Assignment 3A - Project Design

Team members:
Jake Reason (104742474)
Julia Im (102719887)
Mark Saleh (103994313)

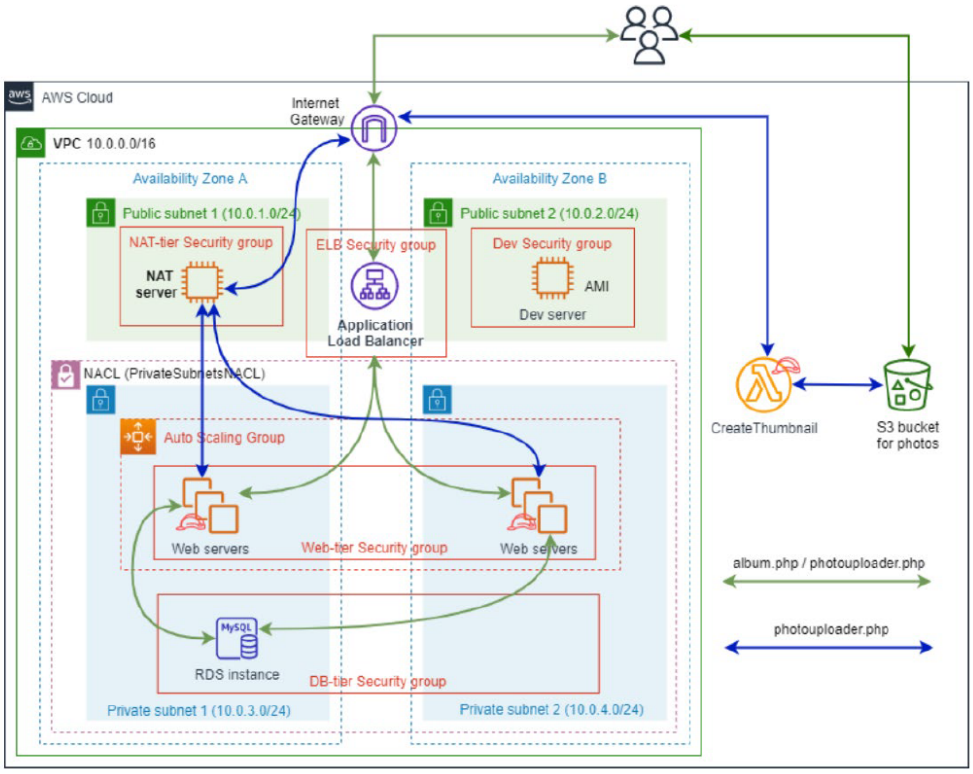

Figure 1 - Architecture diagram

Business, Design & Alternative Rationale

1. *Modify the already existing CreateThumbnail Lambda and other Lambda functions so it includes the Glacier feature.*

Rationale: This allows for long-term data archiving to ensure no data loss/backup.
Design Considerations: This solution is reliable since automatic archiving is less prone to human error. It is cost-effective because lambda functions incur a fee every time a function is triggered, so it is better to integrate tasks into existing ones. It is also secure since data in S3 buckets is highly secure. The cost of adding Glacier to the architecture will be about 0.01$ for retrievals and about 0.03$ pre-upload.
Alternative Rationale: Alternative solutions could be to create another Lambda function instead of integrating the iceberg feature into the CreateThumbnail function, but

having too many Lambda functions could result in redundancy and slow down the server, as well as increase costs. There is also the option of manually archiving but that is error-prone and time-consuming.

2. *Use lambda script to automatically scale architecture. The Lambda function would trigger from CloudWatch when the processors for all instances exceed 80%. The Lambda function will change the properties of the Auto-Scaling Group to have a higher maximum number of instances by 1 for each availability zone. Using CloudWatch, we can also trigger the Lambda function when the processors are below 40% to reduce the maximum number of instances by 1 for each availability zone. This function can not reduce the available instances below 6 (3 in each availability zone). This lambda function can only be triggered every 5 minutes.*

Rationale: This solution addresses the expected increase in usage of the photo album over time.
Design Considerations: This solution ensures the optimal distribution of resources based on current demand. It is cost-effective since the lambda function would rarely be used, and reliable since it reduces the risk of the website's performance degrading with neglect. The cost for the Lambda function is 0.20$ per 1 million requests.
Alternative Rationale: Simply changing the properties of the Auto-Scaling Group was considered, but this will simply push the problem into the future rather than be a permanent solution. It would also not necessarily reflect the actual usage of the photo album.

