180+ Web Application Testing Example Test Cases (Sample Checklist)

Last Updated: November 10, 2019

Web Application Testing Example Test Cases: This is a complete Testing Checklist for both Web-based and Desktop applications.

This is a very comprehensive list of Web Application Testing Example Test Cases/scenarios. Our goal is to share one of the most comprehensive testing checklists ever written and this is not yet done.

We'll keep updating this post in the future as well with more test cases and scenarios. If you don't have time to read it now, please feel free to share this with your friends and bookmark it for later.

About SoftwareTestingHelp

Helping our community since 2006! Most popular portal for Software professionals with **100 million+ visits!** You will absolutely love our tutorials on Software Testing, Development, Software Reviews and much more!

Join Over 200,000+ Testers

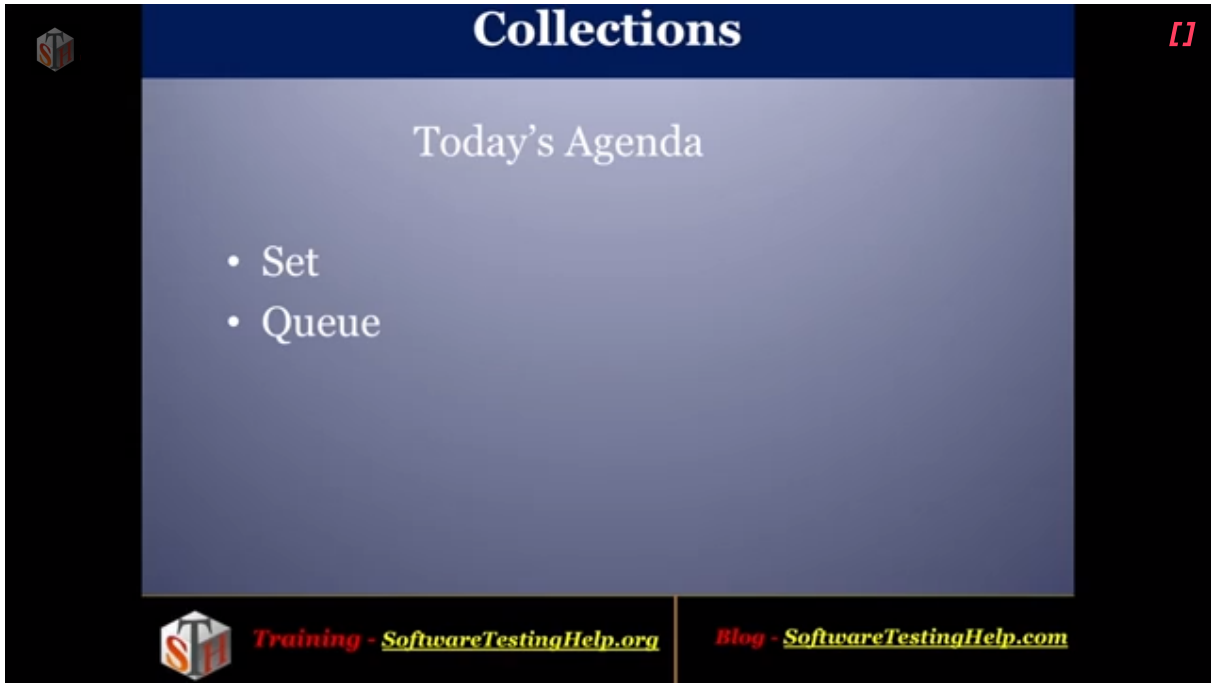
Get premium ebooks and testing tips.

Enter your email here...

SUBSCRIBE NOW!

Join Over 300,000+ Followers!

FEATURED VIDEOS



op FREE Training Tutorials

- Load Testing Buyer's Guide
- ppium Testing Tutorial
- Free QA Training
- Selenium Tutorials
- QTP/UFT Tutorials
- Quality Center QC Tutorials
- LoadRunner Tutorials

Make a testing checklist as an integral part of your Test case writing process. Using this checklist, you can easily create hundreds of Test cases for testing web or desktop applications.

These are all general test cases and should be applicable to almost all kinds of applications. Refer these tests while writing test cases for your project and I'm sure you will cover most of the testing types except the application-specific business rules provided in your SRS documents.

