COS20019 Capstone Project

Ioannis Iliadis – 104010553
Chayan Kapoor – 104202680

22/10/2023

Table of Contents

I. Introduction 3
II. Architecture Overview 3
III. Compute 4
A. AWS Elastic Container Service (ECS) 4
B. AWS Lambda 4
IV. Storage 5
V. Database 5
VI. Networking 5
A. Amazon Route 53 5
B. Amazon CloudFront 5
VII. Security 6
A. Amazon Virtual Private Cloud (VPC) 6
B. Security Groups 6
1) Application Load Balancer SG: 6
2) ECS Container SG 6
3) DAX Node SG 6
C. Amazon Identity and Acess Management (IAM) 6
D. Amazon Cognito 6
E. Amazon Web Application Firewall (WAF) 6
F. AWS Shield 6
VIII. Decoupling 6
A. Amazon Simple Notification Service (SNS) 6
B. Amazon Simple Queue Servcie (SQS) 7
IX. Deployment Orchestration 7
X. Monitoring and Logging 7
A. Amazon CloudWatch 7
B. Amazon Cost Explorer 7
XI. Video Transcoding 8
XII. Fullfilment of Business Scenario 8
XIII. Justification 8
A. Performance 8
B. Scalability 8
C. Extensibility and Decoupling 9
D. Reliability 9
E. Security 9

F. Cost 9
XIV. Alternative Solutions 9
A. EC2 vs ECS 9
B. RDS vs DynamoDB 9
C. 2-Tier vs 3-Tier Architecture 10
D. Elemental MediaConvert vs Elastic transcoder 10
XV. Summary 10
XVII. Work Distribution 10
A. Ioannis 10
B. Chayan 11
XVIII. References 11

I. Introduction

The goal of this project is to research and develop a new highly available cloud-based architecture based on the photo album website. The new architecture will be designed in accordance with the customer's business scenario and will aim to resolve existing problems, meet new requirements, and be extensible for the addition of new functionality in the future.

In the business scenario, it is stated that the website has been experiencing a high amount of traffic as a result of the increasing demand. A solution is requested that will incorporate managed AWS services and serverless computing. The slow and costly relational database will need to be addressed and global response times have to be improved. Furthermore, the system will be expected to handle video media in the future.

In this proposed solution, out team has developed a robust and scalable architecture by leveraging managed services and serverless computing capabilities. A justification of the design and the rationale between each service will be explained in detail in this document.

