International Journal of Management and Digital Business
Vol.5 No.2 (2026)
ISSN 2962-4762
pp.68-82
https://journal.adpebi.com/index.php/ijmdb

Transition to Automated Regression Testing: Testing Capacity, Stability, and IT Risk Control in Digital Banking

Wisnu Bramantyo[1], Ratih Hendayani[2]
[1,2]Fakultas Ekonomi dan Bisnis, Universitas Telkom, Indonesia,
Email: [1]wisnubramantyo.wb@gmail.com, [2]ratihhendayani@telkomuniversity.ac.id

ARTICLE INFO

Research Paper

Article history:
Received: 25 June 2026
Revised: 15 July 2026
Accepted: 17 September 2026

DOI:
https://doi.org/10.54099/ijmdb.v5i2.1882

ABSTRACT

This study uses a descriptive-comparative quantitative approach to examine regression testing before and after automation. Secondary data were obtained from the Lumos Dashboard and Google Spreadsheets, covering application versions v1.5.0 to v1.36.0. The analysis focused on regression test case growth, statistical comparison between the manual and post-automation periods, and the Weighted Automated Regression Test Pass Rate. The results show that the number of regression test cases increased by 426.39%. The average number of test cases increased from 1,034.08 during the manual period to 2,272.10 after automation. The Mann-Whitney U Test confirmed a statistically significant difference between the two periods ($U = 0.00$, $Z = -4.67$, $p < 0.001$). Meanwhile, the Weighted Automated Regression Test Pass Rate remained stable at 98.36%–100%. These findings indicate that the post-automation period was associated with broader regression testing capacity while maintaining high testing stability. Automated regression testing may also support IT risk control as a preventive internal control mechanism by helping identify potential functional errors before application release. However, the findings should not be interpreted as direct causal evidence of reduced IT operational risk.

Keywords: Automated Regression Testing, Mobile Banking, Software Quality Assurance, Testing Capacity, IT Risk Control

INTODUCTION

Since its public launch on June 19, 2024, Superbank has served more than 5 million customers through continuous product innovation and the strengthening of its digital ecosystem, reflecting a high level of public trust in secure and relevant digital banking services (Superbank, 2025). This growth reflects the importance of maintaining application quality, especially in a mobile banking service that must be stable, fast, and reliable. Application quality is one of the main reasons why customers continue to use a mobile banking application. Superbank users rely on the application for transactional activities such as money transfers, deposits, and loans, as well as non-transactional activities such as onboarding, login, logout, and profile updates. Financial losses are among the most significant consequences of unmanaged IT operational risks, as discussed by Wayan et al. (2016) in the banking sector. Information technology disruptions may result in transaction delays, data processing failures, and increased operational costs due to repeated system corrections. Furthermore, organizations may face regulatory pressure when IT failures create the potential for violations of data security and confidentiality standards (Nafisah Balqis et al., 2025). Software testing becomes more important when the application continues to grow and change. Each new feature, enhancement, or bug fix may affect existing features. Because of this, regression testing is needed to make sure that the application still works as expected after changes are made. In a mobile banking context, this process is not only a technical activity, but also part of maintaining customer trust and reducing operational risk. This practice serves as an important foundation for the success of digital banking, as even minor system errors can significantly affect customer trust and business stability (Pressman, 2020).

According to the study conducted by Alam et al. (2023), many technology organizations still face challenges in fully integrating Quality Assurance or QA into the Software Development Life Cycle or SDLC process. Inadequate testing processes and poor testing planning often lead to a higher number of defects after release, lower software quality, and increased reputational risk when systems become unstable. This challenge is relevant to Superbank, which must continuously ensure the quality of its mobile banking application through repeated regression testing following every feature enhancement or bug fix. The quality of a mobile banking application also has a direct impact on customer experience and trust. Kumalasari et al. (2022) emphasized that system stability, information accuracy, and digital service responsiveness are among the primary factors influencing customer satisfaction and loyalty. As mobile applications have become the primary channel of interaction between banks and customers, digital quality is no longer merely a technical concern but also a strategic factor that influences the competitiveness of digital banking institutions.

Safaat and Tjhin (2024) also found that many organizations still rely on manual regression testing despite its limitations. Manual testing typically requires more time, is more susceptible to human error, and becomes increasingly inefficient as the number of test cases grows. In contrast, automated regression testing enables parallel execution of test scenarios, produces more consistent results, reduces repetitive testing efforts, and accelerates feedback to development teams.