Though this is a common checklist, I recommend preparing a standard testing checklist tailored to your specific needs using below test cases in addition to application-specific tests.

Recommended Tool:

Before continuing to the test case writing process, we recommend downloading this Test case Management tool. This will ease your Test Plan and Test case writing process mentioned in this tutorial.

=> Download TestRail Test Case Management Tool

Importance of Using a Checklist for Testing

#2) A checklist helps to complete writing test cases quickly for new versions of the application.
#3) Reusing the test cases help to save money on resources to write repetitive tests.
#4) Important test cases will be covered always, thereby making it almost impossible to forget.
#5) The testing checklist can be referred by developers to ensure if the most common issues are fixed in the development phase itself.

Notes:

- Execute these scenarios with different user roles e.g. admin user, guest user etc.
- For web applications, these scenarios should be tested on multiple browsers like IE, FF, Chrome, and Safari with versions approved by the client.
- Test with different screen resolutions like 1024 x 768, 1280 x 1024, etc.
- An application should be tested on a variety of displays like LCD, CRT, Notebooks, Tablets, and Mobile phones.
- Test application on different platforms like Windows, Mac, Linux operating systems etc.

180+ Web Application Testing Example Test Cases

Assumptions: Assume that your application supports the following functionalities

- Forms with various fields
- Child windows
- The application interacts with the database
- Various search filter criteria and display results
- Image upload
- Send email functionality
- Data export functionality

General Test Scenarios

1. All mandatory fields should be validated and indicated by an asterisk (*) symbol.
2. Validation error messages should be displayed properly in the correct position.
3. All error messages should be displayed in the same CSS style (**For Example,** using red color)
4. General confirmation messages should be displayed using CSS style other than error messages style (**For Example,** using green color)
5. Tooltips text should be meaningful.
6. Drop-down fields should have the first entry as blank or text like ‘Select'.
7. ‘Delete functionality' for any record on a page should ask for a confirmation.
8. Select/deselect all records option should be provided if page supports record add/delete/update functionality
9. Amount values should be displayed with correct currency symbols.
10. Default page sorting should be provided.
11. Reset button functionality should set default values for all fields.
12. All numeric values should be formatted properly.
13. Input fields should be checked for the max field value. Input values greater than the specified max limit should not be accepted or stored in the database.
14. Check all input fields for special characters.
15. Field labels should be standard e.g. field accepting user's first name should be labeled

16. Check page sorting functionality after add/edit/delete operations on any record.
17. Check for timeout functionality. Timeout values should be configurable. Check application behavior after the operation timeout.
18. Check cookies used in an application.
19. Check if downloadable files are pointing to the correct file paths.
20. All resource keys should be configurable in config files or database instead of hard coding.
21. Standard conventions should be followed throughout for naming resource keys.
22. Validate markup for all web pages (validate HTML and CSS for syntax errors) to make sure it is compliant with the standards.
23. Application crash or unavailable pages should be redirected to the error page.
24. Check the text on all pages for spelling and grammatical errors.
25. Check numeric input fields with character input values. A proper validation message should appear.
26. Check for negative numbers if allowed for numeric fields.
27. Check the number of fields with decimal number values.
28. Check the functionality of buttons available on all pages.
29. The user should not be able to submit a page twice by pressing the submit button in quick succession.
30. Divide by zero errors should be handled for any calculations.
31. Input data with the first and last position blank should be handled correctly.

GUI And Usability Test Scenarios

1. All fields on a page (**For Example,** text box, radio options, drop-down lists) should be aligned properly.
2. Numeric values should be justified correctly unless specified otherwise.
3. Enough space should be provided between field labels, columns, rows, error messages, etc.
4. The scrollbar should be enabled only when necessary.
5. Font size, style, and color for headline, description text, labels, infield data, and grid info should be standard as specified in SRS.
6. The description text box should be multi-lined.
7. Disabled fields should be greyed out and users should not be able to set focus on these

8. Upon click of an input text field, the mouse arrow pointer should get changed to the cursor.
9. The user should not be able to type in drop-down select lists.
10. Information filled by users should remain intact when there is an error message on page submit. The user should be able to submit the form again by correcting the errors.
11. Check if proper field labels are used in error messages.
12. Drop-down field values should be displayed in defined sort order.
13. Tab and Shift+Tab order should work properly.
14. Default radio options should be pre-selected on the page load.
15. Field-specific and page-level help messages should be available.
16. Check if the correct fields are highlighted in case of errors.
17. Check if the drop-down list options are readable and not truncated due to field size limits.
18. All buttons on a page should be accessible by keyboard shortcuts and the user should be able to perform all operations using a keyboard.
19. Check all pages for broken images.
20. Check all pages for broken links.
21. All pages should have a title.
22. Confirmation messages should be displayed before performing any update or delete operation.
23. Hourglass should be displayed when the application is busy.
24. Page text should be left-justified.
25. The user should be able to select only one radio option and any combination for checkboxes.

