Cloud-Native Architecture for an Emerging Online Retail Business

Marawan Eldeib - Matrikelnummer: 3764796

st196189@stud.uni-stuttgart.de

Stuttgart University

Abstract. This paper presents a cost-optimized cloud-native architecture for an emerging online retail business, leveraging AWS services for scalability, automation, and operational efficiency. The design incorporates serverless computing, microservices, and event-driven workflows to minimize costs while ensuring future growth and high availability.

1. Introduction

A cloud-native architecture enables scalability, resilience, and operational efficiency, making it ideal for an emerging online retail business. This paper presents a solution for six core services: e-commerce, content management, inventory, payment processing, customer relationship management, and analytics, aligning with AWS Well-Architected Framework principles and industry best practices. While AWS is the primary provider, the design follows a cloud-agnostic approach, ensuring flexibility for multi-cloud integration if required.

2. Cloud-Native Industry Best Practices

Cloud-native best practices prioritize scalability and operational efficiency. The AWS Well-Architected Framework and Cloud Native Computing Foundation establish key patterns including microservices, containerization, infrastructure as code, event-driven architecture, serverless computing, appropriate database selection, and comprehensive monitoring. These enable high availability, automatic scaling, and cost optimization—ideal for an emerging retail business with growth potential [1–8].

3. Component Architecture

3.1 Website for Browsing, Ordering, and Returns

The website architecture is designed for scalability, reliability, and high performance, ensuring smooth browsing, ordering, and returns. Amazon S3 stores static assets, while Amazon CloudFront (Content Delivery Network, CDN) improves load times, reduces latency, and offloads traffic from the origin server by caching content at edge locations [9, 10].

For backend processing, AWS Elastic Beanstalk automates deployment, scaling, and management, integrating Elastic Load Balancing (ELB) and Auto Scaling to ensure high availability [11, 12] . Amazon DynamoDB is used for both transactional data and fast lookups for product catalogues and shopping carts, ensuring high availability, scalability, and cost efficiency without the overhead of relational database management like AWS RDS [13, 14]. AWS Lambda supports serverless execution of business logic, handling processes like cart updates and order validation without maintaining dedicated infrastructure [15]. Amazon CloudWatch provides real-time monitoring and logging, enhancing system visibility and operational efficiency [16].

AWS services were chosen over Microsoft Azure and Google Cloud Platform for their seamless integration, scalability, and cost efficiency. Amazon CloudFront offers better AWS compatibility and global coverage than Azure CDN and Google Cloud CDN [17]. Elastic Beanstalk simplifies deployment compared to Azure App Service and Google App Engine, which require more manual setup. DynamoDB provides better scalability for key-value workloads compared to Firestore and Cosmos DB, while AWS Lambda ensures lower execution costs than Azure Functions and Google Cloud Functions.

3.2 Blog

The blog architecture leverages AWS services for efficient content delivery, scalability, and minimal operational overhead. Amazon S3 and CloudFront (CDN) ensure fast, reliable access to static assets and cached content globally. For dynamic content management, Amazon DynamoDB provides highly available, scalable storage for blog posts and metadata, enabling fast retrieval without manual scaling [14]. AWS Lambda powers on-demand content generation, handling backend logic efficiently [18].

Amazon API Gateway manages and secures incoming requests, enabling authentication, throttling, and request routing for seamless interaction between the frontend and backend services [19]. Amazon

CloudWatch ensures real-time monitoring and logging, providing insights into traffic patterns and performance.

Compared to self-hosted CMS platforms or virtual machines, this serverless, fully managed approach minimizes operational complexity, ensuring high availability, reduced latency, and cost efficiency. For businesses preferring a managed CMS, Contentful or WordPress API can be integrated via API Gateway, enabling headless content management. Cached content is delivered via Amazon CloudFront to ensure low latency and high availability [20, 21].