IT operational risk can also affect an organization’s reputation, particularly when digital services experience repeated disruptions or demonstrate low reliability that negatively impacts customer satisfaction. The modern financial industry is highly sensitive to system failures because customers expect speed, accuracy, and service stability. Such failures may not only reduce customer trust but also weaken an organization's competitive position in the digital marketplace (Nafisah Balqis et al., 2025).

Based on this background, this study examines the transition from manual to automated regression testing in Superbank’s mobile banking application. The analysis focuses on changes in regression testing capacity before and after automation, the stability of automated regression test execution, and the role of automated regression testing in supporting IT risk control. Rather than examining a direct causal effect, this study applies a descriptive-comparative approach to identify observable differences between the manual and post-automation periods and to interpret their operational implications for the Software Quality Assurance process.

LITERATURE REVIEW

According to Fernandez-Sanz (2009, as cited in Sutapa et al., 2019), software quality assurance plays a critical role in ensuring the success of software development projects, as inadequate implementation may lead to substantial organizational losses. One of the primary approaches to minimizing such risks is the implementation of software testing processes. Software testing is widely recognized as a technique for verifying and validating software throughout its development lifecycle. Furthermore, Fernandez-Sanz reported that approximately one-third of the total effort in a software development project is typically devoted to software testing activities, highlighting the significant role of testing in achieving software quality and reliability.

Regression testing is a type of software testing that aims to ensure previously developed and tested software continues to function correctly after modifications have been introduced. These modifications may include software enhancements, patches, configuration changes, or other adjustments. Regression testing is particularly important because modern software systems have become increasingly complex, making it difficult to predict how changes may affect existing functionalities (Mathur, 2013, as cited in Akin et al. 2018). According to Persson and Yilmaztürk (2004), when regression testing is performed manually, the process tends to be time-consuming, costly, labor-intensive, repetitive, and generally less attractive to software testers. In contrast, automated regression testing enables testing activities to be carried out more efficiently and effectively. Finsyen Ilham Erdi and Astari Retnowardhani (2024) stated that one of the main challenges of manual testing is the significant amount of time required and its vulnerability to human error. As software systems become increasingly complex, maintaining testing quality through manual execution alone becomes more difficult.

In mobile banking applications, regression testing is especially important because users expect the application to work reliably at all times. Features such as onboarding, transfer, loan application, deposit, and profile management must remain stable even when new features are added or existing features are improved. For this reason, automated regression testing can help the QA team maintain quality while keeping up with a fast release cycle. The growth of users and features in Superbank's mobile banking application creates a need for a faster and more consistent testing process. Previously, regression testing in Superbank was conducted manually, which required more time and effort from QA Engineers. As the application became more complex, the manual approach became more difficult to maintain. Automated regression testing was then introduced to improve efficiency, reduce repetitive manual work, and support more stable testing results.

Previous studies provide relevant evidence regarding the implementation of automated regression testing. Akin et al. (2018), in a banking industry case study, found that the transition from manual to automated regression testing supported reductions in execution time and human resource utilization while increasing testing frequency and providing economic benefits through cost-benefit analysis and Return on Investment. Menegassi and Endo (2020) further demonstrated the relevance of automated testing for mobile applications across multiple device configurations. More recently, Brasil et al. (2024) identified through a systematic mapping study that empirical evaluations of regression testing in real-world mobile application environments remain relatively limited compared with studies focusing on libraries, frameworks, database systems, and web applications. This study differs from previous studies by focusing on an actual digital banking environment and comparing documented regression testing conditions during the manual and post-automation periods. The study also examines whether broader regression testing capacity can be maintained with stable automated execution and interprets the findings in relation to preventive IT risk control.

From the perspective of IT risk control, automated regression testing can also act as a preventive control mechanism. By running regression tests repeatedly before release, the QA team can identify potential system errors earlier. This helps reduce the possibility of defects moving to the production environment. Therefore, automated regression testing is not only related to technical efficiency, but also to the organization's ability to manage system risk. Based on the theoretical and empirical considerations discussed above, the research framework used in this study is presented in Figure 1.

