COS20009 – CLOUD COMPUTING ARCHITECTURE

1. Introduction

This document outlines the strategic overhaul of an existing Photo Album application to meet increasing user demand and address current challenges. Hosted on Amazon Web Services (AWS), the existing architecture requires an update to enhance performance, reliability, and scalability while reducing the burden of internal management. The transition to an event-driven/serverless architecture is essential. This transformation will utilize AWS services including AWS CloudFront, AWS DynamoDB, AWS Route 53, AWS Simple Notification Service (SNS), AWS Amplify, and Amazon API Gateway. The rationale behind this architectural evolution and its anticipated benefits are thoroughly examined in this report.

1.1 Existing Photo Album Application

1. **Current Infrastructure**

The current infrastructure comprises a photo uploading web application hosted within a Virtual Private Cloud (VPC) located in a specific AWS region. This VPC spans two availability zones, each containing private and public subnets designated as Private Subnets 1 and 2, and Public Subnets 1 and 2, respectively. Key components include:

- *Database Management*: A Relational Database Service (RDS) instance essential for database operations is situated within Private Subnet 1.
- *Image Processing and Management:* A web server instance in Public Subnet 2 handles the creation of Amazon Machine Images (AMI) and facilitates manual interactions with the RDS.
- *Load Management:* A load balancer located in the Public Subnet directs incoming internet traffic to an Auto Scaling Group that can scale between two and three instances, optimizing traffic management.
- *Storage Solutions:* The application offers scalable storage options by storing images in an Amazon S3 bucket located outside the VPC. Integration with AWS Lambda automates the compression and re-uploading of images upon addition, enhancing image management efficiency.
- *Network Connectivity:* A Network Address Translation (NAT) instance in Public Subnet 1 enables secure external connections to services like S3 storage, adhering to stringent security and access control protocols.

This robust infrastructure supports the photo uploader application by balancing scalability, security, and functionality.

1.2 Further Business Scenario and Objectives

As the application anticipates substantial growth, the following objectives guide the architectural enhancements:

- *Demand Management:* Address the escalating demand for the Photo Album app.
- *Managed Services:* Minimize internal system management by leveraging managed cloud services.
- *Future Growth:* Prepare for exponential growth, anticipating a doubling of application demand every six months over the next two to three years.
- *Performance Optimization:* Improve system efficiency by reducing the load on EC2 instances.
- *Architecture Transition:* Shift towards an event-driven/serverless architecture.
- *Database Efficiency:* Explore cost-effective alternatives to the currently slow and costly relational database system.
- Global Accessibility: Enhance response times for users globally.
- *Media Support:* Plan for future enhancements including automated media version production and support for video media.
- *Automated Media Processing:* Implement an extensible and efficient architecture to automate media processing across multiple formats.

