QUARKS

CI/CD Implementation using Azure Dev

A leading SaaS provider for the professional services industry with more than 100,000 users worldwide with a comprehensive end-to-end billing and automated payments processing software. It integrates directly with the legal and accounting practice management software, simplifying the billing and payment experience.

The Challenges

The client was facing several challenges in their software development life cycle that directly impacted the product delivery process in terms of frequency, stability and quality. Some of the major points are highlighted below:

- Missing CI pipeline, and unoptimized CD pipeline due to flat zip deployments
- Lack of configuration management for essential system level dependencies, resulting in manual installation of plugins for each deploy
- No monitoring or alerting setup for critical events such as system crashes, high request failure rate, unresponsive database, out of memory/space issues on the database
- Outdated logging setup which had no facility for bulk operations or log querying.
- No scaling policies in place for high traffic
- Security vulnerabilities reported by Tenable Vulnerability Assessment software Nessus.

Quarks Solution

Container based dynamic and scalable environments for both frontend and backend applications using Docker as the PaaS solution and Azure App Service as the hosting platform for the applications. The response to each of the challenges was as follows:

- CI pipeline setup, for both unit and integration testing, using TestRail for test management and SonarQube for continuous inspection of code quality. In addition, email notifications were set up for test coverage reports to relevant stakeholders. There were also qa and dev approval checks set up to ensure the flow of the review process for a PR
- Containerisation of build and runtime processes which include the critical system level dependencies(phantomjs) eliminating the need of manual intervention by the development team, thereby reducing deployment errors and time to delivery
- Monitoring set up for custom metrics using Application Insights in Azure Monitor- failed requests, server response time, server requests, and availability

- Health check setup for high availability and automated email alerts for application downtime. Multiple other alerts setup on HTTP 5xx responses with thresholds for different severity levels. Database resource limit alerts were also set up to prevent out-of-memory and out-of-storage issues
- Log Analytics setup to enable the client team to query and access logs from within the App Service Azure Portal Interface
- Manual and Automated scaling was available in Azure App Service for the application, but the scaling service was not considered mature in terms of features and control over the scaling process. The team recommended a plan for using Azure Kubernetes Service to further mature the infrastructure and meet the challenges. The client chose to continue with App Service instead
- Security Reports were analyzed and relevant package updates were performed in both development and production environment servers to ensure all vulnerabilities were resolved.

The Benefits

Quarks' innovative solutions helped in:

- Automating the entire CI/CD process. Enabling developers to deploy and test code on multiple environments faster.
- Containerisation also allowed idempotency in build and runtime processes, making them independent of the environment they are being deployed in.
- Monitoring dashboard was created to monitor the essential metrics in one place. Additionally, alerts were set up at app service and database level to ensure no critical events go unreported.
- Health check monitoring allowed for constant monitoring of application in production.
- Up-to-date packages ensured zero critical or severe security vulnerabilities.