Test Scenarios For Filter Criteria

1. The user should be able to filter results using all parameters on the page.
2. Refine search functionality should load the search page with all user-selected search parameters.
3. When there are at least one filter criteria required to perform the search operation, make sure the proper error message is displayed when the user submits the page without selecting any filter criteria.
4. When at least one filter criteria selection is not compulsory, the user should be able to

submit the page and the default search criteria should get used to query results.
5. Proper validation messages should be displayed for all invalid values for filter criteria.

Test Scenarios For Result Grid

1. Page loading symbol should be displayed when it's taking more than default time to load the result page.
2. Check if all the search parameters are used to fetch data shown on the result grid.
3. The total number of results should be displayed in the result grid.
4. Search criteria used for searching should be displayed in the result grid.
5. Result grid values should be sorted by default column.
6. Sorted columns should be displayed with a sort icon.
7. Result grids should include all the specified columns with correct values.
8. Ascending and descending sorting functionality should work for columns supported by data sorting.
9. Result grids should be displayed with proper column and row spacing.
10. Pagination should be enabled when there are more results than the default result count per page.
11. Check for Next, Previous, First and Last page pagination functionality.
12. Duplicate records should not be displayed in the result grid.
13. Check if all the columns are visible and a horizontal scrollbar is enabled if necessary.
14. Check the data for dynamic columns (columns whose values are calculated dynamically based on the other column values).
15. For result grids showing reports check ‘Totals' row and verify the total for every column.
16. For result grids showing reports check ‘Totals' row data when pagination is enabled and the user gets navigated to the next page.
17. Check if proper symbols are used for displaying column values e.g. % symbol should be displayed for percentage calculation.
18. Check result grid data to know if the date range is enabled.

Test Scenarios For A Window

1. Check if the default window size is correct.
2. Check if the child window size is correct

set on the first input field of the screen).
4. Check if child windows are getting closed on closing parent/opener window.
5. If the child window is opened, the user should not be able to use or update any field in the background or parent window
6. Check window minimize, maximize, and close functionality.
7. Check if the window is re-sizable.
8. Check scroll bar functionality for parent and child windows.
9. Check cancel button functionality for the child window.

Database Testing Test Scenarios

1. Check if correct data is getting saved in the database upon a successful page submit.
2. Check values for columns that are not accepting null values.
3. Check for data integrity. Data should be stored in single or multiple tables based on the design.
4. Index names should be given as per the standards e.g. IND_<Tablename>_<ColumnName>
5. Tables should have a primary key column.
6. Table columns should have description information available (except for audit columns like created date, created by, etc.)
7. For every database add/update operation log should be added.
8. Required table indexes should be created.
9. Check if data is committed to the database only when the operation is successfully completed.
10. Data should be rolled back in case of failed transactions.
11. Database name should be given as per the application type i.e. test, UAT, sandbox, live (though this is not a standard it is helpful for database maintenance)
12. Database logical names should be given according to the database name (again this is not standard but helpful for DB maintenance).
13. Stored procedures should not be named with a prefix “sp_”
14. Check if values for table audit columns (like created date, created by, updated, updated by, is deleted, deleted data, deleted by, etc.) are populated properly.
15. Check if input data is not truncated while saving. Field length shown to the user on the page and in database schema should be the same.
16. Check numeric fields with minimum, maximum, and float values

18. Check if the radio button and drop-down list options are saved correctly in the database.
19. Check if the database fields are designed with the correct data type and data length.
20. Check if all the table constraints like a Primary key, Foreign key, etc. are implemented correctly.
21. Test stored procedures and triggers with sample input data.
22. Input field leading and trailing spaces should be truncated before committing data to the database.
23. Null values should not be allowed for the Primary key column.

