“SHIFT ORBIT” APPROACH IN SOFTWARE DEVELOPMENT

In recent years, the term "shift left" has become a buzzword in the software development industry. It refers to the practice of moving quality assurance and testing activities earlier in the software development lifecycle. The goal of shift left is to catch defects and vulnerabilities earlier in the process before they become more costly and difficult to fix.

However, as technology advances and the pace of software development accelerates, some experts are calling for a new approach: shifting orbit.

What does shifting orbit mean? Essentially, it means taking a broader view of the software development process and rethinking the way we approach quality and security. Rather than simply moving testing earlier in the process, we need to focus on creating a more holistic approach to software development that considers the entire lifecycle of a software application.

Firstly, a shift-orbit approach would require a fundamental change in the way we think about software development. Rather than treating quality as an isolated concern, it should be integrated into every stage of the development process. This means thinking about not just testing and validation, but also user experience, design, architecture, and coding practices that prioritize quality. One key aspect of shifting orbit is a renewed focus on collaboration and communication between different teams involved in software development. breaking down silos and working more closely together. This includes developers and testers as well as security professionals, product managers, and others who have a stake in the success of a software application.

Secondly, a shift-orbit approach would place a greater emphasis on continuous improvement. This

means implementing a feedback loop that allows teams to learn from their mistakes and make improvements in real time. This feedback loop should include not just automated testing and validation tools, but also feedback from end users and other stakeholders to gather insights into the software's usability and user experience. This feedback can inform ongoing improvements and help ensure that the software meets the needs and expectations of its users.

Thirdly, a shift-orbit approach would require a greater investment in technology for testing, validation, and monitoring and analysis of the performance of software applications in production, as well as the infrastructure for continuous integration and deployment. As software development becomes increasingly complex, it's no longer possible for humans to catch every possible issue or vulnerability. Instead, we need to leverage the power of technology to identify and address potential problems before they become serious. For example, machine learning algorithms can be trained to identify patterns and anomalies in code, helping to flag potential issues before they become critical. Similarly, automation tools can be used to automatically test and validate code, freeing up developers and testers to focus on more complex tasks.

Finally, a shift-orbit approach would require a cultural shift to prioritizing quality over speed and agility. This means creating a culture of continuous learning and improvement where teams are encouraged to take risks and learn from their failures, and collaboration and communication are valued over individual achievements.

Overall, shifting orbit represents a more modern, holistic approach to software development that treats quality and security like critical components of the process rather than afterthoughts. By embracing this approach, organizations can build software that is not just functional, but also delivers an exceptional user experience and is secure, scalable, resilient, and ready to meet the demands of the future. For organizations that are serious about delivering high-quality software in a rapidly changing landscape, it's an approach worth considering.