* **8. BDD Test Cases For Login Page**
* **The Given statement** sets up the initial context for the behavior and defines the starting point of the system.
* **The When statement** describes the trigger that brings about a change or behavior in the system.
* **The Then statement** defines the expected outcome that should be observed after the event mentioned in the When statement.

Here are 15 test cases for Login page written in Gherkin format:

**Test Case 1: Successful Login**

Given a valid username and password,

When I attempt to log in,

Then I should be successfully logged into the system.  

**Test Case 2: Invalid Password**

Given an invalid password for a valid username,

When I attempt to log in,

Then I should see an error message indicating the incorrect password.  

**Test Case 3: Empty Username Field**

Given an empty username field,

When I attempt to log in,

Then I should see an error message indicating the username field is required.  

**Test Case 4: Empty Password Field**

Given an empty password field,

When I attempt to log in,

Then I should see an error message indicating the password field is required.  

**Test Case 5: Username with Special Characters**

Given a username with special characters,

When I attempt to log in,

Then I should successfully log in.  

**Test Case 6: Locked Account**

Given a locked account due to multiple failed login attempts,

When I attempt to log in,

Then I should see an error message indicating that my account is locked.  

**Test Case 7: Remember Me Option**

Given a valid username and password with "Remember Me" selected,

When I log in,

Then I should remain logged in across sessions.  

**Test Case 8: Multi-Factor Authentication (MFA)**

Given a valid username and password with multi-factor authentication (MFA) enabled,

When I log in,

Then I should be prompted to enter an authentication code.  

**Test Case 9: Password Reset Request**

Given a password reset request,

When I follow the password reset process,

Then I should be able to set a new password.  

**Test Case 10: Account Recovery Request**

Given an account recovery request,

When I follow the account recovery process,

Then I should be able to regain access to my account.

## ****1. Positive Login Page Test Cases****

**Positive test cases** are test cases that follow the “happy path” i.e. testing if the Login page functions as expected under valid inputs. These test cases explore scenarios where users do what they are supposed to do, such as:

1. Valid username and password combination successfully logs the user in.
2. Testing with the minimum allowed username and password length.
3. Testing with a username and password containing alphanumeric characters.
4. Successful login with the "Remember Me" option selected.
5. Testing login with a username that contains both uppercase and lowercase characters.
6. Successful login using a valid email address as the username.
7. Successful login using a valid phone number as the username.
8. Successful login with multi-factor authentication (MFA) enabled.
9. Testing login with a username that includes special characters (e.g., @, #, $).
10. Successful login using social media accounts (if applicable).
11. Successful login using biometric authentication (e.g., fingerprint, face recognition).
12. Testing login after a password reset to ensure the new password works.
13. Successful login after an account recovery process.
14. Successful login with localization settings (testing with different languages).
15. Testing login with different browsers (e.g., Chrome, Firefox, Edge).

**Negative Login Page Test Cases**

In contrast, **negative testing for the Login page** aims to explore scenarios that deviate from that “happy path”. Users don’t always do what we want them to do. Sometimes they do unexpected things, and a good tester understands that unpredictability to test accordingly. Some common negative test cases you should test on your Login page include:

1. ​​Entering an incorrect password for a valid username.
2. Entering an incorrect username for a valid password.
3. Entering an empty username field.
4. Entering an empty password field.
5. Entering a username that does not exist in the system.
6. Entering a password that does not meet password strength requirements.
7. Testing login with excessive length usernames and passwords.
8. Testing login with incorrect case (uppercase/lowercase) in the username.
9. Testing login with expired or deactivated user accounts.
10. Testing login with suspended user accounts.
11. Multiple consecutive failed login attempts triggering an account lockout.
12. Testing login after the session has expired due to inactivity.
13. Testing login with incorrect multi-factor authentication (MFA) codes.
14. Entering invalid characters (e.g., scripts) in the username or password fields.
15. Testing login with CAPTCHA validation failure.

With negative testing, you have to be creative. The more complex your login and authentication process is, the more test cases you will need to perform. Put yourself in the shoes of a user who has never interacted with the system before. Be a person that makes mistakes all the time and simply think of all the “bad” possibilities that can happen in your system, or specifically your Login page.