II. Architecture Overview

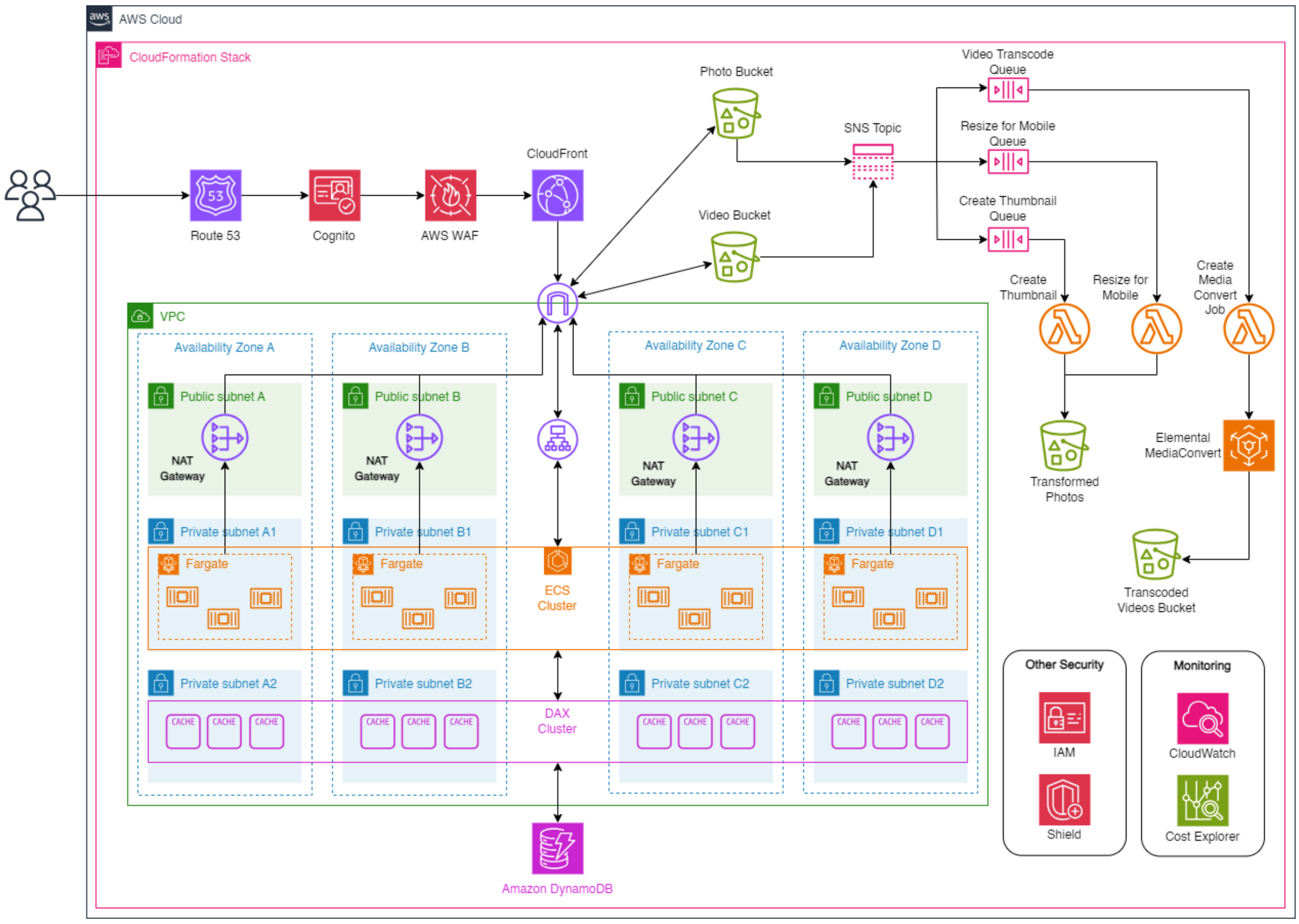

Figure 1 - Diagram Architecture

Below is a brief overview of the components and their functionality in the architecture.

1. AWS CloudFormation will be used for automating multi-region deployment of the application. The main reasons for going multi-region are availability, continuity, and reduced latency. Since CloudFormation is an infrastructure as code service, it allows for fast and consistent deployments.

2. Amazon Route 53 will provide DNS management and DNS routing. Latency-based routing will be used for directing users to different regions where the application has been deployed, that offer the lowest latency.

3. Once users access the website, Amazon Cognito will be used for user sign-up and sign-in.

4. AWS Web Application Firewall (WAF) will be integrated with CloudFront for providing security, blocking unwanted traffic, and protecting against common web attacks such as SQL injections and XSS attacks.

5. AWS Shield advanced option will provide a high level of protection against DDoS and volumetric bots attacks.

6. Amazon CloudFront will be used for edge caching and delivering content with low latency. The application load balancer will be used as the origin domain for the distribution.

7. The VPC will consist of four availability zones for high availability. A NAT gateway will be placed in the public subnet of each AZ, also for high availability. An internet gateway will give internet access to the resources in the VPC.

8. Incoming traffic will be evenly distributed across ECS containers in different AZs with an application load balancer.

9. For compute, an ECS cluster will be used to handle all the websites logic and process user requests. ECS will utilize Fargate as the serverless compute engine, so no servers will have to be managed by the client.

10. DynamoDB will be used as the database for the application. Non-relational databases are known to have much faster read speeds that relational databases and can process large amounts of data. DynamoDB is managed and serverless, so no instances will have to be provisioned. It also offers a feature for fully managed global tables when deploying multi-region architectures, which is very convenient in this case. Furthermore, a DAX cluster consisting of four nodes will be deployed in the private subnets of the VPC. That is one node per AZ for high availability. It will act as a caching service for the DynamoDB tables, further increasing speeds and lowering latencies for frequent requests.

11. For storage, S3 standard buckets will be used for everything. There will be one bucket for photos and another for videos. There will be separate buckets for transformed photos and transcoded videos.

12. When photos and videos are uploaded to the buckets, S3 will automatically generate an event and post it at an SNS topic. The topic will then broadcast the event to SQS queues, based on the media type. An SQS queue will then invoke a lambda function that can either resize a photo or create a media convert job for video transcoding with Elemental MediaConvert. Transformed media will be stored in new buckets.

13. IAM roles will be configured to give resources the permissions they need to access other resources.

14. Amazon CloudWatch will be used for monitoring and logging.

15. AWS Cost Explorer will be used for monitoring and managing the costs.

III. Compute

A. AWS Elastic Container Service (ECS)

AWS ECS is a fully managed compute service used for the orchestration of Docker containers. A container is a lightweight executable software package that consists of all the code and dependencies that an application needs to run. This allows them to be deployed very efficiently, have better portability, and provide much better resource utilization. As a managed service, ECS will automatically scale up or down based on the traffic, effectively reducing operational costs.

