CitiusTech
Shaping Healthcare Possibilities

WHITEPAPER

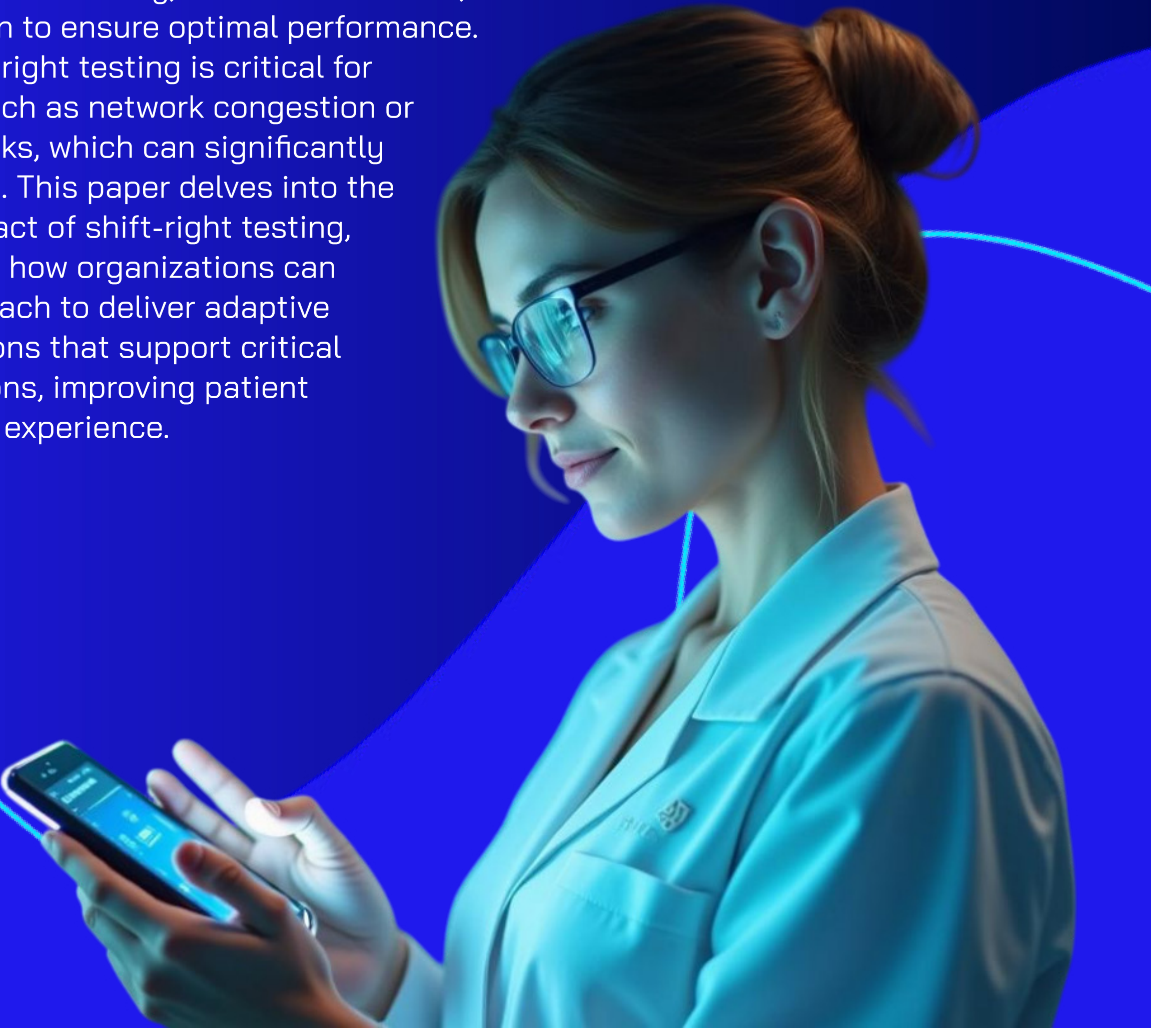
Revolutionizing healthcare: Discover how Shift-right testing ensures quality and real-time solutions

Sandeep Singh
Sr. Tech Specialist, Performance QA Architect, CitiusTech

Jeevan Jadhav
Sr. Tech Specialist, Performance Testing Practice Lead, CitiusTech

Abstract

Traditional testing methods often fall short in addressing key challenges, such as ensuring service availability, maintaining responsiveness, and achieving regulatory compliance, leading to potential risks in patient care and data security. Shift-right testing offers a robust solution by focusing on testing in live production environments post-deployment. This approach enables continuous monitoring, real-time feedback, and real-world validation to ensure optimal performance.
In healthcare, shift-right testing is critical for detecting issues such as network congestion or database bottlenecks, which can significantly impact patient care. This paper delves into the transformative impact of shift-right testing, offering insights on how organizations can embrace this approach to deliver adaptive and resilient solutions that support critical healthcare operations, improving patient outcomes and user experience.



Content

Introduction

"Test smarter, not harder. Bringing the power of proactive quality assurance to production."