3.3 Inventory and Fulfillment

The inventory and fulfillment system leverage Amazon DynamoDB for real-time stock updates, ensuring low-cost, scalable storage without the overhead of relational databases . AWS Lambda automates inventory updates, triggering workflows via Amazon EventBridge to maintain stock accuracy [15, 22]. Amazon SQS queues manage order processing, preventing system overload during peak demand [23]. If third-party logistics services like DHL are used, order status updates are synchronized using webhooks and Amazon EventBridge, ensuring seamless inventory tracking [24]. API Gateway secures external API interactions, while Amazon SQS queues manage delays in fulfillment updates.

For demand forecasting and analytics, inventory data can be streamed to Amazon Kinesis, allowing real-time insights without requiring an expensive data warehouse [25]. Amazon CloudWatch provides monitoring and logging to ensure system reliability. By replacing RDS with DynamoDB, this solution eliminates fixed database costs while ensuring scalable, real-time inventory management at a lower operational expense.

3.4 Payment

I recommend Stripe as the primary payment gateway due to its security, reliability, and ease of integration [26, 27]. It is PCI DSS compliant, encrypts sensitive data, ensures high availability, supports fast transactions, and offers multi-currency support, making it ideal for global transactions [28–30].

To ensure secure and scalable payment processing, the architecture integrates AWS Lambda for executing payment logic, reducing infrastructure management. Stripe processes and securely stores financial transactions externally, ensuring PCI DSS compliance and reducing regulatory overhead. Integration occurs via its REST API with OAuth 2.0 authentication, while Stripe Webhooks triggers real-time order updates. OAuth 2.0 authentication secures API calls, while Stripe Webhooks triggers real-time order updates. Failed transactions are retried using Amazon SQS, preventing double charges [31, 32]. Amazon API Gateway controls access, ensuring a secure payment flow. Amazon DynamoDB stores lightweight transaction metadata, such as order status and payment confirmations, ensuring cost-efficient data management. API Gateway provides controlled access to payment-related APIs, while AWS KMS encrypts sensitive payment data [33, 34]. SQS ensures reliable transaction processing, reducing risks of failed payments due to network or system failures [35].

Using third-party payment providers minimizes compliance risks and development effort compared to in-house solutions, which are costly and complex to secure. While third-party providers incur transaction fees, they reduce infrastructure costs and ensure a cost-effective, secure, and scalable payment system.

3.5 Customer Relationship Services

A scalable CRM system is essential for managing customer interactions efficiently. Amazon Connect provides a cloud-based contact center for voice and chat support, enabling automated routing and AI-driven assistance [36]. Amazon SES handles reliable email communication for transactional and marketing messages, while Amazon SNS ensures real-time notifications via SMS, push notifications, and email [37, 38]. AWS Lambda automates customer service workflows, improving response times and reducing manual effort [15, 18].

While Salesforce Service Cloud offers AI-driven insights and automation, its costs and integration complexity may not be ideal for a cost-constrained business. An AWS-native approach using Amazon Connect, SES, and SNS provides a scalable and cost-effective alternative while still allowing future integration with Salesforce if needed [36, 39, 40]. With pay-as-you-go pricing, this solution reduces infrastructure expenses while ensuring high availability. If Salesforce Service Cloud is used, AWS Lambda synchronizes customer data with Amazon DynamoDB. API Gateway enforces security for outbound API calls, while Amazon Connect remains a fallback option for an AWS-native CRM approach

3.6 Analytics Services

A data-driven analytics system is essential for understanding customer behavior and optimizing operations. Amazon Kinesis processes streaming data from customer interactions and DynamoDB streams, enabling real-time analytics and trend identification [25]. For structured data storage and reporting, Amazon Redshift aggregates information from multiple sources, including transactional metadata stored in DynamoDB, optimizing performance for analytical queries [41]. AWS Glue automates data extraction, transformation, and loading (ETL) processes, ensuring scalability and cost efficiency [42]. Amazon QuickSight provides interactive dashboards for tracking sales and customer engagement, while AWS SageMaker enables predictive analytics, such as churn analysis and personalized recommendations [43, 44].

