iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 11 **Issue:** VII **Month of publication:** July 2023

DOI: https://doi.org/10.22214/ijraset.2023.55066

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue VII Jul 2023- Available at www.ijraset.com

Various Software Testing Techniques, their Implementation and Significance on Banking Applications: In Context of Bangladesh

Sarah Nuzhat Khan[1], Sanjida Akter[2], Shihab Z Hasan[3], Dilruba Sharmin[4]
Banking Solution Department, Leads Corporation Limited, `Dhaka, Bangladesh

Abstract: ***Software testing is the process of identifying bugs or errors in developed software. Consequently, this process offers interested individuals with precise details on the product's quality. Software testing is done not only to ensure that the program behaves correctly or not, but to check whether it satisfies the user's requirements and expectations as well. The testing of banking software is very important since they handle millions of transactions and are related to customers directly. Proper testing of the banking applications ensures that all operations are carried out accurately and remain safe and secure. The main objective of this paper is to provide an overview of various software testing techniques, their implementation and significance on the banking applications of Bangladesh.***

Keywords: Banking Applications, Testing, Banks, Bug, Life Cycle, Quality Assurance, Developers, Cost, Crucial

I. INTRODUCTION

The most important stage of the software development life cycle is software testing or software quality assurance. Quality assurance of developed software has reached new heights because of the complexity of today's software applications rising together with the stress resulting from an expanding competitive environment. One of the most important industries in the world is the banking sector, and the more people talk about it, the more grounds there are for concern. Numerous sensitive pieces of information, including financial, personal, and other crucial details, are stored in banking applications. These applications deal with a lot of money, which is one of the reasons they are so sensitive. Banks deal with enormous amounts of money. It follows that repeated testing is necessary to make sure that these applications are secured enough to be used. Banks have access to a vast amount of information, including consumer personal information, commercial activities, and many other things. The fact that all this data is preserved on a single system makes it simpler for hackers to access it. Banks must take extra security precautions to prevent this. Finding safety issues in banking-related apps is the practice of banking application testing. Testing can be carried out either manually or with the aid of automated tools. Testing of banking applications is also done to make sure that they are accessible to users and provide a secure environment. The testing team must be familiar with banking standards and regulations to evaluate banking applications. Additionally, they must test the application in accordance with the regulations of the financial industry. Financial companies may reject the testing findings if it is not done in accordance with the rules. Since banking apps are extremely sensitive, it is crucial to test them thoroughly by using the appropriate software testing methodologies.

II. IMPORTANCE OF SOFTWARE TESTING IN BANKING APPLICATIONS

Banking applications are essential since they handle millions of transactions and are related to customers directly. Therefore, it is imperative to test these applications. Every update, deployment, or modified project should include testing since thorough testing can reduce risks such as expensive financial breakdowns and time-consuming bug fixes. A project's success and ability to stay within budget are directly impacted by this risk management strategy. Since the banking industry is being closely scrutinized in this area, testing can help banks reduce operational risks inside the system that could have a negative impact on consumer perception or even institutional sanctions. The handling of confidential financial information and transactions by banking apps makes it essential to find and address any possible risks or flaws as soon as possible. Proper testing assists in identifying safety concerns, technical flaws, or performance issues that could put the system and its users in danger. By establishing priority of testing processes banks can uphold their credibility, ensure the security of their clients, and contribute to a secure and reliable financial environment. Banking applications handle confidential client information and financial transactions. The security of these systems must be enforced in order to safeguard client information and stop unauthorized access.

The integrity of the banking software is ensured by rigorous testing, which helps in the detection of threats, defects, and possible security breaches. Testing banking software ensures that it adheres to regulations such data privacy laws, anti-money laundering (AML) laws, and know your customer (KYC) standards. By identifying and fixing bugs before the software is made public, banks can increase the effectiveness and dependability of their operations. By conducting comprehensive testing, banks can lower the likelihood of inaccuracies system failures, and fraudulent activities. Strong, reliable, and secure software solutions are essential in the fiercely competitive banking sector. The organization and its clients could both suffer financial losses as a result of banking software glitches. Inaccurate calculations, transaction problems, system hacking and system breakdowns are just a few examples of the kind of flaws that testing helps to find and fix. By preventing such losses, banks can protect their financial interests and guarantee customer satisfaction. Therefore, bank's revolutionary endeavours must place a strong emphasis on testing their software.

III. IMPORTANCE OF SOFTWARE TESTING IN BANKING SECTORS OF BANGLADESH

One of the most challenging applications in today's software testing and development market is banking software. The banking industry is in jeopardy to several threats, including cyberattacks, data breaches, and internet crimes. To reduce vulnerability software testing and quality assurance are essential. Bangladesh's banking sector is larger in comparison to many adjacent nations with comparable rates of growth and per capita income. Bangladesh's banking industry, which accounts for 55.5% of GDP, is divided into four scheduled bank groups: PCBs, FCBs, NBFIs, and SOBs. The Bangladesh Bank Order, 1972, and the Bank Company Act, 1991 regulate and oversee the 60 scheduled banks in the country. The government owns 43 private commercial banks, with 8 Islamic Shariah-based PCBs and 33 traditional PCBs. The banking industry relies on IT companies to reduce operating costs and offer competitively priced goods and services. Core Banking Solutions (CBS) enables users to access banking services from anywhere, eliminating the need for physical branches.

The banking sector of Bangladesh is making a lot of efforts to support the government in achieving the three main Sustainable Development Goals (SDGs). Due to the active involvement of banks, the RMG industry in the nation has been expanding significantly. Large-scale facilities are being provided by banks for the export-driven garment industry. Following the regulations and guidelines established by the central bank (Bangladesh Bank) for the agricultural business, the banking sector has played a crucial role in agriculture since independence, which in turn contributes to higher farm production. The Bangladesh Bank sets loan disbursement goals for the agriculture sector each year, but most banks exceed the goal since they are eager to lend to the sector. The recent growth of the agricultural sector has raised expectations for export earnings, which is being facilitated by central bank policy assistance for bank lending.

Once again, the Covid-19 outbreak showed that the banking sector was the first to act in the event of a calamity. Banks adopted a number of stimulus initiatives for the agriculture sector that the central bank had declared during the pandemic of the previous two years. Banks are crucial in helping migrant workers send money home by simplifying remittance process. According to a recent World Bank report, Bangladesh maintained its ranking as the country that received the seventh-highest amount of money sent by migrant workers in 2021. The proactive role taken by banks has made this possible.

Nevertheless, banks continue to play a significant part in the expansion and development of the national economy and put in a lot of effort to assist the government in achieving its objective of making the nation a developed one by 2041.

The whole financial system depends on the banking sector. By offering infrastructure, financing, and investment, it has a great impact on the national economy of Bangladesh.

To run banking activities properly, banking apps are essential. In order to provide financial security, legal compliance, operational effectiveness, client confidence, a competitive advantage, and reducing the risk of financial loss, it is crucial to accurately test banking software and ensure that it operates smoothly and error-free.

IV. TOP ASPECTS OF TESTING BANKING APPLICATIONS IN TODAY'S DIGITAL WORLD

A. Business Leaders' Involvement at an Initial Phase

It is crucial for the testing team to work closely with the businesspeople and the business analysts from the start of project's life in order to thoroughly comprehend the business requirements of the application. To ensure that new or existing business procedures are not violated, the requirements are drafted by financial experts or business analysts with their area of expertise. All stakeholders then examine the documentation. Additionally, the testing team must have an adequate understanding of the relevant subject.

B. Impact Evaluation

Analysing the impact of changes to the application or product that has been delivered is what this entails. It pinpoints the system components that require rigorous regression testing since they were unintentionally impacted by the application modification. In general, decisions are made as a team after the analysis is completed. The impacted regions must be located and corrected by the quality assurance team. Due to the fact that it reuses previously run test cases, this technique is known as selective re-testing.