In conjunction with ECS, Fargate will be used as the serverless compute engine, which removes the need of provisioning, managing, and maintaining the underlying EC2 instances that run the containerized applications.

Another reason why ECS was chosen for this design, is that it allows for a microservices approach. Different processes can be handled by creating new services in the ECS cluster and are run and scaled independently of each other. In this manner, the processes are decoupled, and extensibility is achieved, since new services can be created in the future that implement new functionality.

For the client's application, a new cluster will be created with services than span across four availability zones for high availability. User traffic will be distributed to the cluster via an application load balancer. The cluster will be responsible for serving users, updating the DynamoDB database, and storing photos and videos to S3.

In summary, business requirements are satisfied with ECS and Fargate. As managed and serverless solution, this compute option enables high scalability, decoupling, and extensibility.

B. AWS Lambda

Lambda is a serverless compute service used for running code without having to manage the underneath infrastructure. The code is placed in Lambda functions and is executed when invoked by events. Lambda is a highly available and redundant service that automatically scales up or down based on the number of concurrent invocations.

For the client's application, Lambda functions will be used to resize photos or create video transcoding jobs for Elemental Media Convert.

IV. Storage

Amazon S3 standard will be used for storing all photos and videos. S3 provides highly available object level storage with 11 9s of durability. This is achieved by replicating data across multiple availability zones and performing health checks to verify integrity of stored objects.

In this application, appropriate policies will be placed for each bucket following the least privilege rule. Initially, four buckets will be created:

- Photo Bucket: Used for storing user uploaded photos.
- Transformed Photos Bucket: Used for storing photo thumbnails or other transformed photos.
- Videos Bucket: Used for storing user uploaded videos.
- Transcoded Videos Bucket: Used for storing transcoded videos.

Since the application will be deployed in multiple regions, the Multi-Region Access Points feature will be used for replicating bucket data across regions while providing a single global endpoint for accessing the objects. Utilizing the AWS global infrastructure, this feature will help reduce the applications latency.

In summary, the business scenario's storage requirements are met with S3, by providing highly available storage and reducing latency with multi-region access points.

V. Database

Amazon DynamoDB is a fully managed, serverless NoSQL database service for any scale. The reason for going with a non-relational database compared with the relation RDS service, is that non-relational databases are known to offer much higher performance, speed, and scalability. DynamoDB also provides high extensibility since the document schema is dynamic and can easily be modified, unlike relational schemas.

For increased performance, the DynamoDB Accelerator (DAX) service will provide caching for the tables. This service is fully managed, and AWS claims up to 10 times performance improvement. DAX runs within a VPC, so a DAX cluster will be created and will span across four availability zones. The cluster will consist of four nodes, one in each availability zone. A node runs an instance of DAX software and stores the cached data.

For global deployment of the architecture, DynamoDB offers multi-region synchronous data replication with global tables. Global tables are fully managed and can provide better response times and lower latency for users.

For increased reliability, DynamoDB point-in-time recovery (PITR) will be enabled to allow for continuous backup of table data. The backups are encrypted for high security and can be easily restored at any point in the case of data loss or system failures.

In summary, ditching RDS and migrating to DynamoDB enables to application to reach might higher speeds. Using DAX for caching further increases speed, and global tables can help lower latency.

VI. Networking

A. *Amazon Route 53*

Route 53 serves as a highly scalable and reliable Domain Name System (DNS) web service designed to effectively route end-users to internet applications, and in this specific case, to a CloudFront distribution. By utilizing the latency-based routing feature of Route 53, when a user makes a request to the client's application, the request is not merely directed to any server. Instead, it's intelligently directed to the CloudFront distribution that can deliver the content in the least amount of time. As the application has gained widespread adoption, users from different geographical locations access it, and this is where Route 53 shines. It ensures that a user's request is resolved to the nearest CloudFront distribution, thereby significantly reducing the response time, which is especially beneficial for users situated far from the primary server location in Australia.

B. *Amazon CloudFront*

Amazon CloudFront stands out as AWS's content delivery network (CDN) service. It caches content in multiple edge locations spread globally to ensure rapid content delivery to users irrespective of their geographical location. Within the context of the client's application, as users from various global locations access the content, CloudFront delivers this content from the nearest edge location. So, if a European user wants to view a photo from the application, they don't have to wait for the content to load all the way from the primary server in Australia. Instead, CloudFront provides them the content from an edge location situated in Europe, ensuring faster data retrieval. Beyond static content, CloudFront, with its optimized connections between edge locations and the original servers, expedites the delivery of dynamic content, ensuring that even non-cached content reaches the user faster. This seamless integration between Route 53 and CloudFront ensures that users worldwide enjoy a swift and streamlined experience when accessing the client's application.