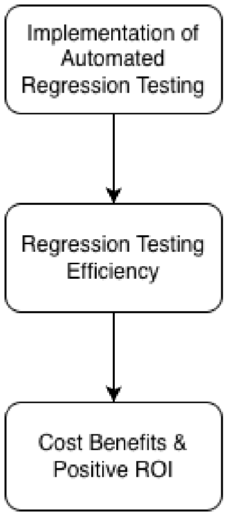

Figure 1. Research Framework

The research framework presented in Figure 1 above was developed based on the study by Akin et al. (2018), which focused on the implementation of automated regression testing and its potential to improve regression testing efficiency. In the final stage of their study, Akin et al. (2018) examined the cost benefits and positive Return on Investment (ROI) generated by the implementation of automated regression testing within the organization. According to Akin et al. (2018), automated regression testing was implemented through the development of an automation infrastructure and automated test suites, as well as the adoption of various test automation best practices to replace the previously manual regression testing process. From a management perspective, improved testing efficiency is expected to optimize resource utilization, reduce testing time and costs, and increase the value of the investment made in implementing automated regression testing.

METHOD

This study uses a descriptive-comparative quantitative approach to describe and compare regression testing conditions before and after the implementation of automated regression testing. The study does not employ an experimental design and therefore does not attempt to establish a direct causal relationship between automation and testing outcomes. Instead, it examines observable differences between the manual and post-automation periods based on documented regression testing data. The data were obtained from Superbank's internal testing documentation, specifically Google Spreadsheets and the Lumos Dashboard. Google Spreadsheets contain regression testing records from the earlier manual-testing period, while the Lumos Dashboard provides automated regression testing records for later application releases. The analysis covers 32 application releases from v1.5.0 to v1.36.0, comprising 12 observations from the manual regression testing period (v1.5.0–v1.16.0) and 20 observations from the post-automation period (v1.17.0–v1.36.0). Testing efficiency in this study is operationally interpreted as the capacity of the regression testing process to accommodate broader test coverage while maintaining stable automated execution. Therefore, efficiency is assessed through test case coverage, differences in regression testing capacity between the manual and post-automation periods, and automation stability. Execution time, monetary cost, and direct human-effort reduction are not measured in this study. IT risk control is examined from a preventive-control perspective. Automated regression testing is assessed based on its role in identifying potential functional failures before an application release. The Automated Regression Test Pass Rate and its corresponding failure rate are therefore used as testing-based indicators and risk proxies rather than as direct measures of realized IT operational risk such as production incidents, financial losses, or system downtime.

Statistical Analysis

The analysis was conducted in four stages. First, descriptive analysis was used to examine the growth in regression test cases across application releases. Second, a statistical comparison was conducted between the manual and post-automation periods. Before conducting the comparison, the Shapiro-Wilk Test was applied to assess the normality of the two groups. Because at least one group did not meet the assumption of normal distribution, the Mann-Whitney U Test was selected as a non-parametric method for comparing two independent groups. A significance level of 0.05 was used.

Third, the Weighted Automated Regression Test Pass Rate was used to evaluate automation stability because the number of executed regression test cases varied across application versions. Finally, the findings were interpreted in relation to the preventive role of automated regression testing in IT risk control. Descriptive and correlation analyses were processed using SPSS, while the additional comparative analysis was conducted using GNU PSPP.

Data Quality Considerations

The study relies on documented internal regression testing records obtained from Google Spreadsheets and the Lumos Dashboard. For analytical purposes, the records were organized according to application release version and testing period to ensure comparability between the manual and post-automation datasets. The analysis was limited to regression testing records available within the selected observation period. As the study uses secondary internal operational data, the findings are dependent on the completeness and consistency of the records available in these systems.

Population and Sample

The population of this study consists of all regression testing results conducted by Superbank's Software QA team during the research period. This includes newly added regression test cases documented in spreadsheets, as well as the results of automated regression test cases recorded in the Lumos Dashboard, which was developed internally by Superbank's QA team. The population was selected because it represents the testing activities used to evaluate the transition from manual to automated regression testing. These data also provide the basis for measuring testing efficiency and the role of automated testing in supporting IT risk control in Superbank's mobile banking application.

The sample used in this study consists of regression testing data considered representative of the efficiency changes after the implementation of automated regression testing. The sample covers application release versions from v1.5.0 to v1.36.0. These versions were selected because they represent the period in which regression testing data were recorded and the transition from manual to automated regression testing became visible. The sample includes manual regression test cases added during the selected release versions, manual and automated regression testing reports, and automated regression testing execution results. Through this sample, the study analyzes the changes, trends, and relationships between automation indicators and regression testing efficiency.

RESULT AND DISCUSSION

Result

The secondary data were collected from two main platforms used by Superbank, namely Google Spreadsheets and Lumos Dashboard. The data include the total number of regression test cases, automated regression test pass rate, and the number of automated regression test cases. After the data were collected, they were compiled in Microsoft Excel and processed using SPSS.