2. Architecture Diagram

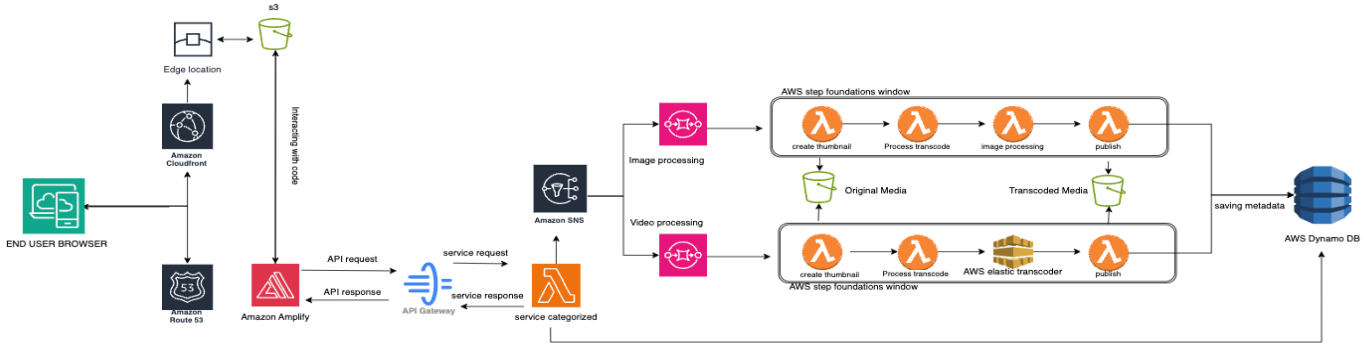

FIG1: ARCHITECTURE DIAGRAM

3. Design Justification and Description

3.1 AWS S3 Bucket

Using Amazon S3 in a serverless AWS Amplify setup aligns well with our goals of leveraging managed cloud services and reducing internal system management overhead. Here are the key benefits:

- *Managed Service*: AWS handles infrastructure management, including data replication, server provisioning, storage management, and backups, allowing our team to focus on application development.
- *Automatic Scalability*: S3 scales storage capacity automatically based on application needs, which is essential in a serverless environment.
- *Security and Access Control*: S3 provides advanced security features like bucket policies, IAM roles, and access control lists (ACLs) for robust data security management.
- *Data Durability and Availability*: S3 ensures high data durability and availability by storing data redundantly across multiple data centers.
- *Cost-Effectiveness*: The pay-as-you-go pricing model helps control costs by charging only for the storage used, aligning with our goal of minimizing expenses while maximizing resource utilization.

AWS Amplify

AWS Amplify is a comprehensive set of tools designed for front-end and mobile developers, simplifying the process of building end-to-end applications on AWS [1]. It addresses key challenges such as traffic load, server management, and scalability, enhancing developer productivity and operational efficiency.

Aspect	Current infrastructure	AWS Amplify
Performance	Manually managed EC2 instances and load balancers require significant optimization efforts.	Uses serverless architecture that auto-scales resources and integrates with Amazon CloudFront for faster content distribution, reducing manual maintenance and optimizing performance.
Reliability	Relies on redundancy across availability zones for reliability, but manual resource management can introduce risks.	Enhances reliability with fault-tolerant, highly available serverless services like AWS Lambda and Amazon S3, minimizing downtime and human errors.
Security	Depends on proper configuration of EC2 instances and effective security group management.	Simplifies security with AWS IAM integration for access control and automatic security patches, enhancing the overall security posture.
Cost	Manually managing EC2 instances can be cost-effective but requires continuous management, increasing operational costs.	Operates on a pay-as-you-go model, reducing costs associated with infrastructure management and maintenance.

Table 1. Current infrastructure and AWS amplify comparison

3.3 Amazon API Gateway

Amazon API Gateway is a fully managed service provided by AWS that enables developers to create, publish, maintain, monitor, and secure APIs at any scale [3]. The integration of Amazon API Gateway into the Photo Uploader application brings numerous benefits that streamline API management, enhance security, enable scalability, and ensure the reliability of the API layer, crucial for maintaining robust application architecture [2].

Integrating Amazon API Gateway into the Photo Uploader application provides significant benefits in API management, security, scalability, monitoring, and development.

1. **Simplification of API Management**:
   - Existing Challenges: Managing APIs manually is time-consuming and error-prone, lacking centralized security and monitoring.
   - Benefits: API Gateway simplifies API management with a user-friendly interface, reducing setup time and effort, and providing centralized security and monitoring.

2. **Security and Access Control**:
   - Previous Approach: Inconsistent security measures and disparate access controls across API endpoints.
   - API Gateway Implementation: Integrates robust security features, including authentication and authorization, with simplified, standardized access control via IAM policies.

3. **Scalability**:
   - Manual Scaling Limitations: Traditional scaling requires extensive planning and can cause performance issues during high traffic.

- API Gateway Scalability: Offers auto-scaling capabilities to dynamically adjust resources, ensuring optimal performance and reliability.

4**. Monitoring and Analytics**:
- Conventional Monitoring Challenges: Complex setups with third-party tools hinder real-time visibility into API performance.
- API Gateway Monitoring: Provides integrated monitoring and analytics for detailed insights, enabling proactive issue resolution and enhanced reliability.

