## Test Plan

##### Testing objectives

The main objectives of this software testing are to check for any critical errors (bugs) in the program and ensure that it performs various functions correctly to achieve the expected results. We expect the program to handle image files correctly and display processed files accurately in the designated area. Additionally, the program should handle user input correctly and provide appropriate warnings or prompts when invalid information is entered. For critical functionalities, it is important to ensure quick response times to provide a good user experience.

The key testing objective in this software testing is to test the license plate recognition feature of the program. This includes testing the accuracy of license plate recognition for different scenarios, GUI testing to ensure correct prompts when no image is selected, and testing the ability to complete license plate recognition tasks within 1.5 seconds. In addition to license plate recognition, the testing also covers all image processing functionalities in the program, such as convolution, image segmentation, image blurring, and more.

##### Testing Plan

The system will be tested using a unit testing strategy. This testing will be fully automated and will utilize the patch method to simulate user inputs. The testing will be conducted using QTest combined with unittest to write test cases, and XTestRunner will be used to generate test reports.

For this testing, a set of 10 different images will be selected as test resources. These images will include blurred images, images with smaller license plates, images with rotated license plates, images with license plates positioned off-center, images of performance cars' license plates, images with multiple license plates, and normal images, among others. These images will be used to assess the system's performance under various scenarios.

The testing approach will ensure that the system can handle different types of images and accurately recognize license plates. It will also verify the system's ability to respond to user inputs, generate appropriate warnings or prompts, and meet the expected performance requirements.

The use of automated testing and comprehensive test cases will help identify any issues or errors in the system and ensure its functionality is thoroughly tested. The generated test reports will provide valuable insights into the system's performance and assist in identifying any areas that require further improvement or optimization.

Testing work will be carried out by Xu Yuzhuo, who will test all system functionalities. For simple functionalities, a single test case will be written to check if the function can achieve its intended purpose correctly. For regular functionalities, at least three test cases will be written to test if the function can perform correctly under different conditions. For critical functionalities (such as license plate recognition), at least seven test cases will be written. The testing period will be in the 15th and 16th weeks, and all testing and test report generation should be completed before Wednesday of the 16th week.

The specific test case writing plan is as follows:

**Open\_image**:

1. Test whether the function can correctly open a .jpg file and display the image in the corresponding location.
2. Test whether the function can correctly open a .png file and display the image in the corresponding location.
3. Test whether the function can correctly open a .jpeg file and display the image in the corresponding location.
4. Test whether the function can correctly open a .webp file and display the image in the corresponding location.
5. Test whether the function can correctly open a file under a Chinese path.
6. Test whether the function can open non-image files, and if not, whether it provides a prompt.
7. Test whether the function can open larger images.

**Save image：**

1. Test whether the function can save the image in .jpg format.
2. Test whether the function can save the image in .png format.
3. Test whether the function can save the image in .jpeg format.
4. Test whether the function can save the image in a Chinese path.
5. Test whether the function allows the user to save as a non-image file and whether it provides a prompt.

**Vertical flip**：

1. Verify if the image can be vertically flipped, with a size of 640\*480 pixels.
2. Verify if the image can be vertically flipped, with a size of 1600\*1200 pixels
3. Verify if the image can be vertically flipped, with a size of 557\*373 pixels.

**Horizontal flip**：

1. Verify if the image can be **horizontal** flipped, with a size of 640\*480 pixels.
2. Verify if the image can be **horizontal** flipped, with a size of 1600\*1200 pixels.

3）Verify if the image can be **horizontal** flipped, with a size of 557\*373 pixels.

**Rotate image**：

1. Verify if the function can rotate the image by a positive angle.
2. Verify if the function can rotate the image by a negative angle.
3. Verify if the function can rotate the image by a floating-point angle.

**Crop image**：

1. Verify if the function can correctly crop the image with a crop ratio of (100, 100, 200, 200).
2. Verify if the function can correctly crop the image with a crop ratio of (10, 20, 30, 40).
3. Verify if the function can correctly crop the image with a crop ratio of (10, 10, 20, 20）.

