AWS Architecture Monthly

Open Source

November/December 2020

aws

Notices

Customers are responsible for making their own independent assessment of the information in this document. This document: (a) is for informational purposes only, (b) represents current AWS product offerings and practices, which are subject to change without notice, and (c) does not create any commitments or assurances from AWS and its affiliates, suppliers, or licensors. AWS products or services are provided “as is” without warranties, representations, or conditions of any kind, whether express or implied. The responsibilities and liabilities of AWS to its customers are controlled by AWS agreements, and this document is not part of, nor does it modify, any agreement between AWS and its customers.

Editor's note

According to the Open Source Initiative, the term “open source” was created at a strategy session held in 1998 in Palo Alto, California, shortly after the announcement of the release of the Netscape source code. Stakeholders at that session realized that this announcement created an opportunity to educate and advocate for the superiority of an open development process.

We’ve witnessed big changes in open source in the past 22 years, and our expert this month, Richard Sueiras, talks about a few trends he’s noticed, including the shift from businesses only consuming open source to contributing to it. In this month’s issue, we’re also going to learn how AWS open source projects are one of the ways we’re making technology less cost-prohibitive and more accessible to everyone.

I hope you’ll find this edition of Architecture Monthly useful, and my team would like your feedback. Please give us a star rating and your comments on the Amazon Kindle page. You can view past issues and reach out to aws-architecture-monthly@amazon.com anytime with your questions and comments.

In The Open Source issue:

- **Ask an Expert**: Ricardo Sueiras, Principal Advocate for Open Source at AWS
- **Blog**: How Amazon Retail Systems Run Machine Learning Predictions with Apache Spark using Deep Java Library
- **Case Study**: Absa Transforms IT and Fosters Tech Talent Using AWS
- **Reference Architecture**: Running WordPress on AWS
- **Blog**: Simplifying Serverless Best Practices with Lambda Powertools
- **Whitepaper**: Modernizing the Amazon Database Infrastructure: Migrating from Oracle to AWS
- **Quick Start**: Magento on AWS
- **Related Videos:** Wix, Viber, UC Santa Cruz, & Redfin

Annik Stahl, Managing Editor

What are the general trends for open source?

I'd like to mention three trends I have noticed regarding open source.

The first is a shift from businesses simply consuming open source to contributing to it. A recent Synopsis report shows that while open source is ubiquitous in enterprises, with 99% consuming open source software, only 23% are participating and contributing. Developers are helping to shift that, and a Tidelift survey found that 84% view themselves as active contributors.

The second trend is a shift towards more permissive open source licensing (a non-copyleft open source license — one that guarantees the freedoms to use, modify, and redistribute, but that permits proprietary derivative works). According to a recent survey from WhiteSource, 67% of licenses were permissive.

The third and final trend I've observed is that cloud is making it easier for open source projects to provide an easy way for users to consume and start getting value from their open source technology.

When architecture with open source, are there any specific considerations you have to think about?

There are a few considerations we need to think about when using open source. First, when using a piece of open source technology, you need to understand whether you need to full control over that technology or if it makes sense to use it as a managed service. For example, customers love using Amazon Elastic Kubernetes Service (EKS) instead of have to manage Kubernetes on-premises. To operationalize open source technology you need to figure out how to: 1) scale and make it highly available; 2) secure, patch, and harden it; 3) figure out backup and disaster recovery; 4) manage upgrades…and much more. This is a lot of work, and people often use managed services where all that is taken care of for you. It's important to understand the tradeoffs.

With open source becoming ubiquitous, businesses must think about how to secure the digital supply chain while ensuring they are compliance with open source licensing under which open source tools, libraries, and frameworks operate.

Do you see different trends in open source in cloud versus on-premises?

There are a few trends that businesses need to consider when looking at how open source in cloud vs on-premises can help them meet customer needs.

The pace of innovation in cloud has brought new capabilities and tools that simply do not exist in on-premises. AWS introduced the AWS Graviton Processor, which is an ARM-based (Advanced RISC Machines) architecture. You can take your open source projects, re-compile them, and then run them on these new instance types. This can help you reduced the power footprint, improve the performance characteristics and reduce cost of those workloads.

