COS20019 CLOUD COMPUTING ARCHITECTURE

Assignment 3

Architecture Design Report

YOU WEE LIEW
102786467

Table of Contents

Table of Figures

1. 0 Introduction

In today's digital landscape, companies face the challenge of building scalable, cost-effective, and secure systems to meet the needs of a growing and global user base. This report outlines the architecture design for a cloud-based media processing application that leverages AWS services to provide a highly available, serverless, and event-driven solution. The application supports features such as media uploads, automatic processing and integration with third-party AI services for tasks such as chatbot interactions and image generation.

The proposed architecture is designed to meet the following key business requirements:

1. **Scalability**: Handle user growth and unpredictable workloads without manual intervention.
2. **Cost Efficiency**: Reduce operational overhead by leveraging AWS managed services and serverless technologies.
3. **Performance**: Fast response times and low-latency content delivery are essential for a global user base.
4. **Reliability**: Data must remain secure, durable, and recoverable in the event of a disaster.
5. **Extensibility**: Support future integration of additional AI services and features.

This report presents the architectural diagrams, describes the AWS services used, explains the workflow for key processes, and provides a rationale for the design decisions based on business needs. By leveraging AWS's serverless and event-driven capabilities, this solution ensures high performance, security, and reliability while addressing current and future business scenarios.

2.0 Architecture Design

This chapter provide an architecture design of the cloud-based media processing application scenario. It includes the architectural diagram, a detailed description of AWS services and activity diagrams that illustrate the workflows for the proposed solution. The architecture is designed to ensure scalability, performance, reliability, security, and cost-efficiency while meeting the specific requirements of the business scenario.

2.1 Architectural Design

The architectural diagram below illustrates the interaction between AWS services, VPC components, and third-party AI services. It includes services for compute, storage, messaging, monitoring, and content delivery, all designed to support a serverless and event-driven workflow.