5. **Developer-Friendly Environment**:
- Traditional Development Hurdles: Cumbersome manual processes and lack of standardized tools for publishing and documenting APIs.
- Advantages: API Gateway offers intuitive interfaces, built-in documentation tools, templates, and guidelines, reducing time-to-market and fostering better collaboration and innovation.

3.4 AWS Simple Notification Service (SNS)

AWS Simple Notification Service (SNS) is a fully managed messaging service offered by AWS that facilitates the efficient distribution of messages to multiple recipients across diverse protocols. Operating on a robust publish-subscribe model, SNS enables publishers to send notifications to topics, which then distribute these messages to subscribed endpoints, such as Lambda functions, SQS queues, HTTP endpoints, or even direct to user emails or SMS. This service plays a crucial role in the media distribution and processing architecture of the Photo Uploader application.

Justification for Using AWS SNS

1. **Central Role in Media Management:**

- *Event-Driven Integration:* Uploading a media file to Amazon S3 triggers an AWS Lambda function, which processes the file and publishes a notification to an AWS SNS topic..
- *Workflow Efficiency:* AWS SNS orchestrates media processing tasks by pushing messages to relevant processing functions automatically, enhancing efficiency.

2. **Routing Media:**

- *Targeted Delivery:* AWS SNS directs notifications to appropriate subscribers based on predefined topics, ensuring precise processing of each media file.
- *Functional Routing*: Notifications are routed to Lambda functions specialized in tasks like video transcoding or image resizing, ensuring targeted processing.

3. **Enhancing Flexibility and Scalability:**

- *Adaptive Messaging*: SNS's dynamic message routing allows the application to adapt to new requirements easily.
- *Scalable Communication:* SNS supports straightforward modifications to topics or subscriptions, aiding in agile responses to changing needs.
- *Decoupling Components:* NS decouples system components, enhancing independent scalability and resilience, ensuring changes or failures in one part do not impact others.

3.5 Amazon DynamoDB

Amazon DynamoDB, an AWS-managed NoSQL database service, is celebrated for its exceptional scalability and rapid performance. It processes an astonishing volume of over 10 trillion requests daily, making it ideal for high-data-load applications. With robust features like encryption, automated backups, and up to 99.999% availability, DynamoDB ensures secure and reliable data storage. As a fully managed serverless database, DynamoDB adjusts dynamically to match application demands, cutting costs and reducing administrative burdens. Its seamless integration with other AWS services enhances flexibility, making it suitable for diverse industries like software development, retail, and finance.

Justification for Choosing Amazon DynamoDB

- *Scalability and Management:* DynamoDB efficiently handles rapid data volume increases with minimal administrative tasks, supporting application growth without significant management overhead.
- *Cost-Effectiveness:* The pay-as-you-go pricing model ensures payment only for used resources, aligning with financial management goals and avoiding unnecessary costs.
- *Media Data Management:* DynamoDB's flexibility allows effective management of various media formats, crucial for storing and retrieving large media files efficiently.
- *Integration with Serverless Architecture:* DynamoDB fits well into serverless and event-driven architectures, promoting innovation and global scalability without traditional database constraints.

Key Differences from SQL Databases

- *Schema Flexibility:* Unlike SQL databases, which have fixed schemas requiring extensive planning and laborious modifications for changes, NoSQL databases like DynamoDB offer dynamic schema designs. This flexibility allows for easier and quicker adjustments to the database structure, accommodating evolving data models with minimal disruption.
- *Ease of Use and Rapid Data Access:* DynamoDB excels in user-friendliness and quick data access, making it particularly suitable for applications undergoing rapid growth or those operating on modern architectural paradigms such as serverless and event-driven frameworks. The database's ability to provide immediate access to data is vital for maintaining high responsiveness and supporting scalable operations.
- *Complex Data Relationships:* SQL databases excel in managing complex data relationships and supporting intricate queries, prioritizing data integrity and transactional consistency. In contrast, DynamoDB emphasizes high throughput and low-latency data access. While SQL databases are preferred for detailed transactional operations, DynamoDB is better suited for scenarios prioritizing performance and scalability.