C. Verification

Verification is the process of assessing software at different stages to make sure it complies with established standards and meets the requirements that have been set forth. It concentrates on determining whether the software is being developed rightly. Verification operations in the context of banking software testing may involve checking to see if the software requirements are complete, checking the accuracy, completeness, and consistency of documentation such as user manuals, system specifications, and technical guides. It confirms that the software design adheres to the stated specifications and structural principles. It also ensures that the banking software is installed and configured correctly, including the integration of third-party systems, privacy settings, and database connections.

D. Validation

The purpose of validation operations is to ensure that the banking software runs dependably in the intended production environment and satisfies the end-users' expectations and needs. This emphasizes determining whether the appropriate software is being developed. When testing banking software, validation activities may include checking if the software's functions properly compared to the defined requirements of the intended banking functions. To make sure that the program is understandable and user-friendly, the user interface, user experience, and simplicity of use are evaluated. It confirms that the banking application complies with legal mandates such as Know Your Customer (KYC), Anti-Money Laundering (AML) and data protection laws as well.

V. A POPULAR BANGLADESHI BANKING APP "BANKULTIMUS"

The "BankUltimus" brand of Core Banking Solution (CBS) is a software of Leads Corporation Limited, a leading IT industry of Bangladesh. This application offers integrated services for retail, SME, and corporate banking as well as features for trade finance, loan management systems, and many other things. It is an enterprise banking solution that enables banks to offer the full spectrum of banking services and content to customer information files (CIF), know your customer databases (KYC), and transaction profiles (TP) in a reliable and efficient manner in accordance with Central Bank requirements.

"BankUltimus" satisfies both the current and future needs of the market. The transition to Core Banking is now more streamlined than ever with the aid of “BankUltimus”. Blockchain, Artificial Intelligence (AI), mobility (mobile apps), and data analytics are four examples of quickly developing technologies that are undermining banks' traditional business structures. In BankUltimus all areas are provided with service commencing from teller operations and ending with the Treasury. Built on innovative web technologies, “BankUltimus” is a comprehensive Core Banking Solution for Banks.

VI. REAL TIME TESTING CYCLE OF A BANKING APPLICATION (BANKULTIMUS)

In this paper the real time testing steps of a banking software will be discussed. Like many other software, banking applications also follow a software testing life cycle (STLC) to test the product and guarantee that quality standards are maintained. The topic of discussion will be software testing life cycle of ‘BankUltimus’, a renowned banking application of Bangladesh. For ‘BankUltimus’, testing is done manually and automatically, starting from unit testing to system testing and performing maintenance testing for each iteration. As “BankUltimus” is a complex application and handles the Core Banking of many banks and NBFIs, it’s very important to test this accurately. For this following all the steps is very essential for complete testing and delivering a quality product. The steps are as following:

A. Requirement Documentation

This step involves setting requirements down as functional specifications or use cases. Based on customer’s expectations, banking professionals or business analysts collect and document requirements for various banking modules. This phase is very crucial since the accuracy of this step will determine the client’s satisfaction.

B. Requirement Analysis

All parties involved, including development leads, business analysts, and testers, analysis the requirement documentation output. Both the new and the current business workflows are double-checked to ensure no violations occur. Moreover, the central bank (Bangladesh Bank) has a banking compliance that describe a set of guidelines, regulations, and laws that banks must adhere to conduct secure, moral, and lawful business. It is very important to ensure that the documented requirements should not go against this banking compliance set by the central bank. The prerequisites should be validated and verified precisely. Based on this, further steps are taken, and requirement documents are revised.

C. Planning Business Scenario

The Business Scenarios are prepared in this step by Test Engineers from the requirement specifications, making sure that all Business Requirements are covered, and the banking compliance has been followed. Business scenarios are generalized events without specific instructions. Additionally, business analysts analyse these business scenarios to make sure that all business requirements are satisfied. Analysing business scenarios is more accessible than detailed test cases for business analysts.

D. Test planning

Test engineers may start planning tests after completing the business scenario planning. The more requirement analyses and business planning that are mirrored in the related test plan, the better the result will be. The Test analyst or Test team lead creates a testing plan during this phase that aligns with the project goals and works with the technological infrastructure. The testing team and product owner should agree on the testing approach before it is formally established in the form of checklists and feasibility reports.

E. Test Case Development

The real-time test cases need to be prepared as soon as the planning process is finished. The number and nature of test cases might vary significantly depending on the testing types that will be employed on the project and the proportion of manual to automated testing. The entire life cycle of software testing will be determined by how a tester handles the test case design. A feature is likely to contain bugs if it is not thoroughly examined from all sides. The three fundamental methods for creating test cases are experience-based, structure-based, and specification-based. It is advised to concentrate on specification-based testing for banking software specifically, implementing the other two whenever necessary.

F. Setting up Test Environment

To run developed test cases, a test environment setup is required. A test environment is a server that enables the execution of test cases in accordance with the specifications of the software or users being tested. The host is configured, and hardware (RAM, CPU speed), software, and network configurations are all integrated. For the application to perform properly, test environments are used to test and find bugs in the program. For a web-based banking application like ‘BankUltimus’, an ideal test environment for testing is a cross-browser test environment which is testing the web apps on several browsers and browser versions.

G. Test Execution

Executing test cases is the stage of software testing where all of the previously created plans and scenarios are actually put into practice. At that point, the development team receives the results for improvement once the testing team gets chance to run the test cases in the predetermined environment. The bug finding and fixing processes shouldn't be separated, which emphasizes the importance of maintaining outstanding interaction across all the modules working on the software. The quality of the final product and the efficiency of the programming process are substantially better when developers consult their testing team.

H. Test Closure and Reporting

The reporting phase of the software testing life cycle is crucial. A formal document known as a test closure report gives a thorough review of the testing scope, goals metrics, techniques, tools, resources, flaws, challenges, risks, and recommendations. Additionally, an appraisal of the test's effectiveness and efficiency is included, along with a comparison of the planned and actual test schedule, cost, and standards. The testing team generates this before the testing procedure is formally concluded. The precise documentation produced by software testing for banking applications makes the time and effort expended on it worthwhile.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue VII Jul 2023- Available at www.ijraset.com

Testing records offer reliable information on all the product has through, including test plan, test cases, test logs, bug reports, test metrics, upgrades, consumer feedback, failures, and regressions, for such complicated software solutions as banking apps. The users can gain significant idea about the product infrastructure and required upgrades simply by looking at the precise QA documentation for a banking app. For a perfect maintenance and test closure process of a banking software, reporting after every sprint is preferred. Fig. 1 shows software testing life cycle (STLC) of a banking software.

Fig. 2 Software Testing Lifecycle of a Banking Application

VII. MAINTAINING BUG CYCLE

A bug is any flaw in the design, requirements, specifications, or coding that leads to unintended outcomes. Tracking bugs and making them resolved is one of the most crucial parts of a banking application. Every product needs bug tracking to preserve quality, minimize time, and reduce expenses. Making the correct tool selection for bug tracking increases software quality assurance. Azure DevOps tools can be a prominent option to track bugs. The Testing team can identify, record, prioritize and manage software bugs in Azure Boards. A bug goes through a bug life cycle, from which it starts and traverses all of its phases. This begins the moment a tester discovers a new bug and ends when the tester closes it, guaranteeing that it won't be repeated. From project to project, a bug may pass through a different number of states. For a banking application Fig. 2 and the model below shows all potential states of a bug's life.

A. New

A defect is assigned the status of NEW bug when it is initially logged by the testers and accepted by the development team.

B. Assigned

The bug gets assigned to the developers after a tester posts it after getting the test team lead's authorization. While assigning the bugs the QA needs to set the level of bug priority and severity.