Test Scenarios For Image Upload Functionality

(Also applicable for other file upload functionality)
1. Check for uploaded image path.
2. Check image upload and change functionality.
3. Check image upload functionality with image files of different extensions (**For Example,** JPEG, PNG, BMP, etc.)
4. Check image upload functionality with images having space or any other allowed special character in the file name.
5. Check duplicate name image upload.
6. Check image upload with image size greater than the max allowed size. The Proper error message should be displayed.
7. Check image upload functionality with file types other than images (**For Example,** txt, doc, pdf, exe, etc.). A proper error message should be displayed.
8. Check if images of specified height and width (if defined) are accepted otherwise rejected.
9. The image upload progress bar should appear for large size images.
10. Check if the cancel button functionality is working in between the upload process.
11. Check if file selection dialog shows only supported files listed.
12. Check multiple images upload functionality.
13. Check image quality after upload. Image quality should not be changed after upload.
14. Check if the user is able to use/view the uploaded images.

Test Scenarios For Sending Emails

(Test cases for composing or validating emails are not included here)

1. The email template should use standard CSS for all emails.
2. Email addresses should be validated before sending emails.
3. Special characters in the email body template should be handled properly.
4. Language-specific characters (**For Example,** Russian, Chinese or German language characters) should be handled properly in the email body template.
5. Email subject should not be blank.
6. Placeholder fields used in the email template should be replaced with actual values e.g. {Firstname} {Lastname} should be replaced with an individual's first and last name properly for all the recipients.
7. If reports with dynamic values are included in the email body and report data should be calculated correctly.
8. Email sender name should not be blank.
9. Emails should be checked in different email clients like Outlook, Gmail, Hotmail, Yahoo! mail, etc.
10. Check to send email functionality using TO, CC and BCC fields.
11. Check plain text emails.
12. Check HTML format emails.
13. Check email header and footer for company logo, privacy policy, and other links.
14. Check emails with attachments.
15. Check to send email functionality to single, multiple or distribution list recipients.
16. Check if a reply to the email address is correct.
17. Check to send the high volume of emails.

Test Scenarios For Excel Export Functionality

1. The file should get exported in the proper file extension.
2. The file name for the exported Excel file should be as per the standards, **For Example,** if the file name is using the timestamp, it should get replaced properly with an actual timestamp at the time of exporting the file.
3. Check for date format if exported Excel file contains the date columns.
4. Check number formatting for numeric or currency values. Formatting should be the same as shown on the page.
5. The exported file should have columns with proper column names.
6. Default page sorting should be carried in the exported file as well

numbers, etc. values for all pages.
8. Check if the data displayed on a page and exported Excel file is the same.
9. Check export functionality when pagination is enabled.
10. Check if the export button is showing proper icon according to the exported file type, **For Example,** Excel file icon for xls files
11. Check export functionality for files with very large size.
12. Check export functionality for pages containing special characters. Check if these special characters are exported properly in the Excel file.

Performance Testing Test Scenarios

1. Check if the page load time is within the acceptable range.
2. Check the page load on slow connections.
3. Check the response time for any action under a light, normal, moderate, and heavy load conditions.
4. Check the performance of database stored procedures and triggers.
5. Check the database query execution time.
6. Check for load testing of the application.
7. Check for the Stress testing of the application.
8. Check CPU and memory usage under peak load conditions.

Security Testing Test Scenarios

1. Check for SQL injection attacks.
2. Secure pages should use the HTTPS protocol.
3. Page crash should not reveal application or server info. The error page should be displayed for this.
4. Escape special characters in the input.
5. Error messages should not reveal any sensitive information.
6. All credentials should be transferred over an encrypted channel.
7. Test password security and password policy enforcement.
8. Check application logout functionality.
9. Check for Brute Force Attacks.

11. Session tokens should be transmitted over a secured channel.
13. The password should not be stored in cookies.
14. Test for Denial of Service attacks.
15. Test for memory leakage.
16. Test unauthorized application access by manipulating variable values in the browser address bar.
17. Test file extension handing so that exe files are not uploaded and executed on the server.
18. Sensitive fields like passwords and credit card information should not have to autocomplete enabled.
19. File upload functionality should use file type restrictions and also anti-virus for scanning uploaded files.
20. Check if directory listing is prohibited.
21. Passwords and other sensitive fields should be masked while typing.
22. Check if forgot password functionality is secured with features like temporary password expiry after specified hours and security question is asked before changing or requesting a new password.
23. Verify CAPTCHA functionality.
24. Check if important events are logged in log files.
25. Check if access privileges are implemented correctly.