## ****Important Test Cases For Banking Applications****

### **1. Account Management Test Cases For Banking Apps**

An account in a banking application is the repository for all of the critical information about the customer's financial and investment status. This is why it is usually placed in the top priority when developing the [test plan and test strategy](https://katalon.com/resources-center/blog/test-plan).

Some important test cases for this feature are:

1. Verify the ability to create a new account with valid information.
2. Verify the inability to create an account with incomplete or invalid information.
3. Test the process of updating account information (e.g., address, phone number).
4. Verify that account balances are updated accurately after transactions.
5. Test the account closure process and ensure all associated data is removed.
6. Check for account naming conventions and validation rules.
7. Test the ability to link or unlink accounts (e.g., joint accounts).
8. Verify that account statements are generated correctly.
9. Test account recovery options in case of forgotten credentials.
10. Check for account locking and unlocking functionality after failed login attempts.
11. Verify that account data remains confidential and is not accessible to unauthorized users.
12. Test for concurrent access to the same account from multiple devices.
13. Verify that account data is synchronized across multiple platforms (web, mobile, ATM).
14. Test the process of setting up account alerts and notifications.
15. Verify the ability to export account data in various formats (e.g., CSV, PDF).
16. Test the accuracy of interest calculations for savings accounts.
17. Confirm that accounts with different currencies are handled correctly.
18. Check for the ability to set account-specific preferences (e.g., language, notifications).
19. Verify that account histories are archived and retrievable.
20. Test the process of merging or closing duplicate accoun  
     **2. Transaction Processing Test Cases For Banking Apps**
21. Test different types of transactions (e.g., deposits, withdrawals, transfers).
22. Verify that transactions are processed in real-time or within defined timeframes.
23. Test for transaction limits and restrictions on various account types.
24. Verify that transaction fees are accurately deducted.
25. Test the ability to cancel or reverse pending transactions.
26. Check for overdraft protection and notifications.
27. Verify that international transactions are handled appropriately (currency conversion, fees).
28. Test transactions during scheduled maintenance periods.
29. Verify that transaction references and receipts are generated accurately.
30. Test for duplicate transaction detection and prevention.
31. Verify the handling of transactions involving joint accounts.
32. Test the process of setting up recurring transactions (e.g., bill payments).
33. Check for the ability to designate transaction categories (e.g., groceries, entertainment).
34. Verify that transactions are reflected correctly in account statements.
35. Test transactions with offline banking options (e.g., check deposits at ATMs).
36. Verify the ability to dispute and investigate erroneous transactions.
37. Test the integration with third-party payment gateways.
38. Verify the handling of large transactions that may require additional authorization.
39. Test the processing of transactions involving foreign exchange rates.
40. Verify that transactions are time-stamped accurately for auditing purposes.

### **3. User Authentication and Security Test Cases For Banking Applications**

User authentication usually happens at the Login page and Registration page. They are more than just a “portal” to login - but rather an authentication system to prevent unauthorized access into the user’s account. Banking applications, and other applications in high-risk industries, take this process very seriously, and they usually employ several layers of authentication for added safety.

Here are some popular user authentication test cases for you to consider:

1. Test the login functionality with valid credentials.
2. Verify that incorrect login attempts result in appropriate error messages.
3. Test multi-factor authentication (MFA) methods (e.g., SMS, email, token).
4. Verify the account lockout mechanism after a specified number of failed login attempts.
5. Test password reset functionality and security questions.
6. Verify the expiration and automatic logout of idle sessions.
7. Test the ability to change account passwords.
8. Verify secure password storage and encryption.
9. Test the application's response to suspicious login activity (e.g., from different locations).
10. Verify that sensitive data (e.g., passwords, PINs) is masked or hidden.
11. Test the handling of lost or stolen devices for mobile banking.
12. Verify the security of biometric authentication (e.g., fingerprint, facial recognition).
13. Test the logout functionality and session termination.
14. Verify the effectiveness of security certificates for encrypted communication.
15. Test the behavior of the application in case of a known security vulnerability.
16. Verify the implementation of role-based access control for staff members.
17. Test for SQL injection and other common security vulnerabilities.
18. Verify that user data is not stored on the client-side.
19. Test the response to brute force attacks.
20. Verify that security patches and updates are applied promptly.