1) *Priority:* The urgency with which a bug needs to be corrected and removed from a website or app is referred to as its priority. It gauges how high on the debugging hierarchy the bug should be. A successful software release depends on properly allocating bug priorities and establishing a bug handling procedure.

Bug priority levels are as follows:

a) Low: The bug can be solved later. Priority is given to other, worse issues.
b) Medium: The bug can be resolved during routine testing and development.
c) High: The bug must be fixed immediately as it negatively impacts the system and makes it inoperable until fixed.

2) *Severity:* The impact of a bug on the functioning of an application feature when it is used is measured by the severity. It relies on how the bug affects the entire system. There are various degrees of bug severity that correspond to how much of a risk the bug can be to the software:

a) Low: Bug won't cause any obvious system failure
b) Minor: Causes some unusual or inappropriate activity, but not sufficient to interfere with system operation
c) Major: The system could collapse due to a serious bug.
d) Critical: A bug that can cause an entire system breakdown

C. Active

In this state the developer continues to work on investigating and fixing the bug.

D. Resolved

A developer can mark the bug status as "Resolved" when the required code modification is successfully made and verified as well.

E. Retest

At this point, the tester retests the program to see if the bug has been solved by the developer, and then updates the status to "Closed" or "Active" as necessary.

F. Verified

After the bug has been fixed by the developer, the tester re-tests it. If no software bug is found, the problem has been resolved, and the status is now set to "verified."

G. Reactivate

The tester updates the status to "Reactivate" if the bug still persists after the developer has fixed it. The bug goes then puts on its life cycle once more.

H. Closed

If the bug has been fixed, the tester labels it as "Closed." If it’s not, then they make the bug status “Active” again. The bug needs to be “Closed” in case of being “Rejected” for reasons like- Duplicate bug, NOT a Bug, Non-Reproducible, Obsolete and Removed from the backlog.

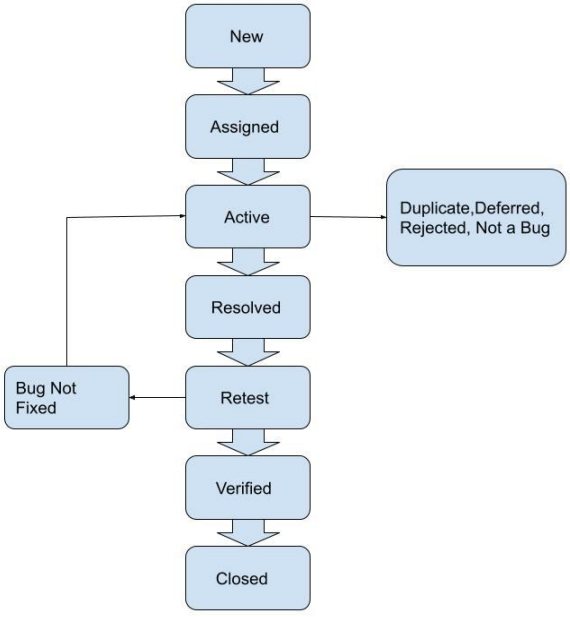


Fig. 2 Life Cycle of a Bug

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue VII Jul 2023- Available at www.ijraset.com

VIII. IMPORTANCE OF EARLY BUG DETECTION OF A BANKING APPLICATION

Bugs are a significant problem in the world of software. Even worse, if they go unnoticed, they can cause a data breach or other significant security event that may severely affect the banking industry. The cost and time of fixing bugs may increase with late bug detection.

It's worthwhile to detect issues before they occur, even though doing so requires a lot of labour. Maintaining an effective bug detection procedure may help an organization remain protected from outside threats.

Any software that drives critical infrastructure or networks that could be connected to key systems, such banks or Fintech organizations, needs to have excellent bug detection capabilities. For instance, it makes sense to spend a little more time and money on detecting and fixing issues while working for a company with extremely important software, such as banking apps. The future avoidance of technical debt can be facilitated by early investment in bug elimination.

With each passing stage of the software development process, the cost of finding and fixing software bugs increases noticeably. It can be exceedingly costly and dangerous to patch bugs that are found after the program has been launched, and it usually costs much more to do so than to catch them sooner. Fixing a defect in the code may also affect how the application functions, necessitating other changes that would add to the cost, time, and effort involved. The cost of the program will increase in proportion to how long it takes to detect a bug. That's why early detection of bugs is very crucial for any software. The Fig. 3 below demonstrates how the cost of bug detection increases as detection time increases.

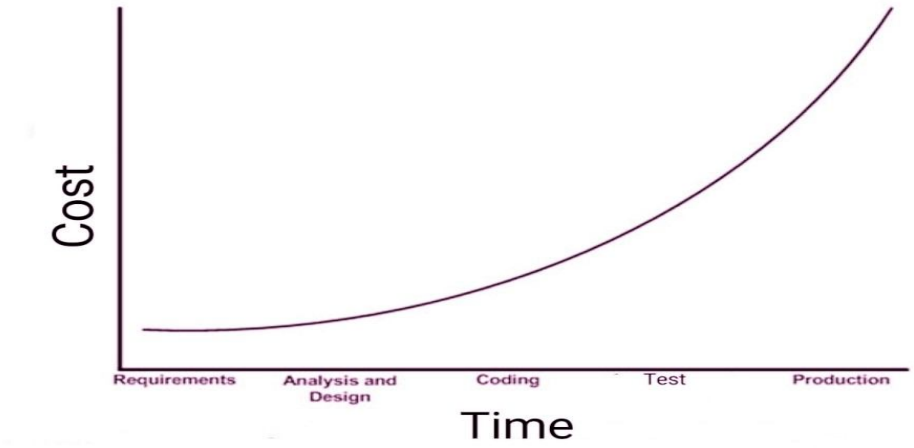


Fig. 3 Cost Increment with Time In Case of Bug Detection

IX. METHODOLOGIES TO ACCELERATE THE BUG SOLVING PROCEDURE

There are numerous approaches for IT companies to fix bugs. The diversity of methods used reflects the team's varying levels of risk tolerance and urgency in releasing new features. Depending on the product's type and level of mission criticality, bug solutions can differ. The severity and priority of a bug can also influence whether or not it receives an immediate solution. Perfect coordination between testers and developers can accelerate the whole bug solving procedure.

The strategies like creating a standardized procedure and plan to rapidly fix any bug, using time management techniques, establishing benchmarks, prioritizing test code, making use of chaotic engineering and adopting a mindset of mission-criticality can be used by software teams to resolve bugs faster. A Focus Group Discussion (FGD) and Key informant interview (KII) between the QA team and developers regarding bug priority, severity and impact of existing bugs in the software can accelerate the bug solving procedure and results in an accurate software release.

X. TYPES OF SOFTWARE TESTING AND THEIR IMPLEMENTATION ON BANKING APPS

Software testing type is a segmentation of various testing activities into groups, each of which has a specific test objective, test plan, and test outputs. Validating the application under test for the specified test objective is the aim of having a testing type. Due to the complexity of banking applications, testers must conduct different types of software testing because this is the only way to assure that banking software are stable and bug-free. In this paper types of testing necessary for a complete banking software test will be discussed.

A. Unit testing

Unit tests are intended to examine a particular system module, such as determining whether a component controller establishes the proper state. It entails disassembling the software's elements into distinct, independent parts in order to assess each one's functionality separately. Depending on the programming syntax and environment, a unit may be a function, method, class, or module. Basically, unit testing is done by the developer. To make sure that the individual units achieve the desired results, developers create short test cases that are especially targeted at those units and simulate various inputs and circumstances. Unit testing is important for evaluating banking apps because they comprise many different units and modules. Therefore, unit testing is an essential procedure in banking software to guarantee the dependability, accuracy, and stability of the code.

