Test Plan

Table of Contents

Scope

The scope off this test plan is to write and implement tests that give feedback to SDC goals described in the document strategy.pdf. To do this the goal of this test plan is to exercise tests that complete milestone 1 as described in the document strategy.pdf.

Milestone 1: “Implemented tests of "My web server" which with concrete clear results can give feedback to SDC so it can be stated if web server "My web server" is worth continuing to invest in as a web server for IoT devices. If these tests show that an investment is worthwhile, these tests should indicate how further development of the project should continue.”

List of what should be tested

The following list of what should be tested are arranged with the highest priority tests first and the lowest priority tests last

- Legacy Requirements section “Use Cases”
  - UC1 Start Server
  - UC2 Stop Server
  - UC3 Request shared resources

- Legacy Requirements section “Supplementary Specification”*

  - Req 1. The web server should be responsive under high load.
  - Req 2. The web server must follow minimum requirements for HTTP 1.1.
  - Req 4. The source code should be released under GPL-2.0.
  - Req 5. The access log should be viewable from a text editor.

- Concerning SDC:s Goal 1, *“To know if the found web-server “My web server” can be used on a wide range of IoT devices”*, described in document strategy.pdf section “Stakeholders Goals” following sub objectives have been deduced and have been decided to be tested to give feedback to this goal.

  1. “My web server” should have high maintainability.
  2. “My web server” should have high usability.
  3. “My web server” should be compatible with different java versions.
  4. “My web server” should be compatible with common used operative systems on IoT devices.

* OBS! From legacy requirement, see Appendix A in document strategy.pdf, supplementary specification “Req 3. The web server must work on Linux, Mac, Windows” will NOT be tested. Instead the requirement will be modified and tested in tests concerning “The compatible with “My web server” and common used operative systems on IoT devices”.

List of what should NOT be tested

- ➢ The integration of “My web server” to IoT devices
- ➢ The adaptability of “MY web server” to IoT devices
- ➢ “MY web server” securability
- ➢ “My web server” accessibility

The test that should NOT be tested is tests that should be fulfilled for completing milestones 2 and 3 which are also previously described in the document "strategy.pdf". These two milestones should be completed at a later time if the tests in this iteration shows that further investments of "My web server" should be done. In order to implement tests concerning milestone 2 and 3 more time, knowledge and technical resources must be available, specially knowledge about IoT.

Milestone 2: “Implemented tests of “My web server” on IoT devices which with concrete and clear results show integration and adaptability between web server and IoT devices and further point out how integration and adaptability can be improved.”

Milestone 3: “Implemented tests of "My web server" on IoT devices which with concrete and clear results show the degree of securability and accessibility and further point out how securability and accessibility can be improved when it comes to using "My web server" on IoT devices.”

Approach to fulfill planned tests

Tests concerning Use-Cases from the legacy requirement

To test the use-cases from the legacy requirements (Appendix A document strategy.pdf section “Use Cases”) the software tester will write test cases and perform manual system tests regarding these test-cases. The software tester may in certain circumstances use static code reviewing or exploratory testing during these tests to get better understanding of the web-server and hence implement test that can provide more value.

Tests concerning Supplementary Specification from the legacy requirement

To test the supplementary requirements (Appendix A document strategy.pdf section “Supplementary Specification”) the software tester going to use different approaches and techniques.

<u>Regarding Req 1 “The web server should be responsive under high load"</u>
The software tester will use JMeter as testing tools and test the web server with different loads.

<u>Regarding Req2 "The web server must follow minimum requirements for HTTP 1.1"</u>
The software tester will write test-cases that tests if the server handles standard HTTP 1.1 request methods and perform manual system tests regarding these test-cases.

<u>Regarding “Req 4. The source code should be released under GPL-2.0”</u>
The software tester will do code-reviewing and investigate if any file that comes with the source code states which license "My web server" is released under.

<u>Regarding “Req 5. The access log should be viewable from a text editor”</u>
The software tester will do exploratory testing and see if this is true.

Tests concerning maintainability for the current state of “My web server”

To test the maintainability for the system the software tester going to use his/hers knowledge about Java and JUnit and perform code reviewing of the source code and test code. Mainly the software tester looking for if the code is clean, readable and build with goo design. The software tester is also going to examine automated units tests and integration tests if such tests exist. Of course the software tester will not be able to learn every function about the code but if high test coverage, clean code and good design is what find when investigating the code it will point to that the system have high maintainability.

Tests concerning usability for the current state of “My web server”

To test the usability of the system the software tester going to start with exploratory testing and using the knowledge the software tester have gathered from the system during the tests of the legacy requirements. New finding about the system will be document and can be used as feedback for further development and tests concerning “My web-server”.

Tests concerning compatibility with “My web server” and different java versions

To test test the compatibility with “My web server” and different java versions the software tester is going to do system testing with docker as a testing tool. The software tester is gong to build different containers with different java version installed and test if the the source code can be compiled and if “My web server” can be started with different java version.

Tests concerning compatible with “My web server” and common used operative systems on IoT devices

The software tester is going to explore how to setup environment for testing the three most used families of operative system for IoT devices according to Eclipse Foundation, reference https://f.hubspotusercontent10.net/hubfs/5413615/2020%20IoT%C2%A0Developer%20Survey%20Report.pdf , to see if “My web server” is compatibility with these families of operative systems.

The three most used operative system for IoT devices

- Linux
- FreeRTOS
- Windows

After more research about IoT OS and how to setup those OS the exact operative system to be tested in this iteration and what virtualization tools to be used for this setup is the following list.

- Windows10 IoT using VirtualBox
- Ubuntu Core using KVM
- RIOT using Docker

Ubuntu Core is a version of Linux’s distro Ubuntu made particularly for large container deployments and IoT devices. It use the same kernel, system software, and libraries as Ubuntu but on a much smaller scale and it is used to power robots, gateways, digital signs, etc.

RIOT is a open source operating system designed for working with IoT devices.

Windows10 IoT is windows 10 designed for use in embedded systems and IoT devices

Note: Previous described tests will be performed on operating systems Ubuntu 20.04 and hence the compatibility between "My web server" and operative systems based on Linux will be tested from the beginning of the tests.

If time over

Automation of tests

If time over and if previous tests in this iteration point to further investment in “My web server” the software tester should try to automate tests that he/she can automate experimenting with Postman/Newman and JMeter as testing tools for automation. If this is done these tests and the result from these tests should be documented in section “Experimental Test-cases” in document testCases.pdf and testRport.pdf

Tests concerning milestone 2 and 3

In this iteration the software tester is not planing to do any tests that concerning milestone 2 and 3, but if time over this is two concerns that the software tester can be started to plan for if the tests in this iteration point to further investment in “My web server” is worth doing.

Risk and Mitigation

Following is a list of the 3 biggest risk regarding this test plan

- Miscalculation of time
- Inadequate knowledge of web-servers, IoT devices and testing tools
- The software tester got sick

Mitigation of this three risks: The software tester will start with the most prioritized tests and keep implement more and more tests.