**Convert to gray**：

Verify if the function can correctly convert the image to grayscale.

**Apply gaussion blur**：

1. Check if the function can apply the Gaussian blur to the image correctly when the input convolution kernel is valid.
2. Check if the function displays a warning dialog when the input convolution kernel is a floating-point number or an integer.

**Sharpen**：

1. Check if the image is correctly sharpened with a size of 640\*480.
2. Check if the image is correctly sharpened with a size of 1600\*1200.
3. Check if the image is correctly sharpened with a size of 557\*373.

**Apply histogram equalization**：

1. Check if the histogram equalization can be applied correctly to the image with a size of 640\*480.
2. Check if the histogram equalization can be applied correctly to the image with a size of 1600\*1200.

3）Check if the histogram equalization can be applied correctly to the image with a size of 557\*373.

**Apply gamma transform:**

1. Test if the gamma transformation can be correctly applied when a floating-point number is entered.
2. Test if the gamma transformation can be correctly applied when an integer is entered.
3. Test if the gamma transformation can be correctly applied when a negative number is entered.

**Conv**：

1. Test if the convolution operation can be correctly executed when an integer is entered.
2. Test if the convolution operation can be correctly executed when a floating-point number is entered.
3. Test if the convolution operation can be correctly executed when a negative number is entered.
4. Test if a warning dialog is correctly displayed when an invalid convolution kernel is entered.

**Upload image**：

1. Test if the function can correctly load and save the image in the corresponding path in Window 2.

**Detect image：**

1) This function can correctly identify the license plate number of different types of pictures.

2) The middle picture and the final result picture can be correctly displayed.

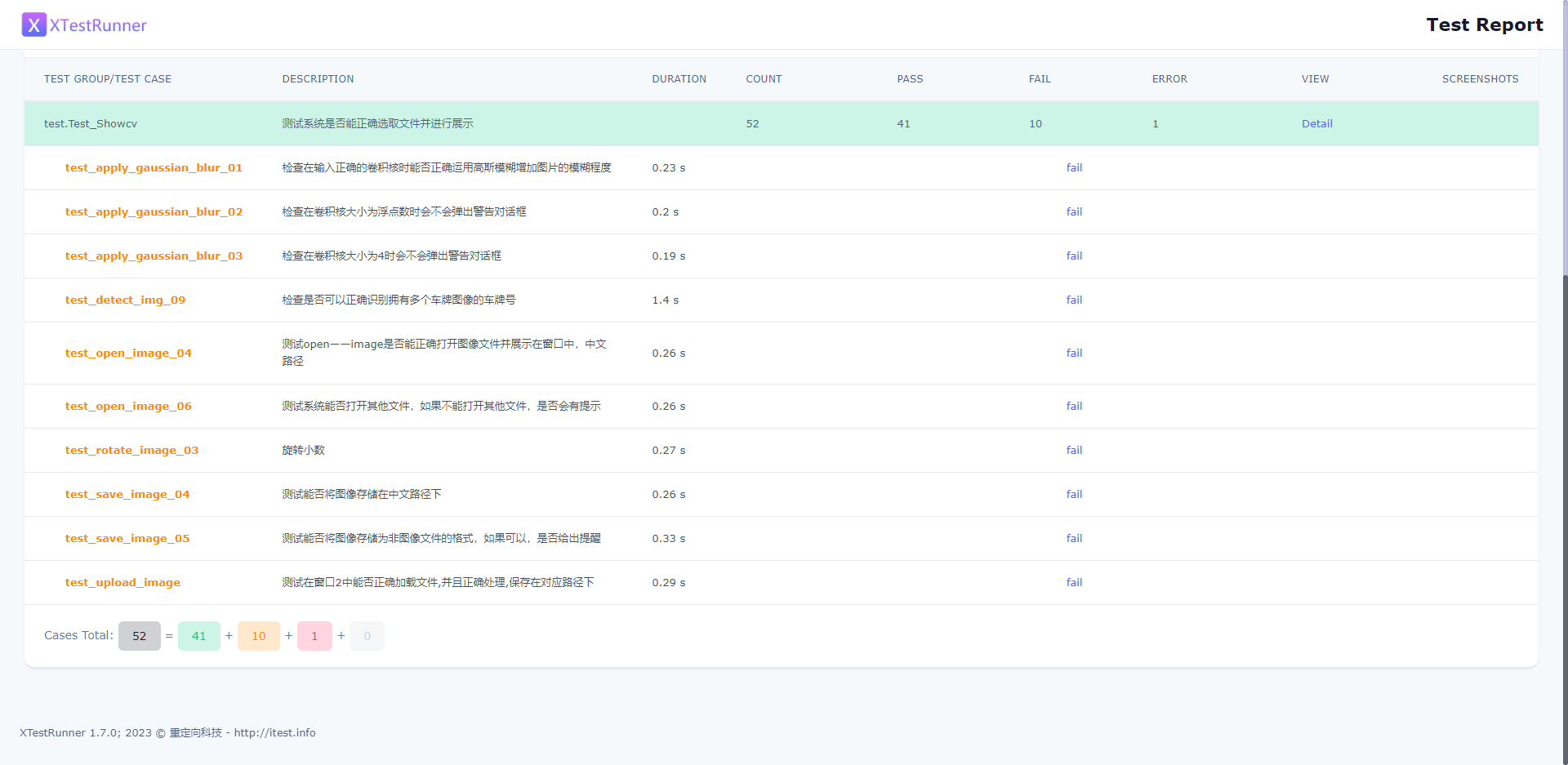
3) The identification time is within 1.1s.