B. Integration Testing

Payment of bills, Money transfer, deposits, and other modules could all be present in a banking application. Numerous elements have been developed as a result. All the components are integrated and validated during integration testing. Integration testing ensures that the banking application seamlessly connects to external banking systems, credit bureaus, and payment gateways for evaluating the application's integration APIs' usability, security, and dependability.

C. System Testing

System testing is testing done on an entire integrated system to determine if it complies with the requirements. All integrated components that have passed integration testing are the input for system testing. The general functionality of the system is verified by system testing. It is a necessary part of testing banking apps because it helps to validate the specifications and make sure that the system fulfils user requirements. This testing also raises the standard and effectiveness of the system.

D. User Acceptance Testing

A crucial stage in the development of banking software is user acceptance testing (UAT). It entails confirming the functioning, usability, and general applicability of the product to end users. User Acceptance Testing should adhere to a few guidelines when developing banking software.

1) *Define UAT Scope:* The User Acceptance Testing phase's scope has to initially be determined as well as which particular functions, features, and user situations will be tested. This will assure thorough coverage and assist in concentrating the testing effort.
2) *Identify User Testers:* The next stage is to choose a set of representative users who will take part in the User Acceptance Testing after determining the UAT scope. Ideally, these users should come from the banking application's intended user base, such as bankers or actual clients.

E. Black Box Testing

To make sure the application's output is accurate for all possible positive and negative inputs, black box testing is carried out. Black box testing may include many different forms, including Equivalence Class partitioning, Boundary value analysis, error guessing, etc. By employing this testing strategy, testers can test a banking app's user interface from the viewpoint of the end user.

F. White box Testing

To prevent software from having duplicate code, white box deals with how code operates internally. Testing of code lines, programs, flows, logic, loops, structures, functions, class communication, and other internal program testing are all included in this. Proceeding with white box testing from early stages ensures code functionality and reduces the time required for subsequent, higher-level testing.

G. Usability Testing

To make sure that banking software is user-friendly, intuitive, and satisfies the demands of its intended users, usability testing is a critical component of the testing process. Customers of all kinds can use a banking application. Some of these users might not have the knowledge and abilities necessary to use the app for financial purposes. Therefore, the banking application will be more useful to a greater number of consumers if the software interface is straightforward and simple to use. It involves assessing how simple it is for corporate users or bank clients to use the program.

The tester will take into account the various user types and roles, such as customers, bank staff, and administrators, who will interface with the software. Each user profile could have particular expectations and requirements for usability. This testing is not performed by the developer or tester but is performed by the business users. The business users conduct this testing rather than the testers or developers. Two different types of usability testing exist. Comparative usability testing and explorative usability testing are two examples.

1) Comparative Usability Testing: In this type of testing, the usability of several websites and applications is evaluated against one another. To deliver the optimum user experience is the goal of such testing.
2) Exploratory Usability Testing: The goal of this testing is to determine what characteristics the new software or application has to have in order to satisfy the needs of the bank's customers.

The banking software should be simple to use, comprehend, and navigate for the end user. Iterative usability testing should be used to continuously enhance the user experience of banking software. To make sure the program is simple to use, effective, and meets user needs, regular user feedback is indispensable.

H. Cross-Browser Compatibility Testing

Millions of customers use thousands of iOS and Android-powered devices, multiple browsers to access banking apps. As a result, it must be compatible with all platforms and have flawless operation across all features on each platform. To assess the banking online application's compatibility with different devices or browsers, QA testers can do a cross-browser compatibility test and receive an in-depth report.

I. Database Testing

Database testing is equally crucial to functional testing since banking applications include complicated transactions that are carried out at both the UI and database levels. It is essential to test the database of banking applications to guarantee data confidentiality, accuracy, and integrity. It employs methods including data loading, data accuracy inspections, data integrity check, and data migration validation. Database testing is mostly done to make sure that the application can store and retrieve data from the database without losing any of it. To make sure that the data contained in the database is accurate, consistent, and in the required format, testers must carry out data validation checks. Conducting tests are needed to validate the accuracy and completeness of the transferred data if the banking software requires data migration from legacy systems or earlier releases. Database testing checks that the data has been loaded, converted, and transferred correctly into the new database architecture. The database's security controls that have been put in place are verified. It also inspects that user privileges, access controls, encryption of sensitive data, and compliance with data protection laws are all appropriate. This testing checks for weaknesses including those caused by SQL injection, illegal access, and potential data leaks. Additionally, it ensures that database backup and recovery procedures are effective.

J. Regression Testing

Regression testing is the process of retesting features that have already been developed and tested to make sure that any updates or changes to the software have not introduced new flaws or undesirable consequences. Banking apps need to be updated frequently with new features to support newly launched services. In this case, regression testing is essential for ensuring a continuous release velocity. To reduce risks related to software upgrades and modifications, banking software must undergo regression testing. It aids in guaranteeing that existing features continue to perform as planned and the general operation and integrity of the banking system are not jeopardized.

K. Alpha Testing

Alpha testing is frequently done when it is still in the early phases of development. It's a very significant testing type to test banking apps. Developers, testers, and other internal team members typically carry out this task. Alpha testing focuses on finding and resolving bugs, confirming fundamental functionality, and making sure the software complies with the requirements, which are the major components of testing banking software.

In order to facilitate close collaboration and rapid feedback loops, the testing is carried out in a controlled setting, frequently on the premises of the development team. Before moving on to more extensive testing, alpha testing enables developers to gather suggestions and make adjustments.

L. Beta Testing

Following the alpha testing phase, external users who are not directly connected to the development team participate in beta testing. Before the software is officially released to the public, beta testing is often carried out when it becomes more reliable and mature. Before being officially deployed in a bank's environment, banking software typically undergoes beta testing. The goal of beta testing is to find any residual bugs that may have been missed during alpha testing, as well as receive user feedback, evaluate the general reliability of the system, and pinpoint any usability difficulties. Before the product goes public officially, beta testing offers the chance to get feedback from actual users, and that feedback helps to validate and improve the software.

M. Security Testing

There are various types of security testing tools and methods used in banking software to identify vulnerabilities, assess risks, and ensure the security of banking systems. Here are some mostly used tools and methods used in banking software for security testing.

1) Penetration Testing: Penetration testing, commonly called ethical hacking, simulates actual attacks on financial software like banking software to find weaknesses and potential access points. The banking sector frequently uses Nessus, Burp Suite, and Metasploit for penetration testing.

2) Vulnerability Scanning: Tools for vulnerability scanning are used to automatically check networks, systems, and banking software for known flaws. These instruments can spot flaws and produce thorough reports on potential security problems. Tools for scanning for vulnerabilities include OpenVAS, Qualys, and Nexpose, as examples.

3) Web Application Security Testing: Web application security testing is essential for all banking-related web applications, including online banking portals. Doing dynamic application security testing (DAST) to find vulnerabilities specific to web applications using tools like OWASP ZAP, Acunetix, and AppScan is common practice.

4) Security Awareness Training: Security awareness training is just as important for banking software security as automated tools and techniques. Bank workers are trained in social engineering attacks, best practices for security, and how to identify and report potential security concerns.

Collectively, these techniques and tools provide risk assessment, vulnerability identification, and security assurance for banking software systems. To conduct thorough security testing and preserve a strong security posture, banks must use a combination of these strategies.

N. Performance Testing

The banking sector relies heavily on performance testing with different types of testing tools like Apache JMeter, LoadRunner, and Web LOAD to guarantee the dependable and effective operation of software systems. The following are the main justifications for why performance testing is crucial in the banking sector:

1) Validating System Responsiveness**:** JMeter performance testing aids in determining whether banking software systems are responsive under various load levels. JMeter tests the system's responsiveness by simulating realistic user traffic and transaction volume, making sure it adheres to the predetermined performance criteria. By ensuring that clients can use financial services without suffering delays or sluggishness, this validation is essential for improving their overall user experience.