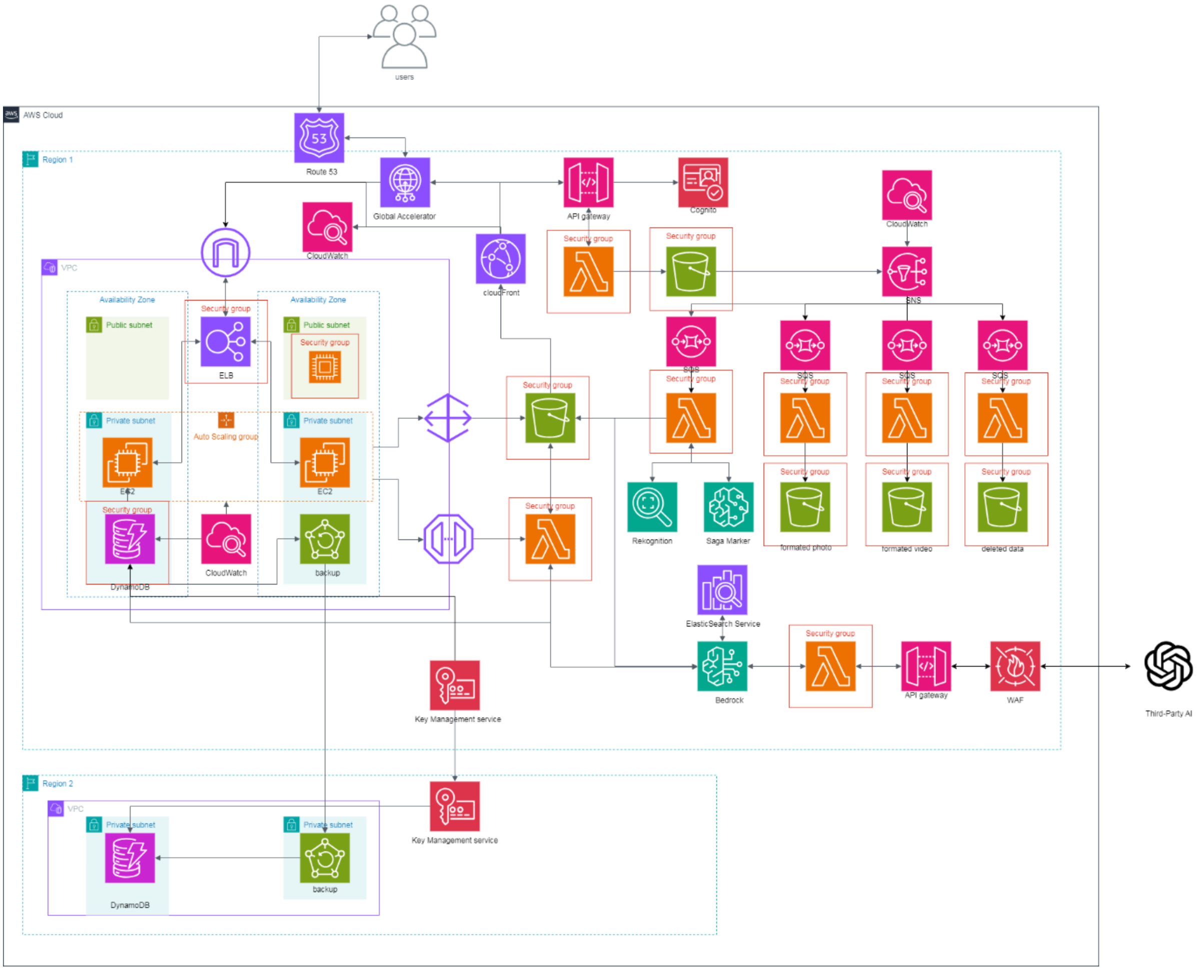

Figure 1: Diagram Architecture

2.2 Description of AWS Services

The architecture leverages a range of AWS services, each fulfilling specific roles to implement the solution. Below is a breakdown of the key services used and their respective functions within the application.

2.2.1 Networking and Traffic Routing

AWS VPC (Virtual Private Cloud)

- The VPC isolates network resources for security and performance. Public subnets host the load balancer and API Gateway, while private subnets secure backend services like EC2 instances and DynamoDB.
- Chosen over a non-VPC setup for enhanced security and fine-grained control over traffic routing.

AWS Route 53

- To the application with high availability and low latency.
- Integrates seamlessly with AWS services like API Gateway and ELB.

AWS Global Accelerator

- Routes user traffic globally over the AWS backbone network to reduce latency and improve performance.
- Selected for its ability to enhance user experience across geographic regions.

2.2.2 Compute Services

AWS Lambda

- Used for running code without having to manage the underneath infrastructure
- Executes serverless tasks, such as resizing images, integrating with third-party AI services, and preprocessing uploaded files.
- Chosen over EC2 for serverless execution and cost-efficiency.

ELB (Elastic Load Balancer)

- Distributes incoming traffic evenly across the Auto Scaling EC2 instances in the backend.
- Provides high availability and fault tolerance by redirecting traffic from unhealthy instances.

Auto Scaling Group (EC2 Instance)

- Handles dynamic backend workloads for applications requiring persistent connections or long-running compute tasks.
- Chosen to support legacy or stateful workloads not suited for serverless.

2.2.3 Storage and Database Services

AWS S3 (Simple Storage Service)

- Stores raw uploaded files and processed files.
- Versioning and lifecycle policies ensure durability and cost-efficiency.

AWS DynamoDB

- NoSQL database used to store metadata for uploaded files, AI labels, and other structured data.
- Chosen over RDS for its serverless nature, automatic scaling, and low latency reads or writes.

2.2.4 Content Delivery and Security

AWS CloudFront

- A content delivery network (CDN) that caches static and dynamic content to reduce latency for global users.
- Chosen for its seamless integration with S3 and its ability to accelerate file delivery.

AWS WAF (Web Application Firewall)

- Protects the application from common web exploits and attacks such as SQL injection and cross-site scripting.

- Configured to filter malicious traffic at the API Gateway and ELB level.

2.2.5 Message Handling and Decoupling

Amazon SQS (Simple Queue Service)

- Handles task decoupling by queuing jobs (e.g., image processing, video transcoding).
- Chosen for its reliable, scalable message queuing capabilities.
- Pull-based processing reduces the risk of overloading the system.

Amazon SNS (Simple Notification Service)

- Sends real-time notifications to users or administrators upon the completion of tasks such as file uploads or processing.
- Enables easy integration with other systems for alerting or automation.

2.2.6 AI and Media Processing

Third-Party AI Services – OpenAI

- Used for chatbot interactions, image generation, and advanced AI features like photo labelling.
- Lambda acts as the integration layer to securely connect to these APIs.

AWS Rekognition

- Provides built-in AI for image and video analysis.
- Used for tagging, object detection, or content moderation.

AWS Bedrock

- Provides access to foundation models for AI tasks like natural language processing, text-to-image generation, and chatbots.
- Ensures scalability and cost efficiency for AI-based features.

AWS SagaMaker

- Supports custom machine learning (ML) model training and deployment for advanced AI tasks.
- Provides flexibility to train and fine-tune AI models for specific use cases, such as video categorization or personalized recommendations.

2.2.7 Monitoring and Observability

AWS CloudWatch

- Monitors all AWS services, logs performance metrics, and triggers alerts for anomalies.
- Essential for ensuring high availability and identifying bottlenecks.

AWS Secrets Manager

- Securely stores API keys and credentials for third-party AI services and database connections.

2.2.8 Identity and Access Management

AWS Cognito

- Provides user authentication, authorization, and user management capabilities.
- Supports features like multi-factor authentication (MFA), federated logins, and user pools to secure access to the application.
- Integrates seamlessly with API Gateway to ensure only authorized users can access APIs.

