International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 11 Issue: 09 | Sep 2024 www.irjet.net p-ISSN: 2395-0072

Exploring the Combined Effort Between Software Testing and Quality Assurance: A Review of Current Practices and Future

Kishan Patel[1]

[1]*Independent Researcher, kpatelscholarnetwork@gmail.com*

---***---

Abstract - *The goal of software testing is to ensure that the final product is free of bugs and meets quality standards. Software testing is a series of steps used to identify and fix bugs in software before releasing it to the public. It provides metrics for evaluating the software's efficacy and accuracy. The role of SQAT in software development will also be studied in this paper with focus on understanding why it is an important aspect in producing quality software products. It suggests the means of work on the basis of properties of SQAT such as the process-oriented approach in Software Quality Assurance (SQA) and the product-oriented approach in Software Testing (ST). A discussion of innovative processes like Agile, DevOps, and automation of testing is made, with a focus on the changes made to enhance software quality. There are also existing trends currently witnessed in the market such as shift-left testing, risk-based testing, and AI, ML, and cloud solutions, as well as future trends such as TestOps, improved security testing, and increased UX testing. It thereby highlights why SQAT is requisite in avoiding failure of software, improving customer's satisfaction, and embracing the standard requirements in whatever form even though there are challenges and costs involved in its execution.*

Key Words: *Software Testing, Software Quality Assurance Testing, Future Trends, Current Practices, Levels of Software Testing, Software Quality Assurance*

1. INTRODUCTION

SQA encompasses a wide range of activities related to software product standards, processes, and procedures; it is a methodical approach to assessing software quality [1][2]. Numerous aspects of our everyday lives are affected by software quality, including economics, health, safety, and security. SQA is crucial to the software development process. However, among SQA's many components, software testing (ST) stands head and shoulders above the others [3][4]. Executing software and watching its behaviour or outputs is the fundamental premise behind this product-oriented activity. To ensure the final product meets all quality requirements, ST is an essential part of the software development life cycle (SDLC). To guarantee that the software program meets the client's requirements, quality assurance techniques and software testing work hand in hand [5][6][7]. The software development process isn't complete without SQAT, despite the fact that it's an expensive but crucial job. Economic harm and catastrophic outcomes could result from inadequate or non-existent SQAT efforts [8][9][10]. Table 1 provides the software Quality Assurance VS testing comparison.

Table -1: Quality Assurance VS Testing Comparison

key phase	Quality Assurance	Testing
Purpose	Establish quality standards to prevent difficulties.	Detect and fix quality issues.
Focus	Development processes	Various facets of the product, including its integration, performance, and usefulness.
Who	Everyone from software engineers and QA specialists to business strategists and external stakeholders	QA engineer, software developers
When	During each stage of a product's development	At any point throughout the development process, even during testing
Doing what	Development process improvement, standardization, and guideline creation	Reviewing code, running test, addressing defects

Development of software requires the use of Software Quality Assurance and Testing (SQAT). A program might have disastrous results if SQAT operations are not performed. Completing SQAT tasks is not without its difficulties, however. It includes both process-oriented and product-oriented methods. Software Quality Assurance (SQA) checks that the development processes are better and more consistent, while Software Testing makes sure that the finished product works the way it should. Checking and validating software, finding and preventing bugs, and making quality processes better all the time are the main goals of SQAT [11] [12]. To reach these goals, methods like human testing, automated testing, and static and dynamic analysis are used. Modern methods like Agile and DevOps have also been incorporated into SQAT. Strategies like shift-left testing, continuous testing, and DevSecOps are now used to make sure quality is kept throughout the development lifecycle. It is important to have SQAT because it lowers the chance of software failures, improves customer satisfaction, and makes sure that industry standards are met. All of these things help make sure that reliable, high-quality software products are delivered.

1.1 Contribution of this paper

A main goal of this paper is to give a proper overview of Software Testing and Quality Assurance with its current practices and future trends. The contribution of this paper is given below:

- This paper explains the fundamental distinction between Software Quality Assurance (SQA) and Software Testing (ST), clarifying their respective roles in enhancing software quality and ensuring effective validation.
- This paper outlines the key elements of SQA, including standards, reviews, error analysis, change management, and security, providing a comprehensive framework for maintaining high-quality software throughout the development lifecycle.
- This paper highlights current practices in Software Quality Assurance and Testing (SQAT), such as shift-left testing, automation, and integration with Agile and DevOps, showcasing modern approaches to improving software quality and efficiency.
- This paper identifies both the benefits and challenges of SQAT, offering insights into how it contributes to high-quality software while addressing the costs, time, and complexity involved in implementing effective quality assurance and testing practices.

