Cloud Computing Architecture - Assignment 3

Serverless /Event -driven Architectural Design Report

Thenura Dulnath Kuruppuarachchi
ID: 103512993
103512993@student.swin.edu.au

Janaka Pruthuvi Muthunayake
ID: 104315180
104315180@student.swin.edu.au

Niketha Dewmini Thota Hewage
ID: 104057903
104057903@student.swin.edu.au

***abstract*- In response to rising demand and shifting requirements, the work required creating an architecture concept for an improved Photo Album application hosted on Amazon Web Services (AWS). The suggested design combines several AWS services, such as Amazon S3, AWS Lambda, Amazon DynamoDB, Amazon CloudFront, Amazon SQS, and Amazon Rekognition, to produce a scalable, trustworthy, and reasonably priced solution. The architecture takes advantage of serverless computing, NoSQL databases, queuing systems, and international content delivery to support both media upload and processing scenarios. This thorough architectural design supports the application's flexibility and reactivity in a changing digital environment while keeping financial constraints in mind.**

1. Introduction

This architectural project will focus on creating the design for an improved version of the photo album application, in response to the growing demand and changing requirements. As the design hosted on the AWS platform there are plenty of services which can be used to enhance the sustainability of the design.

- Handling the rising demand: as mentioned, the statics of the company show that the current situation of the demand seems to be doubling every six months. So, the upcoming user data will be accommodated through this architectural design. Also, the focus on scalability and reliability is a major part of this project to meet the demand of the application.

- Integration of the process: as instructed, to build an effective and scalable environment, there will be using a several kinds of AWS services such as Amazon S3, Lambda functions, DynamoDB services. Using all these services the environment will be enhanced for handle the companies' requirements. In this step we will also be forcing on the media uploading and processing stage.

- Financial boundaries: as a growing company there will be a necessity to maintain, optimize the expenses and maintain an efficient utilization of resources within their budget.

2. Architectural Diagram

The below architectural diagram shows that the final architectural solution, as well as all AWS services used for this project. Also, it includes their interactions and those are detailed as well to get a better understanding. (figure 1)