### **Functional Testing** **Checklist/Test cases**

* Check whether keeping mandatory fields empty shows error messages. For example, while transferring money, ‘Amount’ should be mandatory and cannot be kept empty.
* Check whether all the fields accept valid values and after entering an invalid value system shows error messages. For example – the ‘Account Number’ field should not accept special characters.
* Check whether all the fields have a valid character limit. For example – the ‘Account Number’ field should accept values between 9 to 18 characters.
* Verify that all the links in the application are clickable and land on the desired page.
* Check whether all the buttons are clickable and work in the desired manner.
* Check whether all the calculations are performed in the desired manner. For example – after a debit or credit transaction, the account balance should reflect the correct amount).
* Verify the scrolling functionality of the application.
* Check the application working in flight mode.
* Verify the application working during the ongoing transaction when a phone call, SMS, or any other notifications are received (for mobile applications).
* Check the application installation, uninstallation, and update processes.

Selenium Automation Challenges

**1) Cross Browser Testing**

Sometimes our web application may not response the same in different browsers, and there might be a possibility that our website works fine on Chrome but not on Firefox. Since there are plenty of browsers in the market nowadays, performing test automation on every browser might not be possible. However, we still need to check if the application under test is fully compatible with most used browsers such as Chrome, Firefox, Safari, Edge, and IE. Testing on widely used browsers is still not enough, and we may need to test on widely used versions of these browsers along with different operating systems and resolutions. This strategy sums up to [Cross Browser Testing](https://www.pixelstech.net/article/1556712242-Cross-Browser-Testing-using-Selenium-WebDriver) and makes test automation challenging for testers.

**2) Scalability**

The major challenge faced in automation is test scalability. As discussed in the previous point, running tests on different browsers, OS and resolutions is something significant to perform. Selenium WebDriver provides us a way for running tests sequentially and doesn’t offer an effective way to perform Cross Browser Testing. An application under test may grow concerning multiple features with time, which would lead to more test cases, and running multiple test cases sequentially might become a hassle.

To overcome this, Selenium has overcome with Selenium Grid, where we can test our web application on multiple combinations of browsers and OS. However, Selenium Grid would only help perform Cross Browser Testing on the physical machines or browsers that we have, hence running automation tests on the scale becomes challenging for testers.

**3) Synchronizing Events**

Sync issues are one of the primary reasons for our automation script failure. The sync issues may arise when we expect some event to occur. However, it might get delayed or might not happen due to some unknown reason, which further causes our test case to fail. For example, when we are running an automation script, just after launching the website, we expect a “accept/decline cookies” prompt to come up, but due to some reason, it got delayed, or it didn’t even occur. This type of synchronization becomes the reason for our test script failure.

To handle this, we generally use waits defined in Selenium like Implicit wait and Explicit wait.

**4) Handling Dynamic Elements**

New websites developed using high technologies might have dynamic content that changes over time. Even the sites that are accessible globally might contain content based on geo-locations.

For example, Amazon is an e-commerce application whose catalog changes overtime and reacts differently with different content in different regions. Automating such web applications becomes challenging with Selenium since the locators that we have used might fail to interact with the web elements. Web content based on AJAX sometimes takes time to load, which also becomes a possible reason for test script failure. As mentioned previously, we have waits integrated with Selenium like Implicit wait and Explicit wait, which can be helpful in such cases. We can even create dynamic or custom XPath for handling dynamic web content.

**5) False Positive and False Negative Results**

False Positive and False Negative results have always been a nightmare for automation testers. False Positive refers to the scenario where we get errors in our test cases even though the application under test works fine. Vice Versa, false-negative results refer to the situation where we get passing results for our test cases; yet the application under test has bugs. Such flakiness misleads the testing team and increases the communication gap between the QA team and the development team. Handling flaky tests is itself a challenging task for automation testers.