Table 1. Data Used in the Research

No	*Release Version*	Number of *Test Cases*	No	*Release Version*	Number of *Test Cases*
1	v.1.5.0	667	17	v.1.21.0	1856
2	v.1.6.0	682	18	v.1.22.0	1841
3	v.1.7.0	592	19	v.1.23.0	1897
4	v.1.8.0	775	20	v.1.24.0	1861
5	v.1.9.0	812	21	v.1.25.0	1830
6	v.1.10.0	1344	22	v.1.26.0	2730
7	v.1.11.0	770	23	v.1.27.0	1806
8	v.1.12.0	856	24	v.1.28.0	1782
9	v.1.13.0	1263	25	v.1.29.0	1820
10	v.1.14.0	1485	26	v.1.30.0	1838
11	v.1.15.0	1642	27	v.1.31.0	1842
12	v.1.16.0	1521	28	v.1.32.0	3164
13	v.1.17.0	1911	29	v.1.33.0	3688
14	v.1.18.0	1648	30	v.1.34.0	3341
15	v.1.19.0	1658	31	v.1.35.0	3711
16	v.1.20.0	1707	32	v.1.36.0	3511

Base on table 1 The analysis was carried out using a descriptive-comparative approach. The results are presented in the form of tables and figures. The data were used to analyze the growth of regression test cases, automation stability, and the relationship between test case growth and automated regression testing performance.

$$\Delta TC = TC_{final} - TC_{initial} = 3511 - 667 = 2844$$

Based on the result using the formula above, the number of regression test cases in the initial version, v1.5.0, was 667 test cases. In the final version, v1.36.0, the number increased to 3,511 test cases. This means that there was an increase of 2,844 test cases during the application development period. Furthermore, the growth percentage of the number of test cases is calculated using the formula:

$$\text{TC Growth}(\%) = \frac{TC_{final} - TC_{initial}}{TC_{initial}} \times 100 = \frac{3511 - 667}{667} \times 100 = 426.39\%$$

Based on these results, the number of regression test cases increased by 426.39% from the initial to the final application version. This increase indicates that as the features and complexity of Superbank's mobile banking application continue to grow, more regression test cases are required. The QA team needs to expand the testing coverage to ensure that newly developed features and their related functions are properly evaluated. The growth in regression test cases occurred alongside increasing application complexity and broader testing requirements. The addition of these test cases can also be regarded as one of the preventive measures taken by Superbank's QA team to reduce the likelihood of issues or errors affecting users. On the other hand, this significant growth also shows why manual regression testing became more difficult to maintain. When test cases increase rapidly, manual execution requires more time, more human effort, and more resources. Under these conditions, automated regression testing becomes more relevant because it can help the QA team manage a larger testing scope more efficiently.

Comparison of Regression Test Cases Between Manual and Post-Automation Periods

A statistical comparison was conducted to examine whether the number of regression test cases differed between the manual regression testing period and the period after the implementation of automated regression testing. Descriptive statistics showed a clear difference between the two periods. The manual regression testing period consisted of 12 observations with an average of 1,034.08 regression test cases and a median of 834.00. In comparison, the post-automation period consisted of 20 observations with an average of 2,272.10 regression test cases and a median of 1,849.00. The minimum number of regression test cases in the post-automation period was 1,648, which was slightly higher than the maximum comparable value observed during the manual period. To provide an initial overview of the differences between the manual and post-automation periods, descriptive statistics were calculated for both groups. The comparison includes the number of observations, mean, median, standard deviation, minimum, and maximum number of regression test cases.

Table 2. Descriptive Statistics of Regression Test Cases

Statistics	Manual Regression Testing	Post-Automation Period
N	12	20
Mean	1,034.08	2,272.10
Median	834	1,849.00
Std. Deviation	385.00	756.50
Minimum	592.00	1,648.00
Maximum	1,642.00	3,711.00

As shown in Table 2, the post-automation period had a higher mean and median number of regression test cases than the manual period. The mean increased from 1,034.08 to 2,272.10 test cases, while the median increased from 834.00 to 1,849.00 test cases. These results indicate a clear descriptive difference between the two periods, although further statistical testing is required to determine whether the difference is statistically significant. Because the descriptive statistics only show differences in the observed values, a normality test was conducted before selecting the appropriate statistical comparison method. The Shapiro-Wilk Test was used to assess whether the data in each group followed a normal distribution.

