Test Plan

Company 1 - TDDC88

Contents

Version	Author	Updates	Reviewed by	Date
0.1	E. Sköld	Initial version	A. Telenius	2021-09-13
0.2	E. Parö	Test Levels and more	D. Ma	2021-09-24
0.3	A. Telenius	Fill in last headers	D. Ma	2021-09-24
0.4	A. Telenius	Rewrite every section in chapters 2.3-4.2	-	2021-11-07

1 Introduction

In this document an explanation about our test strategies, process, workflow and methodologies used in the project is given. We follow the V-model which is a model that has a testing phase parallel to the development phases. It is a model that is similar to the waterfall model which means that the requirements have to be clear because it is usually hard to go back to make changes. But the model is simple and easy to follow and works well for smaller projects.

In the project we will use a System usability scale (SUS) which can be used to test the usability of a system. It consists of a questionnaire with ten questions that each has five options. This is to measure the effectiveness, efficiency and satisfaction of the software.

The following test plan is derived from test plan template created by Thomas Hamilton for Guru99. A link to the template can be found here.

1.1 Scope

The scope of the test plan is defined below. This includes what will be tested (In Scope) and the parts that will not be tested (Out of Scope).

1.1.1 In Scope

In accordance with the V-model, testing is to be conducted in unit, module, system and acceptance level testing.

The customer want us to focus on the usability of the system and how to visualise data in a satisfying way. Therefore the testing will focus on that and measure if the developed application achieves requested levels. The testing will also focus on the functionality of the system so that the intended units and integration between units works as expected.

1.1.2 Out of Scope

Due to the limited implementation levels of the project the test plan will diverge from the V-model by not conducting tests at the final level, maintenance.

The customer does not want us to focus on how to store data and database management and therefore such tests will not be conducted. Because the storing of data is not important for this project, security will neither be tested during this project.

1.2 Quality Objectives

något från erik här?

1.3 Roles and Responsibilities

Every tester is part of a cross-functional team (CFT) and is responsible for performing tests for their CFT individually related to the module and system testing. The test leader is responsible for the acceptance testing and will coordinate this with the help of all the testers. The test leader will also be the contact person for non CFT-related inquiries for testing procedures.

2 Test Methodology

2.1 Overview

This part is to be further developed during iteration 1. The testing division needs to coordinate their work from how the development process is to be conducted.

From iteration 3 and further on, a new company structure have been developed with a specialized testing cross-functional team. This team will have all the responsibility of testing from unit testing to acceptance testing. With the new structure, the testers can have better communication and work closer together naturally by working according to scrum.

2.2 Test Levels

In this chapter the different test levels are presented and which test that will be performed to test the levels of the application. The different testings levels are unit testing, module testing, system testing and acceptance testing.

2.2.1 Unit testing

Each tester is responsible for testing the commits of their CFT.(Iteration 1 and 2)

Unit testing will occur when a developer is doing a merge request to their specific cross functional team branch. Together with the unit testing, a peer review of the code will be done by a developer to ensure that the functionality is working and also that the code is living up to company standards.

From iteration 3 and forward the new cross-functional testing team will be responsible for the unit testing. The customer wants the project to have a greater focus on user interface and usability of the system. Therefore, the testing will not have a major focus on black box testing because such a unit will not be developed. Instead the unit testing will focus on the GUI testing by exploratory testing where the tester will try the new unit which, for example, can be trying a new button.

To get further automated test the tool Selenium will be used and in particular Selenium IDE. This will enable us to create test cases with so that when implementing new features the rest of the application will not be affected. This will also help us speed up the testing procedure when the test cases are set up.

2.2.2 Module Testing

The module testing is the test of the integration between several modules which verifies the module design. The company is divided into three cross-functional teams and each team has a tester. Each teams tester is responsible of the cross-functional teams module testing. When a iteration is finished testing is conducted on a group level to confirm that the code is valid. How the test is performed is not yet decided.

To make sure that the module testing is performed a requirement is needed:
Before each merge with master a automated test are conducted and then the merge can be accepted to the master.

2.2.3 System testing

The test team is responsible of the system testing which will verify the system's architecture and high-level design. Each tester will perform this test individually. How the test is performed is not yet decided.

2.2.4 Acceptance testing

To confirm the requirements the test team will perform acceptance testing which also will help the developers to tailor the application to the customer.

The acceptance test will be performed with the concurrent think aloud method. To get a quantitative result and measure the usability of the application a SUS-test will also be part of the acceptance test. To conclude that our developed application has a sufficient level of usability the SUS-test must achieve a score of at least X(not yet decided) for the final product.

The test will be conducted with the customer after each iteration to gather their response. Because the test is done after each iteration this enables the developer to get feedback of the not-yet-complete application by the customer. This will hopefully make the application tailored for the customer.

