Tampere University Unit of Computing Sciences
COMP.SE.610 Software Engineering Project 1
COMP.SE.620 Software Engineering Project 2

Group 4

Snowledge: Pallaksen Lumisovellus

Test Plan

H278851, Katariina Collander
150551124, Abdullah Arif
H275550, Oona Laitamäki
K435282, Emil Calonius
K428358, Lotta Orsmaa
K436244, Tiina Tamminen
K439387, Juho Kumara

Version history

Version	Date	Author	Description
0.1	30.9.2021	Katariina Collander	Create initial document
0.2	2.10.2021	Oona Laitamäki	Introduction, quality assurance process and testing plan
0.3	20.10.	Abdullah Arif	Writing test process
0.4	21.10.	Katariina Collander	Reviewed the document and did some additions
0.5	26.10.	Abdullah Arif	Writing the document
0.6	31.10	Abdullah Arif	Updated unit testing details, wrote about usability testing and goals
0.7	3.11.	Katariina Collander	Wrote more content to many chapters and wrote the last three chapters
1.0	4.11.	Katariina Collander	Finalised the document and returned it

Contents

1 Introduction

1.1 Purpose and scope of document

Purpose of this document is to illustrate the testing process of Snowledge. This document ensures that the project team remembers to take everything necessary into account regarding testing and quality assurance. This document is also an information source for the group and course personnel how testing is planned to be done during the project.

This document covers system, functional and acceptance testing as well as the planned usability assessment. The testing schedule, testing roles and the structure of testing reports are presented.

1.2 Product and environment

Product relating to this project is a web-based application called Snowledge by Pallaksen Pöllöt. The application provides snow information combined with weather conditions in different sections of Pallastunturi to people who are travelling in the Pallas area.

The production version of the application has been running on Tavu Cloud so far, but this might be changed. The application is currently running on test server provided by Tampere University. It can be accessed here: http://itc-pallas.rd.tuni.fi/ (link accessed on 2.10.2021).

1.3 Project constraints related to testing

The largest constraint the project has when it comes to testing is limited time. There are many functionalities to implement to the application and the customer is prioritising the new feature development over very thorough testing.

Another smaller constraint is that the project was already started before our group and it does not have any unit testing implemented. The only usability testing constraint that we have is that we are not able to do the usability testing with users in real life since the users are in Pallas region. We will do the testing remotely instead.

1.4 Definitions, abbreviations, and acronyms

Unit testing	lowest test level that aims to test small sections of code.
UI	user interface
Eslint	tool used for static code analysis

2 Quality Assurance Process

Quality is assured by team members reviewing each other's code before merging the new implementation, by demos and by bi-weekly functional testing.

2.1 Manual Quality Assurance Process

A team member must review and accept a new implementation before it can be merged to the master branch. Here is an example of version control use when the task involves programming:

1. Issue regarding the implementation is created in Trello board
2. Create own branch for development
3. Pull request to the master branch can already be created at this point
5. Code and do some commit on your branch
6. Other group members make a review and theycan comment code in pull request if needed
7. Some corrections can be done based on comments
8. When functionality/issue is ready, you can ask for an approval from other backend developers to pull request.
9. After approval merge your branch on master (or some other branch if needed)
10. The application on test server is updated based on master branch. The application based on state in master branch will be reviewed with the customer in sprint review.

Other tasks, such as design tasks and course documentation, are also reviewed. This is done by sharing the first version of the work in the group's shared folder. The task owner will then notify the rest of the group that there is a design or document to be reviewed. The task will be considered ready for demo only after at least one team member reviews it.

A task is also reviewed by the customer in the sprint review meeting.Task is considered done and demoed when it has been presented to the customer and the customer has accepted it. The customer can also try the application out by themselves on the test server and thus also request changes after the demo via email or Telegram.

In addition, manual functional testing is performed bi-weekly. This is discussed in chapter 3.1. General Approach.

2.2 Automated Quality Assurance Process

Eslint tool is used as static code analyser. Static code analyser has not been used in development in the first section of the project. Adding Eslint tool afterwards required editing previously implemented code to follow Eslint rules.

3 Testing process

3.1 General approach

The general strategy for testing Snowledge is to see if the application and its functional requirement meets the desired specifications of not. Multiple types of testing are going to be done to ensure that the application meets the client's requirements e.g. system testing, acceptance testing and usability testing.

System testing:
System testing will be done in production or in test server as black box testing. System testing makes sure that the system works according to customer requirements. System testing contains.

- Acceptance testing: The customer will do at the end of the project
- Functional testing: Functional testing will be done by our group after each sprint and the results will be documented.

Functional testing:
Functional testing will be done as a part of system testing after each sprint. Through functional testing all the functionalities of Snowledge will be tested and checked that they work according to the requirements.

Acceptance testing:
Acceptance testing's goal is to check whether Snowledge can handle the tasks in real world scenario's or not and if the customer accepts the project. A draft of acceptance tests has been written by one of the group members and these will be verified by the customer.
The customer and/or end users will conduct acceptance testing at the final stage of the project. In the previous part of the project there was acceptance testing done by the project group in every sprint iteration. This type of testing is called functional testing in this project stage.