In today's fast-paced world, the demand for high-quality, secure, and efficient software has never been greater. This is especially true in industries like healthcare, where software solutions directly impact patient outcomes, regulatory compliance, and overall service delivery. One of the most effective approaches to achieving these high standards is **shift-right testing.**

In traditional software testing (often referred to as shift-left testing), most tests are conducted early in the development lifecycle—during the coding and unit testing phases. However, shift-right testing flips this approach by performing tests after deployment, in live environments. This practice involves:

- **Continuous monitoring:** Observing how the software performs in real time after it goes live, by gathering data on user behaviors, system health, and performance metrics.
- **Real-time feedback:** Using telemetry and real-world data to identify performance issues, security vulnerabilities, and other potential bugs that may have been missed during pre-deployment tests.
- **Automated and manual testing post-deployment:** Implementing automated tests for constant verification and manual tests for complex user interactions post-deployment.

Shift-right testing ensures that the application performs as expected under real user conditions, considering variables that can't be replicated in a development or staging environment.

Shift-right testing and Shift-left testing – understand the differences

Shift-right testing and Shift-left testing are two approaches that define when and how software testing is performed during the software development lifecycle (SDLC). While both are integral to modern software delivery, particularly in agile and DevOps environments, they differ in their timing, focus, and objectives. Here are the major key differences between shift-right testing and shift-left testing.

Aspect	Shift-left testing	Shift-right testing
Timing	Performed early in the SDLC (during development and design)	Performed after deployment, in production or near-production environments
Focus	Preventive testing (unit tests, integration tests, etc.)	Real-world monitoring and testing (load, performance, security, user behavior)
Test environment	Controlled, static test environments	Real or production-like environments
Goal	Prevent defects early and improve code quality	Detect and resolve issues in production, ensuring real-world readiness
Feedback type	Early detection and feedback from developers and tools	Continuous feedback from live users and system performance data

Table 1: Key differences between the shift-left and shift-right testing approaches

Importance of shift-right testing for healthcare solutions

Healthcare and life sciences solutions directly impact patient safety, treatment efficacy, and regulatory compliance. A minor software failure can result in significant disruptions in patient care, privacy violations, and even legal consequences. Shift-right testing can help mitigate these risks by ensuring continuous monitoring under real-world conditions and fixing the issues quickly when identified. Some key benefits for healthcare include:

- **Real-time monitoring of patient data systems:** Healthcare applications, such as electronic health records (EHRs) or patient management systems, rely on massive amounts of data to function effectively. Shift-right testing allows for constant monitoring of these systems, ensuring their optimal performance, even under heavy loads. It also provides immediate feedback when issues arise.
- **Proactive security measures:** Security is paramount in healthcare due to the sensitivity of patient data. With shift-right testing, healthcare organizations can continuously monitor for potential security vulnerabilities and breaches, significantly minimizing the risk of data leaks or cyberattacks.
- **Improved user experience:** Both clinicians and patients prefer healthcare applications with engaging user experience. Shift-right testing allows developers to track user interactions post-deployment and make iterative improvements, ensuring that applications remain intuitive and efficient in real-world usage.