Table 3. Shapiro-Wilk Normality Test

Group	Statistic	df	Sig.
Manual Regression Testing	0.86	12	0.05
Post-Automation Period	0.71	20	0.0000526

Table 3 shows that the post-automation group had a significance value of 0.0000526, which is below the 0.05 threshold. This indicates that the data in the post-automation group were not normally distributed. Therefore, a non-parametric test was considered more appropriate for comparing the two independent groups. Based on this result, the Mann-Whitney U Test was used instead of the Independent Samples t-Test. After the normality assessment, the Mann-Whitney U Test was conducted to compare the distribution of regression test cases between the manual and post-automation periods. Table 4 presents the mean rank and sum of ranks for each group.

Table 4. Mann-Whitney Mean Ranks

Group	N	Mean Rank	Sum of Ranks
Manual Regression Testing	12	6.50	78.00
Post-Automation Period	20	22.50	450.00
Total	32		

Base on table 4 The post-automation period had a mean rank of 22.50, which was considerably higher than the mean rank of 6.50 in the manual period. This result indicates that the number of regression test cases in the post-automation period was generally higher than in the manual period. To determine whether the difference between the two groups was statistically significant, the Mann-Whitney U statistic and its significance value were examined, as presented in Table 5.

Table 5. Mann-Whitney U Test

Statistic	Value
Mann-Whitney U	0.00
Z	-4.67
Significance	$p < 0.001$

The Mann-Whitney U Test produced a U value of 0.00 with a significance level of $p < 0.001$. Since the significance value is below 0.05, the result indicates a statistically significant difference in the number of regression test cases between the manual and post-automation periods. This finding supports the descriptive results presented earlier, which showed a higher regression testing capacity during the post-automation period.

From an operational perspective, the findings show that the implementation of automated regression testing occurred alongside a substantial expansion in regression testing capacity. During the manual period, an increase in test cases required a corresponding increase in manual execution activities by QA Engineers. After automation was implemented, part of the regression test execution could be performed automatically, allowing a larger testing scope to be managed without requiring every test case to be executed manually.

However, the statistical difference should not be interpreted as direct causal evidence that automated regression testing alone caused the increase in the number of test cases. Test case growth may also be associated with the addition of new features, increasing application complexity, and broader regression testing requirements across releases. Therefore, the result more appropriately indicates that the post-automation period had significantly greater regression testing capacity than the manual period.

Boxplot

To provide a clearer visual comparison between the manual and post-automation periods, a boxplot was used to illustrate the distribution, median, and range of regression test case counts in both groups. This visualization complements the descriptive statistics and the Mann-Whitney U Test by showing how the distribution of regression test cases differs between the two testing periods, as presented in Figure 2.

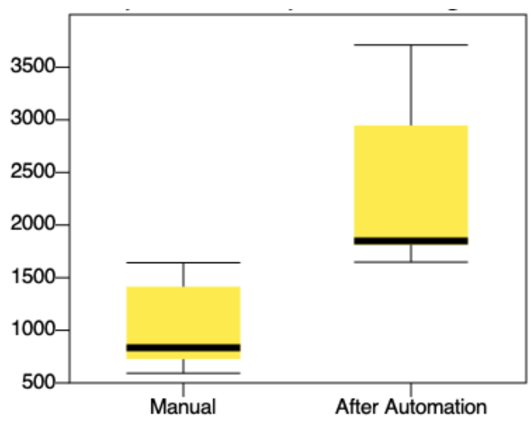

Figure 2. Distribution of Regression Test Cases During Manual and Post-Automation Periods

Figure 2 illustrates the difference in the distribution of regression test cases between the two periods. The median increased from 834 test cases during the manual period to 1,849 test cases during the post-automation period. The post-automation distribution also shows a wider range, which is consistent with the continued expansion of regression testing coverage during later application releases. The visual pattern is consistent with both the descriptive statistics and the Mann-Whitney U Test.

Automation Stability and Weighted Automated Test Pass Rate Analysis

To measure the stability and quality of automated regression testing, this study uses the weighted automated test pass rate. This approach was selected because the number of test cases is different in each release version. Therefore, each version needs to be considered based on the number of test cases executed. Automation stability is represented by the percentage of successful automated regression test execution. The pass rate shows how many automated test cases produced results that matched the expected outcome. A high pass rate indicates that automated regression testing can run consistently and reliably. The development of automation stability and regression test case growth across application releases is presented in Figure 3.

