gslab | GAVS

CASE STUDY

Google Cloud Partner

High Performing Labs Platform through Serverless Architecture on GCP

Customer Overview

The customer is an industry leader in cutting edge computer networking products. Leveraging advanced networking technologies, they deliver high performance, agility, availability, automation, and security through their award-winning platforms.

The Business Context

The customer had a platform featuring their products as labs to enable potential clients to get familiar with product features and functionalities through hands-on interactions. However, the platform was on legacy code base – a deprecated version of Ruby on Rails that was not supported, running on a single Virtual Machine (VM), and based on monolithic architecture that lacked auto scaling and auto healing. It had no monitoring or logging and could not be enhanced with new features. This resulted in missed opportunities, higher operational costs, and poor user experience. The customer was looking to immediately replace the platform with a highly scalable and cost-efficient solution built using modern technologies.

Type of Service Provided

Product Engineering

Technologies Used

Cloud Run, Cloud Scheduler, Serverless VPC Access Connector, Cloud SQL, Cloud CDN, Global External Application Load Balancer, GitHub Actions CI/CD Pipeline, Cloud Logging and Error Reporting, Cloud Deployment Manager, AWS CloudFormation

Solution Summary

The team of product engineering and cloud experts from GS Lab | GAVS designed, architected, and developed a serverless, multi-cloud platform from the ground up, using cloud native technologies. The IaC (Infrastructure as Code) platform features the customer's products as virtual sandboxed labs in isolated cloud environments. The new high performing solution is highly scalable, cost efficient, and capable of launching product labs on both AWS and GCP.

Challenges

- Monolithic architecture that lacked auto scaling resulting in poor performance and bad user experience during heavy usage
- Missed opportunities due to inability to add new features such as SSO authentication or new labs to legacy code base; Version upgrade not possible due to unresolvable dependencies
- Frequent application crashes and manual restarts due to absence of auto healing
- Higher operational costs from billing of even idle application due to continuously running VM
- Dependence on user complaints to detect/fix issues because of lack of monitoring, logging, or error reporting

Solution Highlights

- Highly cost-efficient, scalable, and secure platform with minimal operational overheads, built using cloud native technologies
- Significant speed of deployment through implementation of GitHub Actions based CI/CD pipeline with automation test cases
- Drastic reduction in time, effort, and human error due to complete test automation
- High performing application capable of launching up to 80 lab instances at the same time

Solution Outcomes

- Significant cost advantage of ~20% due to serverless, auto scaling application
- Native support for both GCP and AWS, enabling labs for all product offerings
- Easy enhancements through use of cutting-edge technologies
- SAML based SSO authentication for seamless login for existing users
- Enhanced UX through use of React and Cloud CDN
- Reduced business impact of issues due to auto healing
- Proactive issue detection and resolution through error logging, monitoring, and reporting

To find out how GS Lab | GAVS can help your organization, please visit **www.gavstech.com**

gslab | GAVS

CASE STUDY

High Performing Labs Platform through Serverless Architecture on GCP

Google Cloud Partner

Solution Details

The serverless application was created using the following components:

- Cloud Run - Compute platform
- Postgres on Cloud SQL - Database instance
- Cloud Scheduler – To run cron jobs (periodic background tasks)
- Serverless VPC Access Connector – To connect privately to DB from Cloud Run instance
- Deployment Manager - To deploy labs on GCP
- AWS CloudFormation – To deploy labs on AWS
- Global External Application Load Balancer - As a load balancer for auto scaling and custom domain mapping
- Cloud CDN – To enhance user experience

Cloud Run provides auto scaling and auto healing features which ensures that the application can effortlessly handle varying user traffic and is charged based only on usage. React was used as the frontend development framework and Node for development of backend REST APIs. The frontend is used to create product labs on AWS or GCP. The REST APIs use CloudFormation to create labs on AWS and Deployment Manager to create labs on GCP.

Major Challenges Faced During Implementation

- Since Cloud Run is container based, a private IP cannot be obtained for Cloud Run applications which results in these issues:
  - Cannot directly connect from Cloud Run to the DB over an internal secured network
  - Although Cloud Run can be connected to the DB using public IP, it requires that the DB is exposed to the internet, which is not recommended

The issues were resolved using the Serverless VPC Access Connector which acts as a bridge between the Cloud Run app and the Cloud SQL DB instance, allowing secure private access to the DB from the application.

- Cloud Run containers can perform an action only during the request response cycle, so any background tasks need to be handled exclusively by using the Cloud Scheduler service.
  - Whenever a product lab is launched, background tasks are required to monitor the progress of the launched lab - which sometimes takes 15-20 minutes
  - If a permanent Cloud Scheduler task is created to check the status of the lab every minute, it will solve the problem, but it would also mean that the serverless application is always running (hence losing the cost advantage)

The problems were resolved by creating a temporary Cloud Scheduler task which is deleted once the task is complete, for example, if a Cloud Scheduler task is created to track lab status, once the lab is completely launched, the Cloud Scheduler task is deleted automatically, hence keeping the application cost optimized.

Outcomes Delivered by the Solution

- **Cost Efficiency:** Since the application is serverless, there is no VM that is continuously running providing a significant cost advantage of approximately 20%. Since auto scaling has been configured, the application scales up and down based on user traffic and can even scale down to 0 instance if there is no traffic, during which times the application is not billed.
- **Multi-Cloud Support:** The application provides native support for both GCP and AWS, which is imperative as some of the customer's major product offerings require AWS while the others require GCP
- **Easier Enhancements:** Since the application uses cutting-edge technologies such as the latest versions of Node, React, and cloud native offerings, feature additions and extensions are much easier to develop than on the legacy application.

gslab | GAVS

CASE STUDY

High Performing Labs Platform through Serverless Architecture on GCP

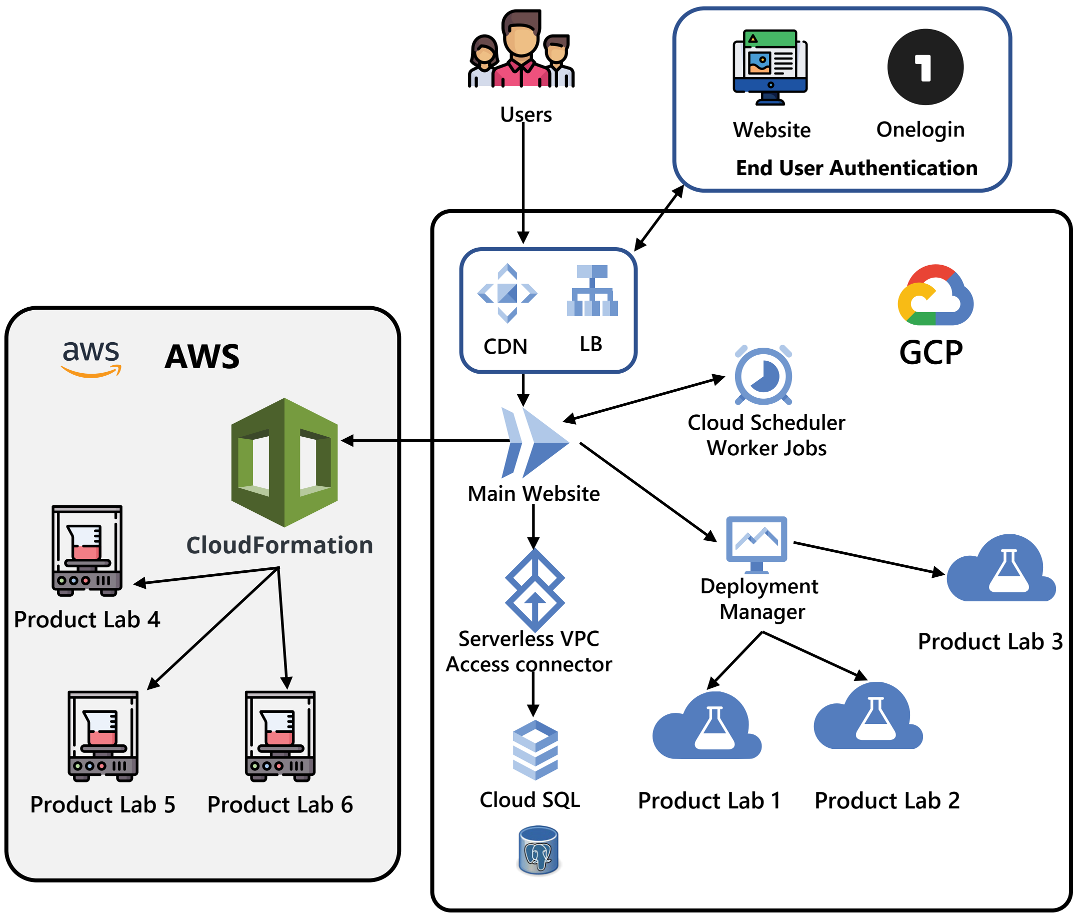
Google Cloud Partner

- **Seamless Authentication:** SAML based SSO authentication was implemented so that a separate user database wasn't required to be maintained and existing users of the customer could seamlessly login.
- **Enhanced User Experience:** React, which is known for its fluid UI experience is used as the frontend framework. Cloud CDN has been configured to enable faster loading of static resources such as images and videos to enhance user experience.
- **Serverless Architecture with Auto Scaling and Auto Healing:** This ensures that even if the application crashes, it automatically restarts so that there is no or minimal impact on business.
- **Error Logging, Monitoring, Reporting:** All errors are logged and reported via email, to help immediate resolution.

Value-Adds

- Use of cloud native technologies to develop this app makes it very cost efficient, scalable, easy to maintain, and secure with minimum operational overheads.
- Implementation of GitHub Actions based CI/CD pipeline with automation test cases for both backend APIs and end-to-end functional testing using frameworks such as Playwright and Jest/SuperTest allows quick movement from feature development to deployment as the entire manual intervention has been automated.
- Significant time saved and human error reduced by automating the entire process since some of the test cases used to take several hours to complete and running them manually was tedious, time consuming, and error prone.
- Performance testing to see the limits of scalability of the application showed that up to 80 different lab instances could be launched at the same time, which is much higher than what the legacy application could handle without breaking down.

Solution Architecture



To find out how GS Lab | GAVS can help your organization, please visit
www.gavstech.com