

INDEX

Sr.No	Exercise	Remark	Signature
1.	Find out defects (layout inconsistencies, spelling errors, and the like) in the image below: [Old Google Homepage Test]		
2.	How to write Test Cases for a Login Page 1. Email/Phone Number/Username Textbox 2. Password Textbox 3. Login Button 4. Remember Me Checkbox 5. Keep Me Signed In Checkbox 6. Forgot Password Link 7. Sign up/Create an account Link		
3.	Write Test Cases for ATM (Test Scenarios ATM Machine)		
4.	Write Test Cases for Log in Page.		
5.	Write the Test Scenarios of Gmail inbox functionality.		
6.	White box testing and Black box testing.		
7.	Retesting and regression testing.		

Exercise No: 1

Q. Find out defects (layout inconsistencies, spelling errors, and the like) in the image below: [Old Google Homepage Test]



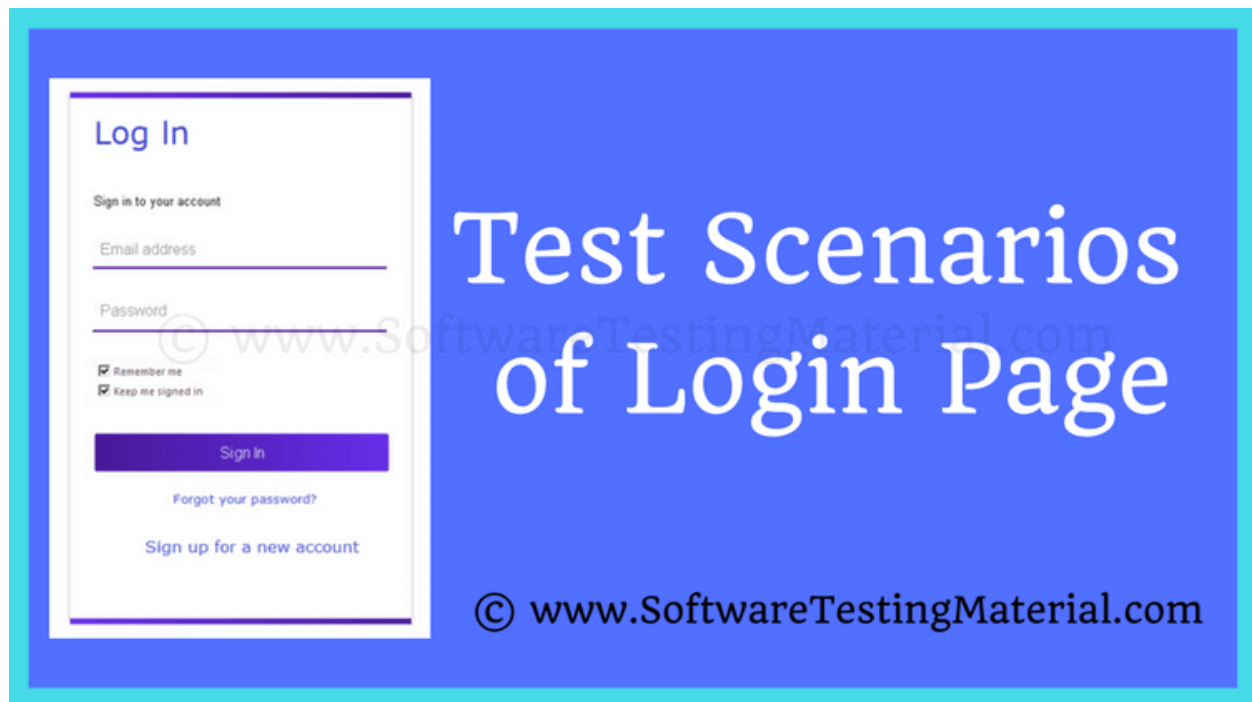
Ans:

- 1) Layout of the Old Google Homepage is not appropriate in nature.
- 2) There is Spelling mistake like Gooogle.
- 3) Status bar showing two times Done word.
- 4) The Search button is having Unlucky word instead of Lucky.
- 5) The fonts are not having proper color scheme and alignment.
- 6) The home page has not properly Layout.