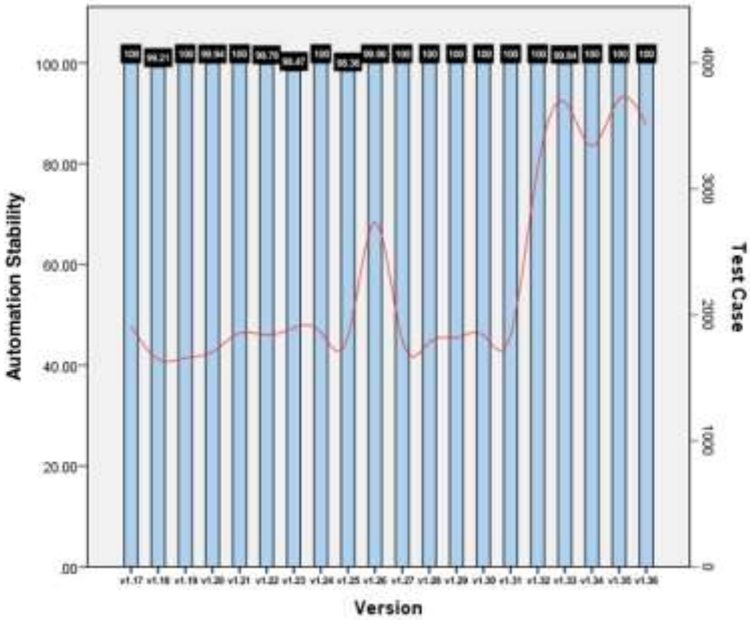

Figure 3. Automation Stability and Regression Test Case Growth

Figure 3 presents the automation stability percentage for each release version together with the number of automated regression test cases executed. The bars represent automation stability, while the line represents the number of regression test cases.

Based on the chart, the number of automated regression test cases increased significantly in several release versions, particularly from version v1.26.0 to v1.36.0. Despite the substantial increase in the number of test cases, automation stability remained very high, ranging from approximately 98.36% to 100%.

These findings indicate that automation stability remained consistently high even as testing coverage expanded. The high level of automation stability also suggests that the testing process could be executed automatically with a very high success rate and minimal failures. The following section presents the correlation analysis conducted to examine the relationship between automated regression testing and regression testing efficiency in Superbank's mobile banking application. To examine the relationship between regression test case growth and automation stability, the results are visualized in Figure 4.

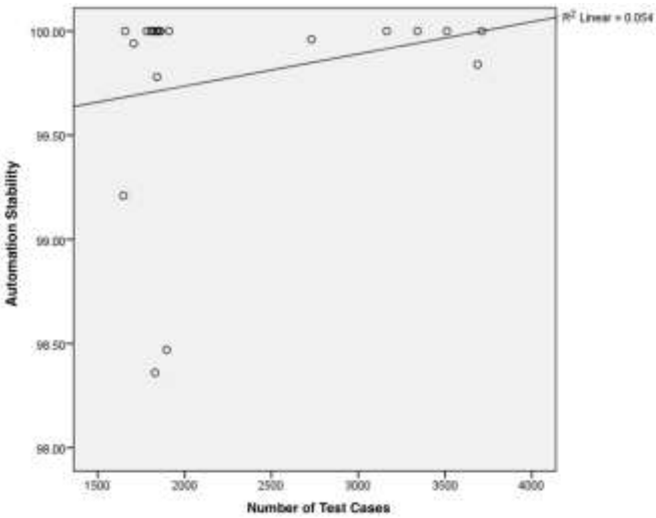

Figure 4. Relationship Between Regression Test Cases and Automation Stability

The scatter plot shows a slight positive trend between the number of automated regression test cases and automation stability. However, the coefficient of determination was only 0.054, indicating that the number of test cases explained approximately 5.4% of the variation in automation stability. Therefore, the relationship between test case growth and automation stability should be considered weak.

This finding suggests that the growth in the number of regression test cases alone is not sufficient to explain automation stability. Other technical and operational factors outside the scope of this study, such as automation script quality, testing environment stability, application changes, test data, and testing pipeline configuration, may have a greater role in determining automated testing stability.

Despite this weak relationship, automation stability remained consistently high while regression testing coverage expanded. The result therefore provides evidence of stable automated test execution during a period of substantial test case growth, rather than evidence that increasing the number of test cases directly causes higher automation stability.

Role of Automated Regression Testing in IT Risk Control

The role of automated regression testing in IT risk control was examined using a qualitative-interpretive approach based on the regression testing results. In this study, automated regression testing is not treated as a direct measure of realized IT operational risk. Instead, its role is interpreted as a preventive internal control mechanism within the software testing and release process. Automated and repeated regression testing enables potential functional failures to be identified before an application is released to users. The consistently high automated regression test pass rate indicates that the automated test suite was executed with a high level of stability during the observation period. From a preventive-control perspective, this provides the QA team with additional information to identify regression failures before deployment. This interpretation is particularly relevant in digital banking, where application failures may affect transactions, service availability, and customer trust. However, this study does not directly measure production incidents, system downtime, financial losses, customer complaints, or defect leakage. Therefore, the findings should be interpreted as evidence of the role of automated regression testing in supporting preventive IT risk control, rather than as direct empirical evidence that automation reduced realized IT operational risk.

