Cloud Computing Architecture

Assignment 3

Serverless/Event-driven Architectural Design Report

Due date: 23:59 (VN Time) Sunday, August 4 (end of Week 12), to Canvas.
Late submissions will not be accepted for this assignment.

Weighting: 15%. The assignment needs to be completed to attain a High Distinction in this unit and it is recommended to students with an average mark higher than 50%, achieved from assignment 1 and 2 to submit this assignment.

Presentation Interview (required): Taking place in Week 12. Appointments for presentation will be scheduled and announced during week 11. Go to **Canvas | Quizzes | Assignment 3 Interview** and complete the questions *by end of week 10*.

Prerequisite requirements:

- Successfully completed Assignments 1A, 1B, and 2.
- Completed all ACF labs.
- Explored various AWS cloud services not covered in the labs or lectures.

All supporting materials mentioned in this document can be found in the corresponding assignment page on Canvas.

This assignment can either be completed individually (if repeating student) or in a group of minimum two(2) and maximum three(3) students.
It will be beneficial to have multiple perspectives on a problem when discussing design alternatives.

Additional requirements will apply to group submissions, as indicated below in this document.
To form your group, go to *Canvas/People/Assignment 3* tab and drag and drop your name to the next available group.
Only one student per group should submit the assignment.

Objectives

Examine alternative solutions to develop a design for a scalable highly available Web Site and justify your chosen solution with a design rationale.

NOTE: You may find the Architecture Review Questions in the Week 11 page in Canvas helpful in evaluating your architectural design. There are also many references available on good architectural practice in the cloud such as:

https://aws.amazon.com/architecture/
https://d1.awsstatic.com/whitepapers/aws-web-hosting-best-practices.pdf
https://d1.awsstatic.com/whitepapers/AWS_Cloud_Best_Practices.pdf
https://docs.aws.amazon.com/wellarchitected/latest/framework/wellarchitected-framework.pdf
http://en.clouddesignpattern.org/index.php/Main_Page

Business Scenario

The Photo Album application you developed has met with amazing success and needs to be further developed to meet increasing demand. In particular, the following problems/requirements have been identified by the company:

1. Where possible the company would like to use managed cloud services to minimise the need for in-house systems administration. Photo and other media will be stored in AWS S3.
2. The company is not sure how demand for its application will grow in the future but over recently it has been doubling every 6 months. It expects this trend will continue for the next 2 or 3 years at least and it wants the architecture to be able to cope with this growth.
3. The current system EC2 instances are running on t2.micro. The compute capacity is regularly exceeding the 80% performance limit with 6 instances running. The desired load needs to decrease to between 50 and 60%. Ignore this requirement if your solution does not involve EC2.
4. The company would like to adopt a **serverless/event-driven** solution.
5. The relational database is relatively slow and costly to run. Given the simple table structure, the company would like to explore more cost-effective options.
6. The application has had wide uptake around the world but response time in countries other than Australia has been relatively slow. Global response times need to be improved.
7. It is expected the system will be extended to handle video media in the future.
8. The media can be uploaded by users in all sorts of formats. The company would like various versions of media to be automatically produced (e.g. thumbnails, low resolution versions suitable for mobile phones, or video transcoding). The process for reformatting/transcoding/reprocessing media should meet the following criteria:
   a. When a media item is uploaded to the S3 bucket, the creation of these alternative versions should be triggered automatically. Transformed media will also be stored in S3.
   b. The architecture for processing media should be extensible. For example, in the future it may be desirable to add the ability to automatically identify tags in photos using AI.
   c. Different processing services should be able to be run on the most suitable platform – e.g. EC2 instance, Lambda, or other AWS managed services. Given cost and performance constraints it is assumed that all services will be provided in the AWS ecosystem.
   d. The reprocessing/reformatting of media is often a time-consuming task. The architecture should be designed so that the application does not become overloaded and is effectively **decoupled**. For example, multiple 'worker' nodes can process transformation jobs that have been placed on a queue. The worker nodes may specialise in particular tasks. For example, one node may specialise in video transcoding which is much more processor and memory intensive than reformatting a photograph.

You need to create a report to the client outline a design for the new system and justifying why that design is best.

