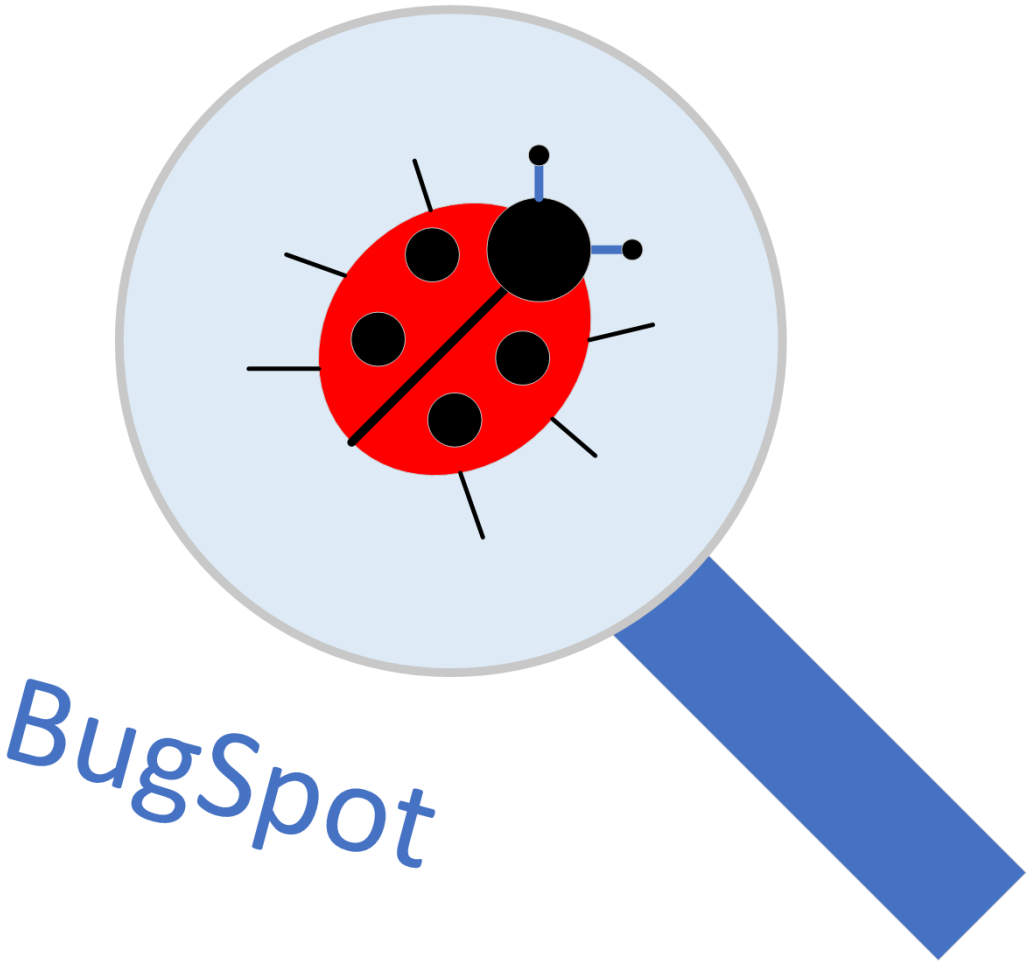
Software Testing Plan



By USN Software AS

Table of Contents

1. Introduction

This document showcases the testing execution and test documentation for the BugSpot bug tracking application. The tests will check the functionality of the different application modules and compatibility to different uses of the application, such as consistency across different browsers and web page hosting.

2. Test Resources, Personnel and Environment

This chapter presents the personnel and resources used for testing purposes during project testing as well as a description of the testing environment.

2.1 Personnel structure

The general personnel testing structure is that each member of the project will perform tests on the entirety of the project to assure that they are within acceptable margins of quality, performance, etc. “Software Requirements and Design” document will be used as a base reference; however, it is not strictly limited to it.

2.2 Resources and Testing Environment

Resources used in testing consist of the following:

- Different devices with available web browser application(s)
- Virtual Machine(s) for server testing

One of most prominent issues for a web-based application that may occur is that the application may perform differently on different web-browsers and device types. An example would be a web-shop that is available on a PC but doesn’t work as intended on mobile phones/tablets. Another example would be that the application works as intended only on a specific browser, where it was intended to work on multiple. For this kind of testing multiple PC’s (laptop, desktop), mobile devices (Android, iOS) and browsers (MS Edge, Chrome, etc.) will be utilized.

