A 30-60-90-Day Plan for QA Leaders

Implement software test automation

eBook

KEYSIGHT

Introduction

Quality assurance (QA) in software testing is pivotal in today's world, where the demand for rapid and reliable software solutions is ever-increasing. New software releases, updates, and features launch daily, and end-users expect these offerings to be bug-free, user-friendly, and dependable. In the competitive landscape of the software industry, testing is not just a phase; it is a mission-critical function.

The role of a software tester has evolved over the years. Testers are no longer solely responsible for manual testing; they are now instrumental in championing automation, which is the key to accelerating testing processes, increasing test coverage, and ensuring high-quality software.

This eBook presents a comprehensive 30-60-90-day automation plan for QA leaders looking to champion an automation strategy for their teams. The following chapters outline the steps a QA manager should take to successfully implement automation in today's software development landscape.

Contents

CHAPTER 1

The Role of a QA Leader in Today's Software Landscape

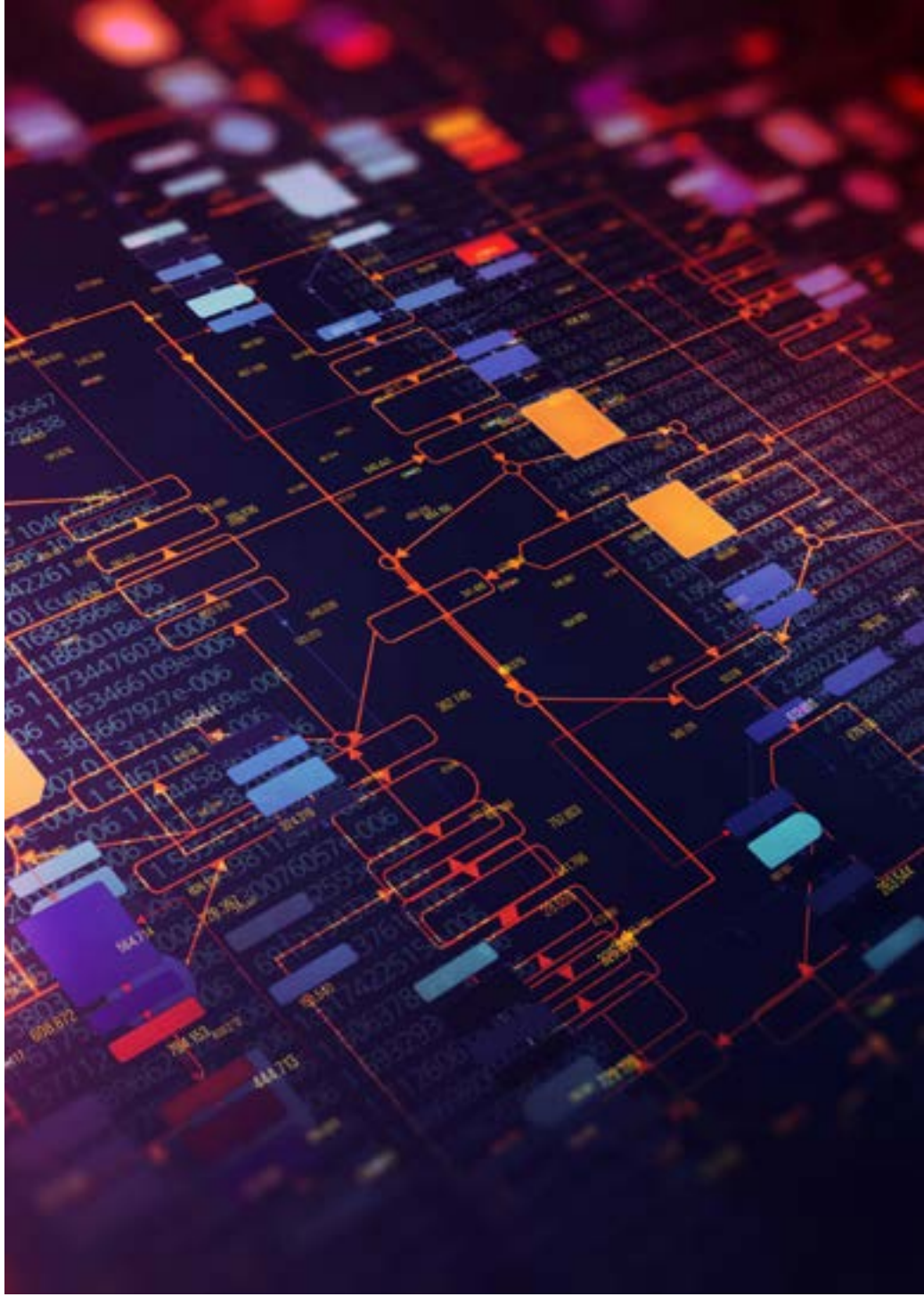
CHAPTER 1

The Role of a QA Leader in Today's Software Landscape

Before we dive into the intricacies of a 30-60-90-day automation plan, it is necessary to underscore the pivotal role of a QA leader.

With the rise of agile methodologies, continuous integration, and DevOps practices, the demand for rapid and high-quality software development has reached unprecedented levels. As such, QA leaders are responsible for maintaining traditional testing practices and spearheading automation initiatives that align with the agile development philosophy.

QA leaders ensure software meets its functional and non-functional requirements while keeping pace with rapid development cycles. They must strike a balance between thorough testing and the need for quick releases. This dynamic environment is ideal for automation, which offers a solution to the challenges of speed and quality.



QA leaders are today's automation champions

Automation is the cornerstone of efficient testing in modern software development. Manual testing alone is insufficient to meet the demands of agile development and continuous delivery. QA teams can significantly expedite testing by automating repetitive and time-consuming tasks. Automation lends itself to improved test coverage as it executes many test cases in a fraction of the time it would take manually.

In addition, automation ensures consistent testing, reducing the potential for human error. It allows for continuous testing throughout the software development life cycle (SDLC), making it an integral part of DevOps pipelines. Integrating new code at the same time as automated tests deploy enables you to offer immediate feedback to development teams. By doing so, you can catch defects early, reducing the cost and effort required to fix them.