Design Rationale for Amazon DynamoDB

Amazon DynamoDB serves as the backbone of our application's database management system, providing a robust, secure, and scalable environment that aligns seamlessly with our cloud-first strategy. The choice of DynamoDB over traditional database systems or other NoSQL services is driven by several key factors that cater to our operational and strategic needs.

1. Efficiency and Expandability

- DynamoDB offers minimal latency for data retrieval and storage, even under varying loads.
- Automatic scaling ensures resource allocation matches demand, maintaining performance stability.

2. Dependability

- Comprehensive replication across AWS regions ensures high availability and durability.
- Routine backups and point-in-time restores protect against data loss.

3. Security

- Fine-grained access control via AWS IAM maintains strict security standards.

4. Scalability

- DynamoDB handles large-scale operations without performance degradation.

5. Cost-Effectiveness

- Pay-as-you-go pricing aligns costs with actual usage, eliminating upfront investments.

6. Serverless and Event-Driven Architecture Compatibility

- Integration with AWS Lambda enables efficient, responsive, event-driven systems.

7. Flexibility for Media Handling

- DynamoDB efficiently handles diverse data types, including images and videos.

8. Reduction in System Administration

- As a managed service, DynamoDB automates tasks, freeing developers for innovation.

3.6 Amazon Route 53

Amazon Route 53 is a vital element in our application's architecture, functioning as a highly available and scalable Domain Name System (DNS) service offered by AWS. It plays a critical role in optimizing user request routing, improving global response times, and ensuring service continuity. Route 53's implementation is particularly essential in overcoming latency and accessibility challenges for users outside Australia.

Role of Amazon Route 53 in Optimizing Global Traffic Management

Route 53 effectively routes user requests to the nearest endpoint, which is critical for a globally distributed application like ours. It manages traffic globally through a network of DNS servers

located around the world, enhancing the user experience by reducing the time taken for DNS query resolutions and subsequently speeding up the overall access to the application.

Detailed Analysis and Advantages

1. **Current DNS Configuration:**

- *Challenges:* Previously, our application relied on a traditional DNS setup that was not optimized for handling high volumes of international traffic, resulting in slower response times and less reliable access for users located far from the origin servers.

2. **Integration of Amazon Route 53:**

- *Enhanced Performance:* Route 53 utilizes a global network of DNS servers that ensures queries are answered with the minimal possible latency by routing them to the closest geographical DNS server.
- *Improved Availability and Reliability:* Amazon Route 53 is designed to offer high availability and robust fault tolerance. It provides a 100% uptime guarantee, ensuring that the application remains accessible at all times, even in the event of a failure in one or more DNS servers.

Key Benefits of Amazon Route 53

1. **Latency Reduction:**
- *Prior Setup:* Global users often experienced delays due to inefficient DNS resolutions.
- *With Route 53:* Utilizes latency-based routing that directs users to the endpoint that provides the fastest response time, drastically reducing latency.

2. **High Availability:**
- *Prior Setup:* The DNS service used previously could not guarantee the same level of availability, potentially leading to access issues during peak times or outages.
- *With Route 53:* Offers built-in redundancy and automated routing policies that adjust to real-time internet conditions, ensuring optimal performance and reliability.

3. **Scalability:**
- *Prior Setup:* Scaling DNS infrastructure to accommodate global reach required significant manual intervention and administrative effort.
- *With Route 53:* Automatically handles large volumes of DNS queries without any need for manual scaling, accommodating sudden spikes in traffic seamlessly.

4. **Sophisticated Traffic Management:**
- *Prior Setup:* Limited flexibility in managing traffic dynamically based on changing internet conditions or user demographics.

- *With Route 53:* Supports a variety of routing policies such as Geolocation, Geo proximity, and Latency-based routing that can dynamically respond to user locations and network health, optimizing the flow of traffic.

3.7 Amazon CloudFront

Amazon CloudFront serves as a key component in our global content delivery strategy, leveraging its extensive network of edge locations to significantly enhance response times and content accessibility for users worldwide. As a content delivery network (CDN), CloudFront plays a crucial role in reducing latency, increasing content delivery speed, and ensuring that users receive a consistent and reliable experience, regardless of their geographic location.