1.2 Organisation of the paper

The following paper organised as: Section II provides elements of software quality assurance, with its benefits and disadvantages. Sections III and IV provide software testing and its different methods. Sections V, VI, and VII provide the software quality assurance and testing (SQAT) with its current practices and future trends. In section VIII some previous researchers' study is explained through literature review and in last section IX conclusion is given of this paper.

2. FUNDAMENTAL OF SOFTWARE QUALITY ASSURANCE

SQA refers to the process of making sure software is good. The project-specific and properly executed standards, procedures, and processes are ensured by this set of measures [13][14]. Software QA is an activity that occurs in the background while software is being developed. To prevent minor problems from escalating into major ones, it aims to streamline the software development process.

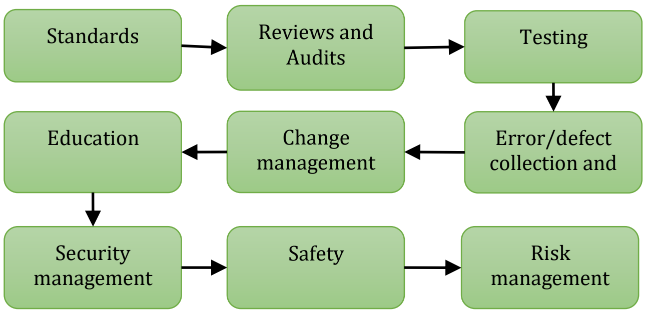

Fig -1: Elements of Software Quality Assurance

2.1 Elements of Software Quality Assurance

Here are some elements of software quality assurance that were explained below shown in Figure 1 [15]:

- **Standards:** Publications and standards developed by organisations like the IEEE and ISO cover a broad spectrum of software engineering. The QA team's job is to check that everyone is following the rules and that the final goods are up to par.
- **Reviews and audits:** One way that software engineers ensure the quality of their colleagues' work is by conducting technical reviews. As a kind of review, SQA experts (those working for an organisation) do audits to ensure that software engineering work is meeting quality requirements[16].
- **Testing:** Error detection is the main objective of software testing, a quality control procedure. Assurance of appropriate planning and effective execution of testing for the main software objective is the responsibility of SQA.
- **Error/defect collection and analysis:** Through the collection and analysis of error and defect data, SQA gains a better understanding of the causes of errors and the software engineering activities that contribute most to their removal.
- **Change management:** SQA checks to see whether the right protocols have been put in place for handling changes.
- **Education:** A common objective throughout all software firms is to improve software engineering procedures. Education has a significant positive impact on the management of software engineers and other stakeholders.
- **Security management:** SQA makes ensuring that the right tools are utilised to make software secure [17].
- **Safety:** It is possible that SQA is responsible for identifying the repercussions of software failure and launching the necessary preventative measures.
- **Risk management:** The SQA organisation oversees the creation of backup plans connected to risks and the proper implementation of risk management protocols.

2.2 Benefits of Software Quality Assurance

Some benefits of SQA are:

- SQA ensures that only excellent software is developed.
- Better apps save time and money.
- SQA contributes to enhancing reliability.
- SQA is beneficial if there are prolonged periods of no maintenance.
- A company's market share grows when its commercial software is of superior quality.
- Improving the procedure for creating software.
- Increase the overall quality of the program.
- It lowers maintenance expenses. Should the launch go off without a hitch from the beginning, your company may go on to the next big thing and leave this behind. A costly, time-consuming, and never-ending cycle of repairs is thrust upon your business with the launching of a product that has ongoing issues.