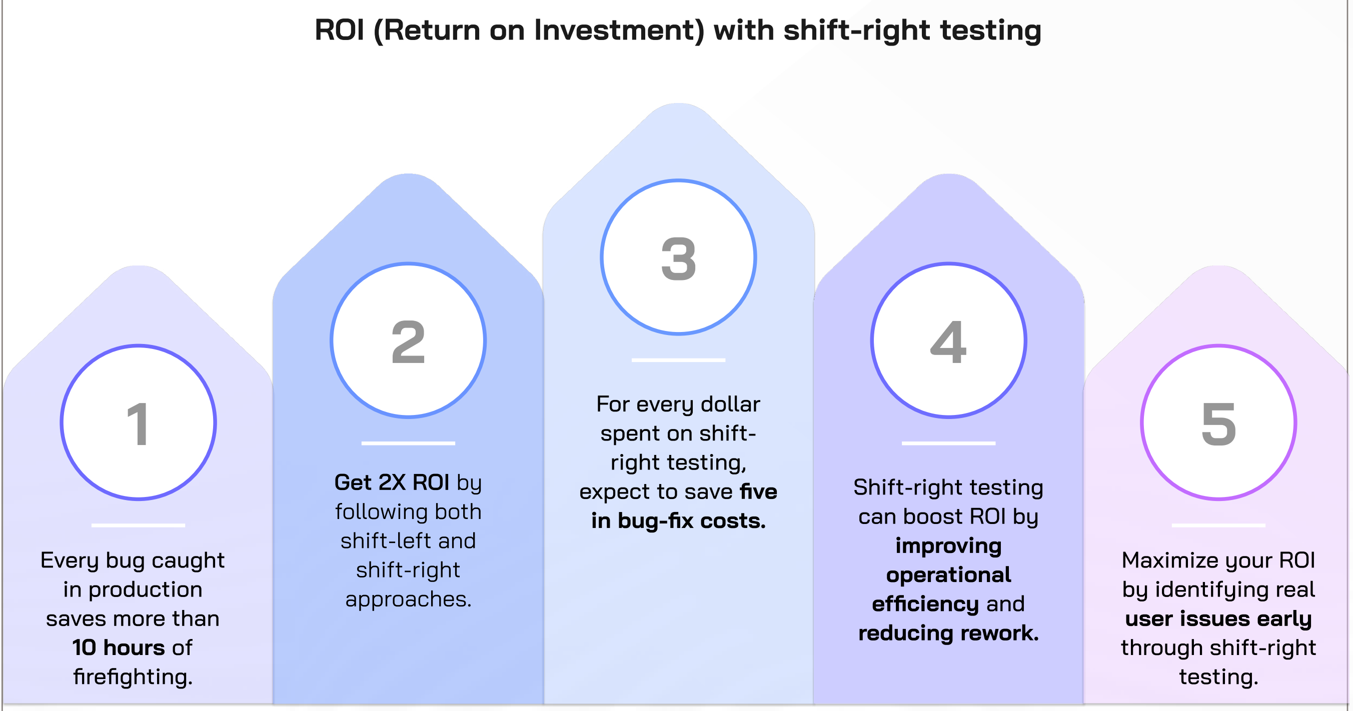

ROI (Return on Investment) with shift-right testing
1
Every bug caught in production saves more than **10 hours** of firefighting.
2
Get 2X ROI by following both shift-left and shift-right approaches.
3
For every dollar spent on shift-right testing, expect to save **five in bug-fix costs.**
4
Shift-right testing can boost ROI by **improving operational efficiency** and **reducing rework.**
5
Maximize your ROI by identifying real **user issues early** through shift-right testing.

Figure 1: ROI with shift-right testing

Key techniques for implementing shift-right testing

Shift-right testing can be implemented in various ways and here are some key techniques to embrace this approach:

1. Production application monitoring and feedback loop: This offers essential practices for ensuring post-release performance. By continuously tracking performance and user behavior in the live environment, teams can quickly detect issues, improve the product, and respond to user needs. The process is as below:

   - **Set up monitoring tools:** Use platforms like Datadog, New Relic, or Prometheus to track key performance metrics such as error rates, response times, and uptime. Include logging and tracing for deeper insights.
   - **Gather user feedback:** Collect user feedback through in-app surveys or tools like Qualaroo and Zendesk to understand their experiences and issues.
   - **Track key metrics:** Monitor performance metrics like load times, error rates, and user engagement with tools like Google Analytics, Mixpanel, or Heatmaps.
   - **Identify issues and anomalies:** Look for performance drops or unusual user behavior and use real-time alerts from monitoring tools to quickly detect issues.
   - **Analyze data and act on insights:** Use data from monitoring and user feedback to prioritize fixes, optimizations, and UI improvements.

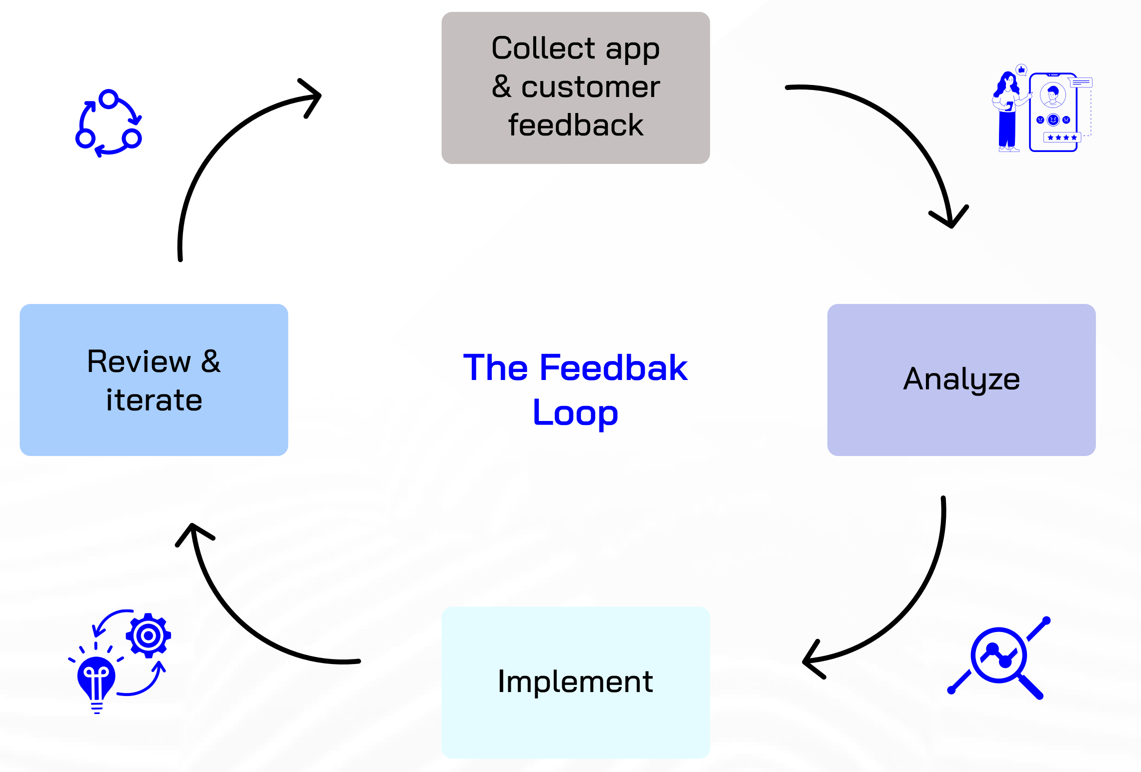


Figure 2: Production application monitoring and feedback loop strategy

2. A/B testing: This involves comparing two or more variants of a feature, design, or system by exposing them to different sets of real users to determine which performs better in terms of user engagement, performance, or satisfaction. This can be implemented by following the steps below.