2.2.9 Backup and Disaster Recovery

AWS Backup

- Centralized backup solution to automate data backups for S3, DynamoDB and EC2 instances.
- Ensures data can be restored in case of accidental deletion or disasters.

AWS KSM (Key Management Service)

- Encrypts backups and data at rest in S3 and DynamoDB for enhanced security.
- Ensures compliance with industry standards for data protection.

2.2.10 Search and Querying

Amazon Elasticsearch Service

- Provides full-text search capabilities and real-time indexing for media metadata.
- Enables users to quickly search for media files based on attributes such as:
  - AI-generated labels (e.g., "beach," "mountain").
  - Metadata like upload date, file type, or user-generated tags.
- Integrates with DynamoDB to index data stored in the database.

2.3 Activity Diagrams

This section outlines the activity diagrams for the media upload and reformatting/transcoding workflows, as well as the chatbot interaction process. These diagrams represent the interaction between users, AWS services, and third-party AI services, illustrating the event-driven nature of the architecture.

2.3.1 Media Upload and Reformatting Workflow

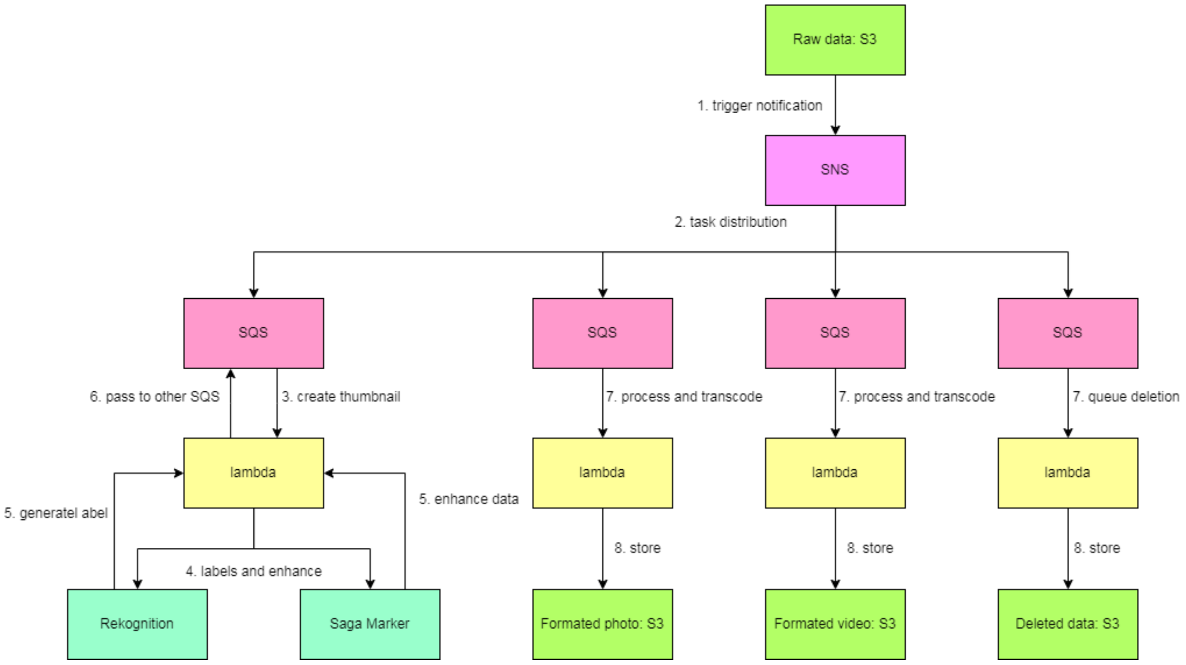

Figure 2: Media Upload and Reformatting diagram

Description

This diagram represents a **Media Upload and Processing System** leveraging AWS services for efficient and scalable data processing. The process begins with raw media data, such as photos and videos, uploaded to an Amazon S3 bucket, which triggers an event notification to Amazon SNS. SNS distributes the notification to multiple Amazon SQS queues, each handling specific tasks like thumbnail creation, media processing, and data deletion. For thumbnail creation, SQS triggers AWS Lambda, which utilizes Amazon Rekognition to generate labels and Saga Marker to enhance or annotate the data. Separate Lambda functions process and transcode photos and videos, storing the formatted outputs in dedicated S3 buckets for photos and videos. Another Lambda function handles deletion tasks, archiving or removing data as needed and storing it in a deleted data S3 bucket. This workflow ensures a modular, secure, and automated system for media processing, with logical task separation and AI integration for labeling and enhancement.

