A Guide to Functional Testing
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>What is Functional Testing?</th>
<th>03</th>
</tr>
</thead>
<tbody>
<tr>
<td>What do you Test in Functionality Testing?</td>
<td>03</td>
</tr>
<tr>
<td>How do you Perform Functional Testing?</td>
<td>04</td>
</tr>
<tr>
<td>Types of Functional Testing</td>
<td>05</td>
</tr>
<tr>
<td>Functional Testing Example</td>
<td>06</td>
</tr>
<tr>
<td>Difference Between Functional and Non-functional Test</td>
<td>08</td>
</tr>
<tr>
<td>Functionality Testing Tools</td>
<td>09</td>
</tr>
</tbody>
</table>
What is Functional Testing?

Functional testing is a type of black box testing that evaluates the conformance of a system or component with stated functional requirements. Functional testing specifies what the system does.

As the name specifies, the test verifies if all the functionalities of the application are working as expected and ready for release. Since it is black box testing, the tester will not know the internal structure of the app or the source code. Development team will create functional unit test cases with respect to user/business requirements and testers specify the functionality requirements based on the user perspective. Functional testing can be carried out by manual test teams or can be automated.

What do you test in functionality testing?

Functional testing also known as black box testing verifies

- Functionality
- End-to-end workflows
- Business Scenarios
- Data storage in the database
Example of a functional test scenario

Sample Test scenario: Verify user is able to add new reminders – SMS

01. Login to application
02. Mouse hover on schedule menu
03. Select Follow-up Reminders’
04. Click ‘New Reminder’ button
05. Select reminder type as ‘SMS’
06. Select the reminder value
07. Click ‘Update Reminders’

How do you perform functional testing?

Functional testing follows a step by step process,

- Identify functions that the application is intended to perform
- Create the input based on the function’s specifications
- Establish the output based on the function’s specifications
- Execute the written test cases
- Compare the final actual and expected outputs
What are the types of Functional testing?

Let's see the different functionality testing types and tools,

- Unit Testing
- Integration Testing
- Smoke Testing
- System Testing
- Regression Testing
- User Acceptance Testing
Examples of functional test cases

Unit Testing

Unit testing in simple terms validates if the separate units of code function properly. For example, it validates if a function, loop, method or a statement in the piece of code is working as intended. It is written by developers.

Example of a Unit test for ‘Login’ will look like,

- Navigate to login page
- In the ‘email’ field, enter the email address of the registered user
- Click the ‘Next’ button
- Enter the password of the registered user
- Click ‘Sign In’
- If success, go to next page
- If not, throw error message and go back to login page
Integration Testing

Integration testing verifies the interfaces or the flow between the units. The focus is given on verifying the integrating links. Sample integration test case for the test scenario: Add to bag -> My bag will look like,

<table>
<thead>
<tr>
<th>Test Case ID</th>
<th>Test Case Objective</th>
<th>Test Case Description</th>
<th>Expected Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Check the interface between the Add to cart and My cart module on the e-commerce application homepage</td>
<td>From homepage select product and click Add to cart button</td>
<td>Selected product should appear in the My cart folder</td>
</tr>
</tbody>
</table>

Here, the objective is to verify the integration between Add to cart and my cart folder.

Smoke Testing

Smoke testing is performed once after developers publish the ‘new build’, usually unstable – to evaluate critical functionalities of the system. The objective of the testing is to verify the most important functionalities of the system and not run an exhaustive testing process. Critical functionality test cases are chosen, run and verified. Both testers and developers perform smoke testing and the testing is ‘build’ focused.
Sanity Testing

Sanity Testing and smoke testing are used interchangeably in testing contexts. This is because both tests are run after the build is received. However, sanity testing, a subset of regression testing, focuses on executing test cases that are related to the changes made to the builds. In simple terms, when a build, usually stable, is received with minor changes, sanity test is performed. It is performed by testers and is ‘release focused’.

**Difference between functional and non-functional test**

<table>
<thead>
<tr>
<th>Functional Tests</th>
<th>Non-Functional Tests</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tests the actions and working functionalities of the application</td>
<td>Tests the non-functionality side of the application. I.e look and feel. Ex., Performance, Security, Usability</td>
</tr>
<tr>
<td>Performed first – applicable in Unit, Integration, System Testing</td>
<td>Performed after functionality tests – applicable in System Testing phase</td>
</tr>
<tr>
<td>Focus lies on user/business requirements</td>
<td>Focus lies on User Expectations/Experience</td>
</tr>
<tr>
<td>Tests ‘What’ the system should do</td>
<td>Tests ‘How’ the system should work</td>
</tr>
<tr>
<td>Can run manual and automated tests</td>
<td>Mostly automated tests</td>
</tr>
<tr>
<td>• Unit Testing</td>
<td>• Performance Testing</td>
</tr>
<tr>
<td>• Integration Testing</td>
<td>• Security Testing</td>
</tr>
<tr>
<td>• System Testing</td>
<td>• Recovery Testing</td>
</tr>
<tr>
<td>• Acceptance Testing</td>
<td>• Usability Testing</td>
</tr>
<tr>
<td><strong>Example:</strong> Verify the 'Place Order' functionality</td>
<td><strong>Example:</strong> Easy to navigate to 'Place Order’ button and 100 number of people should be able to place order concurrently irrespective of OS/browser/device configurations</td>
</tr>
</tbody>
</table>
Functionality Testing Tools

Some widely used functionality testing tools include,

Selenium

Selenium - Open-source, most well-known testing framework used for functional testing automation for browser-based web applications. It supports variety of programming languages for writing test scripts such as PHP, Java, C#, Groovy, Python, Ruby, and Perl. It also supports multiple browsers across OS platforms.

Ranorex Studio

Ranorex Studio - Built with Selenium web driver, it’s used for functional test automation for web, desktop and mobile applications. It supports GUI (Graphical User Interface) and offers record and play back for naïve testers.

TestComplete

TestComplete – Functional testing platform that offers solutions to automate functional testing for web, desktop and mobile applications. It supports a range of programming language such as DelphiScript, C++Script, C#Script, JavaScript, Python, VBScript, and JScript.

UFT

UFT – formerly known as QuickTestProfessional(QTP), by HP. Used to perform automated regression testing as well as functional testing. It offers a full-length feature set for API, web services, and GUI testing of desktop, web, and mobile applications across platforms. It is well known for its user friendliness, image-based object recognition feature, reusable test components, and automated documentation.

Katalon studio

Katalon studio – The software is built on top of the open-source automation frameworks like Selenium, Appium with a specialized IDE acting as a one stop shop for all API, WebUI, Desktop and Mobile testing and the combined capabilities among those.
Final Word

Functionality testing is critical for any software/application. Businesses that consider testing processes to be a gate keeper and skip this crucial stage to launch products quickly in the market will have to face the wrath of its end users when the app doesn’t work as intended.

Therefore, it is very important for teams to pick right combination of manual testing and automation tools to get the most out of functional testing and help businesses deliver a quality product.
About ZUCI

Zuci is a digital organization focused on the craft of assuring quality to software which we have perfected over the years. A perfect blend of test planning, engineering perfection, and customer-centricity in our DNA has enabled us to help small, medium and large organizations with superior quality engineering solutions. Learn More.

www.zucisystems.com
sales@zucisystems.com