2.3 Disadvantages of Software Quality Assurance

Some disadvantages of software quality assurance are given below:

- **Cost:** Incorporating additional resources, which results in a larger budget, and enhancing the product with the addition of additional resources are two examples.
- **Time Consuming:** The project is running behind schedule because testing and deployment are taking longer than expected.
- **Overhead:** Due to the need for documenting, reporting, and monitoring of quality measures, SQA processes can add administrative expenses[18]. Particularly for less substantial endeavours, the advantages could fail to outweigh the extra procedure.
- **Resource Intensive:** SQA necessitates personnel who are proficient in quality assurance practices, tools, and testing methodologies.
- **Resistance to Change:** Implementing SQA methods may face resistance from some team members[19], seeing them as superfluous or bureaucratic.
- **Not Foolproof:** In spite of extensive testing and quality assurance procedures, software may still include bugs or security holes.
- **Complexity:** As the number of stakeholders, dependencies, and interaction points increases in a large-scale project, the SQA process might get more complicated.

3. SOFTWARE TESTING

The term "software testing" describes the steps used to run a system or program in search of bugs. Like any other physical process, software takes in data and produces something new in response. Software varies in the way that it malfunctions. The majority of physical systems malfunction in a limited number of predictable ways [13]. But software may break down in all sorts of strange ways. It is often impossible to identify every potential software failure mechanism. Because software is complicated, it might be difficult to find design flaws in the program. Software or digital system correctness cannot be ensured by checking boundary values alone since they are not continuous [20].

Although every potential value must be examined and confirmed, it is not practical to test them all. Even if tests were run at thousands/second, it would take hundreds of years to thoroughly test a simple program that adds only 2 integer inputs of 32 bits (resulting in 2^64 different test scenarios). The software testing iterative approach includes:

- Designing tests
- Executing tests
- Identifying problems.
- Getting problems fixed.

3.1 Levels of Testing

The 4 phases of software testing are shown in Figure 2: acceptance, system, integration, and unit testing [21].

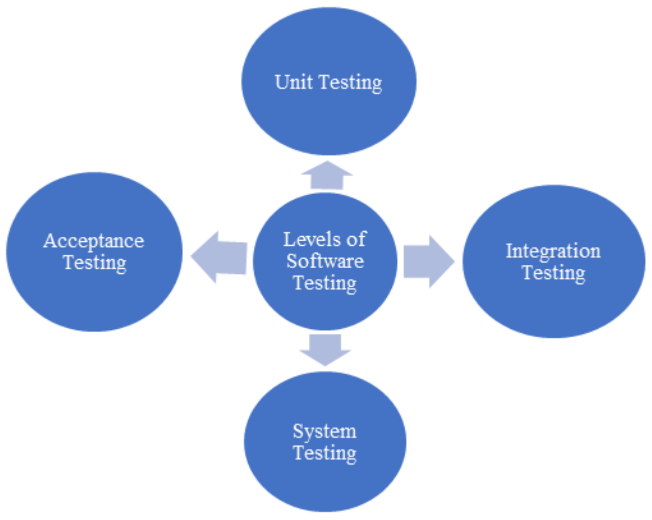

Fig -2: Levels of Testing

3.1.1 Unit Testing/Component Testing/Module Testing

Unit testing is carried out on different system components separately. Additionally, this is utilised to validate whether or not each component is operating in accordance with user needs. Three methods are used to complete this test: unit testing, which is carried out by the program developers, black box testing, and white box testing.

3.1.2 Integration testing

Integration testing involves examining how different parts of the system interact with one another as well as the components and interfaces. The developer also performs this test. Three strategies will be used in this as well: integration testing, bottom-up, and top-down techniques [22].

3.1.3 System Testing

The role of the Tester is to ensure that the program is fully functional by doing thorough tests. There are two main categories of testing in system testing: functional and non-functional. The primary focuses of this testing are the system's security, usability, and external interfaces, as well as its performance.

3.1.4 Acceptance Testing

Software testing culminates in acceptance testing, which occurs after system testing but before the system is made available to end users.