2.3.2 Third-party AI Interaction Workflow

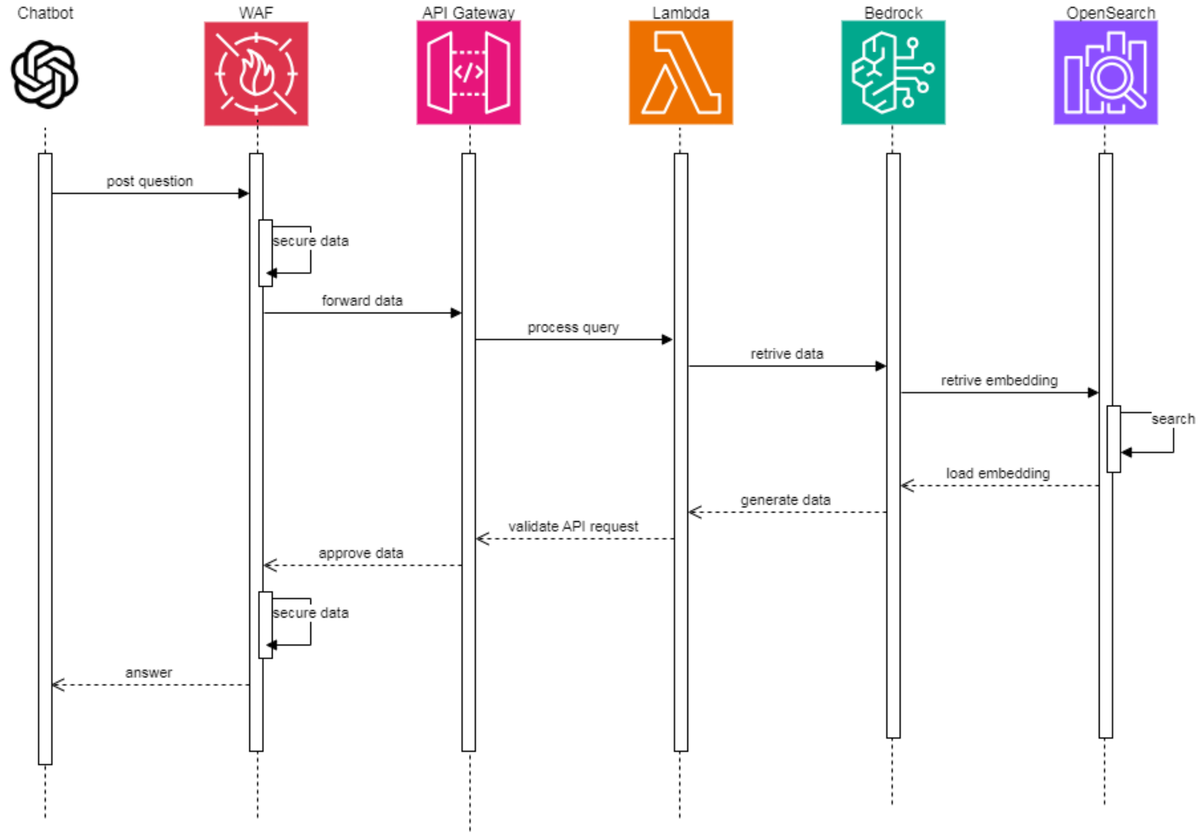

Figure 3: Third-party AI Interaction seqence diagram

Description

This sequence diagram illustrates the process of **Third-party AI Interaction System** for a chatbot system using AWS services. When a user asks a question, it is first sent to AWS WAF for security checks to ensure the input is safe. The query is then passed to API Gateway, which routes it to a Lambda function for processing. Lambda sends the query to Amazon Bedrock, which uses AI models to generate a response. If needed, Bedrock retrieves embeddings from Amazon OpenSearch for additional context or relevance. Once the response is ready, it is sent back through Lambda, API Gateway, and WAF for final validation and security before being delivered to the chatbot. This ensures the system is both intelligent and secure for users.