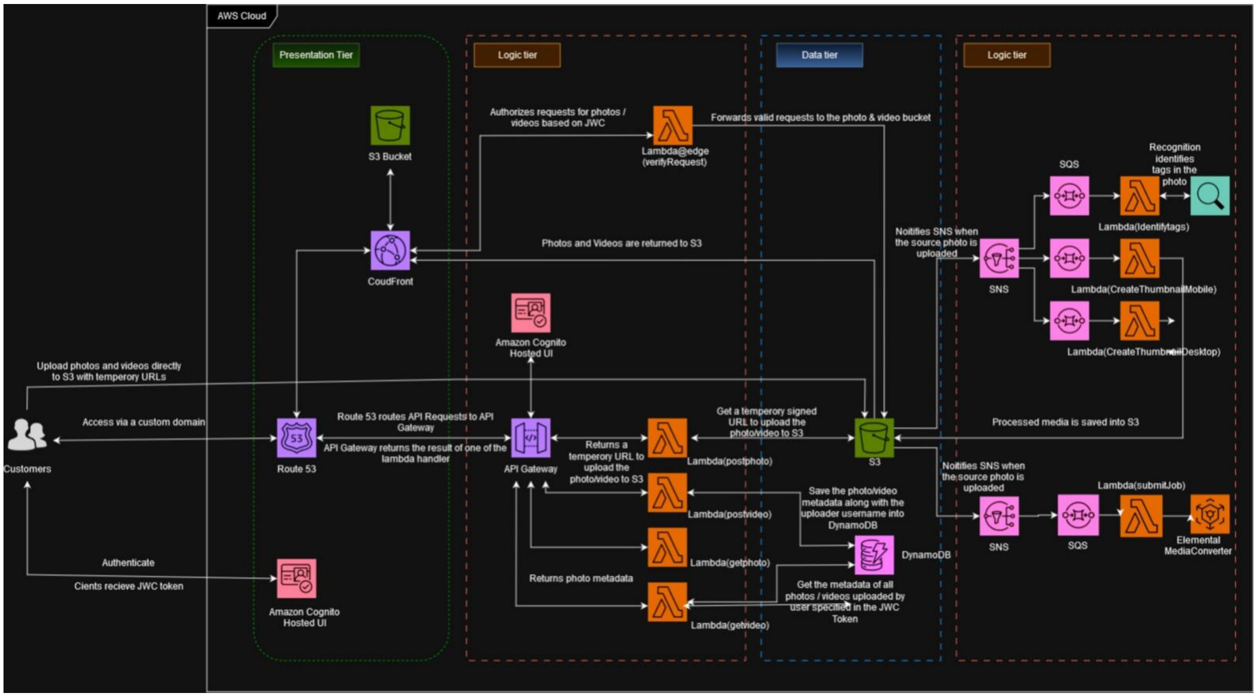

AWS Cloud
Presentation Tier
Logic tier
Data tier
Logic tier
Authorizes requests for photos / videos based on JWC
Lambda@edge (verifyRequest)
Forwards valid requests to the photo & video bucket
S3 Bucket
CoudFront
Photos and Videos are returned to S3
Amazon Cognito Hosted UI
Upload photos and videos directly to S3 with temperory URLs
Access via a custom domain
Customers
Route 53
Route 53 routes API Requests to API Gateway
API Gateway returns the result of one of the lambda handler
API Gateway
Returns a temperory URL to upload the photo/video to S3
Get a temperory signed URL to upload the photo/video to S3
Lambda(postphoto)
S3
Lambda(postvideo)
Save the photo/video metadata along with the uploader username into DynamoDB
Lambda(getphoto)
DynamoDB
Returns photo metadata
Get the metadata of all photos / videos uploaded by user specified in the JWC Token
Lambda(getvideo)
Authenticate
Cients recieve JWC token
Amazon Cognito Hosted UI
Noitifies SNS when the source photo is uploaded
SNS
SQS
Recognition identifies tags in the photo
Lambda(Identifytags)
Lambda(CreateThumbnailMobile)
Lambda(CreateThumbnailDesktop)
Processed media is saved into S3
Noitifies SNS when the source photo is uploaded
SNS
SQS
Lambda(submitJob)
Elemental MediaConverter

Figure 1: The architectural design of the solutions which includes the interactions as well.

UML diagram:

UML cooperation diagrams are shown in this section to illustrate how the suggested AWS services work together to fulfil the use cases. (figure 2, figure 3)

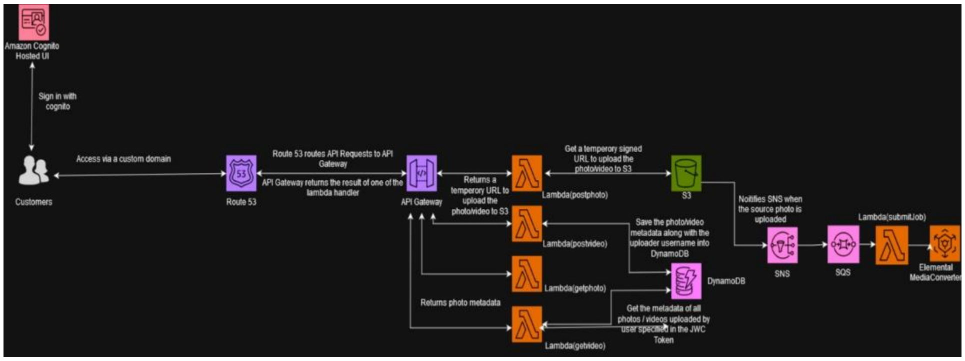
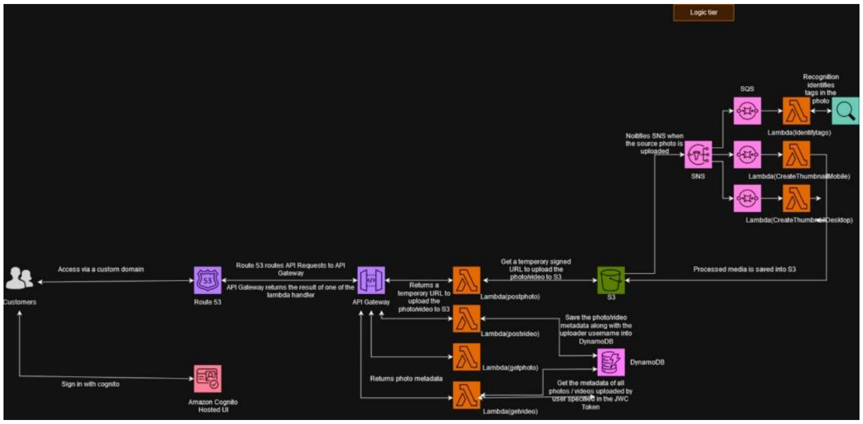