When selecting a data warehouse, Amazon Redshift, Snowflake, and Google BigQuery were evaluated. Redshift was chosen due to its tight AWS integration, optimized query performance, and cost-effectiveness for frequent analytical workloads [45]. Snowflake offers elastic scalability with separate compute and storage, but its pricing is less predictable. BigQuery's serverless model charges per query, making costs unpredictable for high-volume analytics workloads [46]. Given the need for cost control, seamless AWS service integration, and predictable performance, Redshift is the most suitable option. The pay-as-you-go model ensures cost efficiency, and serverless analytics (Glue, Kinesis) reduces operational overhead while enabling real-time insights.

Compared to self-managed solutions, this architecture provides a cost-effective and scalable approach to business intelligence.

4. Security, Availability, and Performance

Security is enforced using AWS IAM for role-based access control, AWS Cognito for authentication, and AWS KMS for encrypting sensitive data [33, 47]. API Gateway applies request authentication and rate limiting, while AWS WAF and AWS Shield protect against web threats and DDoS attacks [48]. High availability is ensured through multi-AZ deployments, Auto Scaling, and Elastic Load Balancer (ELB), distributing traffic efficiently [49]. Amazon CloudFront (CDN) caches static content globally, reducing latency and improving performance [10]. DynamoDB (NoSQL) enables fast, scalable storage without maintenance overhead [14], while CloudWatch provides real-time monitoring and alerts [16]. This approach maintains security, availability, and performance while controlling costs by using serverless and managed AWS services, reducing operational complexity.

5. Operations and Lifecycle Management

Operations are automated using Infrastructure as Code (AWS CloudFormation), ensuring consistency across deployments [8] . CI/CD pipelines (AWS CodePipeline, GitLab CI/CD) enable automated testing and fast, reliable software releases [50, 51].. Automated backups with DynamoDB Point-in-Time Recovery (PITR) protect critical data without additional storage costs [52]. Incident management is handled through AWS SNS notifications for real-time alerts, while Route 53 failover ensures continuity in case of outages [53]. This lean operational model minimizes cost while maintaining scalability, reliability, and security, ensuring smooth system lifecycle management [54].

6. Scalability and Adaptability

The architecture scales automatically through AWS Auto Scaling and serverless functions (AWS Lambda), ensuring efficient resource allocation based on demand. Amazon DynamoDB global tables and multi-region deployments enhance global availability, minimizing latency for international users. Event-driven workflows (Amazon EventBridge, Amazon SQS) enable asynchronous processing for high-volume transactions, improving resilience and scalability. Future AI/ML capabilities, such as AWS SageMaker for predictive analytics, can be integrated without disrupting existing services, allowing for real-time business insights. The microservices architecture ensures independent scaling of components, optimizing cost efficiency and enabling seamless feature expansion based on business needs.

7. Development Process and Organization

Development follows an Agile (Scrum) methodology with cross-functional microservices teams, each responsible for independent service ownership. Teams operate in two-week sprints, ensuring iterative development and continuous feedback. CI/CD pipelines (AWS CodePipeline, GitLab CI/CD) enforce automated testing (unit, integration, and security tests) before deployments, ensuring code stability and reliability. Collaboration is streamlined using Jira for project management, Slack for communication, and Confluence for documentation and knowledge sharing, providing a centralized repository for internal Wikis, API documentation, and onboarding materials. This approach balances standardization with innovation, enabling teams to ship features faster while maintaining well-defined interfaces and shared best practices [54–57].

Exhibits

ChatGPT was used to refine the Operations and Lifecycle Management section, enhancing clarity and structure. All technical content was independently researched and validated against official AWS documentation and industry best practices.