   - **Select test variables:** Choose the specific features, design changes, or performance aspects to test (e.g., new UI layout, A/B variants of a button color).
   - **Segment users:** Divide your users into distinct groups to ensure each group is exposed to different variants.
   - **Track key metrics:** Use tools like Google Analytics, Optimizely, or VWO to measure user interaction, conversion rates, and engagement.
   - **Analyze results:** Evaluate the performance of each variant based on key metrics and user feedback to determine the winning version.
   - **Iterate based on insights:** Once a variant proves successful, implement it widely while continuing to test new changes.

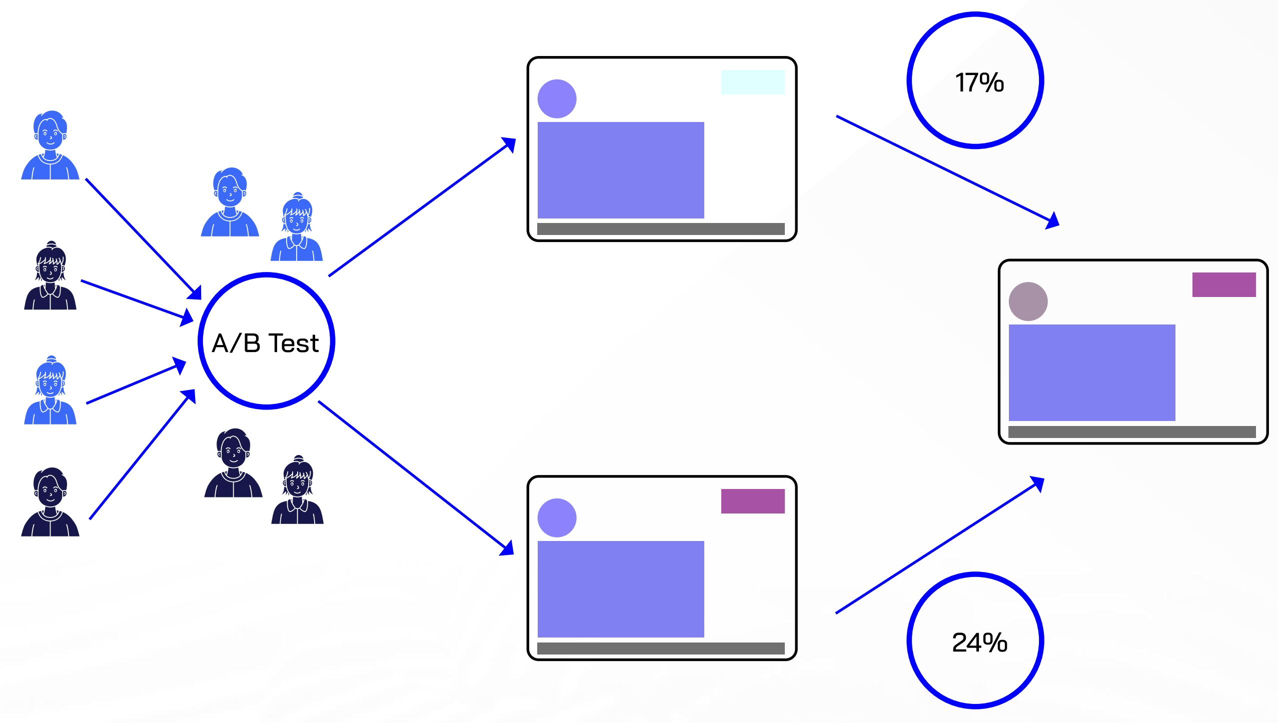


Figure 3: A/B testing strategy

3. Canary releases: This includes deploying a new feature or version of software to a small group of users (the "canary" group) to validate its stability before rolling it out to the entire user base. Below is the process:

- **Select a subset of users:** Begin by releasing the update to a small portion of your user base (usually <10%).
- **Monitor user feedback and metrics:** Use telemetry tools like New Relic, Datadog, or Prometheus to monitor user behavior, error rates, and system performance for the canary group.
- **Gradually scale:** If the canary release performs well, increase its exposure progressively to a larger portion of the user base.
- **Roll back if needed:** If issues are identified, quickly roll back the release for the canary group and fix them before wider deployment.