2) Handling Peak Loads: Banking systems often experience peak loads during specific periods, such as salary disbursements or festive seasons. Performance testing with JMeter allows banks to assess their software systems' ability to handle high transaction volumes efficiently. By gradually increasing the load and analysing the system's performance metrics, such as throughput and resource utilization, JMeter helps identify any bottlenecks or performance degradation. Banks can then optimize their systems to handle peak loads effectively, ensuring uninterrupted services for their customers.

In conclusion, performance testing is crucial for the banking sector. It enables banks to verify system responsiveness, effectively manage peak loads, assure scalability, optimize resource utilization, proactively identify performance concerns, and adhere to SLAs. Banks can build dependable and high-performing software systems that give clients flawless banking experiences while preserving their confidence and pleasure by utilizing JMeter's capabilities.

O. Manual Testing

Software testing that is done manually by a QA Analyst is referred to as manual testing. The purpose of it is to find bugs in software that is still being developed. When performing manual testing, the tester verifies all of the program or application's critical components.

Without the use of any automation software testing technologies, the software testers carry out the test cases and produce the test reports in this method. It aids in the identification of bugs in software systems and is a traditional approach to all testing forms. To complete the software testing process, an experienced tester is typically required.

P. Automation Testing

Testers create code or test scripts to automate the execution of tests in automated software testing. To create scripts for testing and verifying the product, testers need the proper automation tools. The objective is to speed up test execution. To compare actual outcomes with projected results, automated testing solely relies on pre-scripted tests that execute automatically. This makes it easier for the tester to assess whether an application functions as planned. Without the assistance of a manual tester, automated testing enables one to carry out repetitious tasks and regression tests. The initial testing scripts for automation must be created manually despite the fact that all procedures are executed automatically.

Software testing like Black Box Testing, White Box Testing, Unit Testing, System Testing, Integration Testing, Acceptance Testing, Performance Testing, Security Testing and Regression Testing etc. can be carried out in a variety of ways, both manually and with the use of automated tools. Using automated tools for performance testing, security testing, and regression testing yields the most effective outcomes.

XI. SELECTION BETWEEN MANUAL AND AUTOMATED TESTING FOR BANKING APPLICATIONS

From manual testing immediate and precise visual feedback can be obtained. It is less expensive because there's no need to use budget for the automation tools and procedure. An automation test would need to be coded, which may take time while testing a simple change, whereas testers could perform a manual test right away. However, manual testing procedures are less reliable because they are carried out by people. As a result, mistakes and errors are always possible. Since the manual testing procedure cannot be recorded, it cannot be used again.

In comparison, automated testing aids in finding more bugs than manual testers. Testers may have a quick and effective approach since the majority of the testing process is automated and can readily enhance productivity as well. A process of automation can be recorded. This enables testers to repeat and carry out the same testing procedures. In contrast to manual testing, which involves humans, automated testing is carried out using software tools, so it is labour- and fatigue-free. Numerous applications are supported by automated testing. Automation testing tools can boost testing coverage; always remember to examine every single unit. Automated testing has a higher initial cost than manual testing, but that cost per test falls as the number of test cases and builds rises which is shown in Fig. 4.

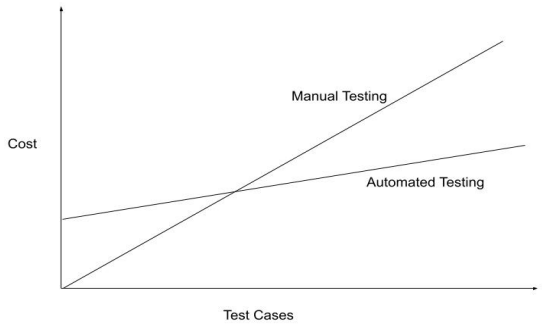


Fig. 4 Cost Effect with Test Case Increment for Manual and Automated Testing

Gaining insight into software UI's visual elements, such as colours, fonts, sizes, contrast, and button sizes, are challenging without the human factor. Due to the potential expense of the automation testing methods, the cost of the testing project may increase. Software testing automation is not yet error-free. The scope of automation is constrained by the limitations of each automation technology.

Script debugging is a big obstacle in automated testing. Test maintenance is expensive, which is every organization's biggest difficulty.

The type of testing will determine how well automation testing and manual testing perform. Exploratory, Usability, and Ad-hoc Testing should all be carried out manually for the best outcomes. Regression testing, load testing, performance testing, and repeated execution should all be done using automation testing for the most effective outcomes.

However, in most instances, due to changes in central bank policies, banking software is often updated. The automated test scripts must be updated in order to reflect these changes. Then it becomes more expensive, requires a lot of work to implement again, and requires constant upkeep. Therefore, manual testing would be beneficial for banking applications most of the time. However, manual testing becomes time-consuming for repetitive tests. Therefore, testers should combine manual and automated testing in a test plan for the best possible outcomes.

XII. INCORPORATE MANUAL AND AUTOMATED TESTING IN THE TEST PLAN

Incorporating Manual and Automated Testing to the test plan could be an appropriate option for the best outcomes. Understanding that no test plan can be carried out entirely by automated tools is the initial phase in this approach. Identifying which parts of the test plan belong in a manual test list and which parts are best suited in the automated test list is the most difficult decision. Choosing the right automation tools is another major challenge.

While testing banking software, 70% of the time goes into manual testing, and the remaining 30% is done using appropriate automated tools. A test team spends about 70% of its time on the 30% of test cases that are utilized most frequently and that have the biggest effects on client retention. The team will have more time to work on other projects due to the 30% of test cases that will significantly shorten total test duration. That could serve as a decent starting point. Each day, all releases, and all builds spend several hours running these test cases. The result tends to remain constant. Despite being repetitive and dull, these test cases are indispensable as these are the paths to accomplish activities successfully. The company and the test team's existence are thus supported by these tasks. Even though the test scenarios are time-consuming, they are crucial.

It is essential to cut down on the time needed to design and run these tests because the remaining 70% will need to be done manually. Initially, testers may utilize technology and the advantages of automation rather than having to manually build test cases step by step. The time it takes to write a test case will be reduced by more than 90% even though a tool will obviously never be able to write a perfect test case. So the testers will therefore need to make some alterations and modifications.

XIII. TEST METRICS AND THEIR SIGNIFICANCE ON BANKING SOFTWARE

Client satisfaction can be increased by verifying the quality of software goods and services using the correct set of trustworthy software testing measures. When offering software solutions and producing applications, software companies must deal with a number of requirements, customer demands, and business limitations. When there are restrictions of production times and costs on projects like banking applications, metrics become even more helpful. Only using widely accepted and efficient software testing metrics can increase the effectiveness and progress of the software testing project.

The following is a list of some of the most significant Software Testing Metrics for banking apps:

Based on how the testing team accumulates data, methods of determining defect percentages, and other relevant dynamics of test case coverage, testing metrics can be identified.

A. Test Efficiency

Test effectiveness is a crucial metric for banking apps, evaluating a test set's bug-finding capabilities and comparing defect numbers to product faults, indicating the test's efficacy and value.

B. Test Coverage

Metrics evaluates software testing, especially continuous testing, in high-quality banking software. Test coverage determines the scope of testing, ensuring high-quality products by covering multiple application objects or components.

C. Test Effort

Test, team abilities, quality objectives, and strategy. As a result, banking software remains cost- Effort encompasses software testing and costs, influenced by factors like maturity, object quality, infrastructure effective.

D. Quantitative Metrics

Software testing techniques use number-based representations of test efforts, recording statistics like test cases, defects, test hours, and bugs discovered at various phases. This quantitative metrics has a great impact on release procedure.

