CSCI 5409 - ARCHITECTURE CRITICAL ANALYSIS AND RESPONSE

Group Name – Apes Together Strong

Instructor – Dr. Lu Yang

Dhrumil Amish Shah (B00857606)
Jaspreet Kaur Gill (B00879409)
Nachiket Niranjanbhai Panchal (B00882397)
Samiksha Narendra Salgaonkar (B00865423)

1. Diagram the architecture of your system, make sure your diagram includes:
   1. The flow of data through the system

Response:

The data for the **TravelMemories** application will include the user profile data for all the users of the application and the details of the posts uploaded by the user. Figure 1 displays the data flow in the **TravelMemories** application. The user-related data will be managed by AWS Cognito and the data related to the posts made by the users will be stored in the AWS DynamoDB database. The posts in the database will be linked to the users using userID. The images that the users will upload in the web application will be stored in the AWS S3 buckets and the URL of that image will be stored in the DynamoDB database to link the image to the user post.

   2. All of the components that make up your cloud software

Response:

The components of the **TravelMemories** application will use the following cloud services by Amazon:

1. **Amazon Elastic Load Balancer (ELB):** The Elastic Load Balancer by Amazon is a service that balances the traffic to access the application. In a scenario where traffic is high towards a particular EC2 instance, the load balancer will redirect the traffic to another EC2 instance. If the number of EC2 instances is exhausted, auto-scaling will be performed to scale up the number of instances available to access the **TravelMemories** application.
2. **Amazon Elastic Container Registry (ECR):** The container registry service by Amazon is a Docker container registry that creates the container images and easily stores and deploys them for building containers.
3. **Amazon Elastic Container Service (ECS):** Amazon's ECS helps to build containerized applications from the container images created using the Amazon Elastic Container Registry.
4. **Amazon Elastic Compute Cloud (EC2):** Amazon EC2 will hold the launched containers generated from the AWS Elastic Container Service on various EC2 instances.
5. **Amazon API Gateway:** API Gateway is a service by Amazon that supports the creation, deployment and management of RESTful APIs. In the **TravelMemories** application, we will invoke these created APIs using the frontend HTTP endpoints. We will also integrate our application with our backend HTTP endpoints.
6. **AWS Cognito:** AWS Cognito is a service by Amazon that helps manage user profiles by registering users into the application, logging into the application, changing the password and updating the profile details for the users. The user management module is handled by AWS Cognito.
7. **AWS DynamoDB:** AWS DynamoDB is a database by Amazon that stores all the data related to the posts made by the users of the **TravelMemories** application. AWS DynamoDB is a Non-SQL database that will store the details related to the user posts. Additionally, it will also link the post details to the user who posted it by adding a userID field to the post data. The DynamoDB database will store a URL to the image stored in the AWS S3 bucket.

8. **AWS Simple Storage Service (AWS S3):** AWS S3 buckets will store the images uploaded by the users as a part of their posts. The URL to the uploaded image will be then stored in the DynamoDB database for the respective post.

3. A description of important aspects of the architecture, for example, which components are key and may require high availability?

Response:

AWS Elastic Load Balancer (ELB) and **Amazon Elastic Compute Cloud (EC2)** are two primary services for the **TravelMemories** application as they will govern the performance of the application in terms of handling the load and traffic to the application. AWS ELB will help manage the scaling of EC2 instances of the application depending on the traffic for accessing the application. AWS Cognito and the database components of the **TravelMemories** application are the other important services as they will manage the database and storage component of the application.