Of course, while automation enhances efficiency, it does not replace the critical role of human QA professionals. Testers' unique ability to think creatively, design complex test scenarios, and apply domain knowledge ensures that QA professionals will always play a key role in maintaining the overall quality and user satisfaction of software products. Moreover, the use of automation empowers testers to focus on higher-value tasks, enabling them to engage in strategic test planning, exploratory testing, and continuous improvement initiatives.

The Automation Advantage

Automation in software testing is not just a trend but a necessity. In a world where software development is moving at breakneck speed, automation is the key to keeping up with the pace while maintaining the highest quality standards. Here are some reasons why automation is so important for software testing:

1. **Speed and efficiency:** Automated tests can execute quickly and repeatedly, enabling rapid feedback to developers. Speed is essential in agile and continuous integration / continuous deployment (CI / CD) environments.
2. **Consistency:** Automation ensures consistent test performance, reducing the risk of human error and providing reliable results.
3. **Reusability:** Automation enables the reuse of automated test scripts for regression testing, reducing the effort required for each new release.
4. **Improved test coverage:** Automation allows you to cover many test scenarios, including edge cases and large data sets, which might be impractical to test manually.
5. **Early detection of defects:** Automated tests can identify defects early in development, making them cheaper and easier to fix.
6. **Resource savings:** Automated testing can reduce the time and effort required for testing, enabling QA teams to focus on more complex and exploratory testing tasks.
7. **Parallel testing:** Automation enables the execution of tests in parallel across different environments and configurations, improving efficiency.
8. **Continuous testing:** Automation seamlessly integrates with CI / CD pipelines, ensuring that tests run automatically with every code change.

As a QA manager, understanding these benefits and advocating for automation within your organization is crucial to your role. Now, let's delve into a 30-60-90-day automation plan to help you successfully implement automation in your software testing processes.

CHAPTER 2

Implementing a 90-day Plan for Automation Success

CHAPTER 2

Implementing a 90-day Plan for Automation Success

The First 30 Days

Day 1 – 10: Assess the current state of testing

Take the time upfront to understand your existing test cases so you can identify which are the best candidates for automation. Prioritize test cases based on their importance and frequency of execution.