To overcome this flakiness, we need to make sure that the test plans, test cases, and the testing environment are managed in an appropriate and organized manner.

**ALSO READ**

* [Selenium Integration with Sauce Labs](https://www.nexsoftsys.com/articles/selenium-integration-with-sauce-labs.html)

**6) Pop up and Alert Handling**

Different types of popup and alerts occur while interacting with a web application, below are few listed:

* **Browser level notification:** Examples of such notifications can be; “Allow/Decline camera access,” “Allow/Decline microphone access,” etc. These are browser level notifications that are handled in different ways depending upon the browsers. For the Chrome browser, ChromeOptions can be used. Similarly, for the Firefox browser, FirefoxOptions/FirefoxProfile can be used.

* **Web-based alerts:** Examples of such notifications can be; “This site says….”, “Do you want to leave this site” etc. These are the alerts that can be handled using Selenium predefined class “Alerts,” which provide various methods like accept(), dismiss(), etc.

* **OS level pop-ups:** Selenium capabilities are only bounded with web interaction; hence, handling OS level interactions is something that can’t be performed using Selenium. These types of pop-ups can be categorized as flash objects which can’t be accessed using Selenium Locators. “Robot” class is commonly used for interaction with such pop-ups.

**7) Captcha or OTP Handling**

1. Why did you leave your last job?

Be very cautious in answering this query. Avoid blaming employers for their mistakes and stating that "Salary was less or poor work management." Instead of this, you can say that:

Sample Answer:

It's a professional shift. I've gained much from my last job, but now I'm seeking new challenges to expand my perspectives and gain an entirely new set of skills.

### **2. How do you handle stress, pressure, and anxiety?**

It is a common interview question to determine your performance under pressure and stress.

#### **Sample Answer:**

Under pressure and stress, I usually utilize my soft skills and handle every situation calmly. I also perform physical exercises and mind relaxing activities like meditation to deal with work stress and pressure.

Strengths:

Time management skill well

Willing to learn new technology

I pride my self

Good communication

Ability to resolve what could be difficult switcwations

Weakness:

## 27 Key One-Word Strengths List

Here are the best examples of key strengths you can say in an interview:

1. Accuracy
2. Brevity
3. Clarity
4. Creative
5. Determinate
6. Disciplined
7. Dedicated
8. Enthusiastic
9. Flexible
10. Honest
11. Hard-working
12. Negotiation Skills
13. Organization Skills
14. Patient
15. Trustworthy
16. [**Time-management**](https://leverageedu.com/blog/time-management-for-students/)
17. Team player
18. Quick learner
19. Versatile
20. Optimistic
21. Judiciousness
22. Excellent Communication Multitasking
23. Expressive
24. Respectful
25. Innovative
26. Attentive
27. Empathetic

Always remember, when you are preparing your response to the “What is your strength and weakness?” question, it’s important to ensure your strengths support the job description and set you apart from other candidates. Be specific in your responses and not overly humble.

[**Here are Expert Answers for Where You See Yourself in 5 Years!**](https://leverageedu.com/blog/where-do-you-see-yourself-in-5-years/)

## 10+ One Word Weaknesses List

When it comes to understanding your weaknesses, it is quite a sloppy path. It is at this point where you can place your [**creative thinking**](https://leverageedu.com/blog/creative-thinking/)at play and be smart about it. Your answer needs to reflect your clear analytical understanding of your own weak spots yet it needs to not absolve you of your strengths. Here are the best answer examples for What are Your Weaknesses:

1. Too critical of yourself
2. Perfectionist
3. Procrastinator
4. Introverted
5. In a rush to complete tasks before the deadlines and you stress yourself in the process.
6. It takes time for you to develop trust with your subordinates.
7. Your quest to achieve perfection might hinder you from engaging in the delegation.
8. Change is not really your strong suit. Hence, you take time to adapt to a new environment.
9. Being a shy person, it becomes difficult for me to question others at certain moments or enforce new rules and regulations.
10. Being sensitive to others’ needs, sometimes, people try to be manipulative with me.
11. Being an open communicator, my style might be a bit blunt and brazen for others.