Another trend we are seeing is customers migrating from proprietary on-premises databases and migrating them to open source equivalents using the AWS Database Migration Service. Beyond databases, they are also transforming their .NET applications to the open source .NET Core equivalent and then running those on modern application technologies, such as containers on Amazon Elastic Kubernetes Service (EKS).

What's your outlook for open source and, what role will cloud play in future of open source?

Marc Andreessen famously wrote that software is eating the world. If that's true, open source is consuming software. Developers are helping to steer a future where organizations move beyond consumption and participation. Recent research has shown that organizations that move beyond consumption write higher quality code, see better productivity of their staff, and double the return on investment against those just consuming open source (University of Southern California, Marshall School of Business, Modern Code Review: A Case Study at Google) The key to this is alignment with the business and executive sponsorship.

The outlook for open source and cloud is looking great. In fact, the future of IT is combining the strengths of open source and cloud to better serve the users of those projects and the teams developing and maintaining them. Cloud is the tide that lifts all boats.

Check out AWS Open Source and follow us on Twitter at @AWSOpen.

About the expert

Ricardo Sueiras is a principal advocate for open source at AWS. He helps customers understand the value of open source, from understanding best practices around the consumption of open source to the reasons why they should participate and contribute. He has been working with open source technologies for over 20 years and has helped many organizations build strategies and programs that help them focus on open source as a strategic enabler.

By Vaibhav Goel and Raja Hafiz Affand

Today, more and more companies are taking a personalized approach to content and marketing. For example, retailers are personalizing product recommendations and promotions for customers. An important step toward providing personalized recommendations is to identify a customer's propensity to take action for a certain category. This propensity is based on a customer's preferences and past behaviors, and it can be used to personalize marketing (e.g., more relevant email campaigns, ads, and website banners).

At Amazon, the retail systems team created a multi-label classification model in MXNet to understand customer action propensity across thousands of product categories, and we use these propensities to create a personalized experience for our customers. In this post, we will describe the key challenges we faced while building these propensity models and how we solved them at the Amazon scale with Apache Spark using the Deep Java Library (DJL). DJL is an open source library to build and deploy deep learning in Java.

Challenges

A key challenge was building a production system that can grow to Amazon-scale and is easy to maintain. We found that Apache Spark helped us scale within the desired runtime. For the machine learning (ML) framework for building our models, we found that MXNet scales to fulfill our data requirement for hundreds of millions of records and gave us better execution time and model accuracy compared to other available machine learning frameworks.

Our team consists of a mix of software development engineers and research scientists. Our engineering team wanted to build a production system using Apache Spark in Java/Scala, whereas scientists preferred to use Python frameworks. This posed another challenge while deciding between Java and Python-based systems. We looked for ways where both teams could work together in their preferred programming language and found that we could use DJL with MXNet to solve this problem. Now, scientists build models using the MXNet – Python API and share their model artifacts with the engineering team. The engineering team uses DJL to run inference on the model provided using Apache Spark with Scala. Since DJL is machine learning framework-agnostic, the engineering team doesn't need to make code

changes in the future if the scientists want to migrate their model to a different ML framework (e.g. PyTorch or TensorFlow).