2.3.3 Backup and Disaster Recovery Workflow

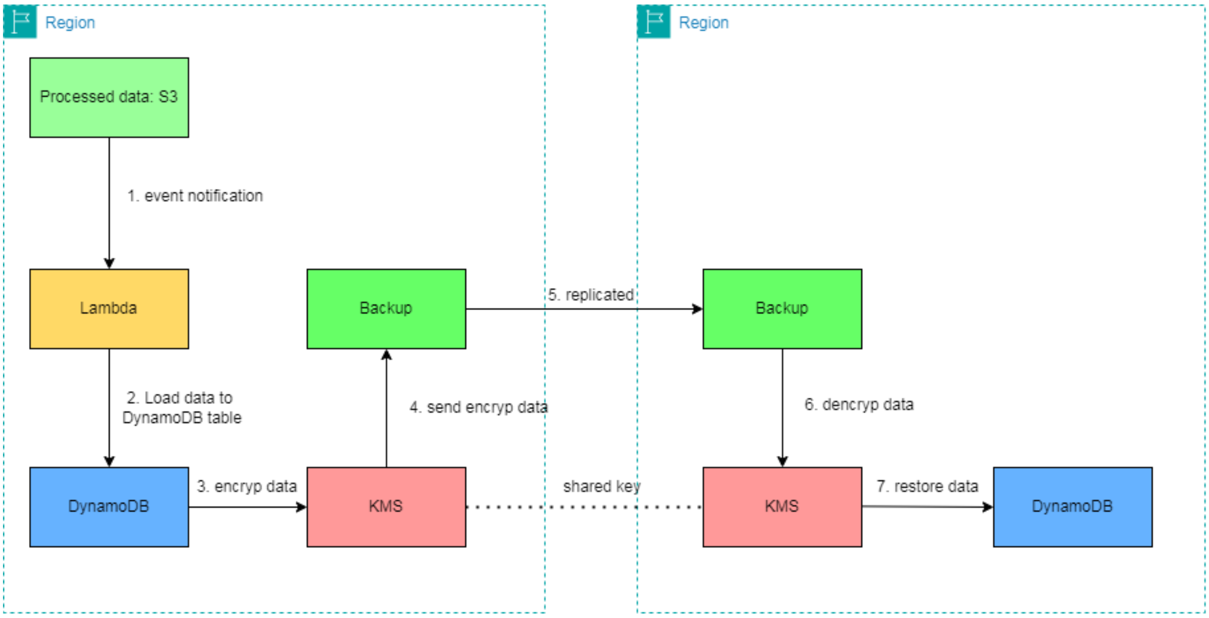

Figure 4: Backup and Disaster Recovery diagram

Description

This diagram represents a robust **Backup and Disaster Recovery System** leveraging AWS services for secure data management across multiple regions. In the primary region (Region 1), processed data is uploaded to an Amazon S3 bucket, which triggers a Lambda function via an event notification. The Lambda function processes the data and stores the structured information in a DynamoDB table. To ensure data security, AWS Key Management Service (KMS) is used to encrypt the data in DynamoDB before it is sent to the AWS Backup service. The encrypted backups are then replicated to the backup service in a secondary region (Region 2) to provide geographical redundancy and disaster recovery capabilities. In the event of a failure in Region 1, the replicated backups in Region 2 can be decrypted using the shared KMS keys and restored to a DynamoDB table in Region 2, ensuring business continuity. This workflow emphasizes secure data encryption, cross-region replication, and high availability, adhering to AWS best practices for disaster recovery and fault tolerance.