E. Derivative Metrics

Derivative metrics improve testing accuracy by identifying issues in software components, aiding team leads, test managers, and customers by applying algorithms to absolute measures.

F. Efficiency Metrics

Software testing metrics track and improve effectiveness through derived metrics like passed, failed, and blocked test cases, as well as bug resolution and fix time. This enables determining test efficiency and overall efficiency.

G. Economic Measures

Maintaining a project's budget, especially in banks, is crucial for retaining testing costs. Factors like employees, facilities, and tools impact testing costs. Understanding projected and actual expenditures is essential, using metrics like real expenses, budget deviation, and bug fixing averages.

H. Team Metrics

These metrics assess workload distribution fairness among test team members, identifying those needing additional explanations. These metrics should never be used to isolate people or assign blame; rather, they should only ever be used as a teaching tool.

I. Status of Test Execution

Test execution status chart displays total executions in passed, failed, blocked, incomplete, and unexecuted categories, aiding daily status meetings with expanding and contracting bars.

J. Tracking of Test Execution and Defect Discovery Rates

Graphs visualize testing and defect detection rates compared to target levels, using error counts and execution rates, indicating early warnings for testers to adjust.

K. Change Metrics

Banking software undergoes frequent changes due to central bank policies. As modifications are made to the system, it is necessary to assess its stability to ascertain their consequences. Usually, changes lead to more errors, lessened program stability, missed deadlines, degraded quality and many more. Understanding required modifications is very important in this regard.

XIV. PROJECT MANAGEMENT METHODOLOGIES FOLLOWED IN BANKING SOFTWARE RELATED INDUSTRIES

A. Agile Scrum

Agile is a project management approach that places a strong emphasis on adaptability, teamwork, and iterative development. It prioritizes producing high-quality software by segmenting large projects into manageable chunks known as sprints. Leads Corporation Ltd., the creator of “BankUltimus” at Bangladesh strictly adheres to the agile process to provide clients with high-quality products.

The agile technique is fundamentally comprised of scrum meetings. They are quick, time-limited meetings that are held every day to keep the development team on schedule. Team members of Leads Corporation Ltd. and many other software companies who follow Agile scrum, begin each day with a scrum meeting conducted by a scrum master to exchange updates on progress, talk about any problems or difficulties, and plan the following day's activities.

Retrospective meetings provide a space for the team to consider how they worked together, their procedures, and the tools they employed during the sprint. Like Leads Corporation Ltd. Many other software organizations in Bangladesh encourages continual learning and improvement by having team members share what went well and what may be improved in subsequent sprints.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue VII Jul 2023- Available at www.ijraset.com

Each sprint begins with a planning meeting that includes the whole development team from specific departments of the software companies. The team determines collectively at these meetings which feature, or user stories will be tackled during the following sprint. They create a reasonable plan for attaining the sprint objectives and estimate the work needed to complete each task. In the Agile scrum methodology, software testers play a crucial role. Before a project is considered completed, specifically before it is delivered to a client, the requested tasks must be checked. Testers evaluate what has been created by contrasting the outcome with what was anticipated.

Scrum meetings, review meetings, retro meetings, and planning meetings are just a few examples of Agile practices that are essential to promoting cooperation, openness, and adaptation in the banking software sector. They support teams in producing high-quality software solutions that satisfy the changing demands of the sector. In Fig. 5 stages of Agile scrum is presented.

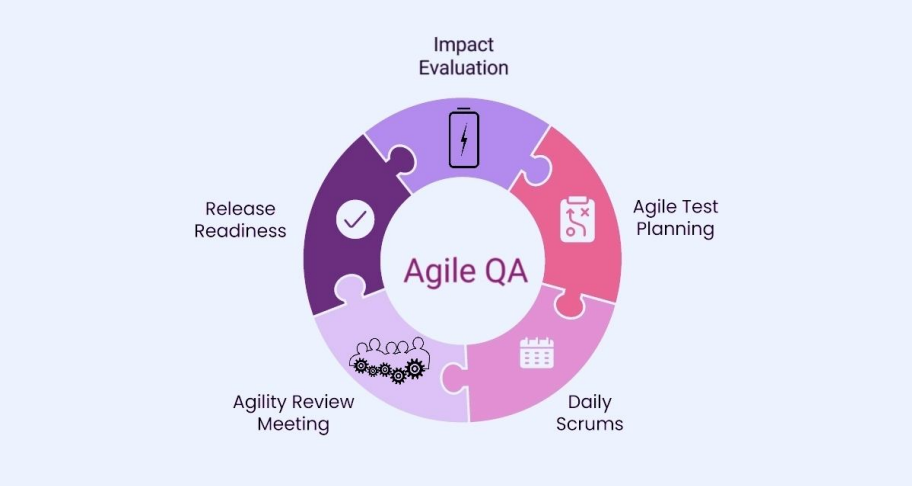


Fig. 5 Stages of Agile Scrum

1) Importance of QA in Agile Scrum: A tester's role in Scrum is typically to design and execute test cases to verify program compliance with required documents. But conducting tests and reporting bugs are just a small part of QA work. The QA engineers engage and carry out a number of tasks alongside other team members in a Scrum team. They collaborate closely with business analysts and developers on projects where they are involved from the very beginning. The QA position in Scrum does not have a separate team that tests the application that is being developed. Instead, the Scrum team consists of engineers, business analysts, and quality assurance specialists who all collaborate. In addition to developing test cases, testers can act as a proxy product owner to assist their product owner in writing acceptance test cases. When the product owner is unavailable, testers step in to act as their representative, which keeps the team moving forward at all times. To assist with explaining the business needs, they can also engage in conversation with the product owner by posing queries and refuting presumptions.

2) Implement QA in Agile Scrum: The actions mentioned below should be taken by testers in order to successfully include themselves into an Agile scrum for a banking application:

a) Risk Evaluation and Impact Analysis: Along with intended end users, the QA team also solicits input from stakeholders at all levels. They list the potential risks and problems and classify them according to the outcomes they expect. Although the analysis might not detect everything, it will allow room to address unforeseen problems if they arise.

b) Make a Test Strategy: A test strategy is created by testers and stakeholders. Every aspect of the testing strategy is included in the plan, including the goals, schedule, hierarchy of deliverables, expected cost, and resources for automated as well as manual tests. The goals for each testing technique (such as unit, acceptance, integration, and smoke tests) are also explicitly stated in the strategy. This ensures that every test is run at the appropriate time and in the proper testing environment, and that it collects useful information for enhancing the quality of software.

c) Collaborate with the End user and The Developers: The QA team must get feedback from the customer after each sprint and consistently offer testing feedback to the customer in Agile model. Agile testers must consider the customer, developer, business and support viewpoint, and to do this, they must work closely with each of these groups. To assist them in creating the acceptance criteria for their user stories, the testers occasionally double as the proxy product owner. Developers and testers occasionally work together to create test cases that serve as the acceptance criteria in the Agile approach.

The team's collaboration, particularly between developers and quality assurance, eliminates confusion and makes sure that everyone is on the same path.

d) Estimate the Release Date: The build undergoes testing each time an update or function is added. The component or feature has been extensively tested before being added to the source-code repository. It passes if it is prepared for release. The test findings are noted and forwarded to the development team if they are not. The developers will then update the feature or function and submit it again for evaluation. The team will become more aware of the most likely release date with each successful Agile sprint. The customer can then be informed of this. The majority of clients will value this degree of openness and feel relieved knowing that their project is being handled carefully.

B. CI/CD Pipeline

The banking industry is rapidly changing, with rapid service changes and faster release cycles. To beat rivals, companies need to launch products quickly. CI/CD, or Continuous Integration and Continuous Delivery, is adopted by software organizations to improve quality and speed to market. By employing the best technologies to hasten deployment, the DevOps technique, CI/CD streamlines Agile development. This strategy enables automated software delivery, code development, testing, and secure software release, reducing manual defects, increasing release rates, reducing costs, and removing risks. The CI/CD pipeline comprises delivery, deployment, integration, and testing phases, ensuring continuous monitoring and automation.

