kasperanalytics.com

+918130877931

Banking Application Testing Process

hr@kasperanalytics.com

kasper-analytics

A typical process of testing a banking application can have the following steps-

Requirements Gathering

The testing process starts from the first phase of SDLC i.e. requirements gathering. Usually, in agile projects, along with business analysts, the testing team is also involved in the requirements gathering process. In traditional models like a waterfall, the testing team comes into the picture later once the requirements have been collected.

For finance-related, banking projects, the testers should have adequate domain knowledge. Testers should be able to think from both the stakeholder and end-user's points of view. In this phase, analysts gather requirements, understand and review the requirements. In addition, they gain as much domain knowledge as possible by the research, and/or taking help from the SMEs.

Test Planning

In this next phase, the testing team proceeds with the detailed planning of the testing process. Here, a test plan is created. This includes the scope of the testing, roles, and responsibilities, test deliverables, testing levels, testing tools, test environments, etc.

Test Scenario and Test Case Creation

Here, based on the requirements document, test scenarios, and test cases are created. The testing team should be careful in covering each scenario. Not only functional but also security, performance, and other nonfunctional characteristics should be covered.

Functional Testing

After test case creation, the testing team conducts functional testing to make sure the features are implemented as per the requirements.

Functional Testing Checklist

- Check whether keeping mandatory fields empty shows error messages. For example, while transferring the 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.

Database Testing

Database testing includes testing the front-end layer, business logic layer, and database. In this type of testing, queries are executed and data flow in different tables is observed. The testing team responsible for the database testing should have in-depth knowledge of the database.

Database Testing Checklist

- Check whether the data is logically organized in the database.
- Check each field's data type and see whether it is as per the desired specification.
- Verify the field length on the front-end side and in the database is the same.
- Check the values of the computed field. (e.g. DoB and Age)
- Check whether the naming conventions used in the database are uniform. (e.g. Names of tables, columns, stored procedures, indexes)
- Verify that the data values are stored in the correct columns and tables.
- Check whether each table has a primary key and foreign key constraints.
- Check for any redundant and duplicate data.
- Insert null values in the columns that should not accept null values.
- Check if the data is correctly stored in the database when fetched from the front end.
- Verify that the database is access is restricted to authorized users only.
- Check whether in the event of failure data is rolled back or not.
- Check application working when the database server is down.
- Verify that the database backup is taken regularly.
- Check database performance when multiple users use the application at the same time (e.g. multiple users try to transfer money simultaneously).

Security Testing

For a banking application, this is one of the most important steps. In

security testing, the goal is to check the system's ability to fight against the attacks.

Also, it is checked that only authorized and authenticated users are allowed to access the application and their data is secured.

Security Testing Checklist

- Check whether the application has a proper process of authenticating the users on the platform. For example – apart from User Id and Password, users can be asked to enter the CAPTCHA.
- Check whether multiple invalid logins are allowed.
- Verify that the 'Forgot Password' and 'Forgot User ID' features have proper mechanisms to recover the password and User ID.
- Check whether the password policies are strong.
- Check whether user IDs and passwords are encrypted.
- Verify that the application has a secure protocol e.g. HTTPS.
- Check whether sensitive information is displayed encrypted at the client-side.
- Check whether input validations are performed at the server-side.
- Verify if the passwords are stored in cookies or not.
- Check whether information stored in cookies is in encrypted form.
- Check whether proper session timeout mechanisms are placed. (e.g. after a few minutes of inactivity, the user should be logged out automatically).
- Check application working when the user clears the cache. (e.g. user should be logged out).
- Check whether the URLs have sensitive information.
- Verify the application for SQL, XML, HTTP header, HTTP parameter, XPath, regular expression injections.
- Check whether the 'Back' icon of the browser is disabled for certain pages. Usually, in banking applications, the 'Back' icon click is not allowed.

Usability testing

Any banking application will have a broad spectrum of users, from tech- savvy people to non-tech friendly people. The goal of usability testing is to check the application's working from all the users' perspectives and see if the application is user-friendly for all types of users.

Usability testing Checklist

- Check whether the user is able to navigate through the application smoothly.
- Check whether buttons and links have uniform fonts, color, and size.
- Verify that the whole application has uniform fonts, text size, and color combination.
- Check whether important and critical, messages and buttons are visible clearly on the page. For example – in the transaction process, any important message should be displayed clearly with the bigger fonts and dark colors, if possible.
- Verify that critical information is displayed without any grammatical errors.
- Check whether critical processes of the application have smooth and easily understandable steps. For example – for deposit accounts, there can be a separate menu item where users can manage all the deposit accounts.
- Check whether each page has a title.
- Verify that all the links and buttons have text that is easy to understand.
- Check whether the error messages of the fields are easy to understand.
- Check whether all the important fields have placeholders and tooltips.

Performance Testing

The goal of performance testing is to check application performance under severe load and stress.

Performance Testing Checklist

- Check application performance when multiple users login at the same time and use different functionalities.
- Check application performance when multiple users login at the same time and use the same functionality. For example – all the users try to transfer the money.
- Verify the application performance when the device battery is low.
- Check application performance when the device battery is high.
- Check application performance when the device is charging.
- Verify the application performance when a user tries to use the same function multiple times in a defined period. For example – the user tries to transfer money from the same account to different accounts within an hour.
- Check application performance when network connectivity is low. For example – during an ongoing transaction if the network has a sudden fluctuation.
- Check application performance when network connectivity is zero. For example – during an ongoing transaction if there is no network.
- Check application performance when the network keeps fluctuating between low and high speed.
- Check application performance during a sudden crash.

User Acceptance Testing

In this step, the client or end-users of the system (usually, representatives of the client) test the system to get an assurance that the application will run smoothly as per desired behavior in real-world scenarios.

Apart from the above-mentioned phases, there are many other important phases. We include these in the testing process of a bank application such as regression testing, compatibility testing, etc.

User Acceptance Testing Challenges

- Bank applications have multiple users in the range of millions. Simulating such a high number of users may create a challenge for the testing team. Testing teams can go for automation tools used for performance testing.
- As there are multiple users, all the users will be using different devices, connections, and operating systems. Testing on each combination of devices, OS, and connections is a complex, time-consuming process.
- Testing bank applications need extra care and time as it deals with money and sensitive information.
- Checking from all the security perspectives and covering all the security testing scenarios is very critical.
- Another important challenge is to test the application for rollback mechanisms if there is a failure in the system.
- Bank applications allow utility payments and many other third-party integrations. Testing them will be a challenge for the testing team as each integration is different.

Conclusion

Banking applications are in demand even in developing countries as nowadays people prefer online services and avoid physically going to the branch.

Moreover, in this fast-paced life, people will always choose fast online transactions for any service or product they are purchasing. There are many other reasons why banking applications are gaining popularity.

Given the sensitive and complex nature of these applications, it is critical to test them thoroughly. As mentioned in the above sections, there are various phases in banking application testing. Of course, many challenges are faced during the testing process. But it becomes the responsibility of the testing team to find as many bugs as possible to achieve the highest quality.