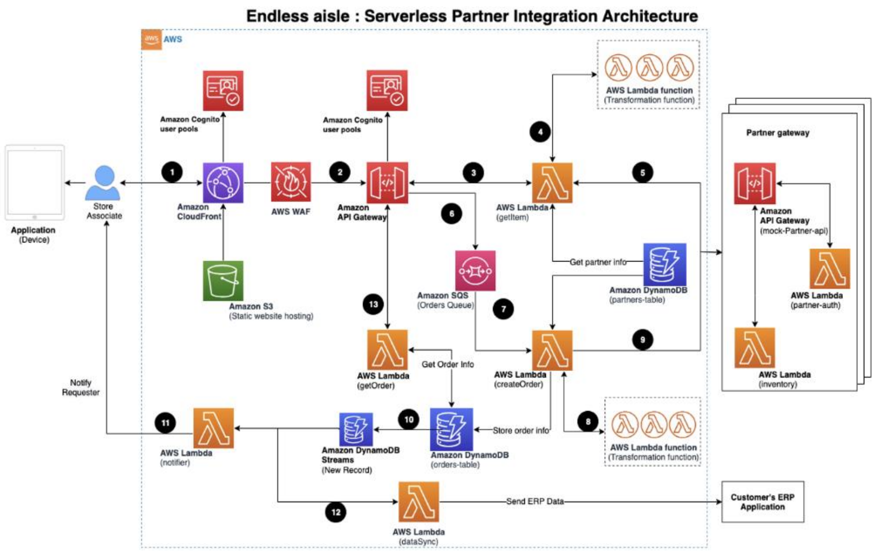

Figure 1 : Microservices & Event-Driven Architecture

This diagram illustrates a serverless architecture where AWS services handle authentication (Cognito, WAF), API management (API Gateway), business logic (Lambda, DynamoDB), and order processing (SQS, EventBridge).

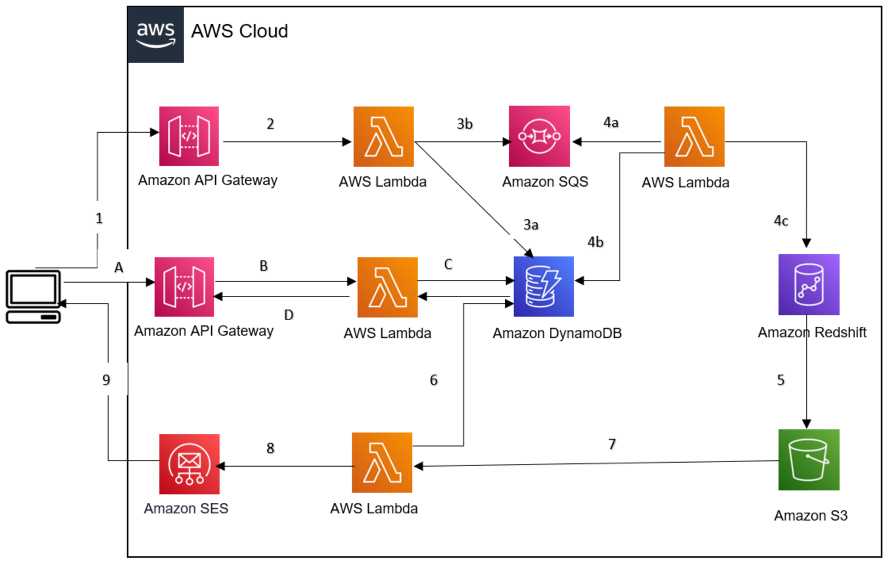

Figure 2: AWS Data Processing Architecture

This diagram shows data flow from API requests to analytics, using Lambda, DynamoDB, SQS, Redshift, and S3 for event-driven processing, data storage, and reporting. Amazon SES handles automated email notifications.