3. *Create a lambda function to change the instance type to the next availability type. Once it exceeds 10 per availability zone, it scales to a higher instance type. The instance type has been changed from t2.micro to t2.medium as a default.*

   (Scenario 1: if demands decrease down to 6 (3 in each availability zone, <40% - just below ideal standards).
   (Scenario 2: if demands increase up to 50 (25 in each availability zone, >60% - just above ideal standards).

Rationale: A lambda function is a quick and effective way to trigger when a change occurs and moves on to the next rational step.
Design Considerations: This solution allows for increased performance and scalability by moving to the next availability type and therefore creates reliable and secure ways to have available backup. It is also cost-effective as the lambda function is a use-based cost. The cost for the Lambda function is 0.20$ per 1 million requests.

Alternative Rationale: Implement auto-scaling groups to adjust the number of instances and auto-configure policies in response to CPU changes to avoid exceeding the performance limit.

4. *All solutions described are serverless/event-driven solutions.*

Rationale: Serverless architecture offers greater scalability, flexibility and efficient resource management.
Design Considerations: Serverless architecture scales automatically, is more flexible since it responds to specific triggers, and cheaper since only what is used is paid for.
Alternative Rationale: Traditional server-based solutions are not as efficient, require more maintenance and are more costly.

5. *Use Amazon DynamoDB to avoid having to use the relational database.*

Rationale: DynamoDB is designed for large capacities and flexible data arrangement and is cost-effective with rapidly changing data.
Design Considerations: DynamoDB is designed for large volumes of data with simple table values. It is cost-effective, (costing 100 times less than RDS) and can hold data for long-term archiving which makes it secure and reliable. For the cost of using DynamoDB, it would cost about 50$ if we buy 100GB of storage.
Alternative Rationale: Amazon Aurora Serverless is another serverless solution that can scale based on demand and is highly available, suited for scaling databases instantly without disruption of application requests.

6. *Improve global availability by creating a CloudFront distribution.*

Rationale: This solution addresses the need to improve global response times.
Design Considerations: CloudFront delivers content globally by saving it in edge locations, which offers high performance. It also protects from DDOS attacks. It delivers data with high speed and low latency. It is also paid with use which makes the cost low. It will cost us about 0.114$ if we have the 10TB plan.
Alternative Rationale: Without CloudFront, response times depend solely on the server's location. This would read to slower response times outside Australia.

7. *Create a new .php that would serve to be a video uploader. The files would be stored on the same S3 bucket but in a different file location. The meta-data will be stored on DynamoDB in a different table called Videos.*

Rationale: This solution allows for video content to be uploaded and managed efficiently, storing the media in S3 and its metadata in DynamoDB.
Design Considerations: This solution is integrated since it maintains structural integrity whilst adding a separate page for different functionality. Therefore performance, reliability and security are linked to the main foundations of all the AWS components. The addition of the .php does not affect cost. If we had 1.65GBs of video data uploaded an hour, had an average of 300 new connections per minute, and each connection was on the website for an average of 120 minutes, it would cost around 86.51$.
Alternative Rationale: Embedded 3rd party media usage such as YouTube links etc.