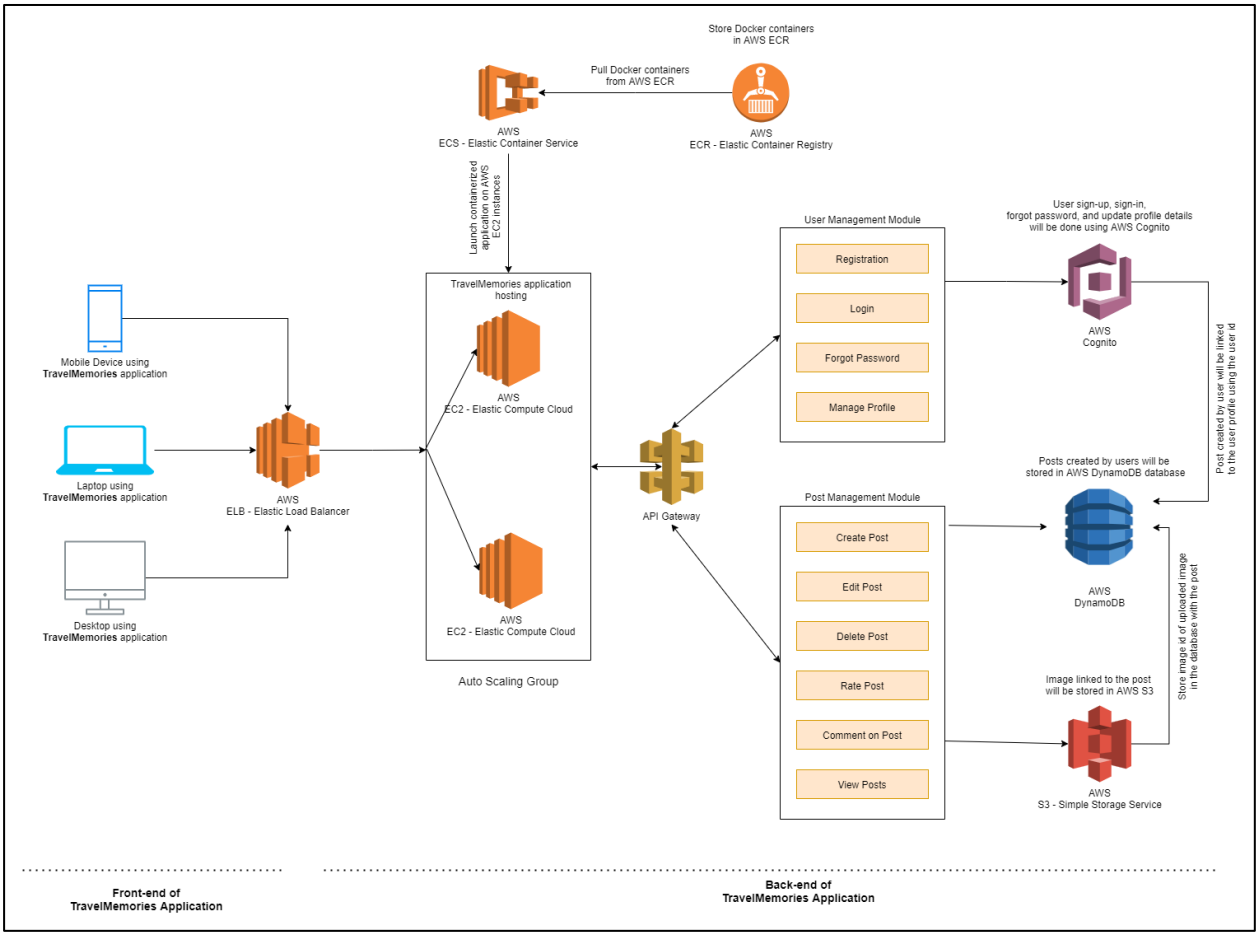

Figure 1 - TravelMemories Application Architecture[1]

2. Determine which architecture you've learned about in class your group's project is most similar to. List it, and then describe any differences you see between your design and the fundamental or advanced architecture from class.

Response:

Our web-application **TravelMemories** is a travel application to help people find tourist attractions of their choice. Therefore, our platform requires the best availability of data stored at an affordable price. We have analyzed all architectures from the lecture and found that they are either used for better availability or low cost.

After a deep dive into architectures, we found that **Cloud Balancing Architecture** is the most relevant with our designed architecture. The **Cloud Balancing Architecture** focuses on the performance, availability and scalability of cloud services. However, this architecture uses an automated scaling listener and uses the Amazon Auto Scaling service to scale our application automatically for optimization.

Rather than depending on the triggers the AWS Auto Scaling service constantly monitors our application and scales the resources frequently based on the usage. This type of scaling helps to increase the resources automatically if our application receives higher traffic. In addition, the resources will be automatically scaled down to normal usage after the heavy traffic is reduced.

3. Are there more appropriate architectures for your program? Discuss alternatives, if there aren't any then compare your architecture to others.
    1. Are there architectures that would be more performant than yours?
    2. Are there architectures that would produce higher availability than yours?
    3. Are there architectures that can scale more easily than yours?

Response:

The most appropriate architecture is the Cloud Balancing Architecture that provides load balancing