VII. Security

A. Amazon Virtual Private Cloud (VPC)

A VPC is an isolated section on the cloud where resources can be securely deployed. It allows for CIDR block selection, subnet creation, and configuration of route tables.

In this architecture, a VPC will consists of four AZs. Each AZ will have a public subnet and two private subnets. An internet gateway will be attached to allow internet access. NAT gateways will be created in the public subnets for letting resources in private subnets access the internet. There will be one NAT per public subnet for high availability. There will be one public route table consisting of a local route and an internet route. There will be four private route tables, each one consisting of a local route and a route to the NAT in its availability zone.

In summary, VPC will allow for securing the application's resources in a secure and isolated section of the cloud.

B. Security Groups

Security groups provide virtual firewall protection to resources within a VPC. These work at the instance level. It allows for configuring inbound and outbound rules for controlling traffic flow. The following rules will be configured for this architecture:

1) Application Load Balancer SG:

- HTTP: All traffic permitted.
- HTTPS: All traffic permitted.

2) ECS Container SG

- All TCP: From the load balancer SG.

3) DAX Node SG

- Custom TPC: Port 9111 from ECS Container SG.

C. Amazon Identity and Acess Management (IAM)

IAM is a service that allows for managing which users or resources have access to AWS resources. In this architecture, the ECS containers and Lambda functions will be provided with the necessary roles for accessing the S3 buckets.

D. Amazon Cognito

Within the framework of the client's application, Amazon Cognito plays a pivotal role in managing user identities. When a user interacts with the application, they engage in a registration or sign-in process. Amazon Cognito handles this authentication, ensuring that user credentials are validated securely. Once authenticated, users are granted token-based access, allowing them to seamlessly navigate through the application. Every time a user returns or uses a different device, Cognito ensures their preferences and data are consistently available, offering a cohesive experience throughout the application.

E. Amazon Web Application Firewall (WAF)

For the client's application, security stands paramount, especially given its expansive user base. AWS WAF acts as the primary line of defense, inspecting incoming traffic to the application. As users interact and engage, any data they submit, from comments to media uploads, is meticulously screened by AWS WAF. This ensures that malicious payloads, hidden in user inputs or requests, are detected, and blocked. The application, thus, remains fortified, ensuring that users can trust the platform with their interactions and data.

F. AWS Shield

In the context of the client's application, AWS Shield serves as the bulwark against external malicious threats, particularly DDoS attacks. Given the platform's popularity and high traffic, it becomes an attractive target for adversaries. AWS Shield's advanced capabilities monitor the application's incoming traffic in real-time. If a sudden surge of requests is detected, Shield swiftly discerns if it's a genuine spike or a DDoS attack, ensuring that legitimate user requests are serviced while malicious ones are stymied. Users, therefore, experience uninterrupted access even in the face of cyber onslaughts.

VIII. Decoupling

A. Amazon Simple Notification Service (SNS)

SNS is a publish/subscribe messaging service. In the pub/sub model, a topic is defined as a logical access point where publishers can post messages that will be broadcast to subscribers. This message communication system is asynchronous and decoupled since a topic is used rather than a point-to-point connection between a publisher and every other subscriber. Furthermore, SNS is completely managed and provides automatic scaling, retry handling, and failure logic.

In the proposed architecture, an SNS topic is created where the S3 buckets will publish notifications when an image or video is uploaded. The topic will be configured with an SNS filter policy that will push the notification to the appropriate SQS processing queue depending on the media format. The queues will be polled by Lambda functions which will be executed after the message is extracted from a queue.

In summary, SNS helped develop a decoupled architecture and allowed for future extensibility, since new publishers and subscribers can be added to the topic when implementing new functionality.

B. Amazon Simple Queue Servcie (SQS)

SQS is a fully managed messaging queuing service used for communication between distributed systems and microservices. It can be used for decoupling directly connected components. This allows for achieving an architecture where components don't directly depend on each other. As a result, when modifications need to be made or new functionality needs to be implemented to the architecture, there will be no impact to the rest of the components.

In the proposed architecture, SQS queues will be used for asynchronous processing of messages between the SNS topic and the lambda functions. Each queue will be associated with a different consumer Lambda function based on the task. For example, one queue for creating thumbnails, another for creating video transcoding jobs. This allows for managing different components of the architecture independently.

