Web Application Deployment Techniques

A software development firm will need to change or upgrade its software at some point. It might be a total version change or a bug fix.

As the new version is ready, here is the real challenge for the firm's DevOps team: How do we deploy this new upgrade seamlessly without complicating things, keeping the user experience in mind?

To solve this humongous task, the DevOps team integrates deployment strategies into their operating procedures. The development of deployment strategies has helped software firms successfully deploy new versions of applications and code. These strategies are known as deployment strategies.

Our aim in this article is to explain what deployment strategies are, the different types of deployment strategies available, and their methods of implementation. We will also cover the advantages and disadvantages of each technique, as well as when to use each in your work as a DevOps engineer.- Cost-effective compared to dedicated servers.

Deployment Strategies: What Are They?

To successfully launch a new version of the software solution they provide, DevOps teams use deployment strategies. Using these techniques, network traffic in a production environment is transitioned from the old version to the new version based on the firm's specialty. Deployment strategies can influence downtime and operational costs based on the company's specialty.

Deployment Strategies of Different Types

Now that we know what deployment strategies are, let's explore the different types of deployment strategies.

Deployment in blue and green

This deployment strategy involves running the new version of the software alongside the old version. Note that this is also known as a red/black deployment strategy.

Stable or older versions of the application are always blue or red, while newer versions are green or black.

> When the new version has been tested and certified to meet all the requirements, the load balancer automatically switches traffic from the older version to the newer one.

> In addition to offering a quick update or rollout of a new application version, this strategy has the disadvantage of being expensive since both the new and old versions must run simultaneously. Engineers mostly use this method in mobile app development and deployment.

Canary Deployments

During canary deployment, the team responsible for deployment gradually redirects traffic from the older version to the new one. For instance, at a certain stage in the process, 90% of production traffic may still go through the older version while only 10% goes through the newer one. This approach allows DevOps engineers to evaluate the stability of the new version by using live traffic from a subset of end-users at varying levels throughout production.

Canary Deployments enables better performance monitoring as well as faster and easier software rollbacks. However, it has a slow deployment cycle and requires more time

Recreate Deployment

The development team shuts down the old version of the application completely, deploys the new version, then reboots the whole. This deployment method produces a system downtime between shutting down the old software and booting the new one.

As there is no traffic shifting from one version to another in the live production environment, it requires no load balancer.

The downtime in this strategy, however, affects the end user dramatically. Users must wait until the application is live again to use it. As a result, not many developers use this strategy unless it is their only option.

Ramped Deployment

The ramped deployment strategy gradually changes the older version to the new version. Unlike canary deployment, the ramped deployment strategy replaces instances of the old application version with instances of the new application version one instance at a time, as opposed to canary deployment. The rolling upgrade deployment strategy is another way to refer to this method.

As soon as developers replace all instances of the older version, they shut it down. After that, the new version is in charge of all production traffic.

The downgrading process to the initial version follows the same cycle, one instance at a time. However, the rollback duration is long in case of an unexpected event.

Shadow Deployment

In this deployment strategy, developers deploy the new version alongside the old version. However, users cannot access the new version immediately. The latest version hides in the shadows. To test how the new version will handle the requests when live, develop pers fork or copy a copy of the old version to the shadow version.

As a result, the DevOps engineer should be careful to ensure that the forked traffic does not create a duplicate live request since two versions of the same system are running simultaneously.

It's expensive and complex to set up, and it can create serious problems. Shadow deployment allows engineers to monitor system performance and conduct stability tests.

A/B Testing Deployment

Developers deploy the new version alongside the older version in A/B testing deployment. However, this new version is only available to a limited number of users, who are selected according to certain conditions and parameters. Location, device type, UI language, and operating system can serve as parameters for selecting these users.

After gathering statistics from performance monitoring, developers deploy the version of the application that yielded the best results.

The use of real-time statistical data can help developers make informed decisions about their software. However, A/B testing is difficult to set up and requires a high-end load balancer.

Tools for better deployment

Developing new features with any deployment strategy isn't easy. DevOps teams need to understand what values they're shipping to their users. Plutora offers tools that make it easier for DevOps teams to launch or release new features with any deployment strategy.

In addition to implementing a deployment strategy and tracking analytics, release management tools allow firms to plan and manage their release. The deployment planning tool is another great tool to have when planning a deployment. By using this tool, risk can be minimized during the deployment process.

Software deployment tools save time as they can manage CI/CD (continuous integration/ continuous delivery), automating deployment. These tools can also enhance security as they can integrate role-based access control.

In conclusion

Explore the various deployment strategies available to update your software. Each has its own advantages and disadvantages, making them suitable for different situations. The main consideration is finding the most suitable option for your DevOps team based on

your team's, project's, and company's requirements and business objectives. Consider factors like downtime tolerance and cost limitations. Enhance your deployment process with Plutora's deployment management tools and release management platform. Book a personalized demo now!

It is inevitable that software development firms will have to change or upgrade their software at some point, whether it is due to a complete version change or a bug fix.

The Timeless Charm of Cricket: An Enduring Sport Across Continents

Cricket, a sport originating in England during the 16th century, has transcended its modest beginnings to become a global phenomenon. The sport is characterized by its distinct format, strategic depth, and cultural significance. From the lush green fields of England to the dusty pitches of India, cricket's allure lies in its ability to adapt and thrive in diverse environments, appealing to both players and spectators.

Historical Evolution

Cricket's evolution from a rustic pastime to a professional sport is a testament to its enduring appeal. The earliest known reference to cricket dates back to a court case in 1598, but it was in the 18th century that the sport began to formalize. The establishment of the Marylebone Cricket Club (MCC) in 1787 played a pivotal role in standardizing the rules and popularizing the game. The first recorded Test match took place in 1877 between England and Australia, marking the beginning of international cricket.

Formats and Gameplay

Cricket is unique in its diverse formats, each offering a distinct experience. Test cricket, the oldest and most traditional format, is played over five days, allowing for a complex and nuanced contest. One Day Internationals (ODIs), introduced in the 1970s, condense the game into a single day, balancing strategy with explosive action. The advent of Twenty20 (T20) cricket in the early 21st century revolutionized the sport, making it faster-paced and more entertainment-driven. Each format has its own set of rules and strategies, adding to the richness of the sport.

Cultural Impact

Cricket's cultural impact is profound, especially in countries like India, Australia, England, Pakistan, and the West Indies. In India, cricket is more than a sport; it's a unifying force that transcends socio-economic boundaries. The Indian Premier League (IPL), a T20 competition, has become a global spectacle, attracting top talent and vast audiences. In England, cricket remains a symbol of tradition and heritage, with events like The Ashes series embodying a historic rivalry.

In Australia, cricket is an integral part of national identity, celebrated with the same fervor as other major sports. The West Indies, with their flamboyant style of play, have produced some of the most charismatic and talented cricketers, contributing significantly to the sport's global appeal. Pakistan's passion for cricket is evident in its vibrant domestic scene and fierce international competition.

Strategic Depth and Skill

The strategic depth of cricket is unparalleled. A cricket match is a chess game of tactics, where decisions on field placements, bowling changes, and batting orders can significantly influence the outcome. Bowlers employ a variety of techniques, from fast bowling to spin, each requiring distinct skills and strategic acumen. Batsmen must adapt to different styles of bowling and pitch conditions, showcasing versatility and resilience.

The role of the captain in cricket is crucial, akin to a general leading his troops. A captain's

decisions during a match can be the difference between victory and defeat, highlighting the sport's emphasis on leadership and tactical thinking.

Global Expansion and Future Prospects

Cricket's global expansion continues to gain momentum. The International Cricket Council (ICC) has been instrumental in promoting the sport in non-traditional regions like the United States, China, and various European countries. Women's cricket has also seen significant growth, with increasing participation and viewership, leading to greater recognition and investment.

Technological advancements have further enhanced the sport, with innovations like Decision Review System (DRS) improving the accuracy of umpiring decisions. The use of analytics and data-driven strategies has also become prevalent, adding a modern dimension to traditional tactics.

Conclusion

Cricket's enduring charm lies in its rich history, diverse formats, cultural significance, and strategic complexity. It is a sport that has evolved with time while retaining its core essence. As cricket continues to expand globally and embrace new audiences, it remains a beloved pastime and a testament to the unifying power of sports. Whether it's the traditional allure of Test matches or the high-octane excitement of T20 games, cricket's legacy as a timeless and captivating sport is secure.