Tasks:

- √ List the operating systems and device families you need to test. Depending on the application and your user base, this can determine what types of tests you want to automate.
- √ Identify the tests most critical to your application and those that require the most manual effort. For example, highly complex tests such as visual verification can be the most time-consuming.
- √ Examine which tests are most sensitive to risk. Any manually performed tasks are vulnerable to human error. Pinpoint the tests that are so critical that even a minor defect could result in substantial business loss.

Day 11 – 20: Define your automation strategy

Decide what percentage of your test cases you want to automate. They will not all be candidates. Many organizations start in the 20 - 30% range for functional testing. The percentage will increase as teams become more comfortable with automation tools and processes.

Tasks:

- √ Consider your software environment. Agile and DevOps environments often aim for 70 to 85% of functional test automation. More high-risk environments will aim for 40 to 60% of functional test automation.
- √ Define specific automation objectives, such as reducing testing cycle time, improving test coverage, and enhancing test reliability. Each objective should contribute to improving overall operational efficiency.
- √ Determine which types of tests you want to automate first — for example, regression, smoke, new feature / exploratory testing, and functional testing. Many organizations start with regression testing because that is where they spend the most time.

Day 21 – 30: Select the right automation tool or tools

Choosing the right software test automation tool (or tools) is critical. The selection should align with your applications' technology stack and your team's skills.

Tasks:

- √ Determine what features are essential to the organization. For example, if security is a top concern, try to find a non-invasive testing tool that does not require access to the source code.
- √ Consider tools that integrate your existing testing environment, development frameworks, and CI / CD pipelines.
- √ Revisit your automation strategy and develop a shortlist of tools that align with the priorities of your specific application and business needs. For example, if automating user experience testing is crucial, seek tools specializing in visual verification testing.

Resource: Buyer's Guide to Evaluating Test Automation Solutions | Keysight

The First 60 Days

Day 31 – 40: Establish a budget and build a business case

Implementing automation requires financial resources, but securing those resources is easier said than done. The next step is determining the budget for automation initiatives and building a business case to get management buy-in.

Tasks:

- √ Quantify the return on investment (ROI) for deploying a test automation solution, such as cost savings or cost avoidance due to fewer software defects and decreased re-testing due to re-releases and bug fixes.
- √ Identify the ROI value to the business, such as improved software quality, better user experiences, faster release cycles, improved productivity, and happier customers. Develop models that translate increased team productivity into cost savings.
- √ Secure budget for automation by taking your ROI pitch to your management team, department lead, or even a C-Suite executive, depending on your organization's size and scope.

Resource: How to Sell the Value of Test Automation to the C-Suite | Keysight

Day 41 – 50: Build your test automation framework

An automation framework is essential for maintaining a structured, scalable, and maintainable automation process. Invest time in building a solid foundation.

Tasks:

- √ Choose a suitable automation framework that aligns with your application's architecture and technology stack. Frameworks like Data-Driven, Keyword-Driven, or Behavior-Driven Development provide structures for organizing and executing automated tests.
- √ Establish scripting and coding standards to ensure consistency. These standards include naming conventions, code structure, and guidelines for creating reusable components within the framework.
- √ Emphasize maintainability and scalability. Create modular and easily maintainable automation scripts that will accommodate changes in the application over time. Consider the ability to scale automation efforts as the application grows.

Day 51 – 60: Provide training and skill development

Automation is only as effective as your team's expertise. Invest in training and skill development programs to ensure your team is prepared for automation.

Tasks:

- √ Arrange training sessions for your team on the selected automation tools.
- √ Consider a low code / no code test automation solution if coding experience is a concern.
- √ Encourage team members to obtain relevant certifications or attend workshops like the Learn courses from Keysight.

The First 90 Days

Day 61 – 70: Automate build and deployment processes

Automating build, test, and deployment processes streamlines the software delivery pipeline, enabling shorter release cycles. Automation enables organizations to release new features, enhancements, or bug fixes more frequently.

Tasks:

- √ Use a version control system like Git to manage and track changes to the source code. Having a centralized repository enables collaboration among developers.
- √ Set up an automated build process to compile the code, run unit tests, and generate executable artifacts. Tools like Jenkins, Travis CI, or GitLab CI can automate this process.
- √ Implement containerization with technologies like Docker and orchestration with Kubernetes to ensure consistent and scalable deployment across various environments.