DISCUSSION

The findings demonstrate a clear difference in regression testing capacity between the manual and post-automation periods. The average number of regression test cases increased from 1,034.08 during the manual period to 2,272.10 during the post-automation period, while the Mann-Whitney U Test confirmed that the difference was statistically significant ($U = 0.00$, $Z = -4.67$, $p < 0.001$). This finding strengthens the descriptive evidence that regression testing activities expanded substantially following the transition to automated regression testing.

The finding is consistent with Akin et al. (2018), who reported that automated regression testing enabled more frequent testing and reduced dependence on manual execution in a banking environment. However, unlike Akin et al. (2018), who evaluated execution time, human-resource utilization, cost benefits, and Return on Investment, the present study focuses on testing capacity and stability. Therefore, the results of this study should not be interpreted as direct evidence of time or cost efficiency. Instead, they indicate that a significantly larger regression testing scope could be accommodated during the post-automation period.

The number of regression test cases increased by 426.39% between the initial and final observed application versions. This growth reflects the increasing testing requirements associated with application development and broader feature coverage. The result is also consistent with the findings of Safaat and Tjhin (2024), who noted that manual testing becomes increasingly difficult to maintain as the number of test cases grows. In this context, automation provides operational value because repetitive regression test execution can be transferred from manual QA activities to automated execution.

Another important finding is that automation stability remained consistently high, ranging from 98.36% to 100%, despite the substantial increase in the number of regression test cases. This indicates that broader test coverage during the observation period was not accompanied by a substantial decline in automated execution stability. The study, however, does not empirically determine the specific factors responsible for this high stability. Factors such as script maintainability, testing environment stability, test data, and pipeline configuration may contribute to automation performance but were not directly measured.

The coefficient of determination of 0.054 further supports this interpretation. Only approximately 5.4% of the variation in automation stability was associated with the number of regression test cases, indicating a weak relationship. Therefore, automation stability should not be explained primarily by test case growth. This result highlights that the quality and reliability of an automation implementation may depend on additional technical and operational factors beyond the size of the test suite.

From an IT risk management perspective, the findings support the interpretation of automated regression testing as a preventive control mechanism. Automated regression testing provides repeated validation of existing application functionality before deployment and can therefore assist the QA team in identifying regression failures before they affect end users. This preventive role is relevant to digital banking because system failures may affect transactions, service availability, and customer trust.

Nevertheless, the present study does not quantify realized IT operational risk through indicators such as production incidents, defect leakage, system downtime, customer complaints, security incidents, or financial losses. Consequently, the findings indicate that automated regression testing supports IT risk control but do not establish that automation directly causes a reduction in realized operational risk.

From a managerial perspective, the findings indicate that automated regression testing should not be considered solely as a technical testing tool. Its ability to support a broader regression testing scope while maintaining stable execution can assist management in addressing increasing application complexity. Automated regression testing results may also serve as supporting information in application release evaluation and preventive IT risk control processes.

CONCLUSION

This study examined the transition from manual to automated regression testing in Superbank's mobile banking application using a descriptive-comparative approach. The findings show that the post-automation period had substantially greater regression testing capacity than the manual period. The average number of regression test cases increased from 1,034.08 during the manual period to 2,272.10 during the post-automation period, while the Mann-Whitney U Test confirmed a statistically significant difference between the two groups ($U = 0.00$, $Z = -4.67$, $p < 0.001$).

Across the overall observation period, the number of regression test cases increased by 426.39%, while the Weighted Automated Regression Test Pass Rate remained consistently high at approximately 98.36% to 100%. These findings indicate that the expansion of regression testing coverage occurred while automated test execution remained stable. However, the weak coefficient of determination ($R^2 = 0.054$) indicates that test case growth explained only a small proportion of the variation in automation stability.

the findings therefore suggest that automated regression testing supports greater testing capacity and consistent regression test execution in Superbank's mobile banking application. The statistical difference between the manual and post-automation periods should not be interpreted as direct causal evidence that automation alone caused the increase in test cases, as application complexity, feature growth, and broader testing requirements may also contribute to this development.