Exercise No: 2

Q. How to write Test Cases for a Login Page

8. Email/Phone Number/Username Textbox
9. Password Textbox
10. Login Button
11. Remember Me Checkbox
12. Keep Me Signed In Checkbox
13. Forgot Password Link
14. Sign up/Create an account Link



Ans:

User Interface Test Scenarios of Login Page:

1. Verify that the login screen contains elements such as Username, Password, Sign in button, Remember password check box, Forgot password link, and create an account link.
2. Verify that all the fields such as Username, Password has a valid placeholder
3. Verify whether all the text boxes have a minimum and maximum length.
4. Verify that the labels float upward when the text field is in focus or filled (In case of the floating label)
5. Verify to see if the font style and size of the labels, as well as the text on each object, are clearly visible.

6. Verify that the application's user interface (UI) is responsive, so it will adapt to different screen resolutions and devices.
7. Verify the login page and all the fields in the login page are displaying without any break in different browsers.

Functional Test Scenarios of Login Page:

1. Verify that cursor is focused on the "Username" text box on the page load (login page)
2. Verify that tab functionality is working properly or not
3. Verify that Enter/Tab key works as a substitute for the Sign-in button
4. Verify that the User is able to Login with Valid Credentials
5. Verify that the User is not able to Login with an invalid Username and invalid Password
6. Verify that the User is not able to Login with a Valid Username and invalid Password
7. Verify that the User is not able to log in with an invalid Username and Valid Password
8. Verify that the User is not able to log in with a blank Username or Password
9. Verify that the User is not able to Login with inactive credentials
10. Verify that the reset button clears the data from all the text boxes in the login form
11. Verify that the login credentials, mainly password stores in a database in an encrypted format
12. Verify that clicking on the browser back button after successful login should not take the User to log out mode
13. Verify that validation message is displayed in the case when User leaves Username or Password as blank
14. Verify that validation message is displayed in case of exceeding the character limit of the Username and Password fields
15. Verify that validation message is displayed in case of entering special character in the Username and password fields
16. Verify that the "Keep me logged in" checkbox is unselected by default (depends on business logic, it may be selected or unselected)
17. Verify that the timeout of the login session (Session Timeout)
18. Verify that the logout link is redirected to login/home page
19. Verify that User is redirected to appropriate page after successful login
20. Verify that the User is redirected to the Forgot password page when clicking on the Forgot Password link
21. Verify that the User is redirected to the Create an account page when clicking on the Signup / Create an account link
22. Verify that the User should be able to login with the new password after changing the password
23. Verify that the user should not be able to login with the old password after changing the password
24. Verify that spaces should not be allowed before any password characters attempted
25. Verify whether the user is still logged in after a series of actions such as sign-in, close the browser, and reopen the application.
26. Verify that the ways to retrieve the password if the user forgets the password

Exercise No: 3

Q. Write Test Cases for ATM (Test Scenarios ATM Machine)




- **Test Cases for ATM:**

1. Verify the 'ATM Card Insertion Slot' is as per the specification
2. Verify the ATM machine accepts card and PIN details
3. Verify the error message by inserting a card incorrectly
4. Verify the error message by inserting an invalid card (Expired Card)
5. Verify the error message by entering an incorrect PIN
6. Verify that the user is asked to enter the PIN after inserting a valid ATM Card
7. Verify that PIN is encrypted
8. Verify that there is an action like blocking of card occurs when the total no. of incorrect PIN attempts get surpassed
9. Verify the user is allowed to do only one cash withdrawal transaction per PIN request
10. Verify the machine logs out of the user session immediately after successful withdrawal
11. Verify the message when there is no money in the ATM
12. Verify the language selection functionality
13. Verify the cash withdrawal functionality by entering some valid amount
14. Verify the cash withdrawal functionality by entering an amount less than 100
15. Verify the cash withdrawal functionality by entering an amount greater than the total available balance in the account.
16. Verify the cash withdrawal functionality by entering an amount greater than per day limit
17. Verify the user is allowed to enter the amount again in case the amount entered is not valid. A proper message should be displayed.

