Quality Assurance Plan for Quality Assurance Plan

Project:

A platform of errand delivery Service in the same city based on crowdsourcing model

Prepared by: Zhangyi Shen

Date: 30.4.2019

1. Draft Quality Assurance Plan

1.1 Introduction

This is a brief description of the test objects (components, applications, systems, etc.) and their goals. The information that needs to be included is: the main features and capabilities, the structure of the test object, and a brief history of the project.

1.2 Purpose

Products must be rigorously tested or coordinated to make the product as smooth as possible. In order to ensure product quality, improve test efficiency and improve version quality. By understanding the problem, solving the problem, and coordinating each other, the product effect is finally satisfied.

1.3 Policy Statement

Set a meeting with the client once a week to showcase the project results. At least two small meetings a week in the group, synchronization progress and test feedback.

1.4 Scope

1.4.1.Order Provider Application (Android):
Users with delivery requirements can publish tasks through this application. After the order is placed, the delivery agent APP will receive the notification

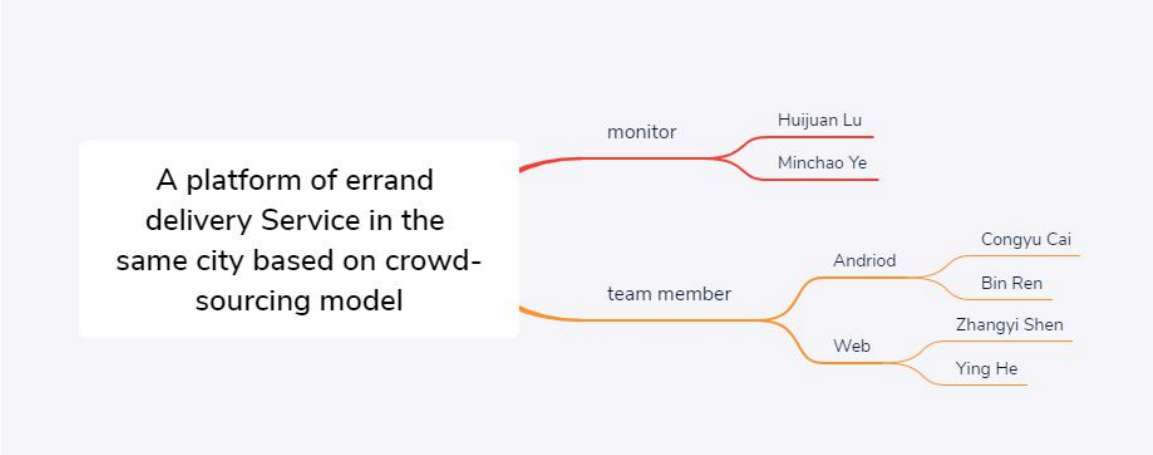
"Come on the order, please grab the order". After the order is received, the delivery staff can be responsible for the delivery of the order.
The order supplier has the following features:
- Set different service types (takeaway, small package or large package) and other important information
- Post commands in the task
- Real-time monitoring, you can see the status of the order
- order has been completed
- Evaluation service

1.4.2.Deliverymen application (Android)
Delivery personnel can download our shipping request and complete orders and rewards by completing these delivery notes.
The deliveryman has the following features:
- Receive orders posted by users
- payment of deposit (by value of goods)
- Receiving and shipping
-Complete order

1.4.3.Platform (Web-Based)
Collect delivery tasks and assign them to order recipients.
- Order management function, you can view orders
- User management function, you can view user information, if you receive a complaint, you can blacklist it
- Blacklist, when receiving user complaints, such as issuing order error messages, items being sent to the wrong place, etc., add them to the blacklist, and if there is a misunderstanding, you can remove them from the blacklist.

2. Management

2.1 Organizational Structure



2.2 Roles and Responsibilities

2.2.1 Technical Monitor/Senior Management

Huijuan Lu and Minchao Ye

2.2.2 Task Leader

Congyu Cai

2.2.3 Quality Assurance Team

Congyu Cai,Zhangyi Shen,Bin Ren,Ying He

2.2.3 Technical Staff

Congyu Cai,Zhangyi Shen,Bin Ren,Ying He

3.Required Documentation

After the requirement review, we estimated the development time, which is reflected in the Gantt chart. Then we make a test plan. Our test cycle is based on the development time point. After one phase is completed, we test (1.30, 2.15, 2.28, 3.13, 3.26, 4.9, 4.19, 5.1). The test covers our product function points.