3.2 Testing Methods

There are two ways in which software testing may be carried out after development:

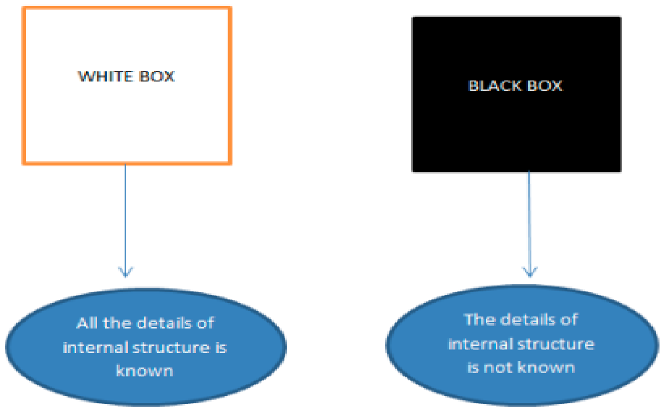
WHITE BOX
BLACK BOX
All the details of internal structure is known
The details of internal structure is not known

Fig -3: Testing methods

3.2.1 White Box Testing:

Figure 3 displays the results of this testing, which is also called structural testing. By this point in the testing process, the program's internal structure and all of its coding details are understood, and the majority of faults will have been caught before the system encounters any problems. In this context, "debugging" refers only to the process of locating software or product defects. There is an element of security testing in this as well.

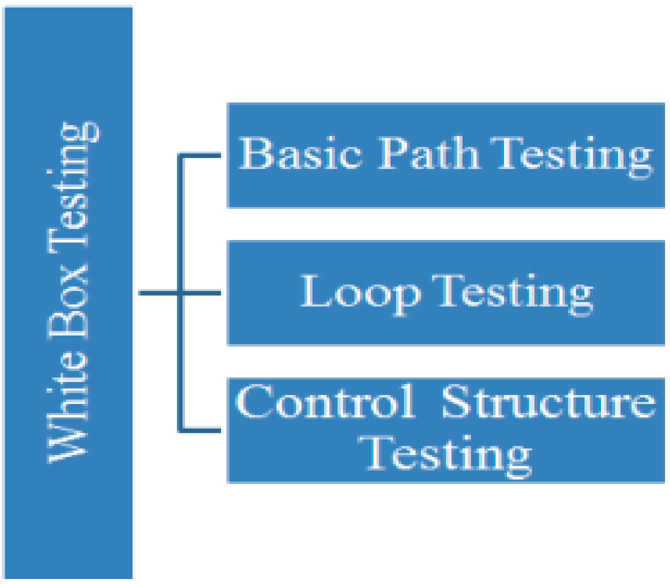

Fig -4: White Box Testing

3.2.2 Black Box Testing:

Figure 4 depicts this kind of testing, which is also called functional testing. This kind of testing conceals all facts from the tester, including the software's or product's fundamental structure and design. In higher-level testing processes like Acceptance and System testing, software testers often do this kind of testing. Additionally, understanding of programming and implementation is not strictly necessary for this examination.

4. CURRENT PRACTICES OF SQAT

Some current practices of the SQA and testing are given below:

- **Shift-Left Testing:** It is the most significant trends in current software testing approach. It involves integrating testing activities early in the software development lifecycle, often alongside the development process. Important parts of this approach include automated testing, TDD, and Continuous Integration (CI), which assist bring testers and developers closer together and find software problems faster [23].
- **Automation in Testing:** It become a cornerstone of modern software testing practices. The use of automated tools to execute tests, compare outcomes, and identify defects has transformed how software is tested. Tools such as Selenium, JUnit, and Jenkins are widely used for functional, regression, and performance testing. The benefits of automation are substantial—it increases test coverage, speeds up the testing process, and allows repetitive tests to be executed without human intervention[19].
- **Agile and DevOps Integration:** The integration of Agile and DevOps methodologies with ST and QA practices is another crucial development. QA is an

inherent aspect of software development in Agile and DevOps settings, rather than a distinct phase. Testers work alongside developers in sprints, and DevOps practices ensure that the software is always in a deployable state.

- **Risk-Based Testing:** It is a strategy that targets the specific area of the software that would likely to cause a problem or develop fault during testing. Thus, based on these areas, one can allocate resources in a way that gives a low probability of critical defects making it to the production phase.