Exercise No: 4

Q. Write Test Cases for Log in Page.

Project Name:	Google Email	 www.SoftwareTestingMaterial.com
Module Name:	Login	
Reference Document:	If any	
Created by:	Rajkumar	
Date of creation:	DD-MMM-YY	
Date of review:	DD-MMM-YY	

TEST CASE ID	TEST SCENARIO	TEST CASE	PRE-CONDITION	TEST STEPS	TEST DATA	EXPECTED RESULT	POST CONDITION	ACTUAL RESULT	STATUS (PASS/FAIL)
TC_LOGIN_001	Verify the login of Gmail	Enter valid User Name and valid Password	1. Need a valid Gmail Account to do login	1. Enter User Name 2. Enter Password 3. Click "Login" button	<Valid User Name> <Valid Password>	Successful login	Gmail inbox is shown		
TC_LOGIN_001	Verify the login of Gmail	Enter valid User Name and invalid Password	1. Need a valid Gmail Account to do login	1. Enter User Name 2. Enter Password 3. Click "Login" button	<Valid User Name> <Invalid Password>	A message "The email and password you entered don't match" is shown			
TC_LOGIN_001	Verify the login of Gmail	Enter invalid User Name and valid Password	1. Need a valid Gmail Account to do login	1. Enter User Name 2. Enter Password 3. Click "Login" button	<Invalid User Name> <Valid Password>	A message "The email and password you entered don't match" is shown			
TC_LOGIN_001	Verify the login of Gmail	Enter invalid User Name and invalid Password	1. Need a valid Gmail Account to do login	1. Enter User Name 2. Enter Password 3. Click "Login" button	<Invalid User Name> <Invalid Password>	A message "The email and password you entered don't match" is shown			

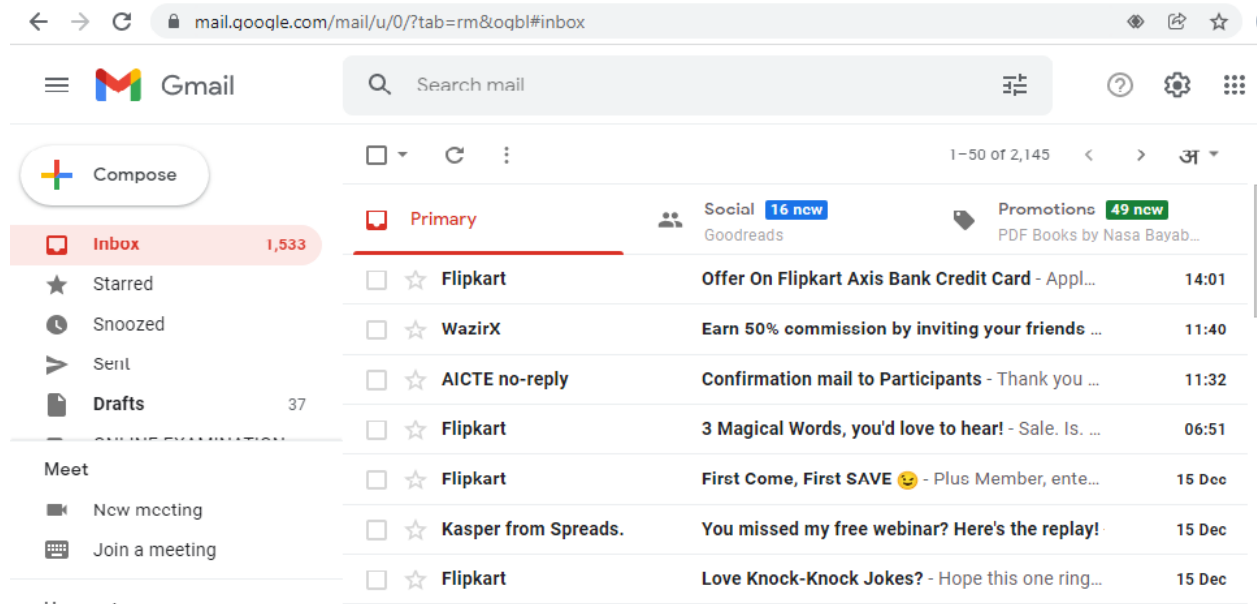
- TEST CASES FOR LOGIN PAGE:

1. Enter valid User Name and valid Password
2. Enter valid User Name and invalid Password
3. Enter invalid User Name and valid Password
4. Enter invalid User Name and invalid Password

1. Verify that clicking on the browser back button after successful logout should not take the User to a logged-in mode
2. Verify that there is a limit on the total number of unsuccessful login attempts (No. of invalid attempts should be based on business logic. Based on the business logic, User will be asked to enter the captcha and try again or user will be blocked)
3. Verify that the password is in encrypted form (masked format) when entered in the password field.
4. Verify the password can be copy-pasted. System shouldn't allow users to copy paste password.
5. Verify that encrypted characters in the "Password" field should not allow deciphering if copied
6. Verify that the "Remember password" checkbox is unselected by default (depends on business logic, it may be selected or unselected)
7. Verify whether the login form is revealing any security information by viewing the page source
8. Verify that the login page is vulnerable to SQL injection.
9. Verify whether Cross-site scripting (XSS) vulnerability works on a login page. XSS vulnerability may be used by hackers to bypass access controls.

Exercise No: 5

Q. Write the Test Scenarios of Gmail inbox functionality.



1. Verify that all the read and unread emails are displayed in the inbox
2. Verify that the recently received email or unread emails are highlighted in bold in the Inbox section.
3. Verify that the recently received email has correct sender's name or email id, subject of the email, its preview and date or time.
4. Verify that the recently received email's sender's name or email id, subject of the email, and date or time should be in bold and preview text shouldn't be in bold.
5. Verify that the attachment icon is displayed next to the preview text of the email, if the email has any attachment.
6. Verify that the Archive, Delete, Mark as read, Snooze options are displaying on hovering the unread email.
7. Verify that the Archive, Delete, Mark as unread, Snooze options are displaying on hovering the read email.
8. Verify that the Email id, Add to contacts, Open detailed view, Send email, Send message, Start video call, Schedule event options are displaying when we hover on the name/email of the read/unread email.
9. Verify that the user is navigated to the email content when clicking on the email in the inbox.
10. Verify that the content of the email is displayed correctly without any formatting issues.
11. Verify that the attachment in the email is downloadable or not.
12. Verify that the attachments can be downloaded as a single zip file.
13. Verify that the attachments can be downloaded individually.
14. Verify that the attachments can be viewable in the browser itself without downloading.

15. Verify that the attachment is downloading in zip format, if the attachment size is more than 1 MB.
16. Verify that the attachments are scanned for viruses once we try to download the file.
17. Verify that the Reply and Forward buttons are displaying in the bottom of the email content.
18. Verify that all the read emails are not highlighted.
19. Verify that unread emails count is displayed beside 'Inbox' text in the left sidebar of Gmail.
20. Verify that unread emails count is increased as per the number of new emails we received.
21. Verify that the unread emails count is increased when we mark an email as unread.
22. Verify that the unread emails count is decreased when we mark an email as read or opened.
23. Verify that email recipients in CC are visible to all the users whose emails are present.
24. Verify that email recipients in BCC are not visible to other users in the TO, CC or BCC section.
25. Verify that email can be received from other domains like Hotmail, Outlook, Yahoo mail or any other company domains.

Exercise No: 6

Q.White box testing and Black box testing.

Ans:

White Box testing

The term 'white box' is used because of the internal perspective of the system. The **clear box or white box, or transparent box** name denotes the ability to see through the software's outer shell into its inner workings.

It is performed by Developers, and then the software will be sent to the testing team, where they perform black-box testing. The main objective of white-box testing is to test the application's infrastructure. It is done at lower levels, as it includes unit testing and integration testing. It requires programming knowledge, as it majorly focuses on code structure, paths, conditions, and branches of a program or software. The primary goal of white-box testing is to focus on the flow of inputs and outputs through the software and strengthening the security of the software.