Design Use Case: Delivery verification is carried out according to product process, interface use case, function use case and use case to ensure product quality: through mainstream process use case - development self-test, unit test, module test, regression test, release online, online verification, communication summary.

4. Quality Assurance Procedures

4.1 Walkthrough Procedure

For this walkthrough, no changes can be made to the web application created in the walkthrough. To create a simple web application, you need the address of the application.

Create a test project, in a new Visual Studio instance, on the File menu, point to New, and then click Project. The New Project dialog appears. Under Projects, expand Visual Basic and select Test Node. Record the web test. Looking at the request properties, the URL tree in the Web Test Editor is called the request tree and looks at the properties associated with each request.

We need to log and run web tests.

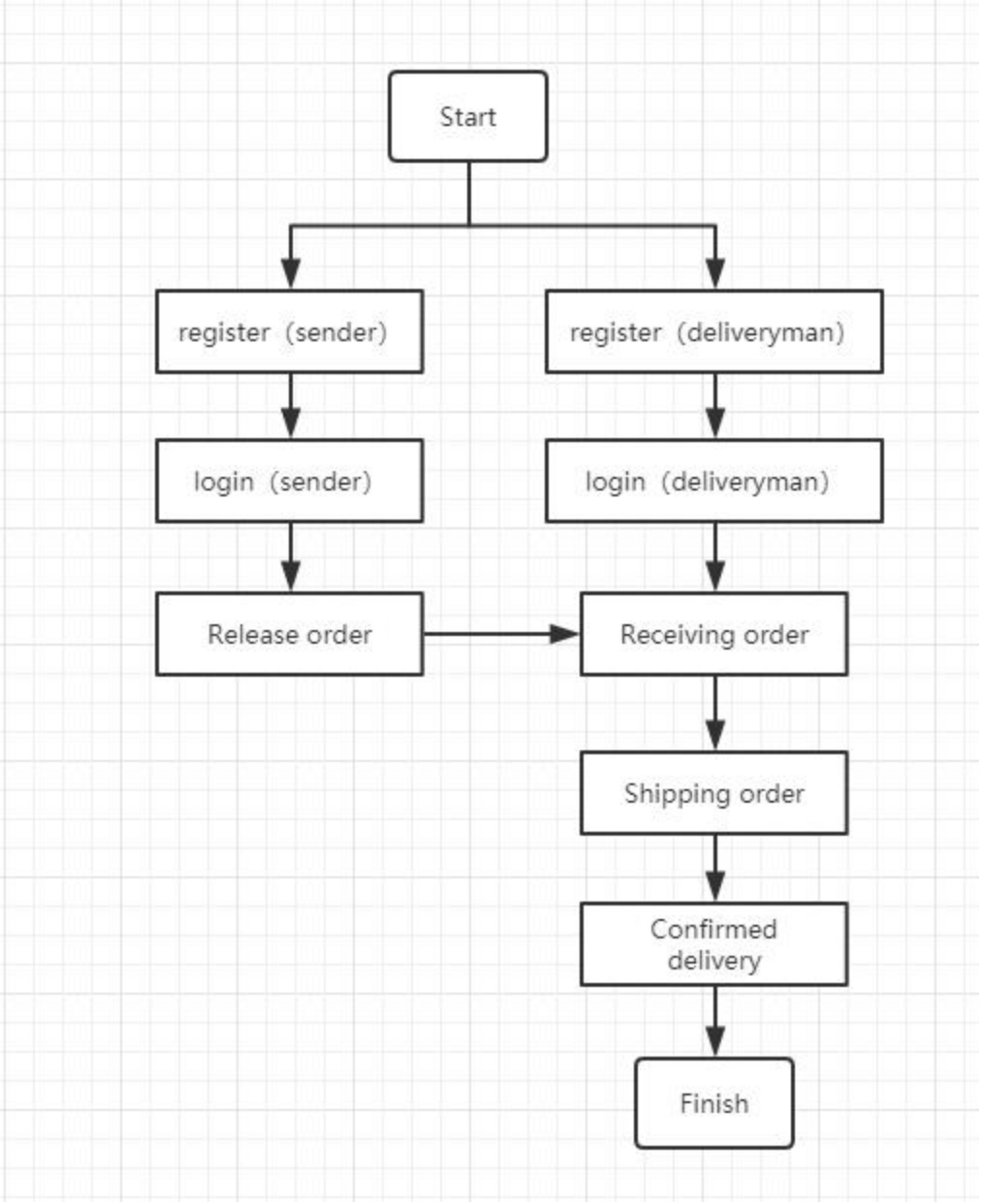
4.2 Review Process

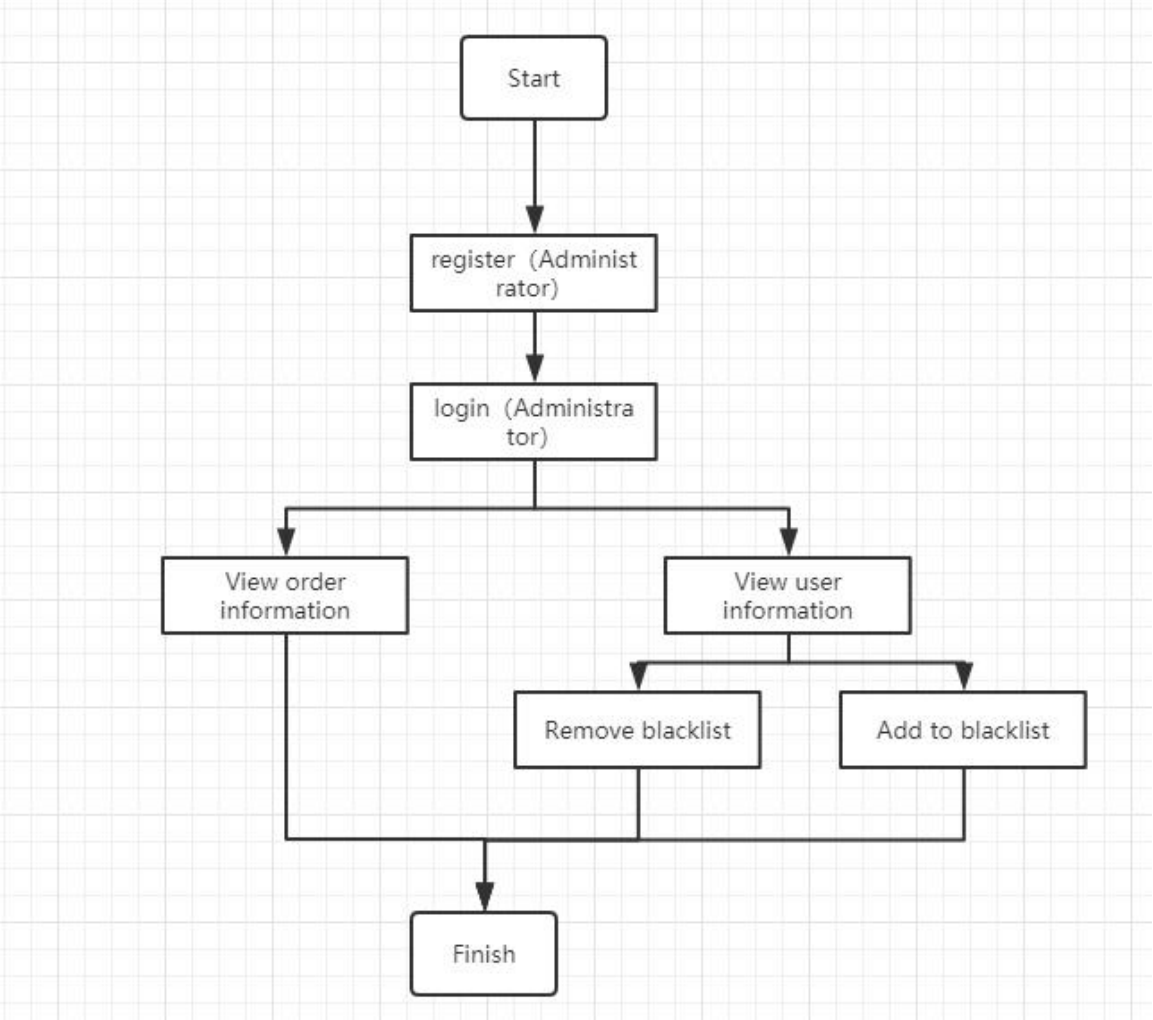
4.2.1 Review Procedures

We conduct weekly review meetings to identify needs loopholes and develop loopholes. Correct in time and conduct second test

