |
|---|---|
| Test items | Exit function |
| designer | Si-min wang |
| Test environment requirements | You can connect to the server normally<br>Software: IE 7.0 browser and above |
| test method | Unit testing was performed using an automated test framework |
| Enter the instructions | Send a POST request with the following parameters:<br>User _ name: the user name of the user<br>User _ loginState: login status of the user (for example, "inactive" indicates exit status) |
| Output standard: | Verify that the HTTP response status code is 200<br>Verify that the response meets expectations |

test case

| number | test case | import | expected result |
|---|---|---|---|
| 1 | Test exit was successful | user_name: "TestUser"<br>user_loginState: "inactive" | The response status code is 200 |

| | | | |
|---|---|---|---|
| | | | The response content is "Logout success" |
| 2 | Test exit failed (user does not exist) | User _ name: "NonexistentUser" (a user name that does not exist)<br>user_loginState: "inactive" | The response status code is 200<br>The response content is "Logout wrong" |

### 4.2.1.4 Add the user information function

Plan the form

| Field name | description |
|---|---|
| Test items | Add the user information function |
| designer | Si-min wang |
| Test environment requirements | You can connect to the server normally<br>Software: IE 7.0 browser and above |
| test method | Unit testing was performed using an automated test framework |
| Enter the instructions | Send a POST request with the following parameters:<br>User _ name: the user name of the user<br>User _ email: a user's email<br>User _ tel: the user's phone number<br>User _ sex: the gender of the user<br>User _ old: the age of the user<br>User _ avatar: user's profile file (optional, pass file path or set to None) |
| Output standard: | Verify that the HTTP response status code is 200<br>Verify that the response meets expectations |

test case

| number | test case | import | expected result |
|---|---|---|---|
| 1 | The test successfully added the user information | user_name: "TestUser"<br>user_email: "test@test.com"<br>user_tel: "12345678901"<br>user_sex: "male"<br>user_old: "30"<br>User _ avatar: "" (set to empty) | The response status code is 200<br>The response content is "User Information Success" |
| 2 | Test failed to add user information (user does | User _ name: "NonexistentUser" (a user | The response status code is 200 |

| | | | |
|---|---|---|---|
| | not exist) | name that does not exist)<br>user_email: "test@test.com"<br>user_tel: "12345678901"<br>user_sex: "male"<br>user_old: "30"<br>User _ avatar: "" (set to empty) | The response content is "User Information Error" |

## 4.2.2 Movie module test

### 4.2.2.1 Train movie related information

Plan form

| Field name | description |
|---|---|
| Test items | Get the movie-related information function |
| designer | Qu Ruyun |
| Test environment requirements | You can connect to the server normally<br>Software: IE 7.0 browser and above |
| test method | Unit testing was performed using an automated test framework |
| Enter the instructions | Send a GET request to get the basic movie information |
| Output standard: | Verify that the HTTP response status code is 200 (status.HTTP_200_OK)<br>Verify that the movie information returned is as expected |

test case

| number | test case | import | expected result |
|---|---|---|---|
| 1 | Test to movie basic information successfully | not have | Response status code is 200 (status.HTTP_200_OK)<br>Verify that the movie information returned is as expected |

### 4.2.2.2 Visualization function of movie category information

Plan form

| Field name | description |
| --- | --- |
| Test items | Movie category information visualization function |
| designer | Qu Ruyun |
| Test environment requirements | You can connect to the server normally<br>Software: IE 7.0 browser and above |
| test method | Unit testing was performed using an automated test framework |
| Enter the instructions | Send a GET request to get the movie category information |
| Output standard: | Verify that the HTTP response status code is 200 (status.HTTP_200_OK)<br>Verify that the movie category information is returned as expected |

test case

| number | test case | import | expected result |
| --- | --- | --- | --- |
| 1 | Test to get the movie category information successfully | not have | Response status code is 200 (status.HTTP_200_OK)<br>Verify that the movie category information is returned as expected |