5. FUTURE TRENDS OF SQAT

In this section the future trends are given below of software quality assurance and testing:

- **AI and ML in Testing:** It is set to transform the said field when integrated with Artificial Intelligence and ML. AI and ML are ideal to predict areas that could potentially fail as well as come up with test cases and even carry out complex testing. Applying of AI and ML to testing is a transition to more intelligent and flexible testing processes in regards to increasingly complex software systems.
- **TestOps:** Another one is the so-called application of the DevOps principles with a focus on the testing phase. Testing TestOps is the process of building pipelines for testing to ensure that testing is continuous, integrated, and automated. TestOps will remain a vital component as organisations embrace DevOps strategies because the rate of development will inflate while the quality of the software must rise to the occasion.
- **Cloud-Based Testing:** It is also developing into the future trend. In this way, organisations can scale testing activities, obtain different testing tools and environments and save infrastructure expenses if they use cloud environments. Cross platform and cross configuration testing is possible with cloud based testing resulting in much more comprehensive scenarios.
- **Security Testing Integration:** Security testing is emerging more as a core component of Quality Assurance and mainly driven by the emergence of DevSecOps. Focusing on security, it is essential to guarantee that software will be protected during the entire development process. Implementing security testing within the QA helps in minimising risks, enhancing the software's robustness and guaranteeing that security is put into consideration at the initial design stages rather than being an add-on.
- **Increased Focus on user Experience (UX):** The integration of UX testing into the QA practices promotes both the proper functionality of the software and its UX. This trend was launched based on the understanding that with the customers' satisfaction, software products are very likely to perform very well. In this way organisations derive better customer satisfaction, less churn rates, and enhanced software that users find useful.

6. COMBINED EFFORT INTEGRATION OF SOFTWARE TESTING AND QA

A combination of Software Testing and QA means the cooperation of the process at different phases of the SDLC to incorporate quality into the product under development instead of examining it at the final stage. Key areas of collaboration include:

- **Test-Driven Development (TDD):** The early inclusion of testing through measures such as TDD that entails writing tests before the actual code. This helps in making sure that quality is a key consideration right from the time when the project is being authored.
- **Automated Testing and Continuous Integration (CI):** It is, therefore, certain that through integration with continuous integration practices, automated testing shall ensure all the modified codes conform to set standards. This enables fast feedback to developers and also helps to avoid introducing bugs.
- **Quality Metrics and Reporting:** QA teams set up objectives and measurable indicators called KPIs that facilitate tests. These can be defect density, code coverage and mean time to failure among others.
- **Risk Management:** The primary methods of early risk assessment and control are created by the QA and testing groups. Risk assessment plays a pivotal role for test prioritisation which in turn targets on the areas most likely to affect the quality of the software.
- **User Acceptance and Usability Testing:** QA makes certain that the interfaces and/or features that the users would expect or like are indeed there while testing teams perform all the necessary tests in order to confirm these qualities.
- **Feedback Loops and Continuous Improvement:** QA procedures involve the collection of data on testing results and using it for changing development procedures, testing approaches, and quality requirements. This feedback process is ongoing, and thus promotes a constantly evolving culture.

6.1 Benefits of Combined Effort

- **Enhanced Product Quality:** Ensuring that the QA and testing are incorporated into the project means that quality is brought into the process at every stage thereby arriving at a better final product free of defects.
- **Reduced Time to Market:** Detection of defects at an early stage and constant testing mean that the problems are resolved much quicker, thus minimising the time taken in the last phases of testing.
- **Cost Efficiency:** Minimising defects in the early stages of the SDLC is done by implementing QA

measures, hence avoiding repeated work and bug fixing in the later phases.

- **Customer Satisfaction:** Delivering a high-quality product that meets user expectations ensures higher customer satisfaction and loyalty.

This collaborative approach ensures that quality is not just the responsibility of a single team but is a shared objective across the entire development and delivery process.

7. LITERATURE REVIEW

In this section some previous study on SQAT is given below:

Deshpande et al., (2023) studied that the quality of software is a precise characteristic that identifies the characteristics that every software should have. Quality is the primary determinant of the success or failure of a software project and a software-related company. Software quality has been the subject of several studies. To produce high-quality software, organisations involved in the software industry adhere to requirements set out by Capability Maturity Model Integration. Therefore, in this research, we'll talk about the concepts and quality standards that are applied to software projects in the Bangalore software industry, as well as how these standards are monitored and maintained[24].