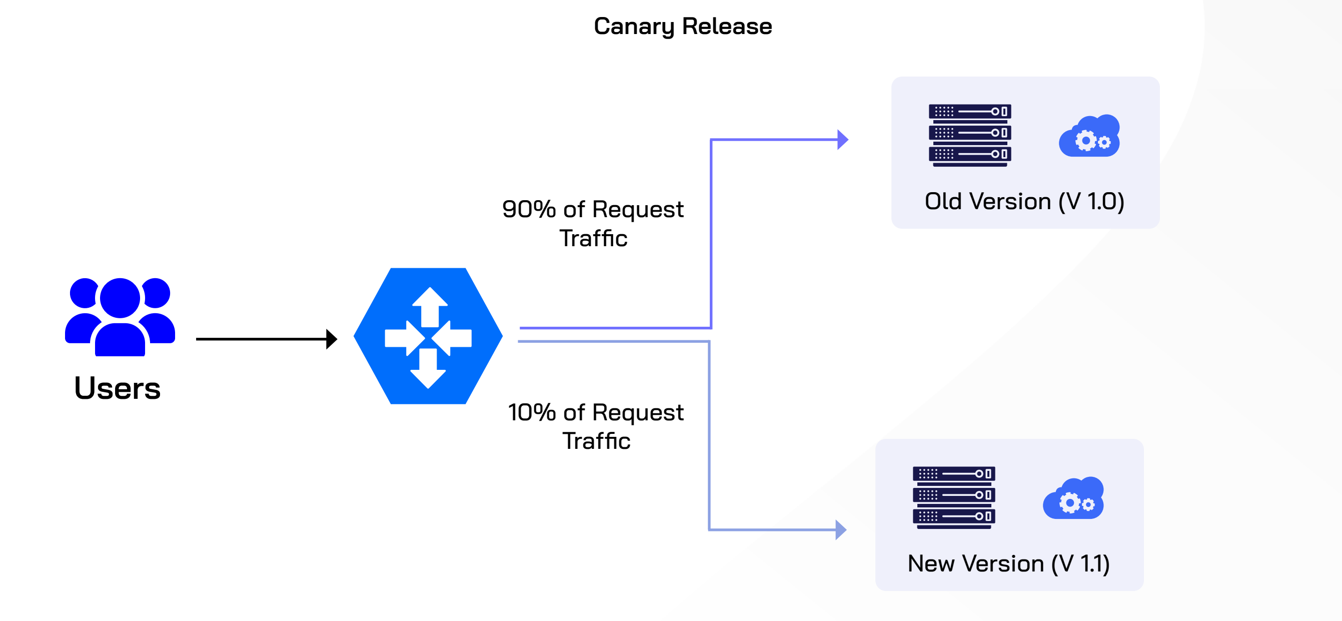


Figure 4: Canary release strategy

4. Real user monitoring (RUM): It tracks how real users interact with your application in a live environment. It provides detailed insights into performance, user experience, and potential issues directly from the end users' perspective. The strategy is as below:

- **Implement RUM tools:** Use RUM tools like New Relic, Dynatrace, or Google Analytics to collect data on users' interactions with your application.
- **Track key metrics:** Monitor page load times, session durations, click patterns, and error rates to gain insight into how real users experience your app.
- **Analyze and optimize:** Use the data to identify performance bottlenecks, user behavior patterns, and areas for improvement.
- **Prioritize user impact:** Focus on resolving the issues that affect the largest segments of users to enhance overall experience.

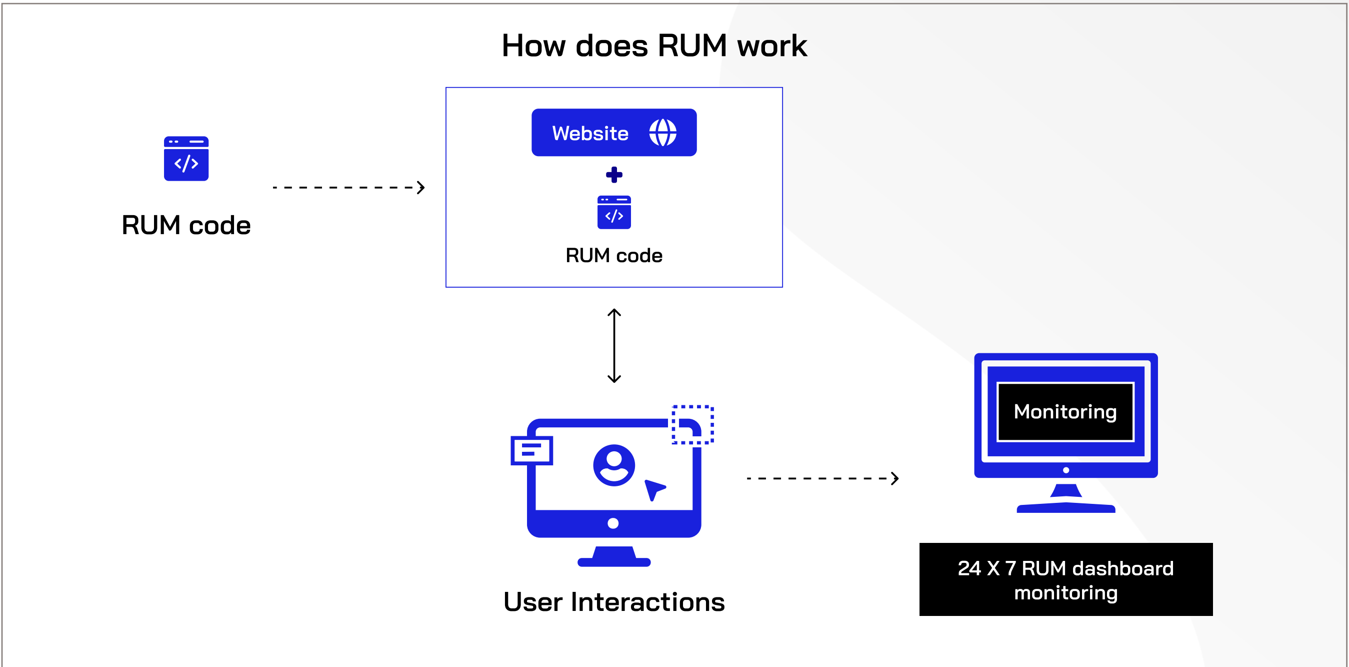


Figure 5: Real user monitoring (RUM) strategy

5. **Feature toggles (aka feature flags):** These allow developers to control the availability of new features without deploying new code. This method enables dynamic enabling/disabling of features for specific users or conditions in real-time. This approach can be followed as below:

   - **Integrate feature flags:** Use feature flag management tools like LaunchDarkly, Optimizely, or Unleash to integrate flags into your codebase.
   - **Define flag criteria:** Set criteria for when features should be enabled, such as user demographics, geographical location, or time of day.
   - **Monitor and adjust:** Monitor the impact of features through analytics and adjust the flag status based on performance or user feedback.
   - **Roll out gradually:** Start with a small group of users, and gradually expand availability as you confirm stability.