2.3.4 User Authentication Workflow

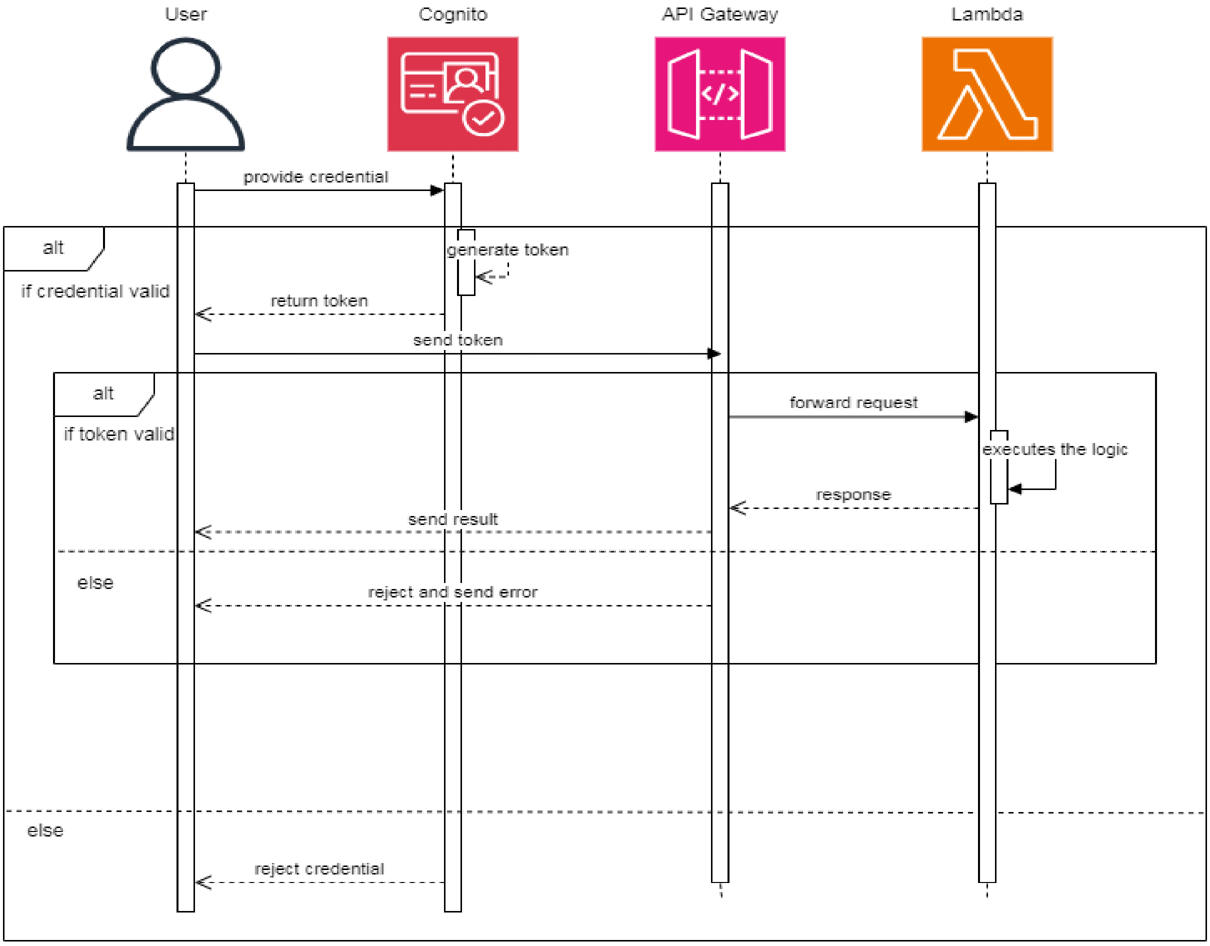

Figure 5: User Authentication sequence diagram

Description

The sequence diagram illustrates the process of secure and scalable **user authentication system** using AWS services such as Cognito, API Gateway and Lambda. The process begins with a user interacting with a web providing their credentials. These credentials are sent to AWS Cognito, which handle user authentication by validating them against its User Pool. Upon successful authentication, Cognito generates a token sends it back to the user. The authenticated user then makes API requests to AWS API Gateway, including the token in the request. API Gateway will verify the token with Cognito ensure that request is authorized. If valid, API Gateway forwards the request to AWS Lambda function, which performs the tasks. Finally, the processed response is sent back to the user through API Gateway. This architecture leverages the strengths of serverless services, ensuring secure, efficient, and scalable user authentication and API handling.

3.0 Design Rationale

This chapter provides the reasoning behind the architecture design and the selection of AWS services, explaining how they meet the requirements of scalability, reliability, performance, cost efficiency, and security. Additionally, it compares alternatives to justify the choices made for each component in the system.

3.1 Fulfilment of Business Scenarios

The proposed architecture effectively addresses the business requirements outlined by the company, leveraging AWS's managed cloud services to deliver a scalable, serverless, and cost-effective solution. Below is a breakdown of how the proposed solution fulfils the key requirements and challenges:

Requirement	Fulfilment
Managed Cloud Services	S3, Lambda, API Gateway, DynamoDB, and other AWS managed services minimize in-house administration.
Scalability for Growth	Serverless architecture with S3, Lambda, DynamoDB, and SQS ensures seamless scaling.
EC2 Performance Limits	Replaced with Lambda for dynamic, scalable compute capabilities.
Cost-Effective Database	DynamoDB offers a scalable and low-cost NoSQL alternative to relational databases.
Global Accessibility	CloudFront and Global Accelerator ensure low-latency access for a worldwide user base.
File Reformatting	Lambda, Rekognition and SageMaker handle reformatting and AI-based processing.
Serverless Solution	Core services are serverless, event-driven, and highly automated.
Data Security and Recovery	KMS, AWS Backup, and cross-region replication provide encryption and disaster recovery.
Third-Party AI Integration	Seamless integration with OpenAI, Bedrock, Rekognition, and SageMaker enables extensible AI features.

1. Managed Cloud Services

- **Amazon S3**:
    - Fulfils the requirement to store files such as images, videos, audio, documents with high durability and scalability.
    - Lifecycle management and cross-region replication eliminate the need for in-house storage administration.
- **AWS Lambda**:
    - Processes uploaded files, creating thumbnails, reformatting images, and triggering other workflows automatically upon events.
    - Reduces operational overhead with serverless execution.
- **Amazon DynamoDB**:
    - Stores metadata for uploaded files in a NoSQL format, providing quick access and low-latency queries without the need for a traditional database server.

2. Scalability for Growing Demand

- **AWS Lambda**:
    - Automatically scales to handle unpredictable and growing workloads, ensuring the system meets user demand as the user base doubles every six months.
- **Amazon S3**:
    - Scales seamlessly to accommodate increasing data storage requirements.
- **Amazon SQS**:
    - Decouples workloads and ensures tasks are processed asynchronously, enabling horizontal scaling of worker nodes for different processing tasks.

3. Compute Limitations

- **AWS Lambda**:
    - Replaces free-tier EC2 instances for lightweight tasks with serverless execution, offering on-demand scaling and cost-efficiency.

4. Cost-Effective Database

- **Amazon DynamoDB**:
  - Offers a scalable and cost-effective NoSQL solution, ideal for storing metadata with a simple structure. Its serverless nature eliminates the need for database provisioning and management.

5. Global Accessibility

- **Amazon CloudFront**:
  - Distributes content to global users with low latency by caching files at edge locations.
- **AWS Global Accelerator**:
  - Routes traffic over AWS's global network for faster and more consistent user experience worldwide.