Proposed dates for test with customer:
After iteration 1: 11/10 - 21
After iteration 2: 8/11 - 21
After iteration 3: 22/11 - 21
After iteration 4/ Final product: 3/12 (Try to push this back further)

Between the tests with the customer, user tests will also be performed on other test objects than the customer. This test will test the progress of the different cross-functional teams and will be done by each tester in that cross-functional team.

2.3 Bug Triage

Bugs found during the testing procedure must be solved or have the underlying code generating the bug removed. This is due to all testing (except acceptance testing) taking place when a merge request is created. If a bug is found, the merge request is not approved by the tester. The merge request must then be edited to solve the bug or it is not to be merged at all. Bugs found outside of the testing procedure (e.g by non-testers) shall be reported to the testing team which will investigate the bug, create a Gitlab issue explaining what the bug is, steps to reproduce it, suspected underlying issue and deadline for when it must be solved. This is usually set to the next iteration deadline as known bugs shall not be present during acceptance testing with the customer. The Gitlab issue is assigned to the author of the code.

2.4 Suspension Criteria and Resumption Requirements

Due to testers only working on testing and the small, limited number of functions in the application the suspension criteria is only a complete system failure rendering the application non-functional. Testers only working on testing means that continued runs after a program is not working correctly is reasonable in our time schedule. This is supported by the fact of a small, limited number of functions making each test short. This means that the only resumption criteria is when a non-functional program is made functional and thereby resuming testing.

2.4.1 Exam period, 19/10-21 → 30/10-21

During the exam period neither developers nor testers will have time to perform testing or programming because they will study for their exams. Because all company participants have this period at the same time this will not create a bottleneck, only a temporary stop in the continuous delivery.

2.5 Test Completeness

A test is considered complete when it passes the Gitlab pipeline, the selenium IDE-tests and exploratory testing for each merge. Not passing one of these tests means the merge request will not be approved.

3 Test deliverables

This chapter regards deliverables created for customer, CEO, company and/or testers.

In the beginning of each iteration we intend to have a meeting with the customer where they will do a System Usability Scale (SUS) test with the concurrent think alound (CTA) protocol. Every iteration is roughly two weeks so we planned to do these test four times with the customer during the project. This is to test the new features that have been developed during the iteration that has been so that the customer can give feedback to what they like and don't like. The feedback from the customer will then be delivered to the developers and UX-designers so that they can continue to improve the features and modules. Each tester was part of a different cross functional team and the tester will inform the team about the results of the test and the feedback that was received and should be implemented. This was changed in iteration 2 where testers joined the same team to work closer together. The first customer meeting did not feature a SUS-test with a CTA protocol as the application was deemed to bare-bones to result in any meaningful feedback from a customer. Instead a customer meeting was conducted where design ideas from the Figma protype and the application were evaluated on a detail level and written down. This interview was done with nurses and IT-staff from the customer. The notes from the meeting were then passed to the UX-team for evaluation.
Unit, integration and regression testing is done at each merge request and bugs found will be commented in code and as a comment in the merge request text fields. The creator as assignees of the merge request are then notified by the automated mail sent to their Gitlab accounts of the discovered bug.
The test plan is created to standardize and inform of the testing procedure. It is written by the the quality coordinator (namely subsection 1.2) and the testers. It is published both in the Gitlab repository as well as the Teams output folder for CEO and company to be informed about how testing is conducted. When a testing procedure is changed or created, it is written down into the test plan by testers so no procedure goes undocumented.

4 Rescource and Enviroments

This chapter regards testing tools and versions used by each tester to ensure the same prerequisites for each test done by the testers.

4.1 Testing Tools

The requriement tracking tool is verbal communication between testers and analysts where either a test is shown proving that a requirement is reflected in the application or by evaluating the application through existing functionality or properties the application possess. The requirement is the marked as done in the analysts document of requirements.

One of the two bug tracking tools is the merge request comment fields where information about what the bug is, steps to reproduce it and suspected underlying issue is relayed to the authors of the code and creator of merge request. The other bug tracking tool is git issues containing the same information as the first bug tracking tool and is used when bugs are found outside the testing procedure.

The automation tools are firstly the automated test done in Angular and implemented in the Gitlab pipeline which is conducted for each merge request. These tests are primarily used to ensure that the application is still having its basic functionality and is not the primary focus of the testing procedure. This is done due to the heavy focus on front-end testing as the customer application is heavily focused on front-end development. The second automated tool is a list of Selenium IDE tests and are run by the testers after they pull the merge request to their local machine.

4.2 Test Environment

- Angular
- Node

- Selenium

Lägger till fler och version senare

5 Terms/Acronyms

¨ A list of terms or acronyms used in the test plan.

- CFT - Cross functional team
- SUS - System usability scale
- CTA - Concurrent think aloud

References