Continuous Integration: Continuous integration involves automatic code modifications, regular testing, and error checks using automated builds for application improvements.

Continuous Delivery: A team builds software applications in short cycles using the continuous delivery approach. It guarantees that the software can be delivered whenever it is required.

Continuous Deployment: This technique uses automatic deployment to supply product features. It assists the testing team in verifying the stability and accuracy of the codebase modifications.

1) Importance of CI/CD in Banking Sector: CI/CD software lifecycle is crucial for software companies in the banking industry to stay ahead in the competitive economic environment. It allows for quick and continuous software updates, providing a strong competitive edge. This flexibility allows for quick and continuous testing, saving resources and reducing time and money spent on ineffective solutions. Automated testing is essential for the entire product development process, identifying and resolving flaws and defects early on. This ensures a more reliable release process and reduces the probability of human mistakes, making the entire process more fruitful.

2) Stages of CI/CD pipeline: To implement CI/CD it's very important to understand the stages of the CI/CD pipeline. The CI/CD process involves four stages: source, build, test, and deployment/release. Source involves sending modification alerts to the CI/CD tool, focusing on source control and tracking changes. Build integrates source code and dependencies, while test runs automated tests to verify functionality and code accuracy. Finally, deploy/release releases the built product after completing test scenarios and meeting client requirements. These stages are shown in Fig. 6.

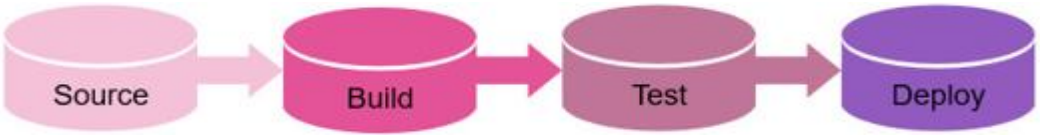


Fig. 6 Stages of CI/CD pipeline

3) Importance of QA in CI/CD: CI/CD pipelines require strong quality assurance (QA) processes for successful creation, testing, and deployment. Manual testing is ineffective sometimes due to iteration delays, slow feedback, and release updates. Despite tools for continuous processes, many CI/CD pipelines are often carried out manually, resulting in iteration delays and slow product rollout. Proper implementation of QA in CI/CD leads to cost reduction and security, allowing developers to focus on developing, releasing updates, and bug fixation instead of software quality.

4) Implement QA in CI/CD: To effectively integrate Quality Assurance (QA) into a banking application's CI/CD pipeline, the following steps are necessary:

a) Execute Automation and Exploratory Testing: Evaluate the banking policy, scope of automation, client requirements, stability, and accuracy of the software. The QA team can start automating their process once everything is set up.

b) Implement Test-Driven Development: Test-driven development ensures that every module in the CI/CD pipeline is in a deployable condition.

c) Set up a Proper Environment for Test Release: Test every module in an environment similar to the client's environment to ensure proper functionality after deployment.
d) Choose the Right CI/CD Tools: Choose tools that suit the requirements and check compatibility with iOS, Android, or Windows.
e) Use Cloud-Based Tools for Cross Browser Testing: Cloud-based cross-browser testing solutions can assist multiple test cases concurrently without relying on hardware requirements.
f) Coordinate Testing with Development: Work cooperatively with developers and testers to improve the project's quality and speed up the release process.
g) Continuously Monitor Test Failures: Early detection of test failures allows for rapid improvements without delaying the process.
h) Implement Negative Testing: Negative testing helps identify flaws and prevent application crashes, increasing the application's testing coverage.
i) Automate Regression Testing: Regression testing ensures that modifications do not result in anomalies or component failures.
j) In summary, implementing automation and exploratory testing, test-driven development, and utilizing cloud-based tools can significantly improve the quality and stability of banking applications.

XV. CHALLENGES AND LIMITATIONS IN SOFTWARE TESTING FOR THE BANKING INDUSTRY

When it comes to software testing, Bangladesh's banking sector has a number of special difficulties. Due to the complexity of financial systems, strict regulatory requirements, worries about data privacy and security, and resource constraints, these problems exist. For the banking industry to adopt efficient software testing procedures, it is essential to recognize and overcome these problems. The solutions to these problems for the banking sector will also be covered in this paper.

A. Dynamic and Complex Banking Systems

Inherently complicated, banking systems are made up of numerous interrelated parts such as core banking systems, platforms for customer relationship management (CRM), payment gateways, and third-party connections. It is essential to test these complicated systems thoroughly to make sure that all of their features and interactions are flawless. Furthermore, banking systems are dynamic, undergoing regular updates, improvements, and the inclusion of new features. Testing these ongoing modifications while preserving backward compatibility poses a substantial challenge because it calls for meticulous preparation and execution to avoid hiccups and guarantee the integrity of the system.

B. Data Privacy and Safety

Data security and privacy are important objectives in the banking industry. Countless sensitive customer data sets, including financial records, transactional data, and personal data, are managed by banks. Testing banking software systems while maintaining the confidentiality and integrity of customer data is extremely challenging. Strict security testing protocols must be implemented in order to identify vulnerabilities, protect against unauthorized access, and halt any data breaches. The need for banks to show the secure handling of cardholder data in order to adhere to regulations like the Payment Card Industry Data Security Standard (PCI DSS) further increases the complexity of testing.

C. Compliance with Regulatory Standards

Numerous regulatory norms and directives, including those enforced by the Bangladesh Bank and global organizations like the Basel Committee on Banking Supervision, are applicable to the banking sector. A significant difficulty in software testing is ensuring compliance with these requirements. The regulatory requirements for data protection, anti-money laundering (AML), know your customer (KYC), transaction monitoring, and reporting must be validated by banks for their software systems. To guarantee that the software satisfies the relevant compliance criteria, testing must be coordinated with these regulatory frameworks.

XVI. RECOMMENDATION AND FUTURE WORK

Banking apps are currently in the top five categories of frequently used apps. That’s why banks need to take a proactive approach to overcome all the obstacles found in banking software.

To overcome these difficulties, testing teams can benefit from investing in training and skill development, implementing strong testing procedures, using automation technologies, performing extensive risk assessments, and more. Collaboration with software providers, government agencies, and trade groups can also offer insightful guidance and assistance in handling the testing requirements of the banking sector. By successfully addressing these issues, banks may secure the dependability, security, and compliance of their software systems, upholding client confidence, reducing risks, and providing greater service. The study is intended as future work where all of the banking apps shortcomings will be discussed as well as the solutions utilizing a variety of tools and technology will be covered.

XVII. CONCLUSION

This paper on software testing discusses various methods for testing software, how they are utilized, and how important they are for banking apps in Bangladesh's scenario. When evaluating banking applications, numerous difficulties could be found. In this paper, challenges have been highlighted along with solutions. These papers address the impact of several software testing types on banking apps, including Unit testing, Integration testing, User Acceptance testing, Cross-Browser testing, Security testing, and Performance testing etc. This article has discussed the significance of manual and automation testing as well as how their integration can speed up testing of banking apps. The complete software testing life cycle for banking apps is described in this paper. The bug life cycle, the significance of early bug identification, and strategies for speeding up the bug solving process for a banking application have also been covered here. Furthermore, the method for accelerating the bug-solving process has been described. For banking software, multiple test metrics and project management methodologies, such as Agile Scrum and CI/CD, have been described, along with how they improve the entire testing procedure. Given the significance of banks to the Bangladeshi economy, this paper inscribes several vital approaches and how they might be put into practice to guarantee quick and reliable quality assurance for banking apps.