### 4.2.2.3 Users shall add the movie information function

Plan form

| Field name | description |
| --- | --- |
| Test items | User adds the movie information |
| designer | Qu Ruyun |
| Test environment requirements | You can connect to the server normally<br>Software: IE 7.0 browser and above |
| test method | Unit testing was performed using an automated test framework |
| Enter the instructions | Send a POST request to add a movie review |
| Output standard: | Verify that the HTTP response status code is 200 (status.HTTP_200_OK)<br>Verify that the movie reviews were successfully added to the movie and to the |

| | |
|---|---|
| | users |

test case

| number | test case | import | expected result |
|---|---|---|---|
| 1 | Test add movie reviews success | Movie _ id: Test Movie movie_id<br>Movie _ comment _ movieName: The movie name of the test movie<br>Movie _ comment _ userName: The user name of the test user<br>Movie _ comment _ userAvatar: The user's avatar URL<br>Movie _ comment _ content: Movie review content | Response status code is 200 (status.HTTP_200_OK)<br>Verify that the movie reviews were successfully added to the movie and to the users |

## 4.2.3 Movie playback function

Plan form

| Field name | description |
|---|---|
| Test items | Movie play function |
| designer | Qu Ruyun |
| Test environment requirements | You can connect to the server normally<br>Software: IE 7.0 browser and above |
| test method | Unit testing was performed using an automated test framework |
| Enter the instructions | Video file path |
| Output standard: | Returns the video file content |

test case

| number | test case | import | expected result |
|---|---|---|---|
| 1 | Movie play function test | Video file path (for example,'/ path / to / your / video.mp4') | Returns the video file content |

## 4.2.4 Movie ticket purchasing function

Plan form

| Field name | description |
| --- | --- |
| Test items | Movie ticket ticket function |
| designer | Si-min wang |
| Test environment requirements | You can connect to the server normally<br>Software: IE 7.0 browser and above |
| test method | Unit testing was performed using an automated test framework |
| Enter the instructions | Provide a valid movie ID and the ticket purchase amount; |
| Output standard: | The status codes and response data were returned as expected |

test case

| number | test case | import | expected result |
| --- | --- | --- | --- |
| 1 | Test of the successful purchase of movie tickets | Valid movie ID and ticket purchase amount | Return status code 201; the response data contains the movie ticket ID, price, and movie information |
| 2 | Test for the case of purchasing a non-existent movie ID | Nonexistent movie ID | Return status code 400; response data contains error message "Invalid movie ID" |
| 3 | Test the shortage of movie tickets purchased | The ticket purchase amount is lower than the movie ticket price | Return status code 400; response data contains error message "Insufficient funds" |

## 4.2.5 The community module

### 4.2.5.1 Get the movie review function

Plan form

| Field name | description |
| --- | --- |
| Test items | Get the review movie feature |
| designer | Si-min wang |
| Test environment requirements | You can connect to the server normally<br>Software: IE 7.0 browser and above |
| test method | Unit testing was performed using an automated test framework |
| Enter the instructions | No input is required |
| Output standard: | Return the correct movie review data |

test case

| number | test case | import | expected result |
|---|---|---|---|
| 1 | Test for obtaining movie reviews | not have | Return the correct movie review data |

### 4.2.5.2 Add the movie review function

Plan form

| Field name | description |
|---|---|
| Test items | Add the Review Movie feature |
| designer | Si-min wang |
| Test environment requirements | You can connect to the server normally<br>Software: IE 7.0 browser and above |
| test method | Unit testing was performed using an automated test framework |
| Enter the instructions | Movie ID, review user information, review content |
| Output standard: | Successfully added comment, return status code 200, return value of "1";<br>Add comment failed, return status code 200 with return value "0" |

test case

| number | test case | import | expected result |
|---|---|---|---|
| 1 | Add comments normally | Movie ID, review user information, review content | Successfully added successfully returned status code 200 with return value "1" |
| 2 | Missing information when adding comments | Movie ID, lack of review user information or comment content | Add comment failed, return status code 200 with return value "0" |
| 3 | Invalid movie ID | Invalid movie ID, comment on the user information, comment on the content | Add comment failed, return status code 200 with return value "0" |
| 4 | Server connection was abnormal | Movie ID, comment user information, comment content, cannot be connected to the server | Add comment failed, returned status code 500 or failed to connect to the server error message |
| 5 | Enter a comment with an illegal character | Movie ID, review user information, comments containing illegal characters | Add comment failed, return status code 200 with return value "0" |