Salahat et al., (2023) the quality assurance process guarantees that the project is executed in accordance with the agreed-upon guidelines, concepts, and objectives. A deeper understanding of the complex software analysis process is still the aim of type and calibration research. Techniques for analysis, user hubris, and logistical culture are only a few examples of the problems with current software practices[25].

Gao, Luo and Xie, (2022) as a knowledge-intensive task, ST is an essential part of ensuring software quality. Not much has been done with the testing institution's mountain of historical test data, which includes a wealth of domain knowledge and test expertise, to address these issues. Supporting engineering techniques in military software testing and enhancing organisational knowledge acquisition and sharing are two of the main goals of their investigation of software testing ontologies and knowledge base systems[26].

Lu et al., (2022) Software testing is a crucial tool for ensuring the quality of software that is vital for safety. The standard of software testing has a direct impact on the software that is vital for safety. Building and using a testing quality assessment model is the primary approach to evaluating software testing quality. This paper proposes a methodology for evaluating the quality of safety-critical software testing that takes into account four factors: effectiveness, adequacy, stability, and compliance. Some measure aspects are present in every facet[27].

Zhao, Hu and Gong, (2021) findings indicate that the software sector is playing a significant role in the advancement of the world economy and is a driving force behind the current wave of technical innovations. Software quality is critical for the software industry's steady expansion, and testing is a certain technique to ensure software meets all of that standard. Developers, testers, independent evaluators, and quality assurance/control staff may all benefit from this as it clarifies and implements the applicable criteria for software testing and quality[28].

Nakahara, Monden and Yucel, (2021) the SDLC includes software quality assurance (SQA) procedures that repeatedly test and validate software products to guarantee their quality. In order to objectively show the beneficial impact of additional QA work, particularly in the early stages of software development, they suggest a simulation model of SQA. By simulating several QA tactics in a specific software development setting, the model helps to determine which strategies are most effective in improving software quality while reducing the amount of time and effort spent on QA and bug resolving. The following Table 2 provide the summary of the related work[29].

Table -2: Summary of The Literature Review on Software Testing and Quality Assurance

Ref	Methodology	Area	Limitations & future work
Deshpande Et Al.	Survey of companies in Bangalore adhering to CMMI guidelines	Software companies in Bangalore	Specific details on the methods of evaluation and quantitative results could be expanded. Future work could include a broader geographical scope.
Salahat Et Al., 2023	Analysis of quality management practices and proposal of solutions to testing problems	Case studies of software projects	The paper focuses on theoretical solutions without extensive empirical validation. Future work could involve implementing and testing these solutions.
Gao, Luo, And Xie	A knowledge base system built on ontologies for the purpose of testing military applications utilising semantic web technologies	Historical test data from military software	The approach is domain-specific to military software. Future work could explore adaptation to civilian software industries.
Lu Et Al.	Construction of a quality evaluation model for safety-critical software using Analytic Hierarchy Process and fuzzy comprehensive evaluation	Practical safety-critical software projects	The model's applicability beyond safety-critical software is not explored. Future work could include validation in diverse software domains.
Zhao, Hu, And Gong	Analysis of international standardisation in software quality	Global software industry standards	The study is broad and may lack depth in specific standardisation issues. Future work could involve more

	and testing		detailed case studies or industry-specific standards.
Nakahara, Monden, And Yucel	Simulation model of software quality assurance to demonstrate the impact of QA effort during development phases	Simulated software development projects	The model is based on simulations rather than real-world data. Future work could validate the model with actual development projects.

8. CONCLUSION AND FUTURE WORK

Software testing's ultimate goal is enhancing the quality of the developed software. The primary objective of testing is to identify issues such as errors, defects, and issues in a system or newly developed software. It is also clear from the above discussions that testing has an important place in software quality control (SQC). This examines the importance of SQAT in software development emphasising its importance in enhancing the reliability of software products. It defines the major aspects of SQAT such as process focus in SQA and product emphasis in ST. Various Agile, DevOps, and automation in testing practices are defined, and their significance in enhancing the availability of quality software is mentioned. The modern trends such as shift-left testing, risk-based testing, AI, ML, cloud-based solutions, and Test Ops that are on the horizon, security testing and UX Testing are also discussed. The paper thus concludes that SQAT is necessary to avoid software failures, increase customer satisfaction, and meet regulatory requirements even though it comes with difficulties and high costs. We want to expand the design pattern's primary use case in the future by investigating its potential use in unit testing and similar contexts. Furthermore, we would want to track certain metrics among a group of test automation experts in order to demonstrate the effectiveness of a framework using this design pattern. This can only be accomplished by giving careful thought to the most crucial details, such the amount of time spent on maintenance, the reusability of methods, and the amount of time required to execute a test case.