Automated regression testing also supports IT risk control by functioning as a preventive internal control mechanism within the application testing and release process. Repeated automated testing can assist in identifying potential regression failures before deployment. However, this study does not directly measure realized IT operational risk through production incidents, system downtime, defect leakage, customer complaints, or financial losses. Therefore, the findings indicate a preventive role in supporting IT risk control rather than direct evidence of reduced operational risk.

From a managerial perspective, automated regression testing can support the organization in managing increasing testing requirements while maintaining consistent execution. Superbank may continue to prioritize automation for business-critical regression scenarios, maintain automation script quality, and use regression testing results as supporting information in pre-deployment quality evaluation. Future research may extend the analysis by including execution time, human effort, cost efficiency, defect leakage, and production incident data.

REFERENCES

Akin, A., Senturk, S., & Garousi, V. (2018). Transitioning from Manual to Automated Software Regression Testing: Experience from the Banking Domain. *Proceedings - Asia-Pacific Software Engineering Conference, APSEC*, *2018-December*, 591–597. https://doi.org/10.1109/APSEC.2018.00074

Alam, M., Priti, T., Fatema, K., & Hasan, S. (2023). Ensuring excellence: A review of software quality assurance and continuous improvement in software product development. *Journal of Software Quality & Continuous Improvement*, *3*(1), 1–15.

Brasil, L., Santos, R. Dos, Ferreira De Souza, É., Endo, A. T., Trubiani, C., Pinciroli, R., & Lankalapalli Vijaykumar, N. (2024). *Performance regression testing initiatives: a systematic mapping*. https://doi.org/10.1016/j.infsof.2024.107641

Finsyen Ilham Erdi, & Astari Retnowardhani. (2024). Optimization of Application Testing in Financial Companies Using Robotic Process Automation (RPA). *Economic Reviews Journal*, *3*(4). https://doi.org/10.56709/mrj.v3i4.595

Kumalasari, R. A. D., Permanasari, K. I., Karismariyanti, M., & Munandar, D. (2022). Mobile Banking: System Quality, Information Quality, Service Quality, Customer Satisfaction, and Loyalty. *Jurnal Administrare: Jurnal Pemikiran Ilmiah Dan Pendidikan Administrasi Perkantoran*, *9*(1), 141–148.

Menegassi, A. A., & Endo, A. T. (2020). Automated tests for cross-platform mobile apps in multiple configurations. *IET Software*, *14*(1), 27–38. https://doi.org/10.1049/iet-sen.2018.5445

Nafisah Balqis, A., Ramadhani, S. N., & Sumual, A. K. (2025). Risiko Operasional dalam Teknologi Informasi: Kajian Literatur Mengenai Mekanisme Identifikasi, Penilaian, dan Dampaknya terhadap Manajemen Risiko. *Scripta Economica: Journal of Economics*, *1*, 67–77.

Persson, C., & Yilmaztürk, N. (2004). *Establishment of Automated Regression Testing at ABB: Industrial Experience Report on "Avoiding the Pitfalls."* https://doi.org/10.1109/ASE.2004.1342729

Pressman, R. S. (2020). *Software engineering: A practitioner's approach* (9th ed.). McGraw-Hill Education.

Safaat, G. I., & Tjhin, V. U. (2024). Analysis of quality assurance performance in the application of manual testing and automation testing for software product testing. *Indonesian Interdisciplinary Journal of Sharia Economics (IIJSE)*, *7*(2), 1987–1996.

Superbank. (2025). Superbank Bukukan Laba Sebelum Pajak Rp122,4 Miliar dan Tembus 1 Juta Transaksi Harian hingga November 2025. In *Superbank*. PT Super Bank Indonesia Tbk. https://www.superbank.id/berita/liputan/superbank-bukukan-laba-sebelum-pajak-rp122-4-miliar-dan-tembus-1-juta-transaksi-harian-hingga-november-2025

Sutapa, F. A. K. P. G., Kusumawardani, S., & Permanasari, A. E. (2019). Automated Test Suite for Regression Testing Based on Serenity Framework: A Case Study. *2019 International Conference of Artificial Intelligence and Information Technology (ICAIIT)*, 138–144. https://doi.org/10.1109/ICAIIT.2019.8834609

Wayan, N., Capriani, W., & Dana, I. M. (2016). *PENGARUH RISIKO KREDIT RISIKO OPERASIONAL DAN RISIKO LIKUIDITAS TERHADAP PROFITABILITAS BPR DI KOTA DENPASAR*. *5*(3), 1486–1512. www.bi.go.id