8. *Create a new lambda function for video processing.*
    A. *To create different versions of the file depending on its format by calling on EC2 instances.*
    B. *The Lambda functions will be saved somewhere secure so that they can easily be edited and reuploaded.*
    C. *There will be an EC2 instance created for each event. For example, video transcoding would be done by an EC2 instance attached to an AWS Elastic Transcoder.*
    D. *To relieve the overload of nodes and processing transformation jobs, AWS SQS Queues is a fully managed service to reliably and continuously exchange volume to near-infinite scalability.*

Rationale: The extension of the lambda function allows another service such as AWS SQS to be activated and allows the stress load to decrease, which in turn queues the video into processing and creates a separate EC2 instance for this functionality.
Design Considerations: The solution creates another separate sector through another lambda function to prevent any more strain on the EC2 instances and securely processes the videos in another auto-scaling group. It would be securely sent through the internet gateway which filters through the allowed security groups. The cost of adding Elastic Transcoder will be $0.0075/minute, and SQS will cost $0.40 per 1 million for the first 100 billion requests.
Alternative Rationale: Implement AWS Step Functions which dictate each media procedural workflow. It allows extensibility and allows ease of integration for future additional processes.

Architecture Design

Main design

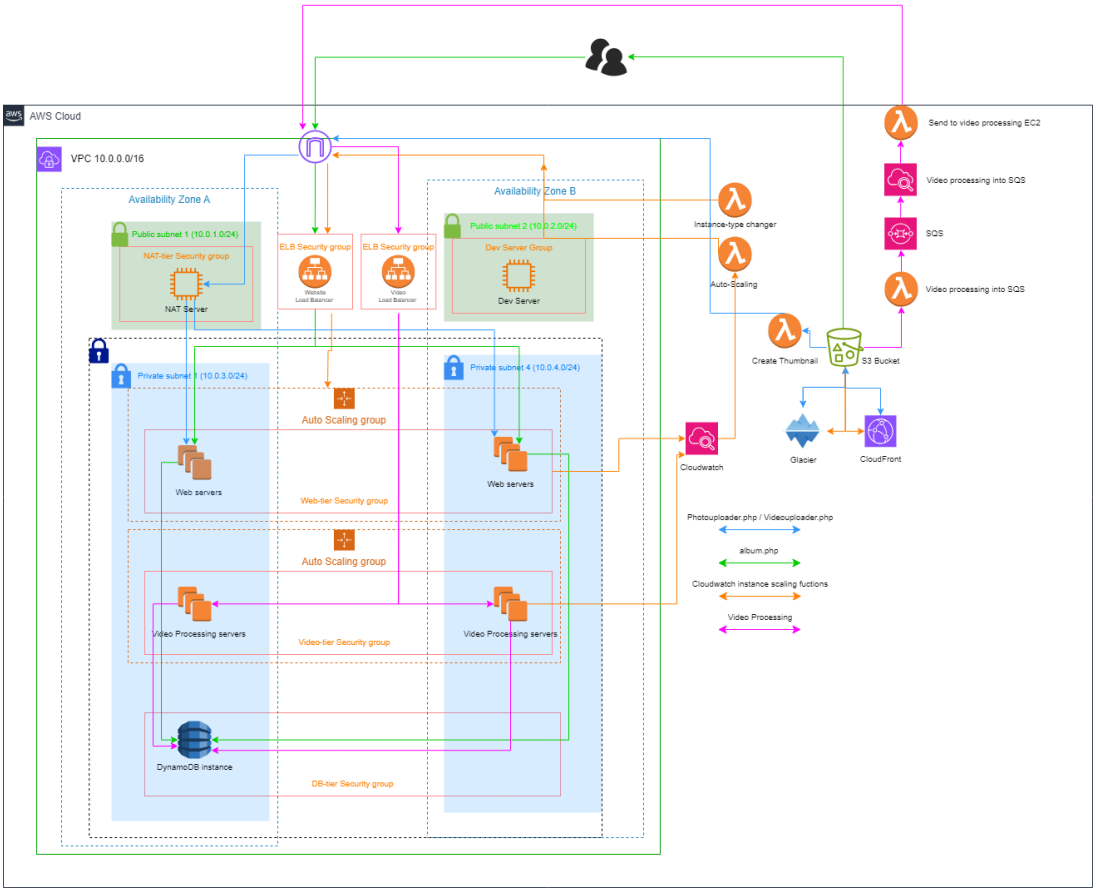

Figure 2. Main Architecture Design