Amazon Cognito Hosted UI
Sign in with cognito
Access via a custom domain
Customers
Route 53
Route 53 routes API Requests to API Gateway
API Gateway returns the result of one of the lambda handler
API Gateway
Returns a temperory URL to upload the photo/video to S3
Get a temperory signed URL to upload the photo/video to S3
Lambda(postphoto)
S3
Noitifies SNS when the source photo is uploaded
Lambda(postvideo)
Save the photo/video metadata along with the uploader username into DynamoDB
Lambda(submitJob)
SNS
SQS
Elemental MediaConverter
Lambda(getphoto)
DynamoDB
Returns photo metadata
Get the metadata of all photos / videos uploaded by user specified in the JWC Token
Lambda(getvideo)

Figure 2: Uploading and processing of user data.

Figure 3:UML for uploading and processing user data


Logic tier
SQS
Recognition identifies tags in the photo
Lambda(Identifytags)
Noitifies SNS when the source photo is uploaded
SNS
Lambda(CreateThumbnailMobile)
Lambda(CreateThumbnailDesktop)
Access via a custom domain
Customers
Route 53
Route 53 routes API Requests to API Gateway
API Gateway returns the result of one of the lambda handler
API Gateway
Returns a temperory URL to upload the photo/video to S3
Get a temperory signed URL to upload the photo/video to S3
Lambda(postphoto)
S3
Processed media is saved into S3
Lambda(postvideo)
Save the photo/video metadata along with the uploader username into DynamoDB
Lambda(getphoto)
DynamoDB
Returns photo metadata
Get the metadata of all photos / videos uploaded by user specified in the JWC Token
Lambda(getvideo)
Sign in with cognito
Amazon Cognito Hosted UI

As mentioned earlier, there are several kinds of services that will be used for this project. This section will delve into more about these services in detail.

- Amazon(S3): Amazon S3 service (simple storage service) is a cloud storage service which is fully managed and maintained by AWS platform. This allows the users to save their media data assets. In this scenario this cloud storage service can be used to store the user data such as photos, transformed photos such as thumbnails and other user uploads.

  This will lead to the application to scale out the media assets without any effort. As it's a cloud-based service the users can access their assets without any interruption such as hardware failure. Also, it makes the assets available through global and with the help of the CloudFront Services, increase the content delivering by reducing the latency. These will make the application much more durable and available globally.

- DynamoDB: Amazon DynamoDB is a database system that stores user media data. This allows the user to create and save the photo meta data, user data and thumbnails of it.

  This will also lead the application to lead the users to read and write their user, media meta data and the thumbnails of the photos. As the DynamoDB function has its own automatic scaling functions, it allows the rising data volumes to accommodate.

- Cloud front services: Amazon CloudFront service is a content delivery service which is provided and maintained by AWS. It acts a major role in delivering contents through out the world in the AWS services. It allows the contents to distribute globally by reducing the latency and improving the overall user experience.

  As the CloudFront services reduces latency and ensures that users can swiftly retrieve material by caching media assets at thoughtfully chosen edge sites. CloudFront's global connectivity ensures that users may access their images and albums with consistency and efficiency from anywhere in the world. Its adaptive scaling and load balancing features support growing user traffic, and DDoS mitigation adds a further degree of security. Additionally, CloudFront reduces the strain on the core infrastructure by distributing content from edge locations, improving cost-effectiveness and resource optimization.

- Lambda Functions: The lambda functions act as a serverless computing system, which allow the users for media uploading and the configure the media processing tasks. Lambda functions are key components for seamless media management. These can be written in various languages such as Python, Javan and Node.

  These functions, inherently event-triggered, diligently execute critical operations. Specifically, when users upload media content, a dedicated Lambda function provides the secure storage of uploads in Amazon S3, optimizing resource usage and enabling rapid data access. For media processing tasks like image resizing, and thumbnail generation, an array of Lambda functions springs into action when new media emerges in S3. This functions triggers by the events happened in the cloud such as HTTP requests. This dynamic, serverless architecture flexibly scales in response to varying workloads, guaranteeing cost-efficiency. Augmented by Amazon SQS queuing, Lambda functions neatly disentangle processing tasks from the core application, endorsing effective, scalable, and dependable media processing. Furthermore, this setup positions the application for future adaptability, accommodating the incorporation of AI-driven features, such as automatic image recognition and tagging, for a richer user experience.