Role of Amazon CloudFront in Enhancing Global Accessibility

CloudFront delivers static and dynamic content—such as images, videos, web pages, and other digital assets—by caching this content in edge locations closer to users. This proximity reduces the distance data must travel, minimizing latency and speeding up access to content. For our photo uploader application, this means that user-generated content, like photo uploads, is quickly and efficiently delivered to viewers around the globe.

Comprehensive Analysis and Advantages

Aspect	Existing Method	CloudFront Implementation
Latency Reduction	Users accessing distant content experienced delays routed to primary servers potentially on another continent	Content cached in multiple edge locations worldwide, minimizing physical distance, reducing latency
Scalability and Global Reach	Scaling involved substantial investment in global data centers, challenging technically and financially	Automatically handles traffic spikes, distributes content across global edge servers, ensures responsiveness
Content Caching Efficiency	Manual management of caching strategies led to operational overhead, suboptimal caching efficiency	Advanced caching mechanisms optimize storage and retrieval, reducing redundant data transfer costs
Security Enhancements	Security managed at origin server level, requiring extensive configurations and updates for DDoS defence	Integrates with AWS Shield for DDoS protection, AWS WAF for defence against attack vectors, HTTPS support
Cost-Effectiveness	Expanding global infrastructure was capital-intensive, involving significant upfront and ongoing expenditures	Pay-as-you-go model, pay only for services used, no costly infrastructure investments needed

Table 2. Comparison between existing method and CloudFront implementation

CloudFront's robust network ensures that our application can efficiently serve a global audience, providing a seamless and engaging user experience that is crucial for the success of our digital media offerings.

3.8 AWS Lambda Functions

AWS Lambda represents a pivotal component in our architecture, facilitating automated and scalable media processing tasks such as thumbnail creation, video transcoding, and other related functionalities. As serverless, event-driven compute services, Lambda functions provide a dynamic response to application events, triggering processes only when needed. This capability aligns seamlessly with the demand-driven nature of our application, supporting a variety of media processing workflows without the overhead of managing server infrastructure.

Role of Lambda Functions in Media Processing

Lambda functions are integral to handling complex media processing tasks. They are triggered by specific events, such as new media uploads to an Amazon S3 bucket. Once triggered, these functions can execute code to perform operations like image resizing or video transcoding in real-time, directly responding to user actions without any manual intervention.

Comparative Analysis and Benefits

1. **Performance:**

- *Existing Infrastructure:* Previously, media processing tasks were handled through manually managed EC2 instances, which required scaling adjustments and could lead to delays or bottlenecks during peak loads.
- *AWS Lambda:* Lambda functions scale automatically with the number of requests, ensuring consistent performance regardless of the load. This auto-scalability enhances the overall responsiveness of our application.

2. **Reliability:**

- *Existing Infrastructure:* Manual setups are prone to human error and can suffer from downtime during updates or scaling operations.
- *AWS Lambda:* Offers high availability and fault tolerance. Lambda functions are replicated across multiple availability zones and are managed by AWS to ensure they execute without any management overhead from our side.

3. **Security:**

- *Existing Infrastructure:* Security depends heavily on the correct configuration of the servers and can be compromised if not updated or managed properly.
- *AWS Lambda:* Enhances security by abstracting the underlying infrastructure management. AWS handles all maintenance, including security patching, freeing our team to focus on application development. Lambda also integrates easily with AWS IAM to manage and restrict function permissions finely.

4. **Cost-effectiveness:**

- *Existing Infrastructure:* Costs are often fixed based on server capacity, which can lead to over-provisioning and underutilization, especially outside of peak times.
- *AWS Lambda:* Operates on a pay-as-you-go model where costs are directly tied to actual usage. You pay only for the compute time you consume, with no charges when your code is not running. This can lead to significant cost savings, especially for applications with variable traffic.