## 4.3 Integrated testing

| | |
|---|---|
| test object | Check the correctness of the business process and data flow in the requirements |
| test specification | Clear business processes in the requirements, or combinations to form a large function without asking for functional modules |
| technology | Expected results are obtained when using valid data.<br>Display the appropriate error message or a warning message when using invalid data. All the business rules are applied correctly. |
| Start the standard | Standards must be met when completing an integration test |
| completion criteria | All of the planned tests have been performed. All of the defects found have been resolved. |
| Test priorities and priorities | In the test process, it is necessary to focus on testing the user's login and video playback, which is also the main business of the system, and the priority is high |
| Special matters to consider | During the video playback test, attention should be paid to whether the video can still play normally after the video progress bar is adjusted. |

## 4.4 User interface test

| | |
|---|---|
| test object | Verify the following contents:<br>The test browsing correctly reflects the functions and needs of the business, including browsing between Windows, fields and fields, and the use of various access methods (Tab keys, mouse movements, and shortcut keys).<br>The objects and features of the window (for example, menu, size, location, status, and center) are all compliant. |
| test specification | The respective formations of the interfaces and their event responses |
| technology | Create or modify tests for each window to verify that each application window and object can be browsed and in a normal object state. |

| | |
|---|---|
| completion criteria | Successfully verified that all windows were consistent with the baseline version or met the acceptable criteria |
| Special matters to consider | Not all features of customized or third-party objects are accessible. |

## 4.5 Performance test

| | |
|---|---|
| test object | Verify the performance behavior of the specified transaction or business function if:<br>Normal expected workload<br>The most heavy workload expected |
| technology | Use the testing process developed for functional or business cycle testing.<br>Increase the number of transactions by modifying the data file, or increase the number of iterations for each transaction by modifying the script.<br>The script should be run on one computer (based on a single user, a single transaction) and repeated on multiple clients (virtual or actual clients, see Special). |
| completion criteria | Single transaction or single user: successfully completed the test script within the expected time of each office without any failure.<br>Multiple transactions or multiple users: Successful test script completion within an acceptable time frame without any failures. |
| Special matters to consider | Comprehensive performance testing also includes adding a background workload to the server.<br>There are several ways to do this, including:<br>Assign transactions directly to the server, usually in the structured language<br>The form of the speech call ^ to implement.<br>Simulate many (usually hundreds of) clients by creating "virtual" users. This load can be implemented through the "" Remote Terminal Simulation (Remote Terminal Emulation) " tool. This technique can also be used to load in a network"rate of flow".<br>Add load to the system using multiple actual |

| | |
|---|---|
| | clients (each client runs a test script).<br>Performance tests should be performed on a dedicated computer or within a dedicated machine to achieve complete control and accurate evaluation.<br>The database used for performance testing should be of actual size or the same scalscale. |

## 4.6 Load test

| | |
|---|---|
| test object | Verify the performance behavior time of the assigned transaction or business reason in the unquestioned workload condition. |
| technology | Use the tests developed for functional or business cycle testing.<br>Increase the number of transactions by modifying the data file, or increase the number of events of each transaction by modifying the script. |
| completion criteria | Multiple transactions or multiple users: completed testing successfully within an acceptable time frame without any failures. |
| Special matters to consider | Load tests should be performed on a dedicated computer or within a dedicated machine time to achieve complete control and accurate evaluation.<br>The database used for load testing should be of the actual size or the same scale. |

# 5 Risk management

## 5.1 Project risk list

| order number | Risk description | resolvent |
|---|---|---|
| 1 | Demand analysis is not comprehensive | Evaluate uncompleted functions and consider waiver in terms of importance and time permit |
| 2 | The development cannot be completed on schedule | Track the development progress and adjust the test schedule in time |
| 3 | Leave for illness or other reasons | Training, increase manpower |
| 4 | Module function change | Actively communicate with the developers and re-assign the test tasks |
| 5 | The test environment is not synchronized with the development environment | Strengthen the version management, implement the database version management, and update the test data regularly |
| 6 | The new start time | Strengthen the training of new recruits in the early stage of the project, and the testers will be familiar with the products as soon as possible |
| 7 | The testers did not communicate enough with the relevant developers | By establishing software test record report and software test report system, and personal weekly report system, make full use of Github, manage the assessment of software system and software test manager. |
| 8 | Because the test work could not be performed as planned due to the delay of the developer release version, the test was not sufficient | Cooperate with the development schedule to make the corresponding overtime work. |