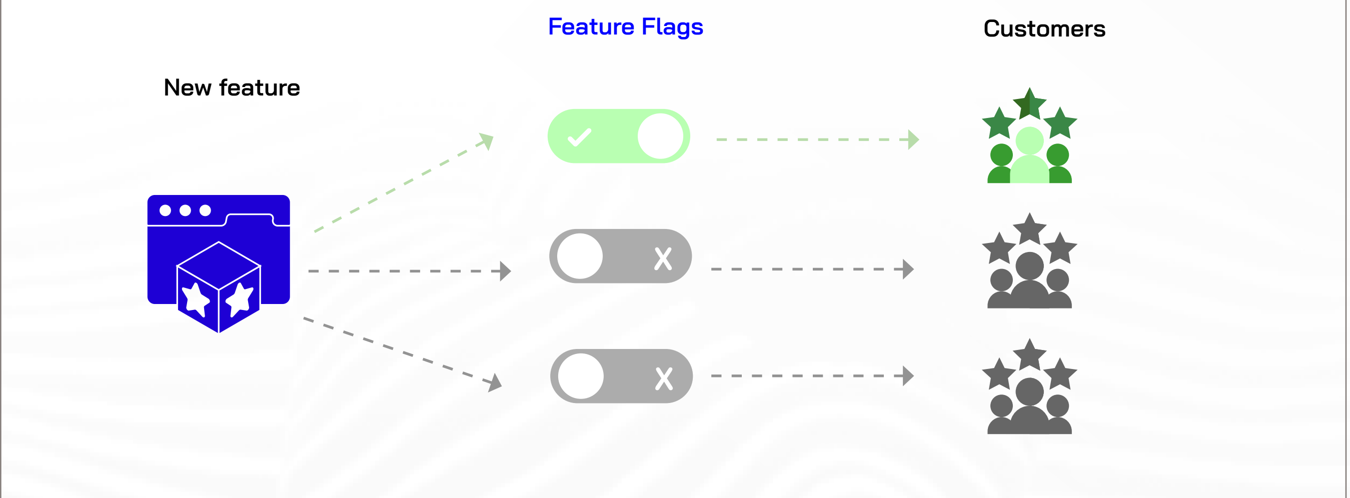


Figure 6: Feature toggles (or feature flags) strategy

Impact of shift-right testing in modern healthcare technologies

Cloud infrastructure and DevOps

Cloud infrastructure and DevOps practices are becoming the backbone of modern software development and deployment. In these environments, shift-right testing plays a crucial role in ensuring continuous quality, scalability, and security. Let's explore the impact of shift-right testing in these areas:

1. **Continuous monitoring in cloud environments:** cloud platforms like AWS, Azure, and Google Cloud provide dynamic and scalable environments that can change rapidly. Software deployed in these environments can be subjected to various load conditions, security challenges, and performance issues that are difficult to replicate in a testing environment. By leveraging cloud-native tools and monitoring services (e.g., AWS CloudWatch, Azure Monitor), shift-right testing allows continuous performance and availability monitoring of cloud-based applications. Issues such as scaling failures, network latency, or resource contention can be detected and addressed in real-time, without waiting for traditional testing cycles.
2. **DevOps integration automating testing and feedback loops:** DevOps emphasizes collaboration between development and operations teams to improve the speed and quality of software delivery. In a DevOps pipeline, shift-right testing can play an integral part; it provides continuous feedback from the live environment. Strategies like Chaos engineering (e.g., Netflix's Chaos Monkey), canary deployments, and feature flags enable testing in production with controlled user subsets. This allows teams to test new features or changes in a safe, real-world scenario, ensuring they don't disrupt the entire system.