- SNS service: Amazon SNS service (Simple Notification Service) is a messaging service which is provided and maintained by AWS. It enables communication between applications to applications and applications to users in scalable, cost-effective, and flexible manner. Mainly it runs as a publish-subscribe model and it can be used to decouple the various architectures.

  In this scenario, SNS service can be used to decouple the photo and other types of media processing pipelines of the application. SNS subtly links multiple application parts, fostering effective communication and information exchange. It plays a crucial function in informing various system components about the status of media processing tasks, fault detection, and real-time alarms. Additionally, SNS's scalability and extensibility allow it to adapt to the application's changing requirements with ease and set the system up for potential upgrades like the incorporation of AI-driven features like automatic image tagging. It effectively informs users of newly tagged media, enhancing the user experience and enhancing the operation of the service.

- SQS service: Amazon SQS is a Amazon based and managed message queue service which enables the queuing and decoupling between the serverless applications. The decoupling enables the components to function separately so that disruptions such as failures in the system, won't affect into other components.
  A number of the project's key needs are met by the upgraded Photo Album application's integration of Amazon Simple Queue Service (SQS). The application's first concern is managing media processing duties effectively, which is essential given that it expects a rise in demand. SQS ensures that media processing is handled fairly, reducing system overload, and adhering to the scalability requirement by queuing and allocating these jobs. Additionally, SQS helps to achieve the goal of decoupling architectural components, improving system reliability and resilience and meeting the requirements for reliability and decoupling. Finally, SQS lays the groundwork for future expansion because it can easily accommodate new activities and components, meeting the need for an architecture that can adapt to future improvements such as features driven by AI.

- Rekognition: Rekognition is a user identity and access management service. Users can be authenticated and their access to apps can be managed using the user identity and access management (IAM) service provided by Amazon Cognito. It can be applied to a wide range of applications, including web-based, mobile, and API-driven ones.
  Cognito can be used to manage user access to the serverless picture album application and perform user authentication. This entails managing user profiles, authorizing users to gain entry to areas o The proposed

architectural design for the enhanced Photo Album application leverages a variety of AWS services to address the growing demand and changing requirements of the business scenario. The design prioritizes performance, scalability, reliability, security, and cost-efficiency while remaining sustainable.

To handle the rising demand, the design focuses on scalability and reliability, accommodating increasing user data and meeting the application's needs. It also utilizes serverless computing, efficient storage, and content delivery to optimize performance.

The design integrates Amazon S3, Lambda functions, and DynamoDB services to create an effective and scalable environment. Amazon S3 provides durable and scalable storage for media files, while Lambda functions enable efficient media processing. DynamoDB offers a NoSQL database solution for handling the application's simple table structure, low-latency access, and scalability.

<u>5. Design Rationale:</u>

The proposed architectural design for the enhanced Photo Album application leverages a variety of AWS services to address the growing demand and changing requirements of the business scenario. The design prioritizes performance, scalability, reliability, security, and cost-efficiency while remaining sustainable.

To handle the rising demand, the design focuses on scalability and reliability, accommodating increasing user data and meeting the application's needs. It also utilizes serverless computing, efficient storage, and content delivery to optimize performance.

The design integrates Amazon S3, Lambda functions, and DynamoDB services to create an effective and scalable environment. Amazon S3 provides durable and scalable storage for media files, while Lambda functions enable efficient media processing. DynamoDB offers a NoSQL database solution for handling the application's simple table structure, low-latency access, and scalability.

Overall, the proposed architectural design is a sustainable solution that meets the growing demand and changing requirements of the business scenario while optimizing performance, scalability, reliability, security, and cost-efficiency.

The design rationale for this project is as follows:

I. Utilization and managing Cloud based services:

Optimize the Requirement for In-House Systems Administration through Utilisation of Managed Cloud Services

Our architectural strategy revolves around utilizing fully managed cloud services to align with our core objective: reducing the workload of in-house systems administration. The selected services for this purpose are as follows:

DynamoDB has been selected as the database solution for efficiently storing and managing the metadata linked to the photos and videos uploaded by users. The schema-less design and key-value data model of this system make it highly suitable for this particular task, providing outstanding performance and scalability capabilities. The automatic scaling, multi-AZ replication, and security features of DynamoDB are in perfect alignment with our requirements.