In summary, SQS used in conjunction with SNS allowed for further decoupling, since each queue and its associated consumer can focus on a specific task.

IX. Deployment Orchestration

Amazon CloudFormation is a service that provides the tools for automating the process of deployment and resource management. It leverages Infrastructure as Code (IaC), which is a term for provisioning entire cloud architectures by defining them in a JSON or YAML template. The benefits of IaC include:

- Deployments of architectures in minutes.
- The configurations are consistent across deployments.
- Version control allow for roll back and modification features.

One of the problems that was faced in the original architecture was slow global response times due to the worldwide traffic uptake. To address that problem, a multi-region deployment of the architecture will be automated with CloudFront. That, in conjunction with CloudFront and other services will aim to greatly reduce the latency for users around the world.

X. Monitoring and Logging

A. Amazon CloudWatch

CloudWatch is a monitoring service that provides system-wide visibility into the performance, health, and status of AWS services or on-premises servers. Metric data and logs are collected from resources and can be viewed in real time through the dashboard using powerful visualization tools. Additionality, resource management can be automated by setting up alarms that will trigger and perform certain actions when a predefined threshold has been breached.

For the client's application, CloudWatch will allow:

- Monitoring ECS container insights and diagnostic information for fault detection and resource utilization.
- Monitoring access logs from S3 buckets and CloudFront for detecting unauthorized access attempts.
- Monitoring DynamoDB metrics like read and write capacity or latency.
- Monitoring invocation count, duration count, and errors of Lambda functions.
- Monitoring VPC network traffic.

In summary, CloudWatch will provide the necessary monitoring capabilities that will allow insights into the application's overall health, performance, and security.

B. Amazon Cost Explorer

In managing the client's application, it's vital to have a clear grasp of the operational costs. AWS Cost Explorer, in this context, becomes an essential tool for financial monitoring. As the application scales, serving more users and processing more data, the underlying AWS services' consumption increases. Cost Explorer helps the client to track these expenses, breaking down costs by service, user interaction, and even specific features. This granular insight ensures that the application remains financially optimized, allowing for funds to be allocated judiciously and ensuring the sustainability of the platform.

XI. Video Transcoding

Amazon Elemental MediaConvert is a fast video processing service that allows for transcoding video files into different formats. It provides advanced features such as HDR video processing, caption support, audio processing, etc.

For the client's application, MediaConvert will be used to transcode uploaded videos into many various formats that will allow for playback on different devices and platforms. This process will be automated with Lambda functions that will create transcoding jobs when media is uploaded to the website. Furthermore, this service is fully managed and automatically handles resource provisioning and scaling.

In summary, the business scenario's video handling requirement is met with Elemental MediaConvert, which will offer high-quality and high-speed video transcoding capabilities for the application.

XII. Fullfilment of Business Scenario

In designing this architecture, our objective was to resolve existing problems and meet new requirements based on the company's business scenario. Each service that was picked serves a specific goal and focused on achieving the following milestones:

1. **Managed services**: Having managed services minimizes the need for in-house systems administration, which allows the company to focus on the application's deployment and innovation. All the services used in this architecture are fully managed, so the underlying infrastructure will be handled by AWS. This will provide features such as automatic scaling, high availability, patching and updates, etc.
2. **Accommodating rapid growth**: To keep up with the rapid growth of the company, our proposed architecture implements the following:
    a. ECS will be used instead of EC2, since containers are fully managed with Fargate, can be deployed, and scaled faster, and have better resource utilization. New functionality can be easily implemented by creating new tasks and services in the ECS cluster, which promotes a decoupled architecture and future extensibility.
    b. Serverless computing is used which scales in accordance with the increased traffic.
    c. CloudFormation is used for quick deployment of the architecture in new regions.
    d. DynamoDB is designed to provide high performance at any scale.
3. **Going serverless**: To meet the client's requirements of going serverless, Fargate is used as the serverless compute engine for container orchestration of the ECS cluster. Lambda functions are used to respond to events and handling media conversion. DynamoDB is also serverless, so no database instances will have to be managed.
4. **Addressing RDS issues**: To address the slow and costly RDS database, DynamoDB was the new choice for the architecture. DynamoDB is a NoSQL service which offers higher read and write speeds than SQL databases. It also automatically scales to meet demand of any magnitude. In addition, for simple data sets, DynamoDB is more cost-efficient.
5. **Reducing global response times**: To adapt to the worldwide traffic uptake of the application, global response times will be improved by the following:
    a. Multi-region deployment with CloudFormation will reduce latency to different parts of the world.
    b. CloudFront will be used as the CDN for caching content at edge locations, which reduces global response times.
    c. Route 53 will utilize latency-based routing as the routing option.
6. **Handling video media**: To meet the requirement for handling video media, Elemental MediaConvert will be used for fast, high quality video transcoding.
7. **Decoupled architecture**: Using an SNS topic enables for configuring new publishers and subscribers in the future when new types of media formats will need to be handled by the application. Additionally, new SQS queues can be created to handle new tasks. This decoupling pattern allows for future functionality to be implemented without having to modify existing components.