Day 71 – 80: Integrate continuous testing

To ensure your automated tests are a seamless part of your development process, integrate them into your CI / CD pipeline. An integration accelerates the delivery of high-quality software, enhances collaboration, reduces manual effort, and promotes a culture of continuous improvement within the development and operations teams.

Tasks:

- √ Develop a suite of automated tests, including unit, integration, and end-to-end tests. Integrate these tests into the CI / CD pipeline to ensure they run automatically with each code commit.
- √ Optimize test execution time by running tests in parallel to reduce the overall testing time, providing faster feedback to the development team.
- √ Automate the deployment process to move code seamlessly from development to staging and production environments, which reduces the risk of human error and ensures consistency in deployment.

Day 81 – 90: Monitor progress and report metrics

As your automation efforts progress, it is essential to implement processes to track and report on the effectiveness of your automation initiatives. Regularly review and adapt these metrics to ensure they align with the evolving needs of your development and testing processes.

Tasks:

- √ Define key performance indicators (KPIs) like test execution time, test coverage, deployment success rate, defect identification rate, and overall improvement in release quality.
- √ Use the feedback obtained from monitoring to inform decisions and address issues promptly. For example, addressing low test coverage can reduce the likelihood of defects going into production.
- √ Track application performance and user experience by gathering user feedback to measure their satisfaction with the application.

Resource: Maximize Software Test Coverage with Advanced Reporting | Keysight Blogs

CHAPTER 3

Day 91 and Beyond — Scaling and Optimizing Automation

CHAPTER 3

Day 91 and Beyond — Scaling and Optimizing Automation

Promote a testing culture

A testing culture goes beyond just performing tests; it encourages everyone involved in the software development process to take ownership of quality. During this phase, gather feedback from your team and stakeholders to refine and enhance your automation strategy. Document the knowledge and experience gained in your automation journey and share insights within your team to ensure continuity and scalability.

Ideas:

- √ Engage with developers, product managers, and other stakeholders to understand their expectations and areas for improvement.
- √ Promote collaboration between the QA team, developers, and other departments to ensure a shared responsibility for quality.
- √ Create a knowledge repository or share documents through a centralized knowledge-sharing platform.

Work towards continuous improvement

Continuous improvement ensures that your automation efforts remain effective and efficient. For best results, regularly review your automation processes and frameworks to identify areas for improvement. Effective communication and collaboration between teams are essential for successful automation. Strengthen these connections as you conduct regular reviews of automation processes.

Ideas:

- √ Conduct periodic code reviews for your automation scripts and framework.
- √ Attend regular meetings or forums for QA professionals to discuss automation challenges and successes.
- √ Recognize and celebrate achievements in testing and automation.

Review and adjust your strategy

Regularly revisit your automation strategy and adjust it based on evolving project requirements, technology changes, and lessons learned. Foster a culture of collaboration between the QA and development teams to resolve issues efficiently.

Ideas:

- √ Review your initial automation strategy and update it to reflect the changing needs of your organization.
- √ Be open to adjusting priorities and resource allocation based on project demands.
- √ Seek feedback from your team and stakeholders to ensure your automation strategy aligns with organizational goals.

Inspire innovation

The final step is to inspire innovation within your QA team and organization. Take opportunities to create a culture of experimentation and learning. Keep an eye on emerging automation trends and technologies and assess their applicability to your organization.

Ideas:

- √ Encourage your QA team to experiment with new tools, techniques, and methodologies.
- √ Allocate time for team members to work on personal or team projects that explore new testing approaches, tools, or ideas.
- √ Organize regular knowledge-sharing sessions where team members can present innovative ideas, discuss new technologies, or share their experiences.

CHAPTER 4

Adopting Today's Automation Trends

CHAPTER 4

Adopting Today's Automation Trends

As QA leaders, it is critical to be aware of the latest trends and technologies in automation that are shaping the software testing landscape. By staying informed and adopting these trends, you can lead your team to a more efficient and productive future in software testing.

Artificial intelligence in testing

Artificial intelligence (AI) and machine learning (ML) are making significant inroads in software testing. AI-driven testing tools can automate test case generation, predict defect-prone areas, and assist in test data creation. ML algorithms can adapt to changing test scenarios, improving test efficiency and accuracy.