REFERENCES

[1] A. Bhanushali, "Ensuring Software Quality Through Effective Quality Assurance Testing: Best Practices and Case Studies," *Int. J. Adv. Sci. Res. Eng.*, 2023.

[2] A. Bhanushali, "Challenges and Solutions in Implementing Continuous Integration and Continuous Testing for Agile Quality Assurance," *Int. J. Sci. Res.*, 2023, doi: 10.21275/sr231021114758.

[3] S. Najihi, S. Elhadi, R. A. Abdelouahid, and A. Marzak, "Software Testing from an Agile and Traditional view," in *Procedia Computer Science*, 2022. doi: 10.1016/j.procs.2022.07.116.

[4] T. K. Abdel-Hamid, "The economics of software quality assurance: A simulation-based case study," *MIS Q. Manag. Inf. Syst.*, 1988, doi: 10.2307/249206.

[5] J. Možucha and B. Rossi, "Is mutation testing ready to be adopted industry-wide?," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2016. doi: 10.1007/978-3-319-49094-6_14.

[6] B. Rossi, B. Russo, and G. Succi, "Modelling failures occurrences of Open Source Software with Reliability Growth," in *IFIP Advances in Information and Communication Technology*, 2010. doi: 10.1007/978-3-642-13244-5_21.

[7] S. Mann, A. Balyan, V. Rohilla, D. Gupta, Z. Gupta, and A. W. Rahmani, "Artificial Intelligence-based Blockchain Technology for Skin Cancer Investigation Complemented with Dietary Assessment and Recommendation using Correlation Analysis in Elder Individuals," *Journal of Food Quality*. 2022. doi: 10.1155/2022/3958596.

[8] S. M., M. Shamsur, A. Z., and M. Hasibul, "A Survey of Software Quality Assurance and Testing Practices and Challenges in Bangladesh," *Int. J. Comput. Appl.*, vol. 180, no. 39, pp. 1–8, 2018, doi: 10.5120/ijca2018917063.

[9] V. Rohilla, D. S. Chakraborty, and D. R. Kumar, "Random Forest with Harmony Search Optimization for Location Based Advertising," *Int. J. Innov. Technol. Explor. Eng.*, vol. 8, no. 9, pp. 1–6, 2019, doi: 10.35940/ijitee.i7761.078919.

[10] V. Rohilla, M. S. S. Kumar, S. Chakraborty, and M. S. Singh, "Data Clustering using Bisecting K-Means," in *Proceedings - 2019 International Conference on Computing, Communication, and Intelligent Systems, ICCCIS 2019*, 2019. doi: 10.1109/ICCCIS48478.2019.8974537.

[11] A. Rath, A. Das Gupta, V. Rohilla, A. Balyan, and S. Mann, "Intelligent Smart Waste Management Using Regression Analysis: An Empirical Study," in *Communications in Computer and Information Science*, 2022. doi: 10.1007/978-3-031-07012-9_12.

[12] K. Patel, "Quality Assurance In The Age Of Data Analytics: Innovations And Challenges," *Int. J. Creat. Res. Thoughts*, vol. 9, no. 12, pp. f573–f578, 2021.

[13] V. Rohilla, S. Chakraborty, and M. Kaur, "An Empirical Framework for Recommendation-based Location

Services Using Deep Learning," *Eng. Technol. Appl. Sci. Res.*, 2022, doi: 10.48084/etasr.5126.

[14] Ritesh Tandon, Aniqa Sayed, and Muhammad Anas Hashmi, "Face mask detection model based on deep CNN technique using AWS," *Ijera*, vol. 13, no. 5, pp. 12–19, 2023, doi: 10.9790/9622-13051219.