XIII. Justification

A. *Performance*

- CloudFront used for caching content and lowering global response times.
- DynamoDB offers greater speeds than RDS.
- Dax increases DynamoDB performance with in-memory caching.
- Route 53 latency-based routing lowers response times.
- Multi-region deployment with CloudFront reduces global response times.
- Load balancer evenly distributes traffic.

B. *Scalability*

- Fargate automatically scales the ECS cluster.
- DynamoDB is fully managed and scales up to meet demand of any magnitude.
- Lambda scales automatically based on the number of concurrent invocations.

- SNS and SQS are fully managed and are designed to scale to accommodate large numbers of messages.

C. Extensibility and Decoupling

- SNS topic facilitates the implementation of new functionality in the future.
- SQS queues act as intermediaries between components, which allows them to operate independently and asynchronously.
- New tasks can be easily implemented with containers in the ECS cluster.

D. Reliability

- Load balancer routes traffic only to healthy instances.
- Resources as spread across four availability zones for high availability.
- S3 provides 11 9s of data durability.
- Multi-region deployment offers high availability when a meteor wipes out a region.
- DynamoDB point-in-time recovery continuously backs up data.
- CloudWatch can be used to monitor the health of resources.

E. Security

- Resources are deployed in private subnets in VPC.
- Security groups are used at instance level.
- Amazon Shield offers DDoS protection.
- Amazon WAF acts as a firewall for CloudFront.
- Amazon Cognito is used for safe user authentication.
- S3 policy blocks public access and follows least privilege rule.
- IAM roles used where necessary and follow least privilege rule.
- CloudWatch logs can be used to monitor access patterns.
- For encryption at rest, S3 automatically encrypts all objects.
- DynamoDB automatically encrypts data at rest.
- An SSL certificate can be created and used with CloudFront for encrypting data in transit.

F. Cost

- Serverless and managed services will be able to scale down when there is limited traffic.
- Amazon Cost Explorer is used to analyze costs.

XIV. Alternative Solutions

A. EC2 vs ECS

ECS was picked over EC2 due to the following reasons:

- A managed service was required by the client.
- For running a serverless containerized application, Fargate will manage the underlying servers and will scale services in the cluster automatically.
- Containers can be deployed and scaled faster and offer better resource utilization.
- Different processes can run on different containers, which promotes decoupling.
- New services or tasks can be created in the ECS cluster to implement new functionality.
- Different services can be scaled independently based on demand.

However, since Fargate will be used to manage the servers that run the containers, this option is more costly, but it is still worth it due to the various benefits it provides.

Pricing example

With a general purpose m5.xlarge instance, the EC2 hourly pricing is at $0.192, and for Fargate it is $0.233. This is 21% price difference. For a single task running for a month (730 hours), the total price would be:

- 730 x $0.192 = $140 per month for EC2.
- 730 x $0.233 = $170 per month for Fargate.

B. RDS vs DynamoDB

DynamoDB was chosen over RDS due to the following reasons:

- DynamoDB offers greater read and write speeds.
- DynamoDB is serverless and can scale to meet demand at any scale.
- It is more cost-efficient for simple data models, which is the case with this application.
- Global tables are convenient to use.
- Data is automatically replicated in different AZs for high availability.

However, since the existing data is in tables, it will have to be converted to documents. Despite this, DynamoDB will be able to facilitate the future growth of the applications.

Pricing Example
Consider the following RDS instance configuration:

- On demand instance.
- Instance type: db.m4.4xlarge with 64gb of memory.
- RDS Proxy enabled.
- 30gb of storage and 30gb of backup storage.

With the above configuration, if the RDS instance was to run for a month at 100% utilization, the cost would be $2230.4.

Consider the following DynamoDB configuration:

- Standard table class.
- On-demand capacity.
- 4x DAX nodes with instance type r5.large.
- 30gb point-in-time recovery storage.

With the above configuration, for 100 million read and 100 million write operations, the cost would be $1408.1.