The use of AI / ML has led to a monumental shift in the role of testing in the software development life cycle. It has empowered testers with a more accurate, efficient, and flexible solution to the ever-changing requirements of modern software development. QA leaders should explore these technologies to augment their automation efforts.

Continuous testing

Continuous testing aligns closely with DevOps and CI / CD pipelines, which involves automated testing at every stage of the software development process, ensuring rapid and frequent feedback. Embracing continuous testing helps catch defects early, reducing the cost and effort of fixing issues and accelerating the release of high-quality software.

Automation software for continuous testing should encompass a combination of functional and non-functional testing as part of the continuous testing process. In addition, the results from test sequences require integration into the CI / CD framework for analysis and reporting.

Any platform testing

Users expect seamless access to apps and websites across multiple devices and platforms. Meeting their expectations requires continuous maintenance of test cases, scripts, and snippet inventory for implementations across many platforms.

With the proliferation of mobile applications and the importance of cross-browser compatibility, QA teams need to embrace testing across any platform or device. Automation tools are evolving to support testing on various browsers, operating systems, and devices, and they help teams ensure their applications work seamlessly across different platforms.

Performance engineering

Performance testing is no longer a standalone phase but is evolving into performance engineering. A performance engineering approach involves an ongoing and integrated approach to maintaining and improving software quality throughout the life cycle rather than addressing quality as a one-time event.

The performance engineering approach advocates for comprehensive testing, where performance requirements are central to shaping product design, development, and delivery within a continuous quality strategy. QA leaders should understand the importance of performance engineering and adopt relevant tools and practices.

Non-invasive testing

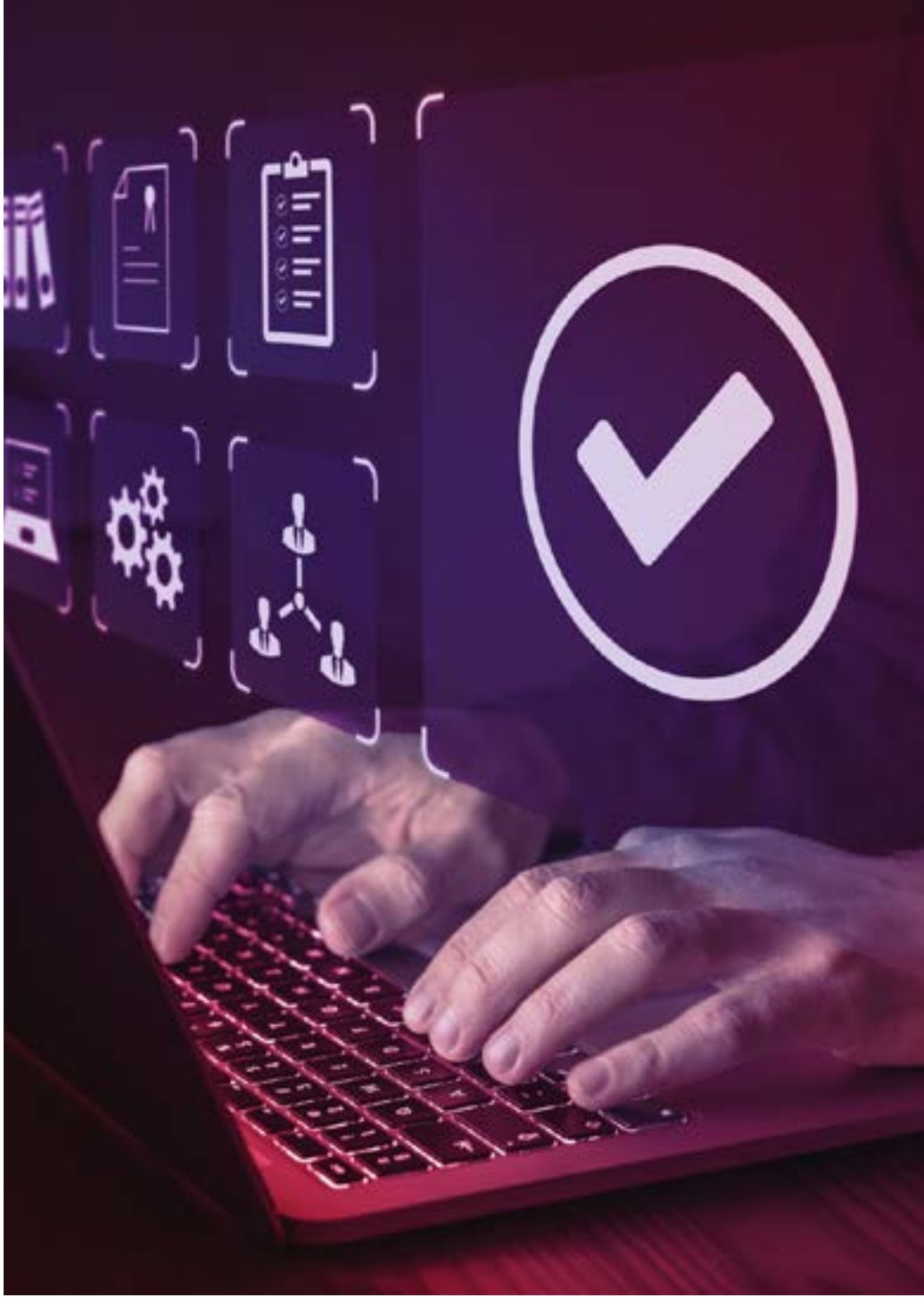
Testing mission-critical systems in a DevOps environment is uniquely challenging due to heightened security and strict regulations. Non-invasive software testing ensures that software gets a rigorous external evaluation to address user needs, security concerns, and integration requirements without relying on detailed knowledge of the software's internal architecture.