Penetration Testing test cases – I've listed around 41 test cases for Penetration Testing on this page.

*I 'd really like to thank **Devanshu Lavaniya** (Sr. QA Engineer working for I-link Infosoft) for helping me to prepare this comprehensive testing checklist.*

I've tried to cover almost all standard test scenarios for Web and Desktop application functionality. But still, I know that this is not a complete checklist. Testers on different projects have their own testing checklist based on their experience.

Updated:

100+ Ready-To-Execute Test Cases (Checklists)

You Can Use this list to test the most common components of AUT

How to test the most common components of your AUT effectively, every single time?

This article is a list of common validations on most widely found elements of AUT – that is put together for the convenience of testers (especially in the agile environment where frequent short-term releases happen).

Every AUT (Application Under Test) is unique and has a very specific business purpose. The individual aspects (modules) of the AUT cater to different operations/actions that are crucial to the success of the business that the AUT supports.

Though each AUT is designed differently, individual components/fields that we encounter on most pages/screens/applications are the same with more or less similar behavior.

Some Common Components of AUT:

- Save, Update, Delete, Reset, Cancel, OK – links/buttons- whose functionality is the label of the object indicates.
- Text box, dropdowns, checkboxes, radio buttons, date control fields – that work the same way every time.
- Data grids, impacted areas, etc. to facilitate reports.

The way these individual elements contribute to the overall functionality of the application might be different but the steps to validate them are always the same.

Let's continue with the list of most common validations for Web or Desktop application pages/forms.

Note: The actual result, expected result, test data and other parameters that are typically a part of a test case are omitted for the sake of simplicity – A general checklist approach is employed.

Purpose of this comprehensive checklist:

The primary purpose of these checklists (or test cases) is to ensure maximum test coverage on field level validations without spending too much time, at the same time not compromise the quality of testing them.

After all, confidence in a product can only be attained by testing every single element to the best extent possible.

The Complete Checklist (Test Cases) For Most Common Components Of AUT

Note: You can use these checklists as it is in Microsoft Excel format (download provided at the end of the article). You can even track the test execution in the same file with pass/fail results and status.

This could be an all-in-one resource for QA teams to test and track the most common components of AUT. You can add or update test cases specific to your application and make it an even more comprehensive list.

Checklist #1: Mobile Testing Checklist

Module Name:	

Module Impact over the application:
Module Flow:
Menu & Submenu:
Spellings and Order & Suitability:
Control for each submenu:

Checklist #2: Forms/Screens Testing Checklist

Form Functionality:
Form Impact over the application:
Form Flow:
Designing:
Alignments:
Title:
Field Names:
Spellings:
Mandatory Marks:
Alerts to Mandatory fields:
Buttons:
Default Cursor Position:

The page before entering any data:
Page after entering data:

Checklist #3: Textbox Field Testing Checklist

Text Box:

	ADD (In add screen)	EDIT (in Edit screen)
Characters		
Special Characters		
Numbers		
Limit		
Alert		
Spelling & Grammar in Alert message:		

BVA (Size) for Text Box:

Min —>—> Pass
Min-1 —> —> Fail
Min+1 —> —> Pass
Max-1 —> —> Pass
Max+1 —> —> Fail
Max —> —> Pass

ECP for Text Box:

–	–
–	–

Checklist #4: List-box or Drop-down List Testing Checklist

List Box/Dropdown:

	ADD (In add screen)	EDIT (in Edit screen)
Header		
The correctness of Existed Data		
Order of Data		
Selection and Deselection		
Alert:		
Spelling and Grammar of Alert message		
Cursor after alert		
Reflection of Selection and Deselection in remaining fields		

Checklist #5: Checkbox Field Testing Checklist

CheckBox:

	ADD (In add screen)	EDIT (in Edit screen)

Default Selection		
Action after selection		
Action after de-selection		
Selection and Deselection		
Alert:		
Spelling and Grammar of Alert message		
Cursor after alert		
Reflection of Selection and Deselection in remaining fields		

Checklist #6: Radio Button Testing Checklist

Radio button:

	ADD (In add screen)	**EDIT (in Edit screen)**
Default Selection		
Action after selection		
Action after de-selection		
Selection and Deselection		
Alert:		
Spelling and Grammar of Alert message		

Cursor after alert		
Reflection of Selection and Deselection in remaining fields		

Checklist #7: Date Field Test Scenarios

Date field:

	ADD (In add screen)	**EDIT (in Edit screen)**
Default date display		
Design of calendar		
Navigation for different months and years in date control		
Manual Entry in date text box		
Date format and uniformity with the overall application		
Alert:		
Spelling and Grammar of Alert message		
Cursor after alert		
Reflection of Selection and Deselection in remaining fields		

Checklist #8: Save Button Testing Scenarios

Save/update:

	ADD (In add screen)	EDIT (in Edit screen)
Without giving any data:		
With only mandatory fields:		
With All fields:		
With Max limit:		
With min limit		
Spelling & Grammar in Confirmation Alert message:		
Cursor		
Duplication of Unique fields:		
Spelling & Grammar in duplication Alert message:		
Cursor		

Checklist #9: Cancel Button Test Scenarios

Cancel:

With data in all fields		
With only mandatory fields:		

With all fields:		

Checklist #10: Delete Button Testing Points

Delete:

	EDIT (in Edit screen)
Delete the record which is not used anywhere in the application	
Delete the record which has a dependency	
Add the new record with same deleted details again	

Checklist #11: To Verify Impacted Areas after Save or Update

After Saving/updating:

Display in View	
Reflection in impacted forms in the application	

Checklist #12: Data Grid Testing List

Data Grid:

Grid Title and spelling	
Form Before giving any data	

Spellings	
Alignments	
S No	
Field Names & Order	
The correctness of Existed data	
Order of Existed data	
Alignment of Existed data	
Page navigators	
Data when navigating with different pages	

Edit Link Functionality

Page after Edit:	
Title and spellings	
Existed data of the Selected record in each field	
Buttons	

While this list might not be exhaustive, it is indeed extensive.

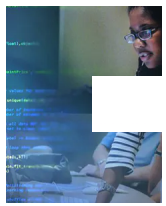
DOWNLOAD ==> You can download all these checklists in MS Excel format: Download in Excel format

1. Depending on your need, additional tests under each category/for each field can be added or existing fields can be removed. In other words, these lists are completely customizable.
2. When in need to include field-level validations to your test suites, all you have to do is pick up the respective list and use it for the screen/page that you would like to test.
3. Maintain the checklist by updating the pass/fail status to make this a one-stop-shop for listing features, validating them and recording the test results.

Please feel free to make this a complete checklist by adding more Test cases/scenarios or negative test cases in the comments section below.

Also, I'd appreciate if you'd share this with your friends!

PREV Tutorial | **NEXT Tutorial**



Recommended Reading

- How to Write Test Cases: The Ultimate Guide with Examples
- Website Cookie Testing & Test Cases for Testing Web Application Cookies
- Sample Test Case Template with Test Case Examples [Download]
- QA Testing Tools
- Web Application Security Testing Guide

- Installing your Application on Device and Start Testing from Eclipse
- TDD Vs BDD - Analyze The Differences With Examples

Database Testing, Testing Tips and Resources, Types of Testing

‹ 16 Characteristics of a Great Software Tester

› How to Use Poka-Yoke (Mistake Proofing) Technique to Improve Software Quality

360 thoughts on “180+ Web Application Testing Example Test Cases (Sample Checklist)”

← Older Comments

amrutha

i need mobile app testing test cases for performance . UI,

Reply

Bhumika Dipak Popat

I want how to write test cases for intigration testing

Reply

Seren

Thank you. This is really helpful

Reply

Cvetelin Borislavov

I need test cases for a testing website and the test cases to cover only UI testing, Functionality testing and Usability testing.

Thanks!!!

Reply

Udhaya

It’s a very useful article

Reply

ravi

After reading this, Truly felt to post this msg.
1001 thopo ki Salaam!!! You guys are awesome

Reply

vj

Very useful.Please upload Functionality testing and SAP data analytics

Reply

Yash Gandhi

I want test plan for Web based railway reservation system

Reply

malika

great info , good thanks .

Reply

Davidbilla

Thank you for the info!

Reply

← Older Comments

Leave a Comment

Name *

Email *

POST COMMENT