C. *2-Tier vs 3-Tier Architecture*

There was no need to implement a 3-Tier architecture since the presentation tier logic and the application tier logic can be implemented as different services in the EC2 cluster and be scaled independently with Fargate.

D. *Elemental MediaConvert vs Elastic transcoder*

Elemental MediaConvert was chosen over Elastic transcoder because:

- It provides more advanced transcoding capabilities and features.
- It does so at half the price.

Pricing example

- 20-minute source file in Mumbai region transcoded to SD output will cost 20 x $0.015 = $0.30.
- 20-minute source file in Mumbai region transcoded with AVC codec to SD will cost 20 x $0.0075 = $0.15.

XV. Summary

To summarize, this project aimed to develop a cloud architecture that would help meet a company's business needs. As solution architects, we had to analyze a business scenario and get insights about the objectives, challenges, and requirements that needed to be satisfied. Ultimately, after researching and considering different services, we proposed a solution that would facilitate the company's future growth and enhance its overall operational efficiency, scalability, security, and cost-effectiveness.

XVII. Work Distribution

A. *Ioannis*

- Introduction
- Architecture diagram
- Architecture overview
- UML diagrams
- Compute section
- Storage section
- Database section

- Decoupling section
- Deployment orchestration section
- Amazon CloudWatch
- Video transcoding section
- Fulfillment of business scenario section
- Justification section
- Alternative solutions section
- Summary

B. Chayan

- Networking section
- Security section
- Amazon Cost Explorer

XVIII. References

[10] AWS, Amazon Cost Explorer, https://aws.amazon.com/aws-cost-management/aws-cost-explorer/ (accessed Oct. 22, 2023).

[11] AWS, Amazon Shield, https://aws.amazon.com/shield/ (accessed Oct. 22, 2023).

[12] AWS, Amazon Web Application Firewall (WAF), https://aws.amazon.com/waf/ (accessed Oct. 22, 2023).

[13] AWS, “Amazon Cognito,” Amazon, https://aws.amazon.com/cognito/ (accessed Oct. 22, 2023).

[14] AWS, Low-latency content delivery network (CDN) - Amazon Cloudfront, https://aws.amazon.com/cloudfront/ (accessed Oct. 22, 2023).

[15] AWS, Amazon Route 53 | DNS service | AWS, https://aws.amazon.com/route53/ (accessed Oct. 22, 2023).

[1] AWS, Using server-side encryption with Amazon S3 managed keys (SSE-S3), https://docs.aws.amazon.com/AmazonS3/latest/userguide/UsingServerSideEncryption.html (accessed Oct. 22, 2023).

[2] AWS, Amazon S3, https://aws.amazon.com/s3/ (accessed Oct. 22, 2023).

[3] AWS, AWS serverless Multi-Tier Architectures with amazon API gateway and AWS ..., https://docs.aws.amazon.com/pdfs/whitepapers/latest/serverless-multi-tier-architectures-api-gateway-lambda/serverless-multi-tier-architectures-api-gateway-lambda.pdf (accessed Oct. 22, 2023).

[4] AWS, AWS Lambda, https://docs.aws.amazon.com/lambda/latest/dg/welcome.html (accessed Oct. 22, 2023).

[5] Yifat Perry, “AWS ECS IN DEPTH: Architecture and deployment options,” NetApp BlueXP, https://bluexp.netapp.com/blog/aws-cvo-blg-aws-ecs-in-depth-architecture-and-deployment-options (accessed Oct. 22, 2023).

[6] Ranadheer raju, “ECS-(Amazon Elastic Container Service),” Medium, https://medium.com/@ranadheerraju11/ecs-amazon-elastic-container-service-ebbb4be7b7dd (accessed Oct. 22, 2023).

[7] I. Rashad, “Modern Web Apps Architecture with AWS Fargate,” Imranarshad.com, https://imranarshad.com/modern-architecture-with-fargate/ (accessed Oct. 22, 2023).

[8] AWS, Amazon Elastic Container Service (ECS), https://aws.amazon.com/ecs/ (accessed Oct. 22, 2023).

[9] AWS, What is Amazon Elastic Container Service? https://docs.aws.amazon.com/AmazonECS/latest/developerguide/Welcome.html (accessed Oct. 22, 2023).

[16] C. Achinga, “Amazon RDS VS DynamoDB: 12 differences you should know,” Cloud Academy, https://cloudacademy.com/blog/amazon-rds-vs-dynamodb-12-differences/ (accessed Oct. 22, 2023).

[17] AWS, Global tables - multi-region replication for dynamodb, https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/GlobalTables.html (accessed Oct. 22, 2023).