Testing non-invasively means evaluating the external behavior of a system, enabling testers to assess its functionality without having explicit knowledge of the internal workings. The non-invasive testing approach is particularly valuable for validating user experience and ensuring the software meets specified requirements.

Model-based testing

In today's software development landscape, where Agile and rapid development methodologies are prevalent, model-based testing provides a flexible and adaptable approach. Model-based testing enables testers to create abstract representations of the software's behavior, functionality, and requirements. The models serve as a basis for generating test cases automatically.

By using models to generate test cases, model-based testing helps ensure comprehensive coverage of the software's functionality. Testers can capture various scenarios and edge cases within the models, addressing different paths and conditions that might be challenging to cover thoroughly through manual testing. Model-based testing contributes to a more robust and reliable testing process.



CHAPTER 5

The Ongoing Journey of Automation Implementation

CHAPTER 5

The Ongoing Journey of Automation Implementation

Implementing automation is not a one-time endeavor but an ongoing journey. After the initial 30-60-90-day plan, continuously improving, adapting, and expanding your automation efforts is essential. Here are some key considerations to keep in mind as you embark on this journey:

Collaboration and communication

Maintain effective communication and collaboration between QA teams, development teams, and other stakeholders. It is paramount to ensure that everyone aligns on automation goals and priorities. Encourage open dialogue and feedback to refine your automation strategy.

Test maintenance

Update automation scripts and frameworks through regular maintenance. As your application evolves, tests may need updates to remain relevant. Establish a process for identifying and addressing maintenance needs to keep your automated tests running smoothly.

Test data management

Implement strategies to manage and provision test data efficiently, ensuring your automated tests can access the correct data when needed. Effective test data management is critical for automation success.

Scalability and parallel testing

Consider the scalability of your test infrastructure as your automation coverage expands. Parallel testing, the ability to run multiple tests concurrently, can significantly reduce execution time. Explore options for scaling your testing infrastructure as needed.

Performance optimization

Optimize your automation scripts and frameworks for performance continuously. Efficient tests run faster and provide quicker feedback. Regularly review and refine your automation code to ensure it is efficient and effective.

Training and skill development

Invest in ongoing training and skill development for your team. Automation tools and technologies evolve, and team members must stay updated to remain effective in their roles. Encourage team members to obtain certifications and pursue continuous learning opportunities.