API Gateway is a crucial component in our system architecture as it acts as a secure intermediary between our frontend and backend logic. Its primary function is to streamline the process of creating, publishing, and managing APIs. By implementing automated load balancing and fault tolerance mechanisms, the system guarantees optimal performance and enhanced reliability. The utilisation of custom authentication methods alongside the integration of AWS Cognito offers a highly resilient mechanism for regulating user access control.

Amazon S3 is the optimal selection for hosting our static website and storing user-uploaded media due to its exceptional capabilities. The scalability, durability, and robust security features of this solution align seamlessly with our specific engineering needs.

Amazon Cognito provides robust identity verification and user management capabilities to ensure the secure and controlled access of our application. The system exhibits high efficiency in managing user registrations, logins, and storage of user data on a large scale. Additionally, it incorporates the capability to facilitate multi-factor authentication.As an engineering student, it is crucial to develop strategies to effectively manage the challenge of coping with a demand that doubles every six months. This exponential growth in demand presents various technical and logistical hurdles that need to be addressed. To tackle this issue, it is essential to analyse the current production and distribution processes thoroughly. By conducting a comprehensive assessment, we can identify potential bottlenecks and areas.

Our approach to addressing the growing demand is in line with our inclination towards utilizing fully managed services and implementing a serverless architecture. The services chosen to fulfil this requirement encompass:

Lambda: The automatic scaling and efficient resource utilisation of Lambda allow us to seamlessly handle increasing workloads. The concurrency management and event-driven nature of our application guarantee efficient handling of surges in incoming requests.

The auto-scaling capabilities of DynamoDB enable our database tables to adapt their read and write capacity in response to real-time traffic patterns, effectively preventing throttling issues during periods of increased traffic.

The API Gateway possesses the capability to dynamically adjust to incoming traffic, thereby facilitating seamless and scalable management of API workloads. Consequently, it serves as a reliable solution for effectively accommodating the growing demand. As an engineering student, it is recommended to consider implementing a serverless/event-driven solution.

The progression towards a serverless architecture entails the deployment of our website in a static manner within an S3 bucket, while incorporating backend functionality through the utilisation of Lambda functions. This methodology is in complete accordance with our event-oriented characteristics, guaranteeing efficient scalability at a reasonable cost as our application expands.

S3, which stands for Simple Storage Service, is an excellent choice for hosting static web content. Its seamless integration with Amazon CloudFront allows for quicker retrieval of content, enhancing the overall user experience. Additionally, S3 boasts impressive durability and security features, ensuring that hosted content remains safe and protected.

Lambda is an exceptional serverless computing service that possesses the remarkable capability to swiftly scale in response to incoming requests. Its distributed nature across multiple availability zones significantly bolsters the reliability of our application.4. Substituting the Relational Database with more efficient and cost-effective alternatives

Considering the straightforwardness of our media metadata, we have made the decision to employ a NoSQL database model, specifically utilising DynamoDB. This selection is in accordance with the prerequisites of achieving high throughput, minimising latency in both read and write operations, and ensuring scalability for a media uploading application.5. Enhance the global response times for users located beyond the geographical boundaries of Australia.

To optimize application performance for users worldwide, we employ Amazon CloudFront to cache various types of content, such as static web files and media uploaded by users. This methodology brings the content in closer proximity to the end users and incorporates an additional level of security by utilising Lambda@Edge functions.As an engineering student, it is crucial to have the ability to effectively manage video media. This entails understanding the technical aspects of video formats, codecs, and compression techniques. Additionally, it is important to be proficient in utilizing video.

By integrating video uploads into our current architecture, we have implemented a system where videos are stored in the S3 bucket alongside photos, while their corresponding metadata is stored in DynamoDB. Elemental MediaConvert, a powerful tool utilised in video transcoding, has been incorporated into the system. This addition provides engineering students with an extensive selection of input and output formats, ensuring compatibility across various platforms. Moreover, the scalability feature allows for efficient handling of large volumes of video data. Additionally, data security measures have been implemented to safeguard the integrity and confidentiality of the transcoded videos. As engineering students, we aim to develop a system that can efficiently process uploaded media using an extensible and decoupled architecture.