The best formula could be to first state your weakness, followed by adding context and a specific story or example of how this trait has emerged in your[**professional**](https://leverageedu.com/blog/code-of-conduct/) life. This will give your employer an insight into your level of self-awareness and commitment to professional growth.

[**Here’s the Right Way to Introduction in an Interview!**](https://leverageedu.com/blog/introduction-in-interview/)

## Banking Domain Testing

**Banking Domain Testing** is a software testing process of a banking application for functionality, performance, and security. The main purpose of testing banking application is to ensure that all the activities and functionalities of a banking software run smoothly with no errors and it remains protected.

The BFSI (Banking, Financial services and Insurance) sector is the biggest consumer of IT services. Banking Applications directly deal with confidential financial data. It is mandatory that all the activities performed by banking software run smoothly and without any error. Banking software perform various functions like transferring and depositing fund, balance inquiry, transaction history, withdrawal and so on. Testing banking application assures that these activities are not only executed well but also remain protected from hackers.

In this tutorial, we will learn

* [What is Domain in Testing?](https://www.guru99.com/banking-application-testing.html#1)
* [Why Domain Knowledge Matters?](https://www.guru99.com/banking-application-testing.html#2)
* [Introduction to Banking Domain](https://www.guru99.com/banking-application-testing.html#3)
* [Characteristics of a banking application](https://www.guru99.com/banking-application-testing.html#4)
* [Stages of testing banking applications](https://www.guru99.com/banking-application-testing.html#5)
* [Sample Test Case for Net Banking Login Application](https://www.guru99.com/banking-application-testing.html#6)
* [Challenges in testing banking domain & their Mitigation](https://www.guru99.com/banking-application-testing.html#7)

[**Join our Live Banking Testing Project for Free**](https://www.guru99.com/live-testing-project.html)

## What is Domain in Testing?

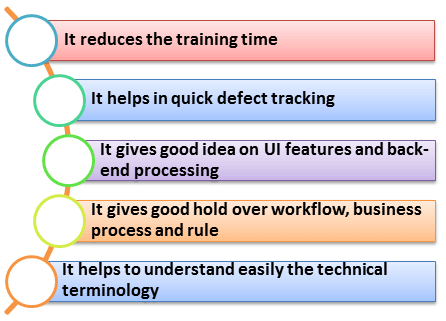
**Domain in Testing** is nothing but the industry for which the software testing project is created. When we talk about software projects or development, this term is often referred to. For example, Insurance domain, Banking domain, Retail Domain, Telecom Domain, etc.

Usually, while developing any specific domain project, domain expert help is sought out. Domain expert are master of the subject, and he may know the inside-out of the product or application.

The video player is currently playing an ad. You can skip the ad in 5 sec with a mouse or keyboardEXPLORE MORELearn Java Programming with Beginners Tutorial08:32Linux Tutorial for Beginners: Introduction to Linux Operating...01:35What is Integration Testing Software Testing Tutorial03:04What is JVM (Java Virtual Machine) with Architecture JAVA...02:24How to write a TEST CASE Software Testing Tutorial01:08Seven Testing Principles Software Testing05:01Linux File Permissions Commands with Examples13:29How to use Text tool in Photoshop CC Tutorial08:32What is NoSQL Database Tutorial02:00Important Linux Commands for Beginners Linux Tutorial15:03

## Why Domain Knowledge Matters?

Domain knowledge is quiet essential for testing any software product, and it has its own benefits like



## Banking Domain Knowledge – Introduction

Banking domain concepts are huge, and basically it is sub-characterized into two sectors

1. **Traditional banking sector**
2. **Service based banking sector**

Below is the table of the services these two sub-sectors of banking encompass

|  |  |
| --- | --- |
| **Traditional banking sector** | * Core banking * Corporate banking * Retail banking |
| **Service based banking sector** | * Core * Corporate * Retail * Loan * Trade finance * Private banking * Consumer finance * Islamic banking * Customer delivery channels/Front end delivery |

Based on the scope of your project you may need to test one or all of the above service offerings. Before you begin testing, ensure you have enough background on the service being tested.