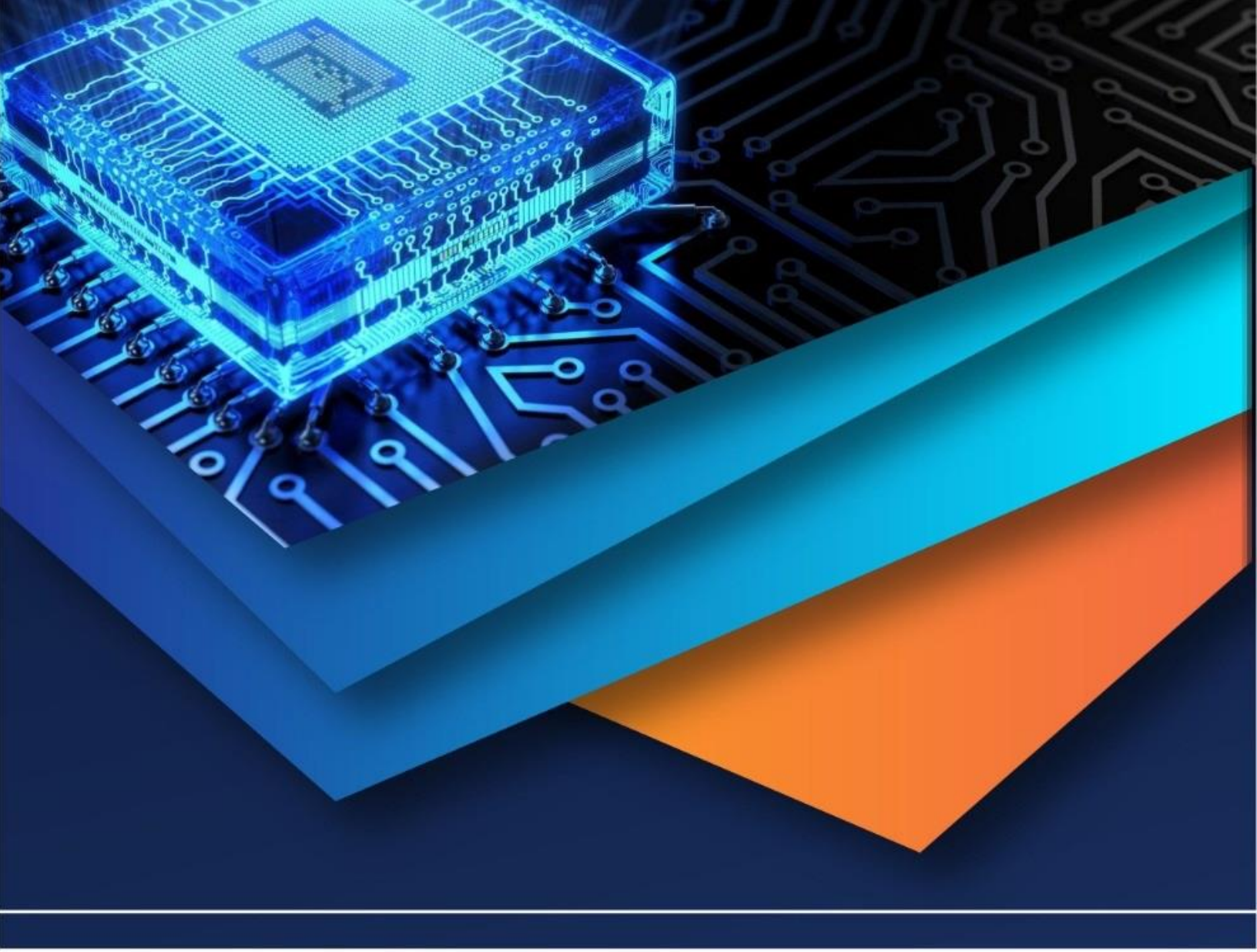
XVIII. ACKNOWLEDGMENT

I appreciate having the opportunity to work on this paper with my teammates. My steering committee's members have all given me a great deal of professional and personal advice and taught me a lot about both technological research and life in general. I want to express my gratitude to everyone who supported and guided me in completing this paper.

REFERENCES

[1] Dhore, L. Wadhwa, P. Shinde, D. Chaudhri, and P. Vyas, "Brief review on different manual software testing approaches & procedure," Journal of Pharmaceutical Negative Results, pp. 455–464, 2023.
[2] F. Sethi, "Automating software code deployment using continuous integration and continuous delivery pipeline for business intelligence solutions," Authorea Preprints, 2020.
[3] I. Hooda and R. S. Chhillar, "Software test process, testing types and techniques," International Journal of Computer Applications, vol. 111, no. 13, 2015.
[4] N. Anwar and S. Kar, "Review paper on various software testing techniques & strategies," Global Journal of Computer Science and Technology, vol. 19, no. 2, pp. 43–49, 2019.
[5] I. Hooda and R. S. Chhillar, "Software test process, testing types and techniques," International Journal of Computer Applications, vol. 111, no. 13, 2015.
[6] I. Jovanović, "Software testing methods and techniques," The IPSI BgD Transactions on Internet Research, vol. 30, 2006.
[7] M. E. Khan and F. Khan, "Importance of software testing in software development life cycle," International Journal of Computer Science Issues (IJCSI), vol. 11, no. 2, p. 120, 2014.
[8] M. Khan et al., "Different approaches to black box testing technique for finding errors," International Journal of Software Engineering & Applications (IJSEA), vol. 2, no. 4, 2011.
[9] R. K. Chauhan and I. Singh, "Latest research and development on software testing techniques and tools," International Journal of Current Engineering and Technology, vol. 4, no. 4, pp. 2368–2372, 2014.
[10] S. S. Isha, "Software testing techniques and strategieg," (Department of computer science) SBMNE College, Rohtak INDI, 2014.
[11] S. Batra, "Improving quality using testing strategies," Journal of Global Research in Computer Science, vol. 2, no. 6, pp. 113–117, 2011.
[12] M. E. Khan, "Different forms of software testing techniques for finding errors," International Journal of Computer Science Issues (IJCSI), vol. 7, no. 3, p. 24, 2010.
[13] Pp_pankaj "Software Testing | Security Testing - GeeksforGeeks." GeeksforGeeks. https://www.geeksforgeeks.org/software-testing-security-testing/
[14] (2018) "Blog-5 Nerve-wracking Challenges Faced While Testing an Online Banking Application." TestingXperts. Available: https://www.testingxperts.com/blog/5-Nerve-wracking-Challenges-Faced-While-Testing-an-Online-Banking-Application
[15] Landy Wingard "Top 10 Banking Industry Challenges — And How You Can Overcome Them." Hitachi Solutions. Available: https://global.hitachi-solutions.com/blog/top-10-challenges-banking-financial-organizations-can-overcome/
[16] (2023) "How to Test Banking Domain Applications: A Complete BFSI Testing Guide." Software Testing Help"Available: https://www.softwaretestinghelp.com/testing-banking-applications/

[17] Anastasiia A. (2023)" How to Test Banking Applications: Complete Guide to Testing in The Banking Domain," Available: https://testfort.com/blog/how-to-test-banking-applications-complete-guide
[18] Dapheny Murphy (2022) “Best Practices for Implementing QA in a CI/CD Software Lifecycle”, Available: https://blog.qasource.com/best-practices-for-implementing-qa-in-a-ci-cd-software-lifecycle
[19] Barkat-e-Khuda (2019) "Economic growth in Bangladesh and the role of banking sector" Available: https://thefinancialexpress.com.bd/views/views/economic-growth-in-bangladesh-and-the-role-of-banking-sector-1547220114
[20] Bangladesh Bank website. [Online] Available: https://www.bb.org.bd/en/index.php



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY