clearscale

aws partner network

Premier
Consulting Partner

San Jose Water Company Expedites New Feature Delivery with DevOps Help from ClearScale on AWS

Executive Summary

Founded in 1866, San Jose Water Company (SJWC) is an investor owned public utility, and is one of the largest and most technically sophisticated urban water systems in the United States. SJWC serves over 1 million people in the greater San Jose metropolitan area with high quality, life sustaining water with an emphasis on exceptional customer service. SJWC also provides services to other utilities including operations and maintenance, billing, and backflow testing. By sharing these services with others, SJWC benefits the local community, lowers the cost of water operations, and improves opportunities.

SJWC is owned by SJW Corp., a publicly traded company listed on the New York Stock Exchange under the symbol SJW. SJW Corp. also owns SJW Land Company, and SJWTX, Inc.

"All the operational heavy lifting has been replaced by integrated tools and services. This is a robust and scalable automated solution that will eliminate dependency on developers when it comes to deployment and provide instead needed development hours back to the SJWC engineering team to focus on innovation. ClearScale took our vision for modern IaaS stack utilizing AWS and turned it into a reality while delivering on time and budget."

Jeff Hobbs,
Director of Web & Geographic Systems
San Jose Water Company

The Challenge

At SJWC, their staff takes technology seriously and have implemented innovative solutions across their business. SJWC had been running their web infrastructure out of a Rackspace datacenter for a number of years. SJWC reached a point where their infrastructure requirements surpassed their current capabilities. As with many organizations, they started to research the possible expansion of their existing infrastructure footprint versus migrating to cloud services. The AWS managed services and elastic infrastructure were very appealing to the team at SJWC as they had been using Docker for development and were impressed with the flexibility, control, and portability of the technology. They envisioned having a development team focused on delivering new products and features, an operations team proactively searching the horizon for game changing technologies, and a scalable infrastructure that was fully automated. SJWC took advantage of their growing pains and used them as a catalyst for migration of their traditional web infrastructure into the AWS Cloud. They also viewed their situation as a perfect opportunity to redesign the infrastructure and incorporate updated tooling. SJWC wanted to maximize their return on the AWS business value proposition.

High level requirements:

- Reduce outages/increase availability
- Scalability and Elasticity of services
- Recoverability/Disaster Recovery
- Infrastructure Automation
- Continuous Integration
- Reduce IT management overhead

The goal: Faster delivery of new application features and functionality to their customers with minimal IT overhead and maintenance cost.

The ClearScale Solution

SJWC partnered with ClearScale to design and implement an AWS optimized solution to deliver maximum ROI based on their requirements. From the first meeting, it was obvious that SJWC wanted to focus on building their application and not worry about managing infrastructure services. Through collaborative requirements gathering and architecture design, ClearScale created an AWS Cloud-native solution to achieve this vision. The new solution was radically different because it offered more than just a place to house their product line, it provided a scalable environment for automated continuous integration efforts. SJWC embraced the new approach without looking back.

The plan included implementing container based PHP and Tomcat applications from the Development environments through to Production. The deployment process would be wrapped with full CI/CD automation leveraging Jenkins, Simple Notification Services (SNS), AWS Lambda, and Elastic BeanStalk. Infrastructure provisioning would be automated through CloudFormation on an as-needed basis. Much of the operational tasks were to be offloaded by implementing AWS components such as Elastic Container Services (ECS), Relational Database Services (RDS), EC2 Container Service (ECS), AWS Lambda, Elastic Beanstalk, Elastic Load Balancers (ELBs), CloudWatch, CloudTrail, Identity and Access Management (IAM), Simple Storage Service (S3).

Infrastructure Design

ClearScale recommended a simple infrastructure design for SJWC. Each environment (Staging, Production, Secondary Production) was deployed in a separate Virtual Private Cloud (VPC). Each VPC spanned two Availability Zones (AZs) for redundancy and each AZ contained only two subnets: a Public subnet and a Private subnet. The Public subnet hosted the Elastic Load Balancers (ELBs) and NAT Gateways. The Private subnet hosted the EC2 Container Services (ECS) clusters (Tomcat and PHP applications) and multi-AZ deployments of Postgres RDS and MySQL RDS database services. Auto-scaling was used to expand and contract services based on utilization. All web services were load balanced to ensure redundancy across the AZs. All core infrastructure components were AWS services, which equated to elastic scalability and availability without the management overhead.

Automation (CI/CD)

Infrastructure Automation — AWS CloudFormation gives developers and systems administrators an easy way to create and manage a collection of related AWS resources by provisioning and updating them in an orderly and predictable fashion. A single CloudFormation template was created to deploy and manage the base infrastructure components. This was made possible due to the simple design and implementation of Elastic Beanstalk services.

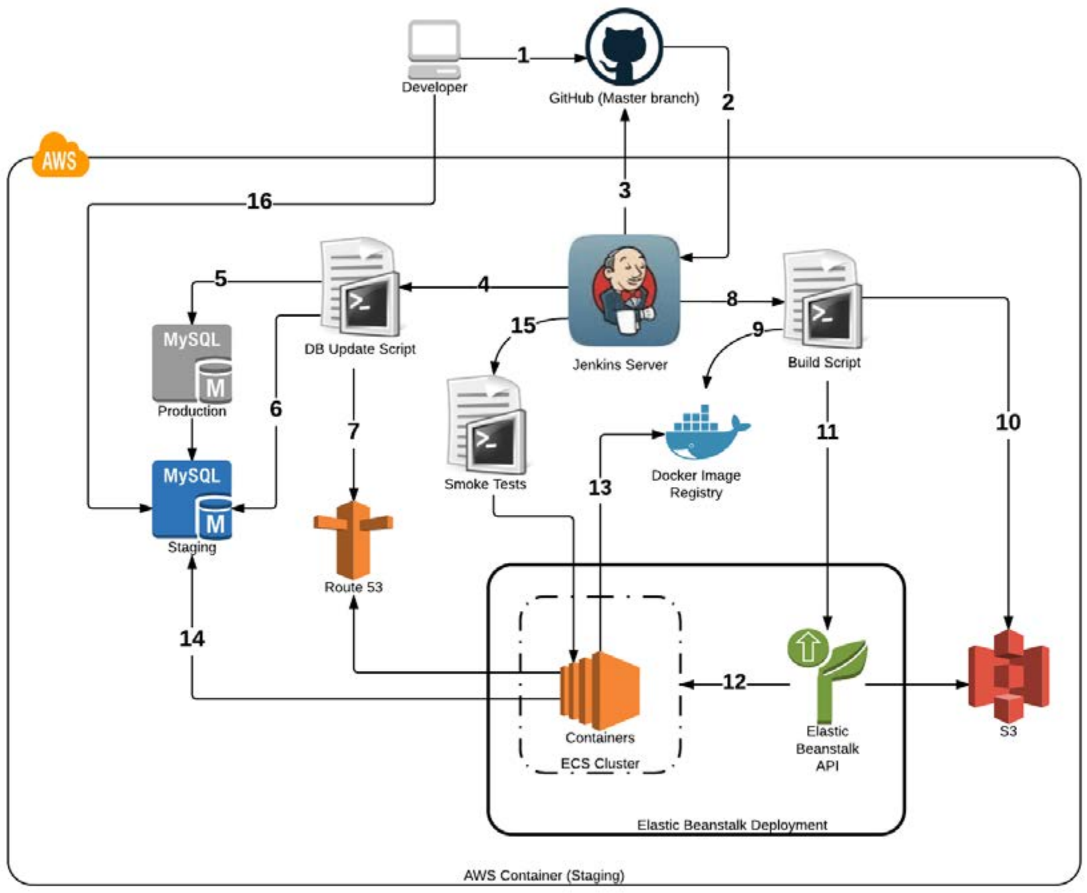
The CloudFormation template included the following:

- VPC with two public and two private subnets in different Availability Zones
- Internet Gateway attached to the VPC
- Routing tables with attached public and private subnets
- Security Groups
- VPN gateway
- NAT Gateway
- Elastic IP addresses

Deployment Automation — Elastic Beanstalk is an easy-to-use service for deploying and scaling web applications and services. SJWC uses Elastic Beanstalk to create ECS clusters, deploy Tomcat/PHP applications (Docker containers), configure auto scaling features, and attach applications to Elastic Load Balancers. This entire deployment process is handled by defining a Beanstalk Application. SJWC has two Beanstalk applications defined — one for their PHP application and one for Tomcat. Deployment automation is completely powered by an AWS service. There is no management overhead for deployment tools support. As an added bonus — it's free to use.

Continuous Integration — Jenkins, Amazon Simple Notification Service (SNS), AWS Lambda, and Elastic Beanstalk create powerful Continuous Integration (CI) flows for SJWC. Each stage of every release has a custom flow based on SJWC process requirements. Some flows are more complex than others, but all were built on this framework. Here is an example of the CI flow for release to the Staging environment that was deployed in AWS:

Staging Environment That Was Deployed in AWS



1. New code is pushed to GIT repository
2. GitHub notifies Jenkins about changes through SNS and Lambda
3. Jenkins deploys new code from GIT repository
4. Jenkins executes DB Update Script
5. DB Update Script creates copy of existing RDS from Production
6. DB Update Script applies changes to existing database
7. Script changes RDS domain alias with new endpoint
8. Jenkins executes Image Build Script
9. Build Script uploads new images to Docker Registry
10. Build Script generates new Beanstalk package
11. Build Script runs Upgrade API Call to Beanstalk application
12. Beanstalk initiates upgrade of Containers in ECS cluster
13. Containers are launched with new Docker images and new applications
14. All Database connections are routed to New Database
15. Jenkins initiates smoke tests for updated applications
16. Developers can query or apply manual updates to Database

Implementation and Migration

Implementing a complex solution like the one ClearScale accomplished for SJWC is not an easy task. Using a tried and true methodology ClearScale developed over years of successful migrations, however, made it a little easier. ClearScale started with an audit of SJWC's infrastructure footprint in their previous datacenter. This allowed the teams at ClearScale to understand what SJWC had and identified a number of weak spots in the existing implementation that were subsequently addressed during the build-out of the new AWS solution. It also allowed for a mapping of existing infrastructure to AWS components; this approach not only met SJWC's requirements that were laid out at the beginning of the project, but it allowed for previous infrastructure to be codified and integrated into an automated solution that would scale the necessary infrastructure needs on an as-needed basis.

Once the basic approach to implementation in AWS had been worked out, CloudFormation was used to deploy the solution to a stage environment within AWS to allow for refinement of the approach. A series of extensive tests were performed against the newly created Stage environment and the findings were then integrated back into the deployment automation toolsets. Once testing was complete to ensure that Stage was being built as expected and it was stable, the Continuous Integration (CI) process was integrated into the Stage environment using Jenkins. It too underwent significant testing and modification until the process of deploying builds automatically from a lower development environment to stage were successful over several mock deployments.

Since the majority of time was spent implementing, testing, and modifying both the automated environmental deployments using CloudFormation and the Continuous Integration process using Jenkins, only a small effort was required to deploy the solution to the production environment in AWS. Once deployed, it too underwent significant end-to-end testing of all of the components and processes to validate that the same process that was used in stage would result in a stable and scalable environment in production.

With this major hurdle overcome, the next major step in migrating SJWC was to start implementing a plan to migrate their data from their existing databases to their new AWS instance. The first step was to enable data replication between the old deployment to AWS so that data in both locations were always in-sync. This then allowed the ClearScale team to focus on setting up mock cut-over exercises to identify any concerns in the approach and fix any data inconsistencies between the two different environments, thus further reducing the risk a large migration such as this always entails. Once the approach had been solidified, a final review and test of each component and process between the environments, including Stage and both the Primary Production and Secondary Production environments, were once again thoroughly tested through successive iterations until all points of risk had been identified and remediated.

When it came time to cutover the client from their old database to AWS, the process could not have been any smoother. By spending the time to fully understand the complexities and risks through iterative testing and subsequent fixes to the process, ClearScale made the actual migration and cutover seamless as well as painless. It is because of this well-proven and easily replicated implementation and migration methodology that ClearScale has been so successful over numerous client implementations in AWS.

The Benefits

SJWC is now able to take full advantage of the value proposition implemented by AWS Premier Consulting Partner: ClearScale. Utilizing Continuous Integration (CI), Continuous Deployment (CD), and infrastructure automation, the quality and repeatability of deployments was significantly improved, resulting in reduced reliance on development resources. The very nature of the solution deployed in AWS meant that the managed services reduced IT overhead and maintenance costs.

The elastic capabilities of auto-scaling delivered cost efficiencies and enabled infinite scaling on-demand. Finally, the new architecture eliminated single points of failure and process bottlenecks that were evident and persistent through SJWC's previous solution in the old datacenter and in turn it provided high availability, highly secured services, and reduced latency.