## 5.2 Bug management

The Bug is managed using the GI thub. GI thub Bug submission and closure of Bug are only limited to the tester in the project team (including the project software test leader). If the source of Bug is not found by the tester of the project, the software project manager is required to notify the test leader of the project in time, and the project test leader shall submit the Bug. For the discovery of pretest version (pretest), the bug cannot be submitted to GI thub, it is required to fill in and submit the software test record report and software test report. If necessary, the relevant personnel (software project manager, software test manager, SQA manager, software test manager, core module software development engineer, General / Deputy General Manager of Software Department) can be organized to review the software test record report and software test report, and confirm the Bug in

the test report.

## 5.3 Definition of the BUG level

Level 1 Bug (AA): causes the system to achieve functional goals and fails to proceed with use. It mainly includes: the abnormal termination of the program, the program crash, the key requirements are not realized, and the software functions are seriously inconsistent with the requirements. And the recurrence rate of 50% above, for the level 1 BUG.

Secondary Bug (A): makes the system unable to achieve functional goals, but knows how to avoid errors. It mainly includes: abnormal termination but avoidable procedure, and wrong understanding of non-key requirements. And the recurrence rate is less than 50%, or the use frequency is not high, for the secondary BUG.

Level 3 Bug (B): The system function objectives are basically realized, and the software functions are basically consistent with the requirements, but some functions have errors or the interface display has errors. For example, a single string display error, the position of the picture and text overlap, not recognizable, etc.

Level 4 Bug (C): the interface display is consistent with the requirements, but the user is not convenient to use, such as the user interface is not very friendly.

## 5.4 Defect report and its treatment

### 5.4.1 Processing process

The treatment process is shown in Figure Figure 2.

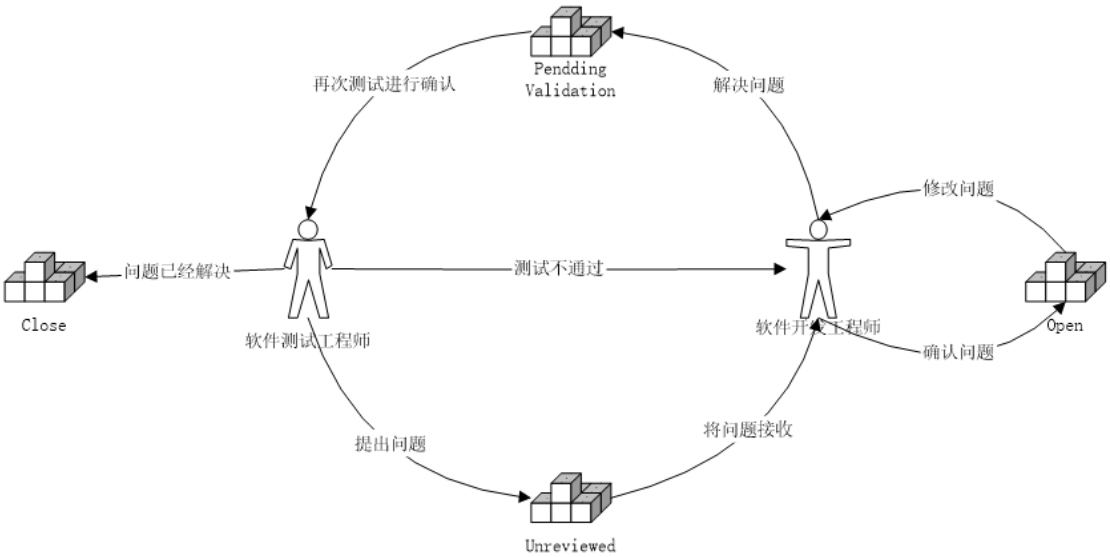


Figure 2 Defect reporting process

## 5.4.2 Defects report

Fault number: 001

| test items | Test category | test method | test data | whack | error state |
|---|---|---|---|---|---|
| Test exit function | function testing | black-box testing | user_name: "TestUser" user_loginState: "inactive" | The response status code is 200 The response content is "Logout success" | The field value used in the query condition does not match the record in the database |
| Description of the abnormal state: | Level III Bug (B) Failed to quit successfully Responsible person: Xu Mengmeng | | | | |
| suggestions on revision | | | | | |
| Review test results | | | | | |