## Characteristics of a Banking Application

Before you begin testing, it’s important to note the standard features expected of any banking application. So that, you can gear your test efforts to achieve these characteristics.

A standard banking application should meet all these characteristics as mentioned below.

* It should support thousands of concurrent user sessions
* A banking application should integrate with other numerous applications like trading accounts, Bill pay utility, credit cards, etc.
* It should process fast and secure transactions
* It should include massive storage system.
* To troubleshoot customer issues, it should have high auditing capability
* It should handle complex business workflows
* Need to support users on multiple platforms (Mac, Linux, Unix, Windows)
* It should support users from multiple locations
* It should support multi-lingual users
* It should support users on various payment systems (VISA, AMEX, MasterCard)
* It should support multiple service sectors (Loans, Retail banking etc.)
* Foolproof disaster management mechanism

## Test Phases in Testing Banking Applications

For testing banking applications, different stages of testing include

* **Requirement Analysis:** It is done by business analyst; requirements for a particular banking application are gathered and documented
* **Requirement Review:**Quality analysts, business analysts, and development leads are involved in this task. The requirement gathering document is reviewed at this stage, and cross-checked to ensure that it does not affect the workflow
* **Business Requirements Documentation:**Business requirements documents are prepared by quality analysts in which all reviewed business requirements are covered
* **Database Testing:**It is the most important part of bank application testing. This testing is done to ensure data integrity, data loading, data migration, stored procedures, and functions validation, rules testing, etc.
* **Integration Testing:**Under [Integration Testing](https://www.guru99.com/integration-testing.html) all components that are developed are integrated and validated
* **Functional Testing:**The usual software testing activities like[Test Case](https://www.guru99.com/test-case.html)preparation, test case review and test case execution is done during this phase
* **Security Testing:**It ensures that the software does not have any security flaws. During test preparation, QA team needs to include both negative as well as positive test scenarios so as to break into the system and report it before any unauthorized individual access it. While to prevent from hacking, the bank should also implement a multi-layer of access validation like a one-time password. For [Security Testing](https://www.guru99.com/what-is-security-testing.html), automation tools like IBM AppScan and HPWebInspect are used while for [Manual Testing](https://www.guru99.com/manual-testing.html) tools like Proxy Sniffer, Paros proxy, HTTP watch, etc. are used
* [**Usability Testing**](https://www.guru99.com/usability-testing-tutorial.html)**:**It ensures that differently able people should be able to use the system as normal user. For example, ATM with hearing and Braille facility for disabled
* **User Acceptance Testing:**It is the final stage of testing done by the end users to ensure the compliance of the application with the real world scenario.

## Sample Test Case for Net Banking Login Application

Security is prime for any banking application. Therefore, during test preparation, QA team should include both negative and positive test scenarios in order to sneak into the system and report for any vulnerabilities before any unauthorized individual get access to it. It not only involves writing negative test cases but may also include destructive testing.

Following are generic test cases to check any banking application

|  |  |
| --- | --- |
| **Sample test cases** |  |
| For Admin | * Verify Admin login with valid and Invalid data * Verify admin login without data * Verify all admin home links * Verify admin change password with valid and invalid data * Verify admin change password without data * Verify admin change password with existing data * Verify admin logout |
| For new Branch | * Create a new branch with valid and invalid data * Create a new branch without data * Create a new branch with existing branch data * Verify reset and cancel option * Update branch with valid and invalid data * Update branch without data * Update branch with existing branch data * Verify cancel option * Verify branch deletion with and without dependencies * Verify branch search option |
| For New Role | * Create a new role with valid and invalid data * Create a new role without data * Verify new role with existing data * verify role description and role types * Verify cancel and reset option * Verify role deletion with and without dependency * verify links in role details page |
| For customer & Visitors | * Verify all visitor or customer links * Verify customers login with valid and invalid data * Verify customers login without data * Verify banker’s login without data * Verify banker’s login with valid or invalid data |
| For New users | * Create a new user with valid and invalid data * Create a new user without data * Create a new user with existing branch data * Verify cancel and reset option * Update user with valid and invalid data * Update user with existing data * Verify cancel option * Verify deletion of the user |