Another point of interest for web-based application testing is to test server-side performance, such as how responsive a web page is to user input, load times and traffic intensity (how many requests can the server handle at once). For this kind of tests, a special environment is required, in this case it will be done using a virtual machine (made using a free version of VMware, a software to create virtual machines on a local computer) to run a server using IIS (Internet Information Services).

3. Overview of different Test Types

Test strategy is about what that needs to be tested, how should the testing be done and when these tests should be performed. During programming each bit of the program is tested when it is added, updated, or removed. This strategy helps to find a significant number of problems that eventually occur in the development process; however, some problems also occur when the module is implemented. Examples being unwanted interactions between modules, visual glitches, and data errors.

For this project the general strategy of testing can be described by two main points:

- Testing the product by using it as originally intended
- Testing the product by actively attempting to break it in any possible way

The point of the first rule is to assess the quality of the main functionality, as described in the Software Requirements and Design document. This is done by using a test case sheet, described in chapter 5: Test Cases. The second rule helps in finding more obscure problems that only occur when the application is being used in an unintended way. This is of key importance as problems may occur quite frequently during deployment without being noticed by the developer.

3.1 What should be tested

The following list specifies what functions and features are being focused on during the test execution.

- Main functionality, functional requirements such as submitting a report/feature, being able to view the content that's been submitted, and viewing status of this reports/features
- Adding additional information to the database such as new software and versions
- Removing reports/features
- Editing the content of reports/features
- Commenting on reports/features
- Changing status of a report
- Assigning a person responsible for a report/bug
- Sorting and searching for reports
- Security, check for common hacks

4. Overview of different test types

This chapter will present the various methods for testing parts of the application.

- Unit Testing
  A method where the smallest testable part is tested individually. Example cases would be function testing of individual buttons, textboxes, etc. Often done during development.
- Regression Testing
  This test consists of simply running a program or a part of a program that has been subjected to change, checking if its original functionality has been altered. Common uses are when fixing issues known beforehand.
- Integration Testing
  A testing method, usually done after its individual components are unit tested. The testing is done by checking how each individual component and function work together with each other. An example would be testing an application which connects to a database where application, the database and the database API are tested at the same time. Passing the test is often decided by how well the system complies with functional requirements.
- System Testing
  Similar to integration testing, the completed system is tested in its entirety with functional requirements serving as points of reference. This type of testing is meant to check for design flaws, behavior, and customer expectations. This type of test is usually done last, after all the smaller parts are tested.
- Acceptance Testing
  This type of test is the final test before the tested product is considered finished. Acceptance test gauges how well does the software meet the end user requirements. It is mostly done by allowing the end users to use the software and give feedback to the developer. Should most end users agree that the quality of the product is sufficient, the test is considered to be passed.

5. Test Cases

This chapter will present an example table, the software modules that is used to test the modules step-by-step with a description of expected behaviour. If any of the steps are failed the module automatically fails, the test case. Table 1 shows an example of test case spreadsheet which is used, in this case, specifically for the bug report part of the reporting module. Spreadsheets for the other modules can be found here.

Table 1 Test case table for Report module, bug report web page

Test Case	OK	Failed	Expected Behaviour	Description/Commentary
Load Bug Report form			No values in input bars, only placeholder commentary indicated with	
Type in title in the title input area			Typed values are displayed in the Title input	
Choose Software from dropbar			Software bar is filled with software names, form is not submitted	
Choose Version from dropbar			Version bar is filled with version numbers, form is not submitted	
Type in the description in the description area			Typed values are displayed in the Description section	
Change Software choice			Title and Description values are retained in their respective fields, Version is reset to placeholder text and must be chosen again	
Select "Send Report" button			Values in the input fields are correctly sent and saved in the database, user is redirected to a "thank you" page	

6. Test Documentation

This chapter contains the documents of the executed test results and how many issues (bugs) have been found in the software. The following links are for the test cases that have been executed with the number of bugs that have been found during the first iteration of testing.

- Report Module Beta test – 7 bugs, 4 fixed.
  - TestCaseReportModule_executed.xlsx
- Management Module Beta test – 11 bugs, 11 fixed.
  - TestCaseManagementModule_executed.xlsx
- Status Module Beta test – 4 bugs, 3 fixed.
  - TestCaseStatusModule_executed.xlsx