4. Interaction in between services and UML collaboration Media showing process:

Media Showing Process:

Interaction between services and UML collaboration is demonstrated in the media showcasing process depicted in Figure 2. This depiction offers insights into the user experience and data flow within the system, outlining the steps users take to access the website and view uploaded media. It emphasizes the pivotal role of AWS services throughout this process.

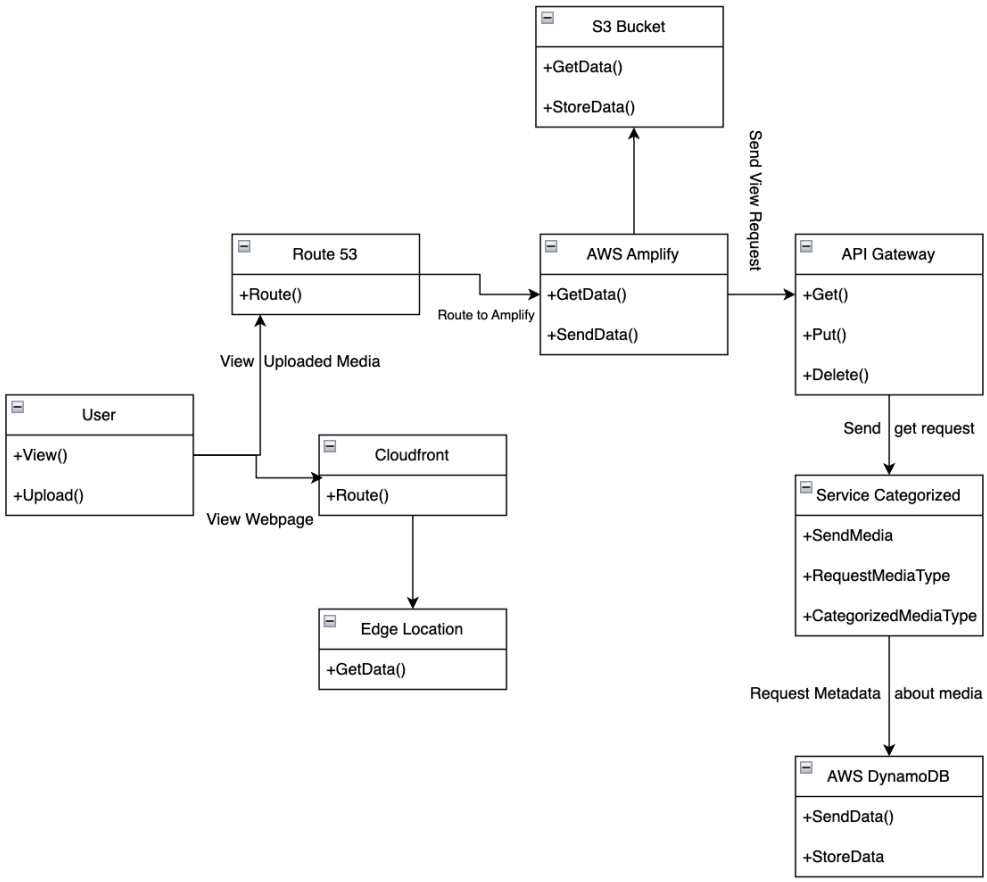

FIG2: UML COLLABORATION CHART.

User Experience and Data Flow

This section offers a comprehensive view of the user experience and the underlying data flow architecture within our system. It details the journey from initial user access to the website to the viewing of uploaded media, highlighting the pivotal role that AWS services play in optimizing and facilitating these interactions.

Process Breakdown

1. **User Browsing:**

- *Initial Access:* As users access the website, they are automatically routed to the nearest Amazon CloudFront edge location. This geographical routing is crucial for ensuring that the latency experienced by users is minimized, thereby enhancing the speed and responsiveness of the website.
- *Content Rendering:* Once the initial routing is complete, AWS Amplify takes over. As a robust frontend service, Amplify is responsible for rendering the website's interface and efficiently displaying the media content stored in Amazon S3 buckets. This setup ensures that the website remains dynamic and responsive, adapting to user interactions in real time.