## Challenges in testing Banking domain & their Mitigation

Challenges tester might face during testing banking domain are

|  |  |
| --- | --- |
| **Challenge** | **Mitigation** |
| * Getting access to production data and replicating it as test data, for testing is challenging | * Ensure that test data meets regulatory compliances requirements and guidelines * Maintain the data confidentiality by following techniques like data masking, synthetic test data, testing system integration, etc. |
| * The biggest challenge in testing banking system is during the migration of the system from the old system to the new system like testing of all the routines, procedures and plans. Also how the data will be fetched, uploaded and transferred to the new system after migration | * Ensure Data Migration Testing is complete * Ensure Regression Test cases are executed on old and new systems, and the results match. |
| * There may be the cases where requirements are not documented well and may lead to functional gaps in test plan * Many non-functional requirements are not fully documented, and testers do not know whether to test it or not | * The test should participate in the project right from Requirement Analysis phases and should actively review the Business Requirements |
| * The most important point is to check whether the said system follows the desired policies and procedures | * Compliance or Regulatory Policies testing must be done |
| * The scope and the timelines increases as banking application are integrated with other application like internet or[Mobile](https://www.guru99.com/mobile-testing.html)banking | * Ensure Time budget for Integration Testing is accounted if your banking application has many external interfaces |

Banking domain is the most vulnerable area for cyber-theft, and safeguarding the software requires precise testing. This tutorial gives a clear idea of what it takes for banking domain testing and how important it is. One must understand that –

* Majority of banking software are developed on [**Mainframe**](https://www.guru99.com/mainframe-testing.html) and**Unix**
* Testing helps to lessen possible glitches encounter during software development
* Proper testing and compliance to industry standards, save companies from penalties
* Good practices help develop good results, reputation and more business for companies
* Both manual and automated testing have respective merits and usability

**Join our** [Live Banking Domain Testing Project](https://www.guru99.com/live-testing-project.html)

**You Might Like:**

* [**What is Software Testing?**](https://www.guru99.com/software-testing-introduction-importance.html)
* [**7 Principles of Software Testing with Examples**](https://www.guru99.com/software-testing-seven-principles.html)
* [**V-Model in Software Testing**](https://www.guru99.com/v-model-software-testing.html)
* [**STLC (Software Testing Life Cycle)**](https://www.guru99.com/software-testing-life-cycle.html)
* [**Manual Testing Tutorial: What is, Types, Concepts**](https://www.guru99.com/manual-testing.html)
* [**Automation Testing**](https://www.guru99.com/automation-testing.html)
* [**What is Unit Testing?**](https://www.guru99.com/unit-testing-guide.html)
* [**Integration Testing: What is, Types with Example**](https://www.guru99.com/integration-testing.html)

Post navigation

[Report a Bug](https://form.jotform.me/72391811797466)

[Prev](https://www.guru99.com/complete-web-application-testing-checklist.html)**[Rajeshwari](https://www.guru99.com/complete-web-application-testing-checklist.html)**[said: 5 days ago](https://www.guru99.com/complete-web-application-testing-checklist.html)

[Firstly, I am telling you about my strength.  
  
My strengths are I am a self-motivated person. I like to gain new skills. With that, I am a quick learner.  
  
And bout my weakness: I am a very emotional person. I trust people very easily and I can't say no if anyone asking me for help.](https://www.guru99.com/complete-web-application-testing-checklist.html)

[(16)](https://www.guru99.com/complete-web-application-testing-checklist.html)

**[Saurav kumar meena](https://www.guru99.com/complete-web-application-testing-checklist.html)**[said: 5 days ago](https://www.guru99.com/complete-web-application-testing-checklist.html)

[Strength:  
My strengths are, I have a positive attitude and easily adapt to new environments. I always looking forward to learning new things.  
  
Weakness:  
I am a very emotional person, I can trust people very easily, cannot say no to anyone asking for help, being too sensitive, and overthinking.](https://www.guru99.com/complete-web-application-testing-checklist.html)

[(11)](https://www.guru99.com/complete-web-application-testing-checklist.html)

**[Rj](https://www.guru99.com/complete-web-application-testing-checklist.html)**[said: 5 days ago](https://www.guru99.com/complete-web-application-testing-checklist.html)

[My strengths are my attitude. I am a self Motivated person and very calm in any situation so i can handle stress. taking challenges.  
  