[18] Alex DeBrie, "The what, why, and when of single-table design with dynamodb: Debrie advisory," DeBrie Advisory RSS, https://www.alexdebrie.com/posts/dynamodb-single-table/ (accessed Oct. 22, 2023).

[19] AWS, A walkthrough of the Amazon DynamoDB Accelerator console, https://aws.amazon.com/blogs/database/a-walkthrough-of-the-amazon-dynamodb-accelerator-console-part-2/ (accessed Oct. 22, 2023).

[20] Amazon DynamoDB Accelerator (DAX), https://aws.amazon.com/dynamodb/dax/ (accessed Oct. 22, 2023).

[21] AWS, DynamoDB: Everything you need to know about Amazon Web Service's NoSQL database, https://aws.amazon.com/dynamodb/ (accessed Oct. 22, 2023).

[22] A. Lovan, Using latency-based routing with Amazon CloudFront for a multi-Region active-active architecture, https://aws.amazon.com/blogs/networking-and-content-delivery/latency-based-routing-leveraging-amazon-cloudfront-for-a-multi-region-active-active-architecture/ (accessed Oct. 23, 2023).

[23] morjoan, "Automating mediaconvert jobs with lambda," GitHub, https://github.com/aws-samples/aws-media-services-simple-vod-workflow/blob/master/7-MediaConvertJobLambda/README.md (accessed Oct. 23, 2023).

[24] AWS, DynamoDB Point-in-time Recovery, https://aws.amazon.com/dynamodb/pitr/ (accessed Oct. 23, 2023).

[25] A. Joshi, "AWS MediaConvert: The Powerful Alternative to Elastic transcoder," CloudThat Resources, https://www.cloudthat.com/resources/blog/aws-mediaconvert-the-powerful-alternative-to-elastic-transcoder (accessed Oct. 23, 2023).

[26] AWS, Video transcoding – AWS elemental MediaConvert, https://aws.amazon.com/mediaconvert/ (accessed Oct. 22, 2023).

[27] T. Schmidt, "SNS to SQS: Fanout topics to queues," AWS Fundamentals, https://blog.awsfundamentals.com/amazon-sns-to-sqs (accessed Oct. 23, 2023).

[28] AWS, Amazon SQS - Fully managed message queuing , https://aws.amazon.com/sqs/ (accessed Oct. 22, 2023).

[29] AWS, Amazon SNS, https://aws.amazon.com/sns/ (accessed Oct. 23, 2023).

[30] G. Tizatto, Amazon RDS vs. Amazon dynamodb: Everything you need to know, https://www.missioncloud.com/blog/amazon-rds-vs-amazon-dynamodb-everything-you-need-to-know (accessed Oct. 22, 2023).

[31] Red Hat, What is infrastructure as code (IAC)?, https://www.redhat.com/en/topics/automation/what-is-infrastructure-as-code-iac (accessed Oct. 23, 2023).

[32] L. Vanthillo, "How to fan-out to different SQS queues using SNS message filtering," Medium, https://betterprogramming.pub/how-to-fan-out-to-different-sqs-queues-using-sns-message-filtering-84cd23ed9d07 (accessed Oct. 23, 2023).

[33] AWS, Containers and Serverless Computing, https://aws.amazon.com/government-education/containers-serverless/ (accessed Oct. 23, 2023).

[34] AWS, Lambda function scaling, https://docs.aws.amazon.com/lambda/latest/dg/lambda-concurrency.html (accessed Oct. 23, 2023).

[35] C. Dekate, "How does Amazon SNS work?," Analytics Vidhya, https://www.analyticsvidhya.com/blog/2022/08/how-does-amazon-sns-work/ (accessed Oct. 23, 2023).

[36] AWS, Web application hosting in the AWS cloud - d1.awsstatic.com, https://d1.awsstatic.com/whitepapers/aws-web-hosting-best-practices.pdf (accessed Oct. 22, 2023).

[37] N. Apicella, “DynamoDB or Aurora/RDS? should we always use dynamodb?,” DEV Community, https://dev.to/napicella/dynamodb-or-aurora-rds-should-we-always-use-dynamodb-51d5 (accessed Oct. 23, 2023).

[38] H. Dhaduk, “AWS fargate vs EC2 pricing comparison: Who wins the pricing war?,” Simform, https://www.simform.com/blog/fargate-vs-ec2-pricing/ (accessed Oct. 23, 2023).

[39] Docker, “What is a container?,” Docker, https://www.docker.com/resources/what-container/ (accessed Oct. 23, 2023).