References:

1. AWS Well-Architected - Build secure, efficient cloud applications, https://aws.amazon.com/architecture/well-architected/, last accessed 2025/02/28.
2. Cloud Native Computing Foundation, https://www.cncf.io/, last accessed 2025/02/28.
3. Event-Driven Architecture, https://aws.amazon.com/event-driven-architecture/, last accessed 2025/02/28.
4. Microservices, https://martinfowler.com/articles/microservices.html, last accessed 2025/02/28.
5. Serverless Architectures, https://martinfowler.com/articles/serverless.html, last accessed 2025/02/28.
6. Google Cloud Well-Architected Framework | Cloud Architecture Center, https://cloud.google.com/architecture/framework, last accessed 2025/02/28.
7. Container Orchestration & Management on AWS | Amazon Web Services, https://aws.amazon.com/containers/, last accessed 2025/02/28.
8. Infrastructure As Code Provisioning Tool - AWS CloudFormation - AWS, https://aws.amazon.com/cloudformation/, last accessed 2025/02/28.
9. Amazon S3 - Cloud Object Storage - AWS, https://aws.amazon.com/s3/, last accessed 2025/02/28.
10. CDN Cloud Service - Amazon CloudFront - AWS, https://aws.amazon.com/cloudfront/, last accessed 2025/02/28.
11. Load Balancer - Elastic Load Balancing (ELB) - AWS, https://aws.amazon.com/elasticloadbalancing/, last accessed 2025/02/28.
12. Web App Deployment - AWS Elastic Beanstalk - AWS, https://aws.amazon.com/elasticbeanstalk/, last accessed 2025/02/28.
13. Managed SQL Database - Amazon Relational Database Service (RDS) - AWS, https://aws.amazon.com/rds/, last accessed 2025/02/28.
14. Fast NoSQL Key-Value Database – Amazon DynamoDB – AWS, https://aws.amazon.com/dynamodb/, last accessed 2025/02/28.
15. Serverless Function, FaaS Serverless - AWS Lambda - AWS, https://aws.amazon.com/lambda/, last accessed 2025/02/28.
16. APM Tool - Amazon CloudWatch - AWS, https://aws.amazon.com/cloudwatch/, last accessed 2025/02/28.
17. Gil, L.: Cloud Pricing Comparison: AWS vs. Azure vs. Google Cloud Platform in 2024, https://cast.ai/blog/cloud-pricing-comparison-aws-vs-azure-vs-google-cloud-platform/, last accessed 2025/02/28.
18. AWS Lambda Documentation, https://docs.aws.amazon.com/lambda/, last accessed 2025/02/28.
19. API Management - Amazon API Gateway - AWS, https://aws.amazon.com/api-gateway/, last accessed 2025/02/28.
20. Reimagine possible | Contentstack, https://www.contentstack.com/, last accessed 2025/02/28.
21. Content that takes you everywhere, https://www.contentful.com/, last accessed 2025/02/28.
22. Event Listener - Amazon EventBridge - AWS, https://aws.amazon.com/eventbridge/, last accessed 2025/02/28.