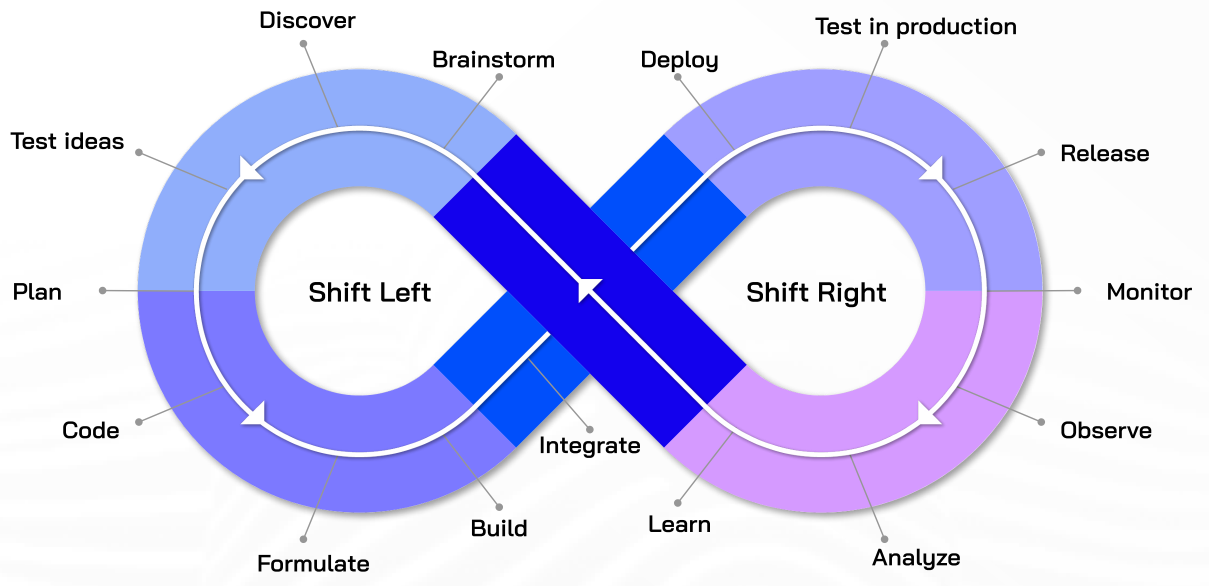


Figure 7: DevOps cycle representation

Anomaly detection and predictive analysis

Anomaly detection and predictive analysis, when integrated with shift-right testing, significantly enhance system reliability and performance in live environments. Anomaly detection identifies unusual system behavior, such as spikes in errors or slow response times, allowing teams to address issues early and prevent them from impacting users, which is especially crucial in high-demand environments. Predictive analysis, on the other hand, uses historical data to forecast potential problems like resource limitations or traffic spikes, enabling teams to take preventive actions and ensure smoother operations. When combined with the shift-right approach, these techniques provide real-time insights into system health, helping teams identify bottlenecks, improve efficiency, and optimize resources. This proactive approach reduces downtime, improves user experience, and minimizes operational costs, ultimately ensuring high system performance and reliable service delivery.

App Security

Security is a critical concern in healthcare software, especially with the rise of data breaches and cyberattacks. Traditional security testing may not always uncover vulnerabilities that could be exploited in a live environment. Shift-right testing addresses this gap by allowing continuous monitoring for new security threats after deployment.

Shift-right security testing helps organizations:

1. **Identify vulnerabilities in real-time:** Continuous security monitoring can detect unusual activity, such as unauthorized access attempts or suspicious data transfers.
2. **Apply automated security patches:** When vulnerabilities are discovered, patches can be applied quickly, ensuring the system remains secure without requiring extensive downtime.
3. **Ensure compliance with regulations:** Monitoring systems for any compliance violations or potential risks can help maintain ongoing adherence to regulations.

Real-world scenarios of shift-right testing in healthcare

Let's explore a few real-world scenarios to understand the impact of shift-right testing.

Scenario 1: Continuous performance monitoring in telemedicine platforms

A telemedicine platform enables virtual consultations between patients and doctors. In a traditional shift-left testing approach, the application may pass performance tests before deployment but fail under real-world conditions when thousands of patients try to access the system simultaneously. With shift-right testing, continuous monitoring of user activity and system performance can identify bottlenecks and slow response times instantly. Based on this feedback, performance improvements can be deployed quickly, ensuring the platform remains responsive during high-demand periods (e.g., during a healthcare crisis).

Scenario 2: Detecting security threats in a healthcare data system

A healthcare provider uses a cloud-based data storage system to store patient records and other sensitive information. After deployment, shift-right testing is implemented to continuously monitor for security threats like unusual login attempts, suspicious IP addresses, or unauthorized data access. When a potential security breach is detected, the system automatically alerts the IT team and takes corrective actions, such as blocking suspicious activities or initiating a security patch.