- **Application load balancer** - This is used to have a highly available web server so it doesn't shut down easily. There is another ELB that handles the video processing so the web server isn't stressed.
- **DynamoDB** - This is used instead of RDS as it is more cost-effective and faster.
- **CloudWatch** - This is used to monitor the instances to determine if they need higher scaling based on the percentage of the processor that's being used. CloudWatch is also being used to determine if there are any available instances for video processing.
- **Lambda -** This is used to deploy different functions using Python to process/deliver files.
- **S3 Bucket** - This is used to store all the uploaded files.
- **Glacier** - This is used for long-term file storage when files haven't been accessed for some time. It will fill up space in the S3 bucket and is more cost-effective.
- **CloudFront** - This is for the user to extend S3 bucket resources to edge locations globally.

- **Simple Queue System (SQS)** - This is used to queue videos before processing. This will prevent the video-processing instances from being overloaded.
- **NAT Server** - This is used to give private subnets access to the internet gateway.

Uploading Architecture

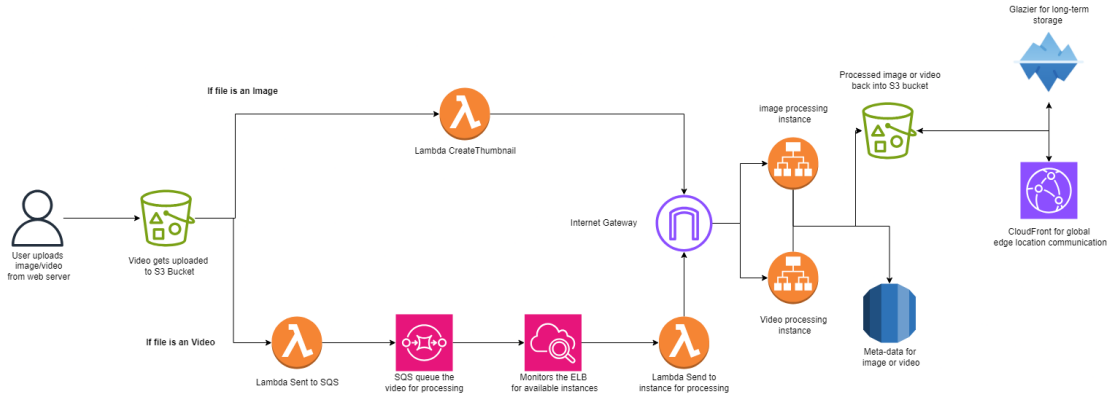

Figure 3. Uploading Architecture

Downloading Architecture

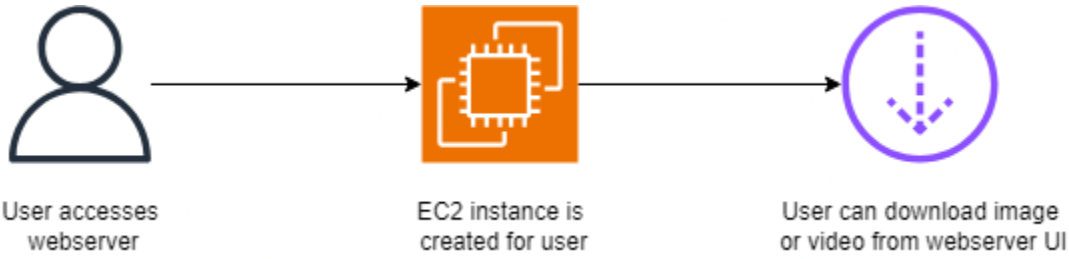

Figure 4. Downloading Architecture

Task Allocation:

Jake Reason

- Architecture Design and Sub designs
- Business Scenario 1-4
- Cost evaluations

Julia Im:

- Business Scenarios 1-5
- Rationale, design considerations and alternative rationale 5-8
- Final evaluation of the document

Mark Saleh:

- Business scenarios 4, 5 and 6, and some of 8
- Rationale, design considerations and alternative rationale 1-4
- Formatting in IEEE Conference Style