2. **Media Viewing:**

- *Metadata Retrieval*: When a user selects a piece of media to view, the request is first processed through Amazon Route 53, which ensures that the request is efficiently routed to optimize performance. The request then reaches CloudFront, where the specific media metadata is required.
- *API Gateway Integration:* At this point, the Amazon API Gateway acts as a critical intermediary by forwarding the metadata request to the appropriate backend systems. The API Gateway's role is to streamline these interactions, ensuring that they are handled efficiently and securely.
- *Data Management:* The retrieval and management of media metadata are handled by a specialized function known as the Categorizer. This function is designed to efficiently categorize and process media data, facilitating quick access and display of the relevant media details to the user.

System Design and User-Centric Approach

The architecture of this system is fundamentally designed to prioritize user experience. By integrating advanced AWS services, the system ensures that all user interactions, from browsing to media viewing, are optimized for speed and responsiveness. The strategic use of AWS CloudFront and Route 53 enhances the geographic routing of requests, significantly reducing latency and improving content delivery times.

The integration of AWS Amplify further enriches the frontend user experience, providing a powerful platform for media content management and display. This comprehensive approach not only enhances the functional aspects of the application but also ensures that the user experience is seamless and engaging.

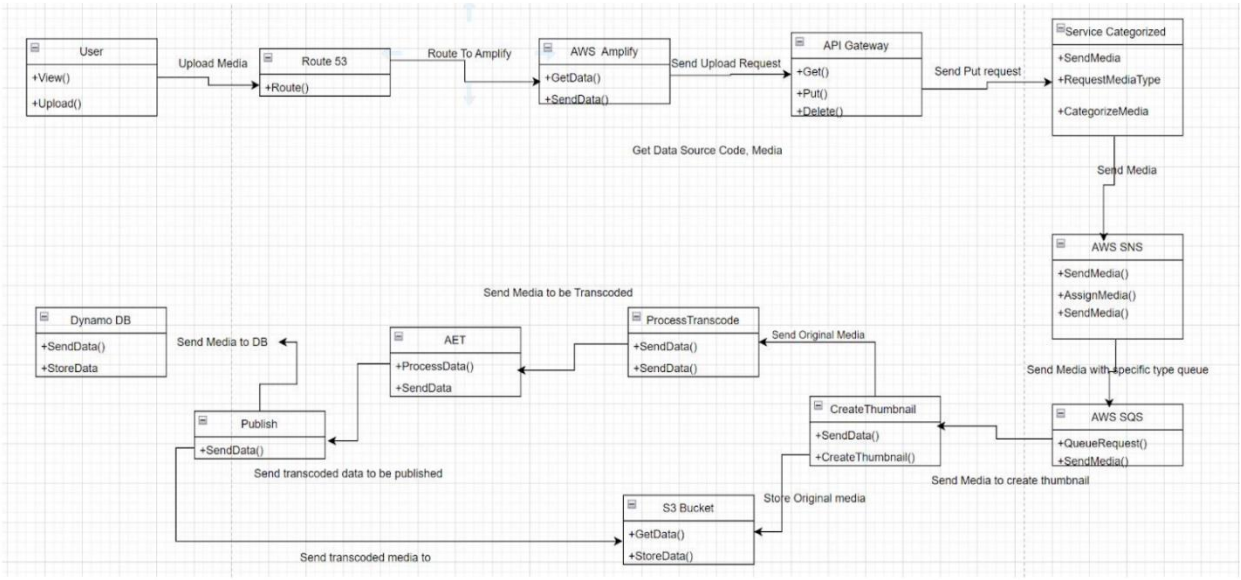

FIG3: MEDIA UPLOAD PROCESS UML

Media Upload Process Overview: Streamlined and Efficient

This section provides a detailed exploration of the media upload process, illustrating the complex steps involved and the crucial role played by various AWS services in facilitating a seamless and efficient user experience.