Data

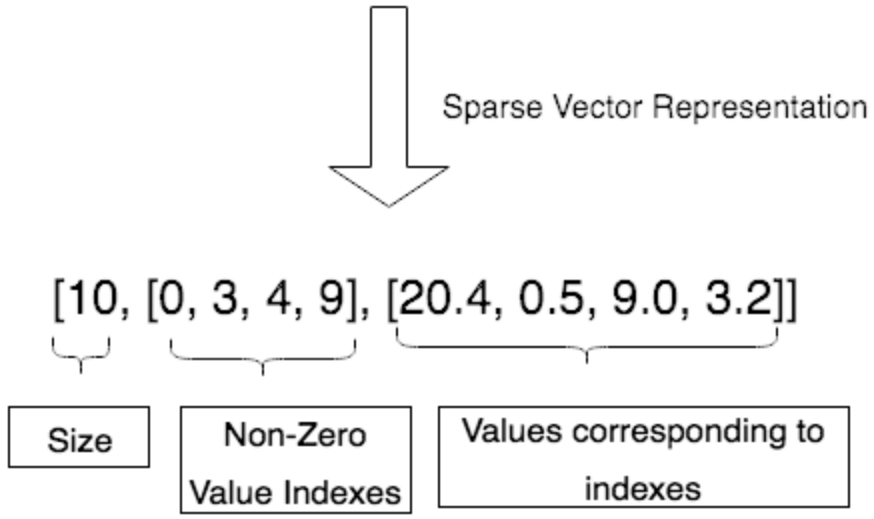
To train the classification model, we need two sets of data: features and labels.

Feature data

To build any machine learning model, one of the most important inputs is the feature data. One benefit of using multi-label classification is that we can have a single pipeline to generate feature data. This pipeline captures signals from multiple categories and uses that single dataset to find customer propensity for each category. This reduces operational overhead because we only need to maintain a single multi-label classification model rather than multiple binary classification models.

For our multi-label classification, we generated high-dimensional feature data. We created hundreds of thousands of features per customer for hundreds of millions of customers. These customer features are sparse in nature and can be represented in sparse vector representation:

Index	0	1	2	3	4	5	6	7	8	9
Feature Value	20.4	0	0	0.5	9.0	0	0	0	0	3.2



Label data

A propensity model predicts the likelihood of a given customer taking action in a particular category. For each region, we have thousands of categories that we want to generate

customer propensities for. Each label has a binary value: **1** if the customer made the required action in a given category, **0** otherwise. These labels of past behavior are used to predict the propensity of a customer taking the same action in a given category in the future. The following is an example of the initial label represented as the one-hot encoding for four categories:

Customer	Category 1	Category 2	Category 3	Category 4
A	1	0	1	0
B	0	1	0	0
C	1	1	0	0
. . .				

In this example, customer A only took actions in category 1 and category 3 in the past, whereas customer B only took actions in category 2.

[Read full post online]

Absa Group (Absa) provides banking services in Africa. The bank offers an integrated set of products and services across personal and business banking, insurance, and wealth and investment management. In 2005, Barclays bought a stake in Absa and later renamed the company Barclays Africa Group. After it reduced its stake in 2017, the bank rebranded as Absa.

Driving Digital Transformation

The rebranding coincided with the transformation of the bank's technology landscape. This included migrating away from existing data centers and building new service capabilities in the cloud for a more agile business and to optimize technology expenditure. The bank chose Amazon Web Services (AWS) to help deliver its aims and, by 2020, had launched more than 100 initiatives on the AWS Cloud. Andrew Baker, chief technology officer for Absa Group, says, "The opening of the new AWS Africa Region means we can now adopt even more AWS services as a result of the Region's proximity and more AWS skilled talent, offer lower latency, and significantly reduce our technology cost base. Additionally, we can achieve our transformational journey with greater velocity, through simple self-service, automation, and skilled DevOps teams."

Ebrahim Samodien, chief information officer for Absa Enterprise Functions, adds, "Our approach to cloud has been to solve real bank problems with an engineering mindset—this is why we work so well with AWS. Like us, AWS has engineering in its DNA and is not pushing products or sales. This partnership has been a key enabler for the culture of innovation and experimentation that we've been driving and embedding across our teams."

Open Source Opens up Opportunities

As part of its innovation and experimentation goals, the bank looked to modernize services and adopt open source, allowing it to move away from expensive proprietary software. Historically, the bank relied on proprietary solutions as well as third-party vendors to support current services and develop new ones. Vendor support, however, created data security challenges. Owing to regulations, the bank was restricted on what data it could share with vendors, limiting development.

The bank has now replaced some of its proprietary software with open source at a fraction of the cost. Moving to open source has allowed the bank to cut reliance on vendors, which has improved data security and given it greater opportunity to design services. Using the non-proprietary software, the bank is building operating systems and databases as well as machine learning and artificial intelligence tooling. What's more, marketing technology across the organization is receiving a major boost. Craig Du Toit, head of technology for Absa Marketing and Legal, explains, "Where previously our focus had to be on core marketing tools, today even non-core services are easily available to us. For example, we are able to quickly and cost-effectively stand up our own fully fledged hosting capability, developing and publishing our own websites on AWS, where we can integrate these with services and rich content management systems. Plus, we have developed our own brand management and marketing tools."

Turning Techies into Chefs

The open-source initiative has allowed the software team to develop its skill sets thanks to the freedom that non-proprietary software brings for creativity. In addition, the team is furthering its skills by making use of the open-source community to deepen its knowledge and crowd source solutions. "Open source and AWS services are the ingredients the bank needs to turn our techies into development chefs," says Du Toit. "We want to drive the development of software talent locally and give them the latest machine learning technologies. These services will be a game changer for Absa."

Eight-Month Analysis Reduced to Three Hours Using Machine Learning

To date, the marketing technology software team has used AWS machine learning services to support the rebrand from Barclays to Absa. The team, working with Synthesis Software Technologies, an Advanced Consulting Partner in the AWS Partner Network (APN), combined Amazon SageMaker, a fully managed machine learning service, and Amazon Rekognition, which makes it easy to add image and video analysis to business applications. The exercise for the marketing department was removing the Barclays logo and other references from bank documents. Comments Du Toit, "This required the analysis of approximately 48,000 pieces of content to identify which ones needed to be redesigned and re-written."

The bank estimated it would take eight months just to manually complete one review cycle of all the documents. Supported by Synthesis Software Technologies, Absa trained Amazon Rekognition using the Amazon SageMaker service to identify the Barclays logo and content. It took just three hours to complete an entire review cycle. Du Toit says, "We identified the documents that included the Barclays logo and copy references with 90 percent accuracy.

Plus, the bank saved R7–8 million ($382,000–$437,000) on the rebrand exercise, while creating an inventory of brand information for marketing teams to use in campaigns."

A Foundation for Innovation

The bank's open-source cloud strategy is now in full swing. The group's technology team continues to support the marketing department's cloud-first strategy and is building several cloud-based services to enhance operations. These include an agency collaboration platform, an online repository for brand guidelines tools, and a content management system. In addition, there is a digital distribution hub for bulk messaging that also offers templates for message creation. So far, the AWS initiatives for the marketing team have delivered projects with projected annual savings of R4.5 million ($246,000)—ensuring the department aligns with the company's larger cost savings goal. Says Du Toit, "People can see the potential of open source on AWS. From here, we want to be bigger and better, building with the talent that exists in Africa."

To learn more, visit aws.amazon.com/machine-learning.

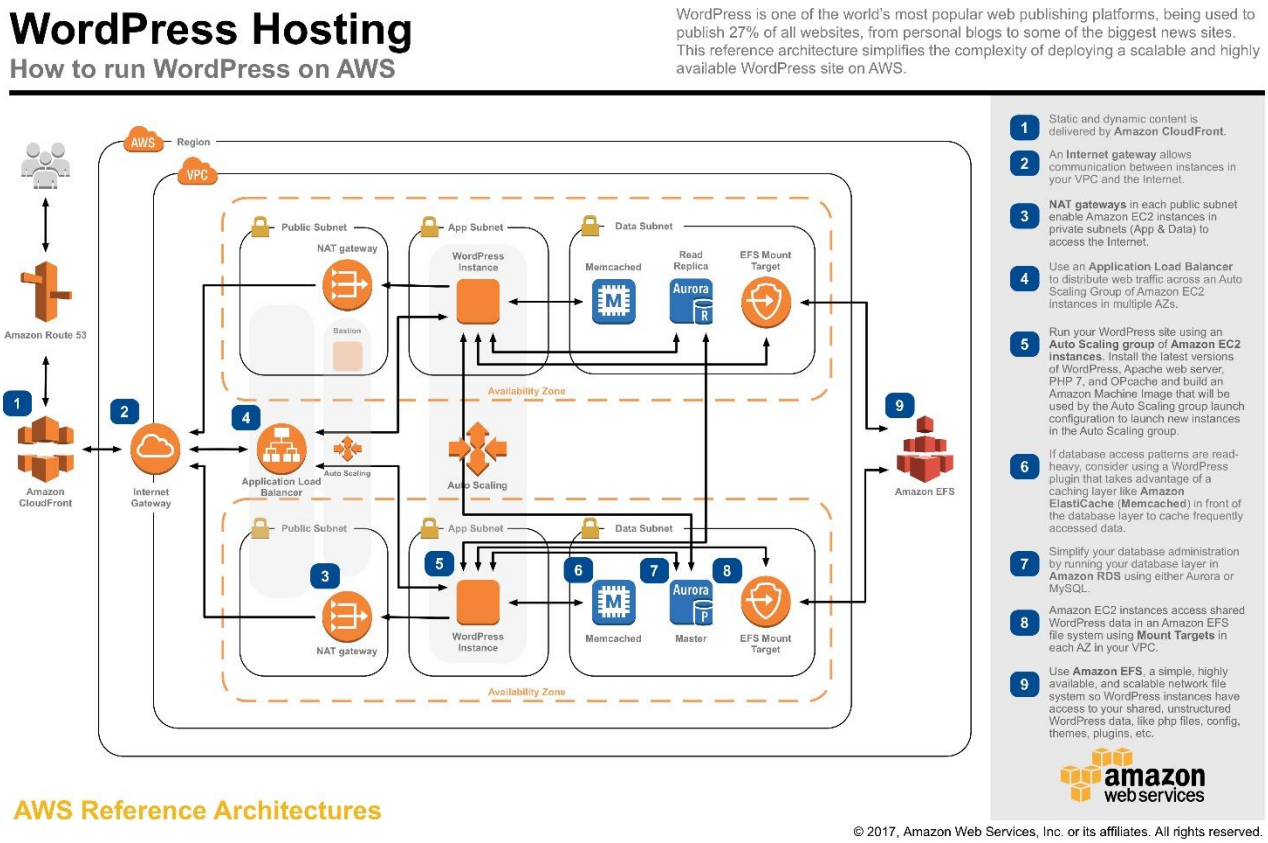
Read case study online

Reference Architecture

Running WordPress on AWS

The WordPress reference architecture provides best practices and a set of YAML CloudFormation templates for deploying WordPress on AWS.

[WordPress reference architecture on GitHub]



See this reference architecture online: [Running WordPress on AWS]

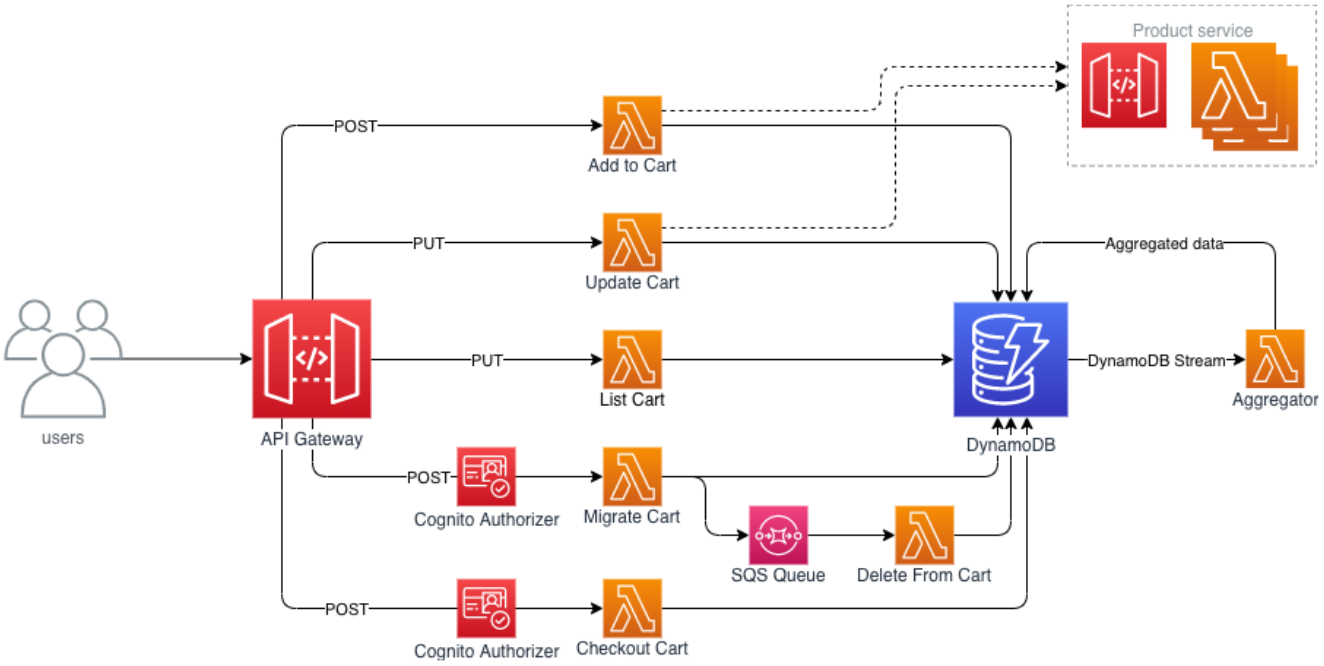
Read the accompanying whitepaper: [Best Practices for WordPress on AWS]

Blog:
Simplifying Serverless Best Practices with Lambda Powertools

By Tom McCarthy

Modern applications are increasingly relying on compute platforms based on containers and serverless technologies to provide scalability, cost efficiency, and agility. Although this shift toward more distributed architectures has unlocked many benefits, it has also introduced new complexity in how the applications are operated. In times past, debugging was as straightforward as logging into the server and inspecting the logs. Now, more thought is required about how to achieve the same level of observability in the applications we design.

In the Serverless Lens for the Well Architected Framework, we suggest several best practices for observability such as structured logging, distributed traces, and monitoring of metrics. In this post, I'll demonstrate how you can use the new open source Lambda Powertools library to implement some of these best practices without needing to write lots of custom code. I'll walk through the process of getting started with the library, with examples of the implementation drawn from a sample shopping cart microservice:



About Lambda Powertools

Lambda Powertools is an opinionated library that will help with implementing observability best practices without the undifferentiated heavy lifting. It currently supports AWS Lambda functions written in Python, with support for runtime versions 3.6 and newer. Lambda Powertools provides three core utilities:

Tracer provides a simple way to send traces from functions to AWS X-Ray to provide visibility into function calls, interactions with other AWS services, or external HTTP requests. Annotations easily can be added to traces to allow filtering traces based on key information. For example, when using Tracer, a *ColdStart* annotation is created for you, so you can easily group and analyze traces where there was an initialization overhead.

Logger provides a custom Python logger class that outputs structured JSON. It allows you to pass in strings or more complex objects, and will take care of serializing the log output. Common use cases—such as logging the Lambda event payload and capturing cold start information—are handled for you, including appending custom keys to the logger at anytime.

Metrics makes collecting custom metrics from your application simple, without the need to make synchronous requests to external systems. This functionality is powered by Amazon CloudWatch Embedded Metric Format (EMF), which allows for capturing metrics asynchronously. Again, convenience functionality is provided for common cases, such as validating metrics against CloudWatch EMF specification, and racking cold starts.

See the code and read the full post online

Whitepaper:

Modernizing the Amazon Database Infrstructure: Migrating from Oracle to AWS

Abstract

This whitepaper is intended to be read by existing and potential customers interested in migrating their application databases from Oracle to open source databases hosted on AWS. Specifically, the paper is for customers interested in migrating their Oracle databases used by Online Transactional Processing (OLTP) applications to Amazon DynamoDB, Amazon Aurora, or open source engines running on Amazon RDS.

The whitepaper draws upon the experience of Amazon engineers who recently migrated thousands of Oracle application databases to Amazon Web Services (AWS) as part of a large-scale refactoring program. The whitepaper begins with an overview of Amazon's scale and the complexity of its service oriented architecture and the challenges of operating these services on on-premises Oracle databases. It covers the breadth of database services offered by AWS and their benefits. The paper discusses existing application designs, the challenges encountered when moving them to AWS, the migration strategies employed, and the benefits of the migration. Finally, it shares important lessons learned during the migration process and the post-migration operating model.

The whitepaper is targeted at senior leaders at enterprises, IT decision makers, software developers, database engineers, program managers, and solutions architects who are executing or considering a similar transformation of their enterprise. The reader is expected to have a basic understanding of application architectures, databases, and AWS.

Overview

The Amazon consumer facing business builds and operates thousands of services to support its hundreds of millions of customers. These services enable customers to accomplish a range of tasks such as browsing the Amazon website, placing orders, submitting payment information, subscribing to services, and initiating returns. The services also enable employees to perform activities such as optimizing inventory in fulfillment centers, scheduling customer deliveries, reporting and managing expenses, performing financial accounting, and analyzing data. Amazon engineers ensure that all services operate at very high availability, especially those that impact the customer experience. Customer facing

services are expected to operate at over 99.90% availability leaving them with a very small margin for downtime.

In the past, Amazon consumer businesses operated data centers and managed their databases distinct from AWS. Prior to 2018, these services used Oracle databases for their persistence layer that amounted to over 6,000 Oracle databases operating on 20,000 CPU cores. These databases were hosted in tens of data centers on-premises, occupied thousands of square feet of space, and cost millions of dollars to maintain. In 2017, Amazon consumer facing entities embarked on a journey to migrate the persistence layer of all these services from Oracle to open-source or license-free alternatives on AWS. This migration was completed to leverage the cost effectiveness, scale, and reliability of AWS and also to break free from the challenges of using Oracle databases on-premises.

Read the full whitepaper online

This Quick Start automatically deploys Magento Open Source (formerly Community Edition) on the AWS Cloud.

Magento is an open-source content management system for e-commerce websites. This automated deployment builds a cluster that runs Magento along with optional sample data, which lets you experiment with custom themes and view the web store.

The deployment uses your choice of Amazon Aurora or MySQL on Amazon RDS for database operations, Amazon EFS for shared storage between EC2 instances, and an Amazon ElastiCache cluster with the Redis cache engine to improve application load times.

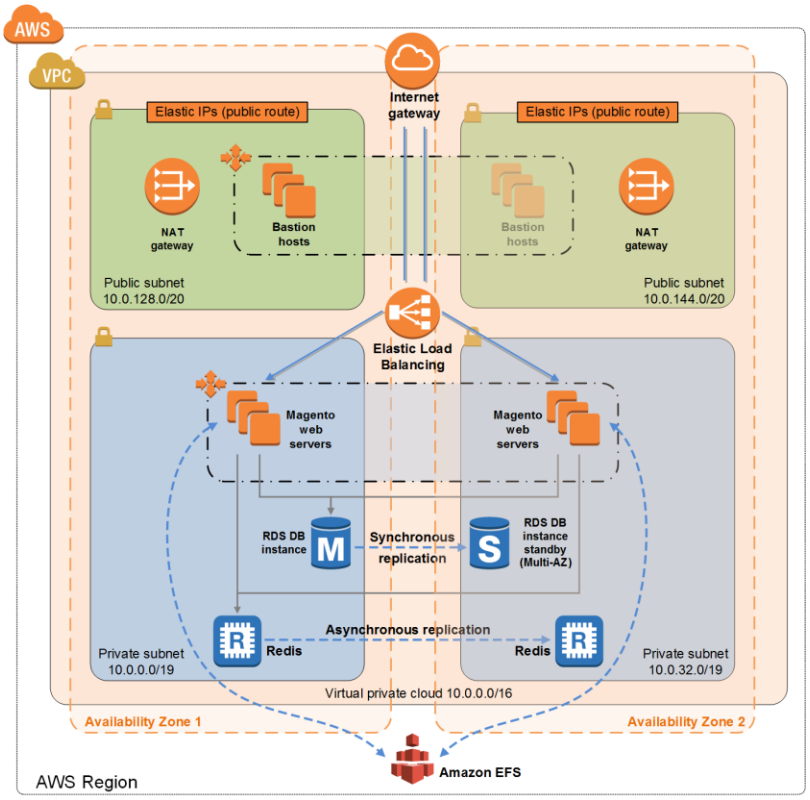
What you'll build

Use this Quick Start to automatically set up the following Magento environment in your AWS account:

- A virtual private cloud (VPC) that spans two Availability Zones, configured with two public and two private subnets.*
- In a public subnet, a bastion host to provide Secure Shell (SSH) access to the Magento web servers. The bastion host is maintained by an Auto Scaling group that spans multiple Availability Zones, and is configured to ensure there is always one bastion host available.*
- AWS-managed network address translation (NAT) gateways deployed into the public subnets and configured with an Elastic IP address for outbound internet connectivity. The NAT gateways are used for internet access for all EC2 instances launched within the private network.*
- Either an Amazon RDS for MySQL or an Amazon Aurora database engine deployed via Amazon RDS in the first private subnet. If you choose Multi-AZ deployment, a synchronously replicated secondary database is deployed in the second private subnet. This provides high availability and built-in automated failover from the primary database.
- An Amazon ElastiCache cluster with the Redis cache engine launched in the private subnets.

- Amazon EC2 web server instances launched in the private subnets.
- Elastic Load Balancing deployed to automatically distribute traffic across the multiple web server instances.
- Amazon EFS created and automatically mounted on web server instances to store shared media files.
- Auto Scaling enabled to automatically increase capacity if there is a demand spike, and to reduce capacity during low traffic times. The default installation sets up low and high CPU-based thresholds for scaling the instance capacity up or down. You can modify these thresholds during launch and after deployment.
- An AWS Identity and Access Management (IAM) instance role with fine-grained permissions for accessing AWS services necessary for the deployment process.
- Appropriate security groups for each instance or function to restrict access to only necessary protocols and ports. For example, access to HTTP server ports on Amazon EC2 web servers is limited to Elastic Load Balancing. The security groups also restrict access to Amazon RDS DB instances by web server instances.

Learn more about [how to deploy and see cost and licenses].



Wix: Serverless Platform for End-to-End Browser Testing using Chromium on AWS Lambda

Tom from Wix describes the company's end-to-end testing platform called SLED. SLED uses open source technologies such as Chromium, Jest, and Puppeteer to orchestrate the parallel run of hundreds of concurrent tests using AWS Lambda as the underlying compute. SLED is integrated into developers' machines or CI servers to deliver a completely seamless developer experience that responds as if the test was locally executed.

Watch on YouTube

Viber: Massive Data Lakes on AWS

When your data lake is in the multiple petabytes, the architecture around it has to be built for massive scale. See how Viber built a data storage and processing solution that handles 10-15 billion events per day, peaking at 300k events per second, for its over 1B users worldwide. You'll see how they use a combination of AWS services, open source tools, and AWS partner solutions to build a flexible, end-to-end solution. We'll discuss their use of Kinesis, Kinesis Firehose, Lambda, Redshift, S3, Athena, EMR, Aurora, Storm, Spark, and more.

Watch on YouTube

Big Data Discount: How UC Santa Cruz Uses Mesos & Amazon EC2 Spot to Enable Low Cost Cancer Research

Mary Goldman, Design and Outreach Engineer at the UC Santa Cruz Genomics Institute, explains how they process genomic sequencing data on AWS. With a need to crunch data measured in petabytes, they designed a low cost solution using a combination of Docker containers and EC2 Spot instances. TOIL, the pipeline management system they built is open source.

Watch on YouTube

Redfin: Full Real Estate Brokerage Platform at Scale

Brian shares how Redfin notifies potential homebuyers with relevant opportunities using open source technologies like Apache Kafka, PostgreSQL, Docker, & Apache Samza on top of Amazon EC2.

Watch on YouTube