The architecture we have implemented utilises S3 Event Notifications to initiate SNS (Simple Notification Service) when there is an upload of media files. This, consequently, results in the implementation of a photo processing pipeline leveraging SNS, SQS, and Lambda functions, enabling concurrent processing and scalability.8. Implement user authentication mechanisms to regulate access to the application.

We employ Amazon Cognito, a robust and readily available solution, to establish verified access to our application. This entails utilising user pools and authentication functionalities to guarantee secure and authorised entry. This selection is in accordance with our objective of prioritising simplicity and ensuring security.

B. Requirements for a Good Layout

I. Scalability and Efficiency

By carefully selecting the appropriate services, our architecture can meet the application's performance and scalability requirements with ease.

Amazon CloudFront: We greatly improve response times for users all around the world by caching material in edge locations globally. This helps with speed and makes the system feel more responsive.

The AWS Lambda service is an event-driven, automatically scalable solution for handling high demand. It optimises resource utilisation when processing media uploads and producing variants like thumbnails and transcoded videos.

Amazon's API Gateway makes short work of receiving API calls thanks to its serverless design and in-built caching features. It can handle spikes in traffic automatically, without sacrificing speed. Smooth scalability is achieved by distributing queries over several backend services.

Our media and data are safely and securely stored on Amazon's Simple Storage Service (S3). Thanks to S3's ability to handle massive volumes of data and multiple requests at once and its global distribution via Amazon CloudFront, speed is maximized, and latency is minimized.

Amazon DynamoDB: DynamoDB is great for storing video and image metadata. It's a great option for storing metadata efficiently due to its simple key-value data type, horizontal scaling potential, and low cost. Two, dependability

We've made it a priority to ensure that our design would pass the dependability tests required by the software. The following service options help to achieve this goal:

Amazon Route 53: By providing a globally dispersed DNS server network, Route 53 guarantees dependable and consistent domain name service routeing. This ensures dependable and constant user access, which improves the application's dependability.

Amazon API Gateway: Capable of handling heavy traffic loads without slowing down. Security is bolstered by measures like API throttling and authentication, leading to a safe and dependable user experience.

Amazon S3 provides users with safe and dependable storage for their uploaded photographs and videos thanks to its large storage space and support for a wide range of media formats. Key elements of its dependability are the data's integrity and availability.

Amazon DynamoDB is a fast and reliable database service that is used to store media file metadata. The horizontal scalability of the application and the simplicity of its key-value data type both add to its dependability.

Decoupling the image and video processing pipelines is made possible with the help of Amazon's Simple Queue Service (SQS). SQS guarantees fault tolerance and improves the reliability of the application's processing tasks by keeping jobs in queues and letting processing nodes pull and process jobs at their own pace.

Decoupling the photo and video processing pipelines with Amazon's Simple Notification Service (SNS) allows for parallel processing and increases dependability. By decoupling various operations, as in a fan-out architecture, system dependability is increased.

C. Safety:

Our architecture uses well-considered service picks to guarantee that the application's security requirements are met.

Amazon Cognito: Cognito is crucial in delivering strong security. Data security is improved by using token-based authentication, claims-based tokens, and encryption for private information. Multi-Factor Authentication (MFA) support, token revocation, and session management all contribute to a more secure application.

Protecting sensitive information is a top priority for AWS Lambda, which is why the service offers security features including IAM roles with granular permissions, fine-grained access control, and data encryption. Lambda's security is bolstered by its use of isolated execution environments and its frequent updates to its security measures.

Protections for data and media assets are robust on Amazon S3. Only authorised users can access a bucket's contents thanks to access control lists (ACLs) and associated policies. Furthermore, sensitive data is protected at rest by server-side encryption's use of permanent encryption.

Data stored and transmitted with Amazon DynamoDB are both encrypted. To protect sensitive information, you can restrict who can access which tables and data by setting policies in AWS Identity and Access Management (IAM).

Access to APIs can be successfully restricted based on user identities and permissions with the help of Amazon API Gateway's authentication and authorisation techniques, such as IAM roles, unique authentication methods, and AWS Cognito. The application's security is bolstered by these measures.4. Cost

Through careful service selection and architectural decisions, our design effectively meets the application's cost criteria:

We only pay for the resources and requests that we use, thanks to the pay-per-use pricing models used by the vast majority of our prefered service providers. This method is perfect for both small and large-scale projects since it saves money by preventing us from being charged for unneeded resources.