[15] M. S. Hossain, "Challenges of Software Quality Assurance and Testing," *Int. J. Softw. Eng. Comput. Syst.*, vol. 4, no. 1, pp. 133–144, 2018, doi: 10.15282/ijsecs.4.1.2018.11.0044.

[16] V. Rohilla, S. Chakraborty, and R. Kumar, "Deep learning based feature extraction and a bidirectional hybrid optimized model for location based advertising," *Multimed. Tools Appl.*, 2022, doi: 10.1007/s11042-022-12457-3.

[17] S. Mathur and S. Gupta, "Classification and Detection of Automated Facial Mask to COVID-19 based on Deep CNN Model," in *2023 IEEE 7th Conference on Information and Communication Technology, CICT 2023*, 2023. doi: 10.1109/CICT59886.2023.10455699.

[18] J. Thomas, "The Effect and Challenges of the Internet of Things (IoT) on the Management of Supply Chains," *Int. J. Res. Anal. Rev.*, vol. 8, no. 3, pp. 874–878, 2021.

[19] J. Thomas, "Enhancing Supply Chain Resilience Through Cloud-Based SCM and Advanced Machine Learning: A Case Study of Logistics," *J. Emerg. Technol. Innov. Res.*, vol. 8, no. 9, 2021.

[20] M. Tuteja and G. Dubey, "A Research Study on importance of Testing and Quality Assurance in Software Development Life Cycle (SDLC) Models," *Int. J. Soft Comput. Eng.*, no. 2, pp. 2231–2307, 2012.

[21] M. A. Jamil, M. Arif, N. S. A. Abubakar, and A. Ahmad, "Software testing techniques: A literature review," in *Proceedings - 6th International Conference on Information and Communication Technology for the Muslim World, ICT4M 2016*, 2017. doi: 10.1109/ICT4M.2016.40.

[22] R. Dwivedi and V. Rohilla, "Empowering agile method feature-driven development by extending it in RUP shell," in *Advances in Intelligent Systems and Computing*, 2017. doi: 10.1007/978-981-10-3770-2_69.

[23] S. Mathur and S. Gupta, "An Energy-Efficient Cluster-Based Routing Protocol Techniques for Extending the Lifetime of Wireless Sensor Network," in *2023 International Conference on the Confluence of Advancements in Robotics, Vision and Interdisciplinary Technology Management, IC-RVITM 2023*, 2023. doi: 10.1109/IC-RVITM60032.2023.10434975.

[24] M. V. Deshpande, P. A. Soitkar, D. R. Tripathi, and S. B. Agarmore, "Ensuring Web Application Quality: The Role of Software Testing as a Form of Quality Assurance," in *2023 IEEE International Conference on ICT in Business Industry & Government (ICTBIG)*, 2023, pp. 1–6. doi: 10.1109/ICTBIG59752.2023.10456277.

[25] M. Salahat *et al.*, "Software Testing Issues Improvement in Quality Assurance," in *2nd International Conference on Business Analytics for Technology and Security, ICBATS 2023*, 2023. doi: 10.1109/ICBATS57792.2023.10111145.

[26] C. Gao, W. Luo, and F. Xie, "An Ontology-based Knowledge Base System for Military Software Testing," in *Proceedings - 2022 9th International Conference on Dependable Systems and Their Applications, DSA 2022*, 2022. doi: 10.1109/DSA56465.2022.00043.

[27] Y. Lu, C. Li, S. Wang, Y. Liu, and J. Dai, "A Quality Evaluation Method for Software Testing about Safety-Critical Software," in *Proceedings of the IEEE International Conference on Software Engineering and Service Sciences, ICSESS*, 2022. doi: 10.1109/ICSESS54813.2022.9930258.

[28] Y. Zhao, Y. Hu, and J. Gong, "Research on International Standardization of Software Quality and Software Testing," in *Proceedings - 2021 IEEE/ACIS 21st International Fall Conference on Computer and Information Science, ICIS 2021-Fall*, 2021. doi: 10.1109/ICISFall51598.2021.9627426.

[29] H. Nakahara, A. Monden, and Z. Yucel, "A Simulation Model of Software Quality Assurance in the Software Lifecycle," in *Proceedings - 22nd IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, SNPD 2021-Fall*, 2021. doi: 10.1109/SNPD51163.2021.9704927.