My weakness is every time I'm finding shortcut to solve any problem. my point of view shortcut is not good way to handle anything.](https://www.guru99.com/complete-web-application-testing-checklist.html)

[(4)](https://www.guru99.com/complete-web-application-testing-checklist.html)

**[Vrushank](https://www.guru99.com/complete-web-application-testing-checklist.html)**[said: 1 week ago](https://www.guru99.com/complete-web-application-testing-checklist.html)

[My Strength Is Quick Learning And Better Understanding.  
  
My weakness is I am an Innocent Person and I trust Everyone.](https://www.guru99.com/complete-web-application-testing-checklist.html)

[(3)](https://www.guru99.com/complete-web-application-testing-checklist.html)

**[Sanket Wagh](https://www.guru99.com/complete-web-application-testing-checklist.html)**[said: 2 weeks ago](https://www.guru99.com/complete-web-application-testing-checklist.html)

[My strengths are I am a Quick Learner and a motivated Person.  
  
My weakness is I am a person who overthinks before doing any task because I want to do everything perfectly.](https://www.guru99.com/complete-web-application-testing-checklist.html)

[(116)](https://www.guru99.com/complete-web-application-testing-checklist.html)

**[Jenny](https://www.guru99.com/complete-web-application-testing-checklist.html)**[said: 2 weeks ago](https://www.guru99.com/complete-web-application-testing-checklist.html)

[My strengths are, firstly I'm an extremely self-motivated person. I'm always trying to find ways to do my work better and also I have a positive attitude.  
  
My weakness is overthinking but now I try to overcome this weakness.](https://www.guru99.com/complete-web-application-testing-checklist.html)

[(9)](https://www.guru99.com/complete-web-application-testing-checklist.html)

**[Miriyala srikanth](https://www.guru99.com/complete-web-application-testing-checklist.html)**[said: 2 weeks ago](https://www.guru99.com/complete-web-application-testing-checklist.html)

[One of my greatest strengths is my dedication to continuous learning and improvement. I actively seek opportunities to expand my knowledge and enhance my skills in the IT field.  
  
As for my weaknesses, I recognize the need to improve my presentation and communication skills, and I am committed to working on them to become a more effective communicator.](https://www.guru99.com/complete-web-application-testing-checklist.html)

[(14)](https://www.guru99.com/complete-web-application-testing-checklist.html)

**[Aishvary](https://www.guru99.com/complete-web-application-testing-checklist.html)**[said: 3 weeks ago](https://www.guru99.com/complete-web-application-testing-checklist.html)

[I have quick grasping power so I can learn anything easily and smartly and also a self-motivated person who tries to improve more and more personally and professionally.  
  
My weakness is Overthinking and I try to overcome it by doing meditation and being mindful.](https://www.guru99.com/complete-web-application-testing-checklist.html)

[(9)](https://www.guru99.com/complete-web-application-testing-checklist.html)

**[Momo](https://www.guru99.com/complete-web-application-testing-checklist.html)**[said: 3 weeks ago](https://www.guru99.com/complete-web-application-testing-checklist.html)

[My strength is I always try to do my best in everything in my life because I don't like anyone to say that I didn't do my work well.  
  
My weakness is overthinking but now I try to overcome this weakness.](https://www.guru99.com/complete-web-application-testing-checklist.html)

[(21)](https://www.guru99.com/complete-web-application-testing-checklist.html)

**[Kandarapu praveen](https://www.guru99.com/complete-web-application-testing-checklist.html)**[said: 4 weeks ago](https://www.guru99.com/complete-web-application-testing-checklist.html)

[My strength is that always try to update myself with new skills whenever I get the chance.  
  
My weakness is whenever I started any I could not leave that work until I completed that work.](https://www.guru99.com/complete-web-application-testing-checklist.html)

[Next](https://www.guru99.com/testing-e-commerce-applications.html)