It is also known as structural testing, clear box testing, code-based testing, and transparent testing. It is well suitable and recommended for algorithm testing.

Black Box testing

The primary source of black-box testing is a specification of requirements that are stated by the customer. It is another type of manual testing. It is a software testing technique that examines the functionality of the software without knowing its internal structure or coding. It does not require programming knowledge of the software. All test cases are designed by considering the input and output of a particular function. In this testing, the test engineer analyzes the software against requirements, identifies the defects or bugs, and sends it back to the development team.

In this method, the tester selects a function and gives input value to examine its functionality, and checks whether the function is giving the expected output or not. If the function produces the correct output, then it is passed in testing, otherwise failed.

Black box testing is less exhaustive than White Box and Grey Box testing methods. It is the least time-consuming process among all the testing processes. The main objective of implementing black box testing is to specify the business needs or the customer's requirements.

In other words, we can say that black box testing is a process of checking the functionality of an application as per the customer's requirement. Mainly, there are three types of black-box testing: **functional testing, Non-Functional testing, and Regression testing**. Its main objective is to specify the business needs or the customer's requirements.

Exercise No: 7

Q.Re-testing and regression testing.

Re-Testing:

Re-testing is a testing technique that is used to verify that defects that were found in a software system have been fixed correctly. This technique involves retesting a specific part of the software that was previously identified as having a defect. Re-testing is typically performed after a bug has been fixed, and the software has been reworked to address the defect. The objective of re-testing is to ensure that the specific issue that was previously identified has been resolved and the software now works as expected.

Retesting is done to make sure that the bug is fixed and whether failed functionality is working fine or not, this is a kind of verification method followed in the testing field for the fixed bugs. Most of the testers have confusion with Regression and Retesting.

Retesting is known as planned testing. Retesting is used to ensure the test cases which failed in last execution are fixed. Retesting is used only for failed test cases. Retesting can not be automated. Retesting has higher priority than regression testing.

It takes less time as it focusses only on the exploration of a certain defect of the product. Retesting is performed in the same environment and with the same data but a new build is used.

Pros-

- Verification of the resolved problem and is functioning as intended.
- Verification time is less because it is only limited to the specific problem or feature.
- The level of the program's or product's is raised.

Cons-

- A new build is needed for the purpose of defect verification.
- No automation of the test cases.
- Time consuming because of the re-testing of failed test cases.
- Test cases can only be obtained after the start of testing and not before that.
- The same data and environment is used to carry out with the new build.

Regression Testing:

Regression testing is a testing technique that is used to verify that changes made to a software system do not have unintended consequences on previously tested functionality. This technique involves retesting the entire system or a part of the system to ensure that the existing functionality of the software is still working as expected after making changes. Regression testing can be performed manually or using automated tools. Regression testing is performed after making any change in the software system, including bug fixes, enhancements, or new features.

Regression Testing is a type of software testing, which is used to verify that modifications in the software or the environment have not caused any unintended adverse side effects.

Regression testing is known as a generic testing. Regression testing is to ensure that changes have not affected the unchanged part of product. Regression testing is used for passed test cases. Defect verification is not coming under regression testing.

Regression testing can be done either in automation or manual testing. Regression testing has lower priority than retesting testing but in some cases it can be done in parallel with retesting. It takes more time as it explores the whole application to uncover the bugs.

Regression testing is only performed after a code modification, in case of implementation of a new feature, or an enhancement.

Pros-

- Significant role in agile environment in ensuring continuity of business operations.
- Aids in identification of bugs in the software
- Constant additions will not hamper the integrity of application
- Helps in achieving higher CSI (Customer Satisfaction Index)
- Reduction in unnecessary expenses
- It can be implemented both ways- manually or automatically.
- Helps in detection and fixing of bugs

Cons-

- Stringent timelines
- Optimization of test cases is difficult
- Requirement of stable environment
- Lack of effective communication between different teams also leads to problems
- Selection of right automated tool is crucial for testing