Integration with new technologies

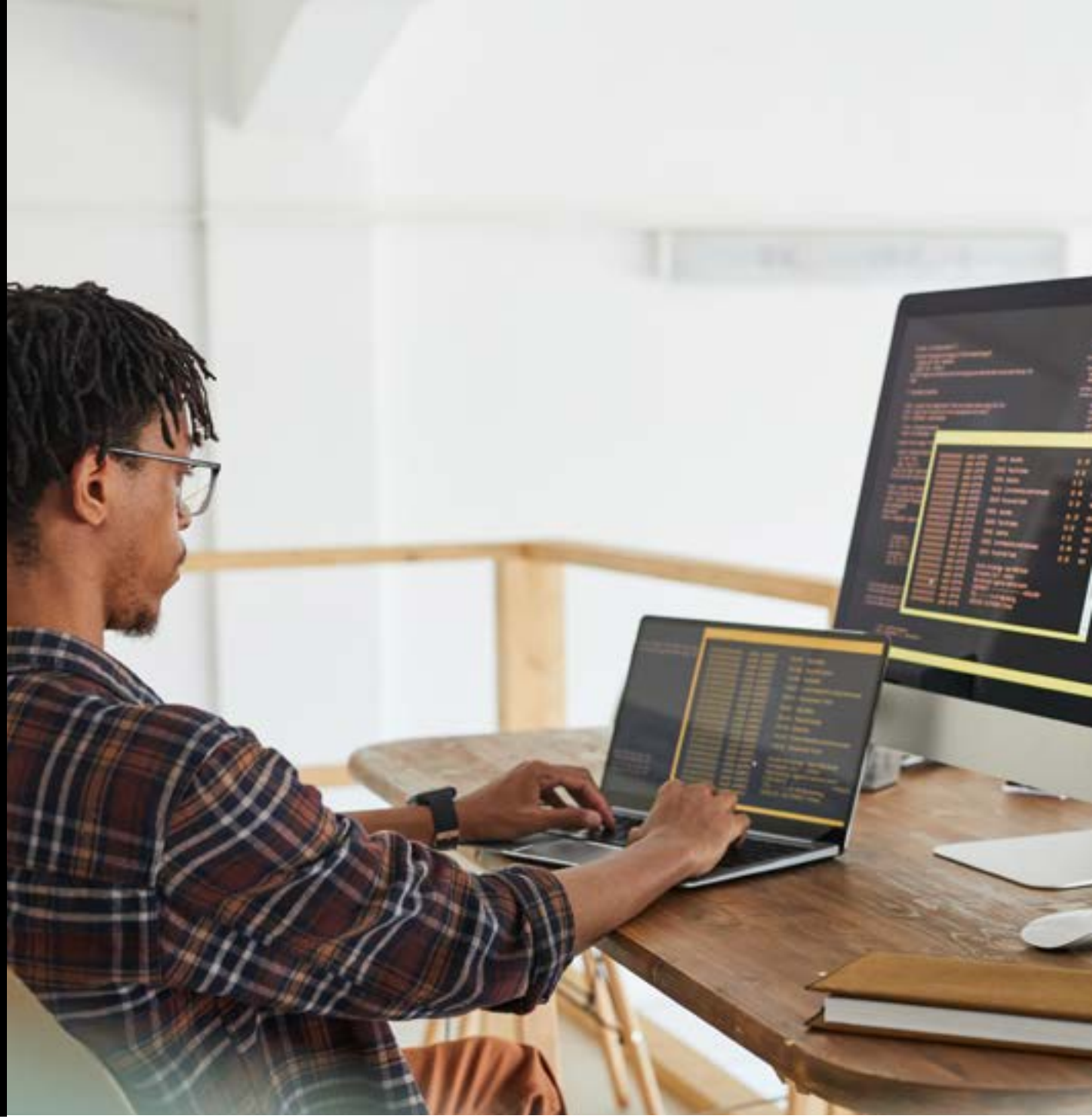
Stay current with emerging technologies and tools that can enhance your automation efforts. Embrace new technologies and integrate them into your automation strategy as they become relevant.

Continuous evaluation and improvement

Evaluate the effectiveness of your automation efforts continuously. Use metrics and KPIs to measure the impact of automation on test coverage, defect identification, and overall product quality. Gather feedback from your team and stakeholders to identify areas for improvement.