NOTE: As you do not have access to the complete details of the current system, or full details of future requirements, you need to document any assumptions you make.

Architecture Design

You will create and document a design that meets the above requirements. Your design document will include the following:

1. An architectural diagram showing cloud services being used and their interactions. To draw an architecture diagram, you could use draw.io (www.draw.io/?splash=0&libs=aws4) or any other tools you see fit (www.aws.amazon.com/architecture/icons/).
2. A description and justification of the services used in your design. For example, some of the AWS services you *might* consider using are VPC, CloudFront, Route 53, ELB, AutoScaling, CloudWatch, Lambda, Kubernetes, S3, IAM, Cognito, DynamoDB, Elastic Transcoder, MediaStore, EMR, Rekognition, SQS, SNS, and so on. Alternatively, you may choose to define a non-AWS custom service. You are more than welcome to use AWS services that are not covered in the lectures.
   For more information, refer to Design Rationale section below.
3. UML collaboration diagram(s) showing the interactions between users and service**s**. Order of messages should be shown. Both the media upload and processing use cases should be documented. Ideally, there should be one diagram for each use case. Alternatively, UML sequence diagrams can be used.

Design Rationale

The function of each service within your application must be clearly defined. If appropriate, discuss why you have selected that service over other alternatives, and how that service helps your solution satisfy the **design criteria** and **business scenario**.

The justification should explicitly discuss:

1. How the business scenario is fulfilled using the proposed services.
2. Alternative solutions. For example, you might discuss alternatives such as:
   a. Virtual machines vs. Containers vs. Serverless computing
   b. SQL vs. NoSQL database
   c. Caching options
   d. Push vs. Pull message handling options to promote decoupling
   e. Number of tiers in the architecture, e.g. 3-tier vs 2-tier.
   f. Etc….
3. Design criteria should include:
   a. Performance, scalability (extensibility, decoupling, etc.)
   b. Reliability
   c. Security
   d. Cost **(group submissions should provide a budget with fixed and variable expenses)**
4. Justification for selection of best solution based on the criteria defined.
5. Etc.

The format of your design document must be in **IEEE Conference Style in either one or two column**

mode. You are a solution architect in this assignment, your document must be able to clearly communicate your solution to other stakeholders such as other executives in the company, developers, system admins, etc.

Submission

For this assignment you must:

- A **single PDF document, maximum 15 pages**, in **IEEE Conference Style in either one or two column mode** submitted to Canvas by the due date
- Attend an **interview**, scheduled for week 12 (interviews timetable will be annouced by week 12).

  NOTE: For Assignment 3 interview, students to be prepared for technical questions about their assignment. Students must be able to answer questions by navigating through their assignment 3 report document and its architectural diagram. Optionally, students could prepare a power point presentation file (maximum 6 minutes can be given to students to present during their interview).

Group Submissions: Groups of **maximum 3 students** can be created via *Canvas/People/Assignment 3* tab. Your submission **must** also include the details of the tasks that have been allocated to each individual group member.

Irrespective of task allocations, both team members are expected to be able to understand, analyse and demonstrate all tasks.

Marking Scheme

Architecture Design (7.5)

- High-level description of the architecture including a well-presented architectural diagram that is complete and correct. All cloud services services shown. Scope of service deployments is correct (e.g. inside/outside the VPC, if applicable).
- Detailed description of the function of each service is provided. Service descriptions are adequate. Explain the functionality of service and why it is used.
- Requirements are met. Services fulfill new requirements in well-designed architecture. Describe how decoupling of the architecture is achieved.
- UML collaboration diagram(s) correct and proposed solution meets requirements. Consistant with architectural diagram. Message content identified. Ordering of messages shown. Illustrate both upload and processing use cases.

Design Rationale (7.5)

- Design rationale - justification and comparison provided in terms of
    - Performance, scalability
    - Reliability
    - Security
    - *Cost (**for group submissions**). A detailed cost estimate of the proposed solution is provided*
- Alternative solutions are compared. Where applicable a quantitive comparison should be made.

Deductions (up to -15)

- Poorly presented report.
- Third-party resources not properly acknowledged.