Usability testing:
Usability testing will be done in sprint 4. The UX group of the project team will be responsible for implementing the usability testing. The testing will be done remotely with the end users of Snowledge and some potential partners of the application. The goal of the testing is to do a usability assessment of the application. The testing is done with a prototype of the full application. The group can improve the usability of the application based on the findings of the usability testing and provide the customer a report on the findings.

Unit testing:
After discussing with client, we came to conclusion that we will not do unit testing as unit testing has not been done from the start of this project. The client's wants us to focus on system testing, acceptance testing and useability testing more.

3.2 Definition of done and goals

The team decided to use "Trello" to keep track of all the tasks that we are going to implement in this project. The tasks to implement must go through multiples steps so that we can be sure that its functionality is as per clients need. Trello is divided in to 5 boards Backlog, Ready to Implement, In progress, In review and finally Ready board.

A functionality is ready to implement when the team decided how a function should perform and what its UX/UI looks like. After that it is given to the suitable member of the team who has expertise in that domain and the task moves to “In progress” board.

Once the functionality is done from that specific person who has been appointed, he/she pushed the changes on to GitHub and put the task “In review” board. The implemented functionality then is reviewed by one of the group members and if everything is looking good then it can be moved to “Ready” board.

A task can be considered done when:

- It has been reviewed and approved by another team member
- It has been merged to the master branch and it is visible on the test server
- Functional testing of that sprint has been done and the tasks related test cases are passing without problems
- The task has been demoed to the customer and the customer has approved it.

The goal of our quality assurance process is to ensure that the usability of the application is as good as possible. We also want to spot potential bugs in the code as soon as possible. The goal of review and demo processes with the customer is to make sure that what we implement is what the customer wanted.

3.3 Testing roles

Functional testing after each sprint is done by Katariina Collander. She is also responsible of writing test reports of the testing sessions.

Acceptance testing is done by the customer (Arto and Juuso) at the end of the project. The test cases of acceptance testing are written by the team and verified by the customer.

Usability testing and assessment will be done by the UX/UI group of the project team. The customer will provide team the contacts of the users taking part of the testing and the UX-group will provide the prototype, the test form and the analysis of the findings.

3.4 Test schedule

Testing Name	Starting Date	Ending Date	Author
Functional Testing	24.9., 7.11., 21.5., 5.12.		Katariina Collander
Acceptance Testing	Start of sprint 6, 6.12.	End of sprint 6, 10.12.	Customer
Usability Testing	Start of sprint 4, 9.12.	Second week of sprint 4, 15.12.	UX group

The sprint 6 has been reserved only for testing. We also reserved time for bug fixing and re-testing in case if things by any chance goes sideways. No new features are implemented in sprint 6 so all time is dedicated to finalising bug fixes and polishing the product.

At the end of the sprint 5, the customer can start the acceptance testing if they need over a week to do it. This will be discussed with the customer. The project will be finalised in sprint 6 and it would be good for the customer to do acceptance testing with as finalised of a project as possible.

3.5 Test documentation

The functional and acceptance test cases are stored in a shared Onedrive folder that the team and the customer can access. A version with test cases for all requirements is available as an excel. Throughout the project as we discuss the tasks more in detail we will write more detailed and complex test cases to verify that we cover many different scenarios.

The bi-weekly functional testing sessions are reported. A report of all the tests ran and the results is stored in the shared folder and presented in our sprint review meetings. The report shows the passing, failing and ignored test cases and any additional notes. We will calculate the percentages of failing and succeeding test cases. The report is used to follow the progress of the project, to notice bugs and to document all known behaviour.

The usability testing test cases are written as a questionnaire that can be sent to the users taking part of the testing. The questionnaire is sent as a Form that will automatically collect the user responses to an excel. The results will be analysed and the group will provide the customer a usability assessment based on them. The details of the testing will be provided in the next sprint.

4 Test cases

The draft of the functional tests can be accessed here:
https://tuni-my.sharepoint.com/:x:/g/personal/katariina_collander_tuni_fi/EXnEZQbOQJRBsSN2PZVoew4BfCgpzGgr0ppOk2Kl3alyVQ?e=xhLqDD
The acceptance testing will be based on the same test cases.

Usability testing test form is provided later.

5 Adopted Tools

Most of the testing is done manually due to the small size of the project and the prioritisation of feature development over testing.

We are using a static code analyser called Eslint. Eslint runs whenever a team member pushes content to Github. The analyser creates a report on the code and potentially shows warnings.

Usability testing is done with a prototype in Adobe XD. The users taking part of the usability testing are given a link to the prototype and a form/questionnaire with the test cases.

Test reports are provided of the functional testing sessions done after each sprint. These reports state the number of test cases and the numbers of passed, failed and ignored tests.

The statistics provided by the usability testing depend on the contents of the usability testing that are decided on later. The goal is to provide meaningful metrics on user satisfaction and about the amount of time needed by users to perform common user tasks.