6. File Reformatting and Extensibility

- **AWS Lambda**:
  - Automatically triggers processing tasks (e.g., image resizing, thumbnail creation) when a file is uploaded to S3.
- **Amazon Rekognition**:
  - Labels and analyzes images using AI for advanced metadata tagging.
- **Amazon SageMaker**:
  - Supports future extensibility by enabling custom AI models for tasks like video categorization or advanced image analysis.

7. Serverless and Event-Driven

- **Amazon SQS**:
  - Provides pull-based messaging for decoupling tasks, ensuring that workloads are processed asynchronously and the application remains event-driven.
- **SNS (Simple Notification Service)**:

  - Sends notifications to trigger workflows, completing the serverless design.

8. Security and Disaster Recovery

- **AWS KMS**:
  - Encrypts data at rest in S3 and DynamoDB, ensuring data security.
- **AWS Backup**:
  - Automates backup management for S3, DynamoDB, and EC2, ensuring data is recoverable in disaster situations.
- **Cross-Region Replication**:
  - Ensures durability and availability of critical data by replicating S3 and DynamoDB data to a secondary region.

9. AI Integration

- **Amazon Bedrock**:
  - Provides foundation models for integrating AI features like chatbots and image generation.
- **Amazon SageMaker**:
  - Enables future extensibility for AI-based features using custom ML models.
- **Amazon Rekognition**:
  - Automates image and video analysis, such as tagging and object detection.

3.2 Comparison of Solutions

This section compares the chosen AWS services and architecture against alternative solutions, evaluating their performance, scalability, reliability, security, and cost-efficiency. The goal is to justify why the selected services are the best fit for the business scenario.

3.2.1 Storage

Chosen Solution: AWS S3

Why Chosen:

- Provides highly durable and scalable storage for all file types (e.g., images, videos, documents).
- Built-in features like versioning, lifecycle policies, and cross-region replication support disaster recovery and cost optimization.

Advantages:

- **Scalability**: Virtually unlimited storage capacity.
- **Durability**: 99% of durability.
- **Lifecycle Policies**: Automatic transition of infrequently accessed data to cheaper storage tiers.

Alternative Solution: AWS EBS

Why Rejected:

- EBS is attached to EC2 instances and lacks the scalability and durability of S3.
- Requires manual management for scaling and backups.

3.2.2 Database

Chosen Solution: AWS DynamoDB

Why Chosen:

- Serverless NoSQL database ideal for storing unstructured metadata.
- Automatically scales to handle varying workloads without provisioning or management.

Advantages:

- **Low Latency**: Fast read or write operations.

- **Global Tables**: Cross-region replication ensures availability and low latency for global users.
- **Cost-Efficient**: Pay-per-use pricing model.

Alternative Solution: AWS RDS

Why Rejected:

- Relational databases are less efficient for unstructured or highly dynamic data.
- Requires provisioning and manual scaling, leading to higher operational costs.

3.2.3 Content Delivery

Chosen Solution: AWS CloudFront and AWS Global Accelerator

Why Chosen:

- **CloudFront**: Delivers static and dynamic content with low latency by caching data at edge locations.
- **Global Accelerator**: Routes traffic through AWS's global network for the fastest possible path.

Advantages:

- **Global Reach**: Improves access speed for users around the world.
- **Security**: Integrates with WAF to protect content delivery.

Alternative Solution: Azure CDN

Why Rejected:

- Lack of seamless integration with other AWS services.
- Requires additional configuration and operational overhead.

3.2.4 Backup and Disaster Recovery

Chosen Solution: AWS Backup and KMS

Why Chosen:

- **AWS Backup**: Automates backup management across services like S3, DynamoDB, and EC2.
- **KMS**: Encrypts backup data for secure storage and transfer.

Advantages:

- **Automation**: Reduces operational overhead.
- **Resilience**: Cross-region replication ensures data availability even in disasters.

Alternative Solution: Manual Backup Scripts

Why Rejected:

- Prone to human error and time-consuming to maintain.

3.2.5 Messaging and Decoupling

Chosen Solution: AWS SQS and SNS

Why Chosen:

- **SQS**: Handles task decoupling with asynchronous, pull-based messaging.
- **SNS**: Distributes notifications for task completion and workflow triggers.

Advantages:

- **Scalability**: Automatically scales to handle increasing workloads.
- **Reliability**: Ensures tasks are processed independently, reducing bottlenecks.

Alternative Solution: Kafka or RabbitMQ

Why Rejected:

- Higher setup complexity and maintenance overhead.
- Not serverless, requiring additional infrastructure management.