Scenario 3: Improving user experience in hospital mobile apps

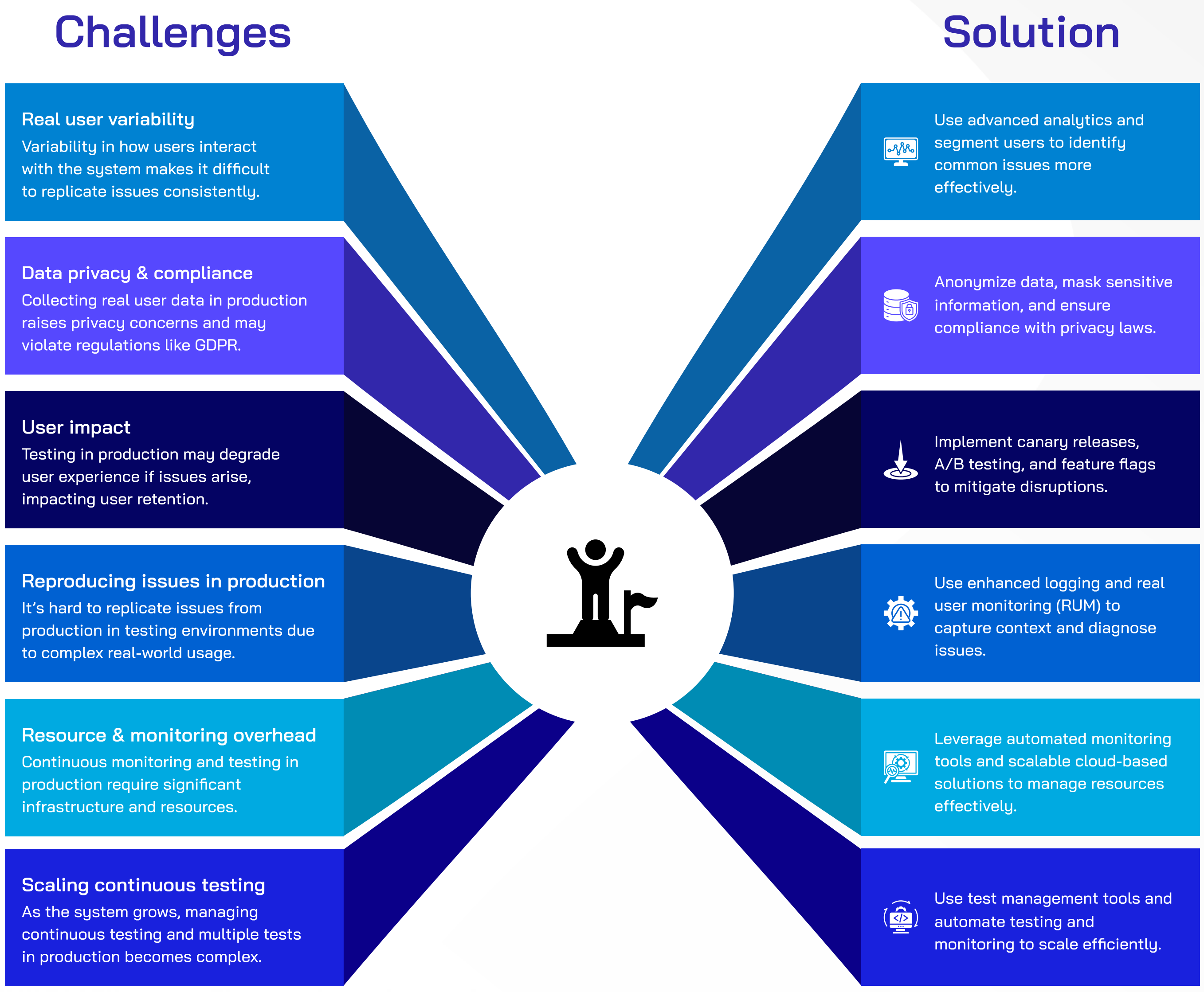
A hospital launches a mobile app for patients to book appointments, receive test results, and communicate with healthcare providers. Post-launch, shift-right testing allows the development team to track how users interact with the app. Feedback from real-world users helps identify areas where the user interface (UI) can be improved, such as simplifying appointment scheduling or improving notification systems. These insights allow the team to make quick iterations and improve the user experience, leading to higher patient satisfaction.

Scenario 4: Hospital management system (HMS)

A HMS integrates features such as scheduling, patient records, and billing. Before deployment, traditional load testing simulates heavy traffic to gauge how the system would handle peak usage. However, once the system is live and being used by thousands of healthcare professionals, issues like network congestion, database bottlenecks, or unforeseen hardware configurations may arise. Healthcare organizations can maintain a high level of service availability and responsiveness, even under heavy usage by leveraging shift-right testing. This helps improve both the user experience for healthcare professionals and the quality of patient care delivered. This feedback loop ensures that the system remains robust and reliable as it scales to meet the demands of a dynamic healthcare environment.

Key challenges & mitigation

Shift-right testing, while offering several benefits, also presents a set of challenges. These challenges arise from performing testing activities after the software is in production, where real users interact with the system under real-world conditions. Here are some of the key challenges and best practices to mitigate them.



Challenges	Solution
Real user variability Variability in how users interact with the system makes it difficult to replicate issues consistently.	Use advanced analytics and segment users to identify common issues more effectively.
Data privacy & compliance Collecting real user data in production raises privacy concerns and may violate regulations like GDPR.	Anonymize data, mask sensitive information, and ensure compliance with privacy laws.
User impact Testing in production may degrade user experience if issues arise, impacting user retention.	Implement canary releases, A/B testing, and feature flags to mitigate disruptions.
Reproducing issues in production It's hard to replicate issues from production in testing environments due to complex real-world usage.	Use enhanced logging and real user monitoring (RUM) to capture context and diagnose issues.
Resource & monitoring overhead Continuous monitoring and testing in production require significant infrastructure and resources.	Leverage automated monitoring tools and scalable cloud-based solutions to manage resources effectively.
Scaling continuous testing As the system grows, managing continuous testing and multiple tests in production becomes complex.	Use test management tools and automate testing and monitoring to scale efficiently.

Figure 8: Key challenges & mitigation

Conclusion

Shift-right testing is revolutionizing how software quality, performance, and security are maintained in live production environments, particularly in high-stakes industries like healthcare. By continuously monitoring applications after deployment, healthcare organizations can ensure their solutions remain secure, efficient, and user-friendly. This approach helps maintain ongoing compliance by monitoring systems for any violations or potential risks. Real-time testing and feedback loops allow teams to identify and address issues before they impact patient care, security, or regulatory compliance.

The healthcare industry's shift-right testing approach is not just about identifying problems, it is also evolving software post-deployment, ensuring that solutions are adaptive, resilient, and capable of supporting critical healthcare operations. With its focus on continuous improvement and real-time monitoring, shift-right testing is a key enabler for delivering high-quality, secure, and patient-centric healthcare solutions.

CitiusTech

CitiusTech is a global IT services, consulting, and business solutions enterprise 100% focused on the healthcare and life sciences industry. We enable 140+ enterprises to build a human-first ecosystem that is efficient, effective, and equitable with deep domain expertise and next-gen technology.

With over 8,500 healthcare technology professionals worldwide, CitiusTech powers healthcare digital innovation, business transformation and industry-wide convergence through next-generation technologies, solutions, and products.

www.citiustech.com

Shaping Healthcare Possibilities