Our programme is scalable because it can adapt to varying loads in real time without requiring additional infrastructure. The cost-effectiveness of services like Lambda and DynamoDB is guaranteed by their dynamic scalability.

We only pay for the time it takes the computer to execute the code when calling an API or running a function, thanks to the serverless architecture we built with Lambda and API Gateway. This saves money because it negates the need for dedicated servers and the associated expenditures.

Our architectural options reduce financial risks because they do not require any upfront payments or contractual commitments. We don't need a big budget to get started, and we can expand if necessary.

Managed Services: Our architecture relies heavily on managed services, which allow us to focus on application development rather than infrastructure administration and save down on operational costs.

Amazon S3 and DynamoDB are two examples of services that provide efficient storage choices, guaranteeing both high availability and durability at low costs. You may save money without sacrificing data security or trustworthiness.

D. Cost:

This section will provide the details about how the proposed design idea will meet the cost effectiveness of towards business.

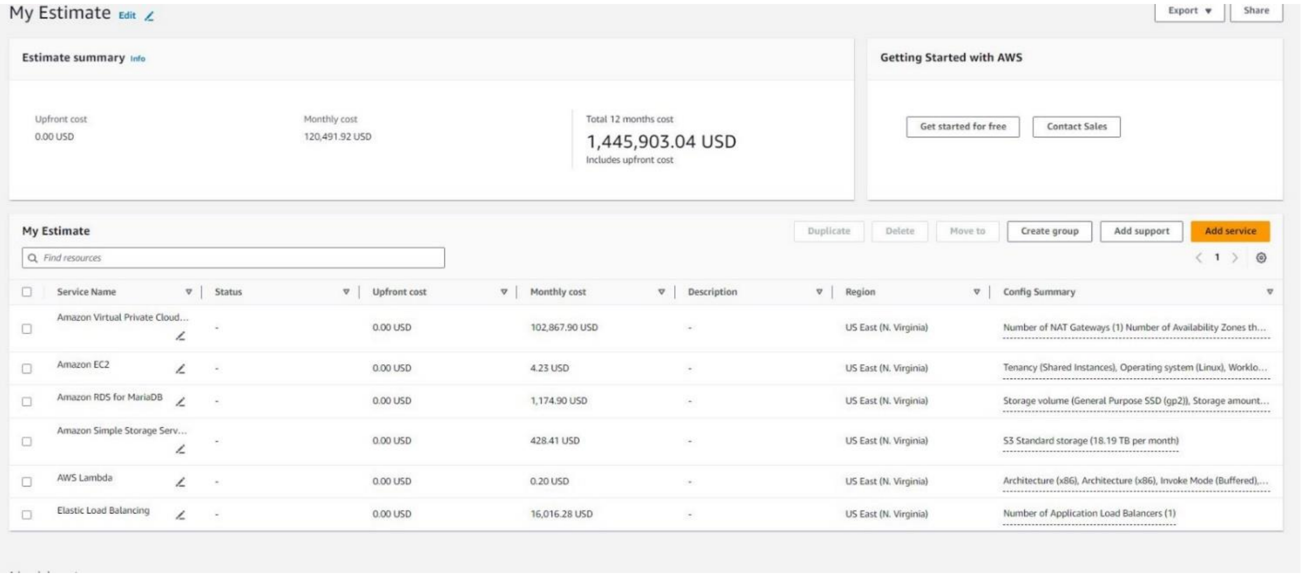
- Pay as you go: The AWS services that have been used in this project are pay-per-use models. This states that the company will only have to pay for what is being used at the time. This method is much more effective for the smaller scale and growing business models as they can fit in with their expected budget.
- Variable costing: The enhanced Photo Album application is scalable and cost-effective due to its serverless architecture and the use of managed AWS services. It can automatically scale up or down in response to demand, so businesses only pay for the computing time and storage they need. There are no up-front costs or long-term obligations, and AWS manages the infrastructure, so businesses can focus on application development.
- Effective storage service: Users can store their assets in more effective

ways, such as in DynamoDB and S3 with high availability and more durability.

E. Final Budget estimation:

This below diagram will demonstrate how the monthly cost will be for the final design solution. These calculations are mainly focused on the assumption that based on the instructions. The prices are up to date and calculated by the AWS pricing calculator and the services that are mentioned below are assumed to be in the Sydney region.