3.2.6 2-Tier vs 3-Tier

Chosen Solution: 3-Tier Architecture

Why Chosen:

- **Presentation Layer**: Managed by **API Gateway**, which routes user requests securely and integrates seamlessly with authentication (Cognito) and WAF for traffic filtering.
- **Application Layer**: Handles business logic using **AWS Lambda** processing tasks.
- **Database Layer**: Data storage and querying are handled by Amazon DynamoDB (for metadata) and Amazon S3 (for media storage).

Advantages:

- **Scalability**: Each layer scales independently, allowing dynamic scaling of the application or database tiers without impacting the presentation layer.
- **Security**: Each tier is isolated, with strict controls over access between layers.
- **Performance**: API Gateway and CloudFront optimize content delivery, while DynamoDB and S3 provide fast and reliable data access.

Alternative Solution: 2-Tier Architecture

Why Rejected:

- **Scalability**: Combining the presentation and application layers limits scalability. A surge in user traffic affects both user-facing services and backend processing.
- **Security**: Exposes the application logic directly to user-facing APIs, increasing the risk of attacks.
- **Maintainability**: Changes to the application logic require downtime or careful deployment planning, as both layers are interdependent.

3.3 Criteria-Based Justification

Criteria	Chosen Services and Justification
Performance	Lambda, DynamoDB, and CloudFront ensure fast response times for processing, querying, and content delivery.
Scalability	Serverless and containerized services (Lambda, S3) scale automatically to meet demand.
Reliability	Cross-region replication, DynamoDB Global Tables, and AWS Backup ensure high availability and disaster recovery.
Security	KMS, Cognito, and WAF protect data and manage user authentication securely.
Cost	Pay-as-you-go pricing for Lambda, S3, and DynamoDB reduces operational costs compared to traditional solutions.

4.0 Conclusion

To summarize, this project aimed to develop a cloud architecture that would help to deliver a scalable, cost-efficient, secure, and reliable solution that fulfils all business requirements. By using a serverless and event-driven design, the architecture minimizes operational overhead and ensures the system can adapt to the company's exponential growth and evolving needs.

This architecture is designed to not only meet the current requirements but also scale and adapt to future business needs, ensuring long-term value for the company. By leveraging AWS's managed services, the company can focus on enhancing user experience and delivering innovative features without being burdened by infrastructure management.

References

AWS 2024, ‘What is a CDN (Content Delivery Network)?’, AWS Web Services, viewed on 30 September 2024, < https://aws.amazon.com/what-is/cdn/>.

AWS 2024, ‘What’s the Difference Between Kafka and RabbitMQ?’, AWS Web Services, viewed on 30 September 2024, < https://aws.amazon.com/compare/the-difference-between-rabbitmq-and-kafka/>.

AWS 2024, ‘What’s the Difference Between Relational and Non-relational Databases?’, AWS Web Services, viewed on 30 September 2024, < https://aws.amazon.com/compare/the-difference-between-relational-and-non-relational-databases/>.

Barr, J 2012, ‘SQS Queues and SNS Notifications – Now Best Friends’, AWS Architecture Blog, 21 November, viewed on 30 September 2024, < https://aws.amazon.com/blogs/aws/queues-and-notifications-now-best-friends/>.

Boopathi, V 2024, ‘Amazon CloudFront vs Azure CDN: What should you choose?’, Whizlabs, viewed on 30 September 2024, < https://www.whizlabs.com/blog/amazon-cloudfront-vs-azure-cdn/>.

Chugh, M Komandooru, A Khanuja, M Meneses, F Nargund, P 2024, ‘Build a contextual chatbot application using Amazon Bedrock Knowledge Bases’, AWS Architecture Blog, 19 February, viewed on 25 September 2024, <https://aws.amazon.com/blogs/machine-learning/build-a-contextual-chatbot-application-using-knowledge-bases-for-amazon-bedrock/>.

Hallett, N 2020, ‘EBS vs EFS vs S3 – when to use AWS’ three storage solutions’, Just After Midnight, September, viewed on 30 September 2024, <https://www.justaftermidnight247.com/insights/ebs-efs-and-s3-when-to-use-awss-three-storage-solutions/>.

Konchada, R Kothurkar, A Keshav, G 2021, ‘Field Notes: How to Back Up a Database with KMS Encryption Using AWS Backup’, AWS Architecture Blog, 19 August, viewed on 27 September 2024, < https://aws.amazon.com/blogs/architecture/field-notes-how-to-back-up-a-database-with-kms-encryption-using-aws-backup/>.

Nemeth, A Vergona, F Sharma, V 2024, ‘Building a Three-tier Architecture on a Budget’, AWS Architecture Blog, 25 September, viewed on 30 September 2024, < https://aws.amazon.com/blogs/architecture/building-a-three-tier-architecture-on-a-budget/>.

Walter, M Pendyala, A Sah, D 2021, ‘Get Started with Amazon S3 Event Driven Design Patterns’, AWS Architecture Blog, 27 September, viewed on 30 September 2024, <https://aws.amazon.com/blogs/architecture/get-started-with-amazon-s3-event-driven-design-patterns/>.