Process Breakdown

1. **User Initiation:**

- *Initial Routing:* When a user begins the media upload process, the media file is immediately routed through Amazon Route 53 and Amazon CloudFront. This utilizes AWS's robust global network infrastructure to ensure the media reaches AWS Amplify efficiently.
- *Content Handling:* AWS Amplify, serving as the frontend orchestrator, manages the rendering and interaction layer of the application, ensuring that the media upload request is seamlessly integrated into the user interface.

2. **Request Handling:**

- *API Interaction:* AWS Amplify forwards the upload request to a designated API endpoint. This endpoint acts as the operational gatekeeper, initiating the upload process by triggering the PUT method, which starts the media upload sequence.

3. **Categorization and Topic Assignment:**

- *Service Categorization:* After the media file leaves the API's domain, it enters the Service Categorizer function. This function is tasked with a critical classification process, assigning the media file a specific category or "topic" based on its type—be it an image, video, or another format. This categorization is pivotal in directing the media to the appropriate processing queue.

4. **Efficient Routing with AWS SNS:**

- *Topic Utilization:* The categorized media file is then channeled to AWS Simple Queue Service (SQS) through AWS Simple Notification Service (SNS). This setup ensures that the media is queued for processing in an organized manner, accommodating efficient handling even during periods of high demand.

5. **Media Processing and Storage:**

- *Transformation Tasks:* At this stage, various media processing tasks take place. AWS Lambda functions might be invoked to generate thumbnails, while AWS Elastic Transcoder handles more complex transformations like video transcoding.
- *Metadata Management:* Concurrently, metadata associated with each media file is meticulously generated and managed, ensuring that all information is accurate and readily accessible.

6. **Media Storage and Management:**

- *Storage Solutions:* The processed media files, along with any generated thumbnails, are stored in Amazon S3, ensuring high availability and robust data integrity. This dual storage strategy enhances accessibility and maintains a consistent user experience.
- *Database Integration:* Additionally, all associated metadata is systematically stored in Amazon DynamoDB. This not only provides a scalable and efficient storage solution but also ensures that the metadata remains structured and easy to retrieve.

Summary of the Upload Architecture

This detailed depiction of the media upload process illustrates the sophisticated integration of AWS services, highlighting how each component—from initial routing to final storage—plays a vital role in the system's overall functionality. The use of AWS Amplify, Route 53, CloudFront, API Gateway, SNS, SQS, Lambda, Elastic Transcoder, S3, and DynamoDB ensures that every aspect of the media upload is handled efficiently and effectively.

References:

[1]

“Welcome to AWS Amplify Hosting - AWS Amplify Hosting,” *Amazon.com*, 2024. https://docs.aws.amazon.com/amplify/latest/userguide/welcome.html (accessed May 19, 2024).

[2]

“API Management - Amazon API Gateway - AWS,” *Amazon Web Services, Inc.*, 2024. https://aws.amazon.com/api-gateway/ (accessed May 19, 2024).

[3]

“What is Amazon API Gateway? - Amazon API Gateway,” *Amazon.com*, 2024. https://docs.aws.amazon.com/apigateway/latest/developerguide/welcome.html (accessed May 19, 2024).

[4]

“Push Notification Service - Amazon Simple Notification Service - AWS,” *Amazon Web Services, Inc.*, 2024. https://aws.amazon.com/sns/ (accessed May 19, 2024).

[5]

“What is Amazon DynamoDB? (1:01),” *Amazon Web Services, Inc.*, 2024. https://aws.amazon.com/pm/dynamodb/ (accessed May 19, 2024).

[6]
“What is Amazon Route 53? - Amazon Route 53,” *Amazon.com*, 2024. https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/Welcome.html (accessed May 19, 2024).

[7]

“Data protection in Amazon S3 - Amazon Simple Storage Service,” *Amazon.com*, 2024. https://docs.aws.amazon.com/AmazonS3/latest/userguide/DataDurability.html (accessed May 19, 2024).

[8]
“What is Amazon CloudFront? - Amazon CloudFront,” *Amazon.com*, 2024. https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/Introduction.html (accessed May 19, 2024).

[9]
“Overview of Amazon Web Services,” 2019. Available: https://d1.awsstatic.com/whitepapers/aws-overview.pdf