The improved Photo Album application is a scalable and affordable solution that satisfies the changing needs of the business scenario and expanding demand. Because of the application's serverless architecture and usage of managed AWS services, it can scale up or down automatically in response to demand, allowing companies to only pay for the computing power and storage they really need. AWS controls the infrastructure, there are no upfront expenditures or commitments, and enterprises can concentrate on application development. (figure 4)



My Estimate

Estimate summary

Upfront cost
0.00 USD

Monthly cost
120,491.92 USD

Total 12 months cost
1,445,903.04 USD
Includes upfront cost

My Estimate

□	Service Name	Status	Upfront cost	Monthly cost	Description	Region	Config Summary
□	Amazon Virtual Private Cloud...	-	0.00 USD	102,867.90 USD	-	US East (N. Virginia)	Number of NAT Gateways (1) Number of Availability Zones th...
□	Amazon EC2	-	0.00 USD	4.23 USD	-	US East (N. Virginia)	Tenancy (Shared Instances), Operating system (Linux), Worklo...
□	Amazon RDS for MariaDB	-	0.00 USD	1,174.90 USD	-	US East (N. Virginia)	Storage volume (General Purpose SSD (gp2)), Storage amount...
□	Amazon Simple Storage Serv...	-	0.00 USD	428.41 USD	-	US East (N. Virginia)	S3 Standard storage (18.19 TB per month)
□	AWS Lambda	-	0.00 USD	0.20 USD	-	US East (N. Virginia)	Architecture (x86), Architecture (x86), Invoke Mode (Buffered),...
□	Elastic Load Balancing	-	0.00 USD	16,016.28 USD	-	US East (N. Virginia)	Number of Application Load Balancers (1)

Figure 4: Final estimation for the business scenario

References:

Amazon Web Services, Inc. (n.d.). Amazon DynamoDB Features | NoSQL Key-Value Database | Amazon Web Services. [online] Available at: https://aws.amazon.com/dynamodb/features [Accessed 20 Oct. 2023].

Amazon Web Services, Inc. (n.d.). Amazon S3 Security Features - Amazon Web Services. [online] Available at: https://aws.amazon.com/s3/security.

Amazon Web Services, Inc. (n.d.). Features | Amazon Cognito | Amazon Web Services (AWS). [online] Available at: https://aws.amazon.com/cognito/details [Accessed 20 Oct. 2023].

Amazon Web Services, Inc. (n.d.). Pricing | Amazon Cognito | Amazon Web Services (AWS). [online] Available at: https://aws.amazon.com/cognito/pricing [Accessed 20 Oct. 2023].

Lambda execution environments - AWS Lambda. [online] Available at: https://docs.aws.amazon.com/lambda/latest/operatorguide/execution-environments.html.

AWS CloudFront Pricing and Cost Optimization Guide | CloudForecast. [online] Available at: https://www.cloudforecast.io/blog/aws-cloudfront-pricing-and-cost-guide/ [Accessed 20 Oct. 2023].

Amazon Rekognition – Video and Image - AWS. [online] Amazon Web Services, Inc. Available at: https://aws.amazon.com/rekognition/.

AWS Fundamentals. (n.d.). Amazon CloudFront Pricing Model Guide. [online] Available at: https://blog.awsfundamentals.com/amazon-cloudfront-pricing [Accessed 20 Oct. 2023].

CloudFront Pricing: How to Approach it and Save Money? [online] Available at: https://www.stormit.cloud/blog/amazon-cloudfront-pricing-how-to-approach-it-and-save-money/.

What is Amazon CloudFront. [online] Available at: https://www.w3schools.com/whatis/whatis_aws_cloudfront.asp.

What is DynamoDB? [online] Available at: https://www.dynamodbguide.com/what-is-dynamo-db/.

What are AWS Fargate? - Amazon ECS. [online] Available at: https://docs.aws.amazon.com/AmazonECS/latest/userguide/what-is-fargate.html.

Data protection in AWS Lambda - AWS Lambda. [online] Available at: https://docs.aws.amazon.com/lambda/latest/dg/security-dataprotection.html.

Amazon Web Services, Inc. (n.d.). Key Features of a Content Delivery Network| Performance, Security | Amazon CloudFront. [online] Available at: https://aws.amazon.com/cloudfront/features/.