CHAPTER 6

Embracing the Future of Automation



CHAPTER 6

Embracing the Future of Automation

In the ever-evolving world of software development, automation has become the linchpin for success. QA leaders in software testing must recognize the critical role automation plays in accelerating testing processes, enhancing test coverage, and ultimately improving the quality of software products.

Automation is a powerful tool representing a strategic shift in how companies develop, test, and deliver software. By efficiently implementing automation in the first 30-60-90 days of their tenure, QA leaders can set the stage for a culture of continuous improvement and excellence in testing.

The importance of automation extends beyond the initial 90 days of your plan; it is an ongoing journey. QA leaders must remain agile, adaptable, and informed about the latest trends and technologies shaping the testing landscape. Collaboration, communication, and a commitment to learning are essential to this journey.

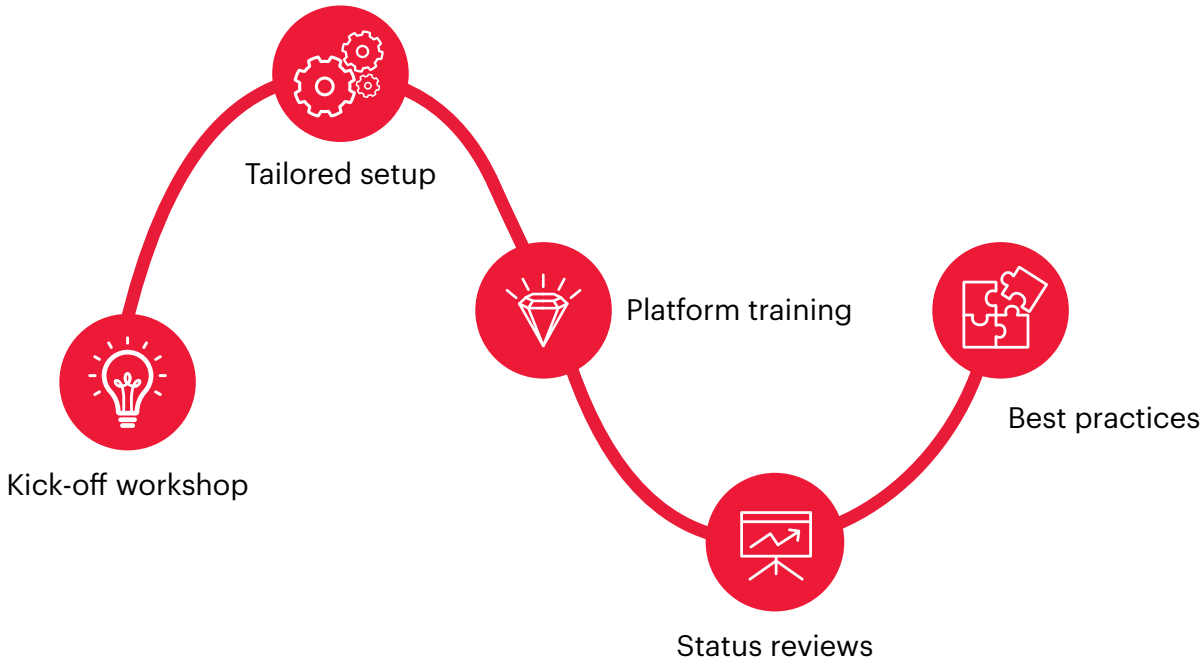
In conclusion, automation is not just a means to an end but the future of software testing. Embracing automation enables QA leaders to deliver high-quality software products quickly and efficiently, ensuring they meet the ever-increasing demands of today's high-tech world. By adhering to the principles and steps outlined in this automation plan, QA leaders can navigate the complex software testing landscape and steer their teams toward a future filled with possibilities.

Get Started in Just 90 Days

At Keysight, we have assisted numerous organizations in achieving their automation goals. We understand the unique challenges of each industry and can tailor solutions to meet your specific needs.

Keysight offers a low-risk approach to prove that automation is right for your organization. For just $10K in 90 days, we will connect our experts with members of your team and build the best automation integration plan for your organization.

Quick start today



Keysight enables innovators to push the boundaries of engineering by quickly solving design, emulation, and test challenges to create the best product experiences. Start your innovation journey at www.keysight.com.