4.3 Audit Process

4.3.1 Audit Procedures


Start
register（sender）
register（deliveryman）
login（sender）
login（deliveryman）
Release order
Receiving order
Shipping order
Confirmed delivery
Finish



4.4 Evaluation Process

The testing process is the content of a project later. At this time, the functions of the system have been realized, and it is necessary to continuously try and return and return to the original requirements of the system.
The test can be deployed in the production environment and run after the test is completed and the review is successful before the system is officially launched.
Test process: code, interface, requirements, user. That is, four main modules: unit test, integration test, system test, acceptance test.

4.4.1 Unit test

Keywords: code.

Unit testing is the most granular part of the test, and it can be as thin as a line of code. Unit tests are generally tested by the developers of our scripts. Test methods generally use white box testing, which is to open the script to see the internal logic.
Module interface test: Accurate and consistent test data inflow and outflow modules. Test whether the actual parameters and formal parameters entered when calling the module are accurate and consistent.
Local data structure test: Test the accuracy and consistency of data temporarily stored in the module during program execution.
Path test: Test each independently executed path in the module.
Error handling test: When the test program fails, the system can output the error result and keep the logic correct.
Boundary condition test: Tests whether the logic on the boundary is valid and correct.
After unit testing, the entire large system is divided into independent small modules. The correctness of the internal logic of each small module ensures that the logic of the most basic layer of the system is correct. After that, you can go to the upper layer and test the logical correctness of the bridge that links each module.

4.4.2. Integration Test (SIT)

Keywords: interface.
Integration testing is often referred to as SIT testing, which is aimed at linking the interfaces of individual test modules, so the test granularity is larger than unit tests. In unit testing, all modules are assembled or spliced according to the design structure of the system to form a subsystem or a large system. Integration testing is the correct process of testing the modules to each other.

4.4.3. System testing

Keywords: demand.

System testing is the function that is included in testing the entire online system. It is closely integrated with the project's SOW or requirements specification. The goal of system testing is clear: whether the test system meets the requirements. Therefore, all the function points in the requirements specification are use cases that need to be tested, which determines that the system test is generally based on black box testing.
Functional test: Test whether the functions in the requirements specification are implemented correctly and consistently in the system. General functional tests include logic tests, interface tests, usability tests, and installation tests. Logic

testing is the main content of system testing, and its use cases generally correspond to the framework of the requirements specification.
Performance test: test the time, speed, concurrency, fault tolerance, space occupancy, etc. of the system.
Security test: Test system's defense against illegal intrusion.
Compatibility test: Test system running on different platforms and different hardware environments.
After system testing, it can ensure that the delivered system meets the needs of the original project design, and can realize the functions of the original design with quality and quantity. After completing this phase, you need to enter the final stage of the test, the acceptance test, and the actual user of the system to test the system.

4.4.4.Acceptance test (UAT)

Keywords: user.

The acceptance test is the final stage of the entire test. Its test case design is similar to the system test, and it must be designed according to the design of the SOW or the requirements specification. The difference between it and system testing is that the person who tests is the user, not the developer. Therefore, most of the stages are tested in black box, and the test cases are less than the system tests. The acceptance test is more focused on whether the function of the system meets the requirements, whether the interface of the system is suitable, and whether the running time of the system is reasonable.
The use cases in the acceptance test need to have been tested by the developer in the system test. This means that the use case for system testing must include the use case for acceptance testing, but not all system test cases must be tested by the user in UAT.
After the user's UAT test is passed, the test work comes to an end and enters the review phase. After the review is successful, the system can be deployed online.

4.5 Process Improvement

I think there are three process improvements: finding problems, analyzing problems, and solving problems. We found problems through customer feedback, log analysis, and team members' brainstorming. We use critical thinking to

analyze problems. We can solve problems by referring to the industry's excellent practical experience.

5.Problem Reporting Procedures

5.1 Noncompliance Reporting Procedures

-

6. Quality Assurance Metrics

Requirements Review - Demand Side Requirements Specification - Design - Development Plan - Quality Plan - Implementation - Test and Confirmation - Acceptance - Copy, Delivery and Installation - Maintenance