4) When the user does not select the picture, will the prompt dialog box pop up

##### 3. Test execution

Use the following images as test case resources:

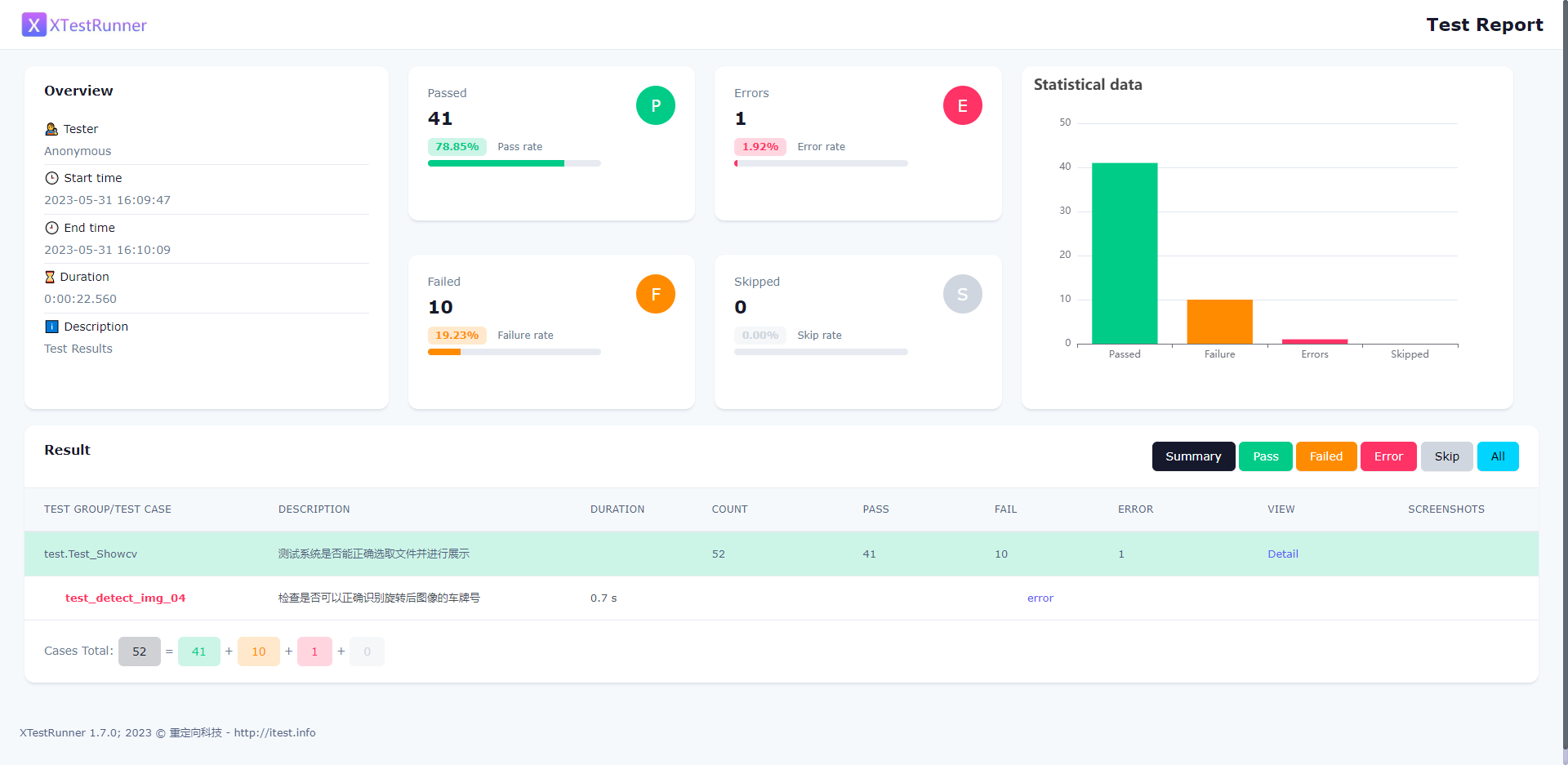


During the test execution, it is necessary to wrap all test cases using the TestSuite method from unittest into a test suite, and execute the test suite using the runner.run method. Out of all the test cases, there were 10 tests that did not pass. The details are as follows: 

Among them, there were significant bugs in the testing of the Gaussian blur functionality. The function did not correctly execute the image blurring process and did not prompt the user for input. In all the functions related to opening and saving images, users are unable to choose paths with Chinese characters. This can cause the program to crash. The specific reason is that the getOpenFileName method in the Dialog is unable to read Chinese paths correctly. Users need to save the images to be opened in paths with English characters in order to load them correctly, but there is no such prompt in the system. In the image rotation functionality, users are unable to enter floating-point numbers. When the user inputs a decimal point ('.'), the window does not read it and only reads the numerical part. Additionally, for the most important functionality, license plate recognition, we used various images to assess the correctness of the recognition feature. In the case of multiple license plates present in an image, the system is unable to recognize them correctly, but it does output the resulting image and provides output content. However, for rotated images, the system is unable to recognize them correctly and cannot generate the final result image. This may be due to the elimination of the part containing the license plate during the image scaling process.

##### Test Report

Use XTestRunner to generate the unit test report for the entire system. The report includes statistics on the pass rate, failure rate, and error rate of all test cases. The report will be saved in the specified folder in the format of an HTML file. Please refer to the following image for the detailed test report:



The Overview section records the details of the testing personnel, testing time, and relevant descriptions. The middle section contains the record of the number of passes, pass rate, number of errors, error rate, number of failures, and failure rate for this test. The table on the right displays a histogram showing the number of passes, failures, and errors for this test. The Results section stores all the test cases for this test and categorizes them as pass, failed, or error.

The pass rate for this test is 78.85%, with 41 test cases passing. The reasons for the other cases not passing or resulting in errors have been analyzed in the previous section in relatively specific terms. Overall, most of the functionalities tested in this round are relatively complete. However, there are still some major bugs in the system that need to be fixed, as well as some minor bugs that need improvement.