23. Message Queuing Service - Amazon Simple Queue Service - AWS, https://aws.amazon.com/sqs/, last accessed 2025/02/28.
24. DHL Group API Developer Portal | DHL API Developer Portal, https://developer.dhl.com/, last accessed 2025/02/28.
25. Data Stream Processing - Amazon Kinesis - AWS, https://aws.amazon.com/kinesis/, last accessed 2025/02/28.
26. Security at Stripe, https://docs.stripe.com/security, last accessed 2025/02/28.
27. How to integrate a payment gateway into a website | Stripe, https://stripe.com/en-de/resources/more/how-to-integrate-a-payment-gateway-into-a-website, last accessed 2025/02/28.
28. Official PCI Security Standards Council Site, https://www.pcisecuritystandards.org/, last accessed 2025/02/28.
29. Stripe global availability, https://stripe.com/en-de/global, last accessed 2025/02/28.
30. What are Instant Payouts and who is eligible? : Stripe: Help & Support, https://support.stripe.com/questions/what-are-instant-payouts-and-who-is-eligible, last accessed 2025/02/28.
31. Stripe API Reference, https://docs.stripe.com/api, last accessed 2025/02/28.
32. Receive Stripe events in your webhook endpoint, https://docs.stripe.com/webhooks, last accessed 2025/02/28.
33. AWS Key Management Service - AWS Key Management Service, https://docs.aws.amazon.com/kms/latest/developerguide/overview.html, last accessed 2025/02/28.
34. Security in Amazon API Gateway - Amazon API Gateway, https://docs.aws.amazon.com/apigateway/latest/developerguide/security.html, last accessed 2025/02/28.
35. What is Amazon Simple Queue Service? - Amazon Simple Queue Service, https://docs.aws.amazon.com/AWSSimpleQueueService/latest/SQSDeveloperGuide/welcome.html, last accessed 2025/02/28.
36. Cloud Contact Center And Customer Service Software - Amazon Connect - AWS, https://aws.amazon.com/connect/, last accessed 2025/02/28.
37. What is Amazon SES? - Amazon Simple Email Service, https://docs.aws.amazon.com/ses/latest/dg/Welcome.html, last accessed 2025/02/28.
38. What is Amazon SNS? - Amazon Simple Notification Service, https://docs.aws.amazon.com/sns/latest/dg/welcome.html, last accessed 2025/02/28.
39. Netzwelt, V.T.: Salesforce vs AWS: Who is the Better Cloud Service Provider?, https://www.vtnetzwelt.com/salesforce/salesforce-vs-aws-who-is-the-better-cloud-service-provider/, last accessed 2025/02/28.
40. Developer Documentation | Salesforce Developers, https://developer.salesforce.com/docs, last accessed 2025/02/28.
41. Cloud Data Warehouse - Amazon Redshift - AWS, https://aws.amazon.com/redshift/, last accessed 2025/02/28.
42. ETL Service - Serverless Data Integration - AWS Glue - AWS, https://aws.amazon.com/glue/, last accessed 2025/02/28.

43. Business Intelligence Tools - Amazon QuickSight - AWS, https://aws.amazon.com/quicksight/, last accessed 2025/02/28.
44. Unified data, analytics and AI – Amazon SageMaker – AWS, https://aws.amazon.com/sagemaker/, last accessed 2025/02/28.
45. Redshift vs Snowflake vs BigQuery: A Comparison of Three Major Cloud Data Warehouses, https://risingwave.com/blog/redshift-vs-snowflake-vs-bigquery-a-comparison-of-three-major-cloud-data-warehouses/, last accessed 2025/02/28.
46. BigQuery enterprise data warehouse, https://cloud.google.com/bigquery, last accessed 2025/02/28.
47. Access Management- AWS Identity and Access Management (IAM) - AWS, https://aws.amazon.com/iam/, last accessed 2025/02/28.
48. Web Application Firewall, Web API Protection - AWS WAF - AWS, https://aws.amazon.com/waf/, last accessed 2025/02/28.
49. Application Scaling - AWS Auto Scaling - AWS, https://aws.amazon.com/autoscaling/, last accessed 2025/02/28.
50. What is CI/CD?, https://about.gitlab.com/topics/ci-cd/, last accessed 2025/02/28.
51. CI/CD Pipeline - AWS CodePipeline - AWS, https://aws.amazon.com/codepipeline/, last accessed 2025/02/28.
52. Amazon DynamoDB point-in-time recovery (PITR), https://aws.amazon.com/dynamodb/pitr/, last accessed 2025/02/28.
53. DNS Service - Amazon Route 53 - AWS, https://aws.amazon.com/route53/, last accessed 2025/02/28.
54. ChatGPT, https://chatgpt.com, last accessed 2025/02/28.
55. Slack: AI Work Management & Productivity Tools, https://slack.com, last accessed 2025/02/28.
56. Confluence | Your Remote-Friendly Team Workspace | Atlassian, https://www.atlassian.com/software/confluence, last accessed 2025/02/28.
57. Jira | Issue & Project Tracking Software | Atlassian, https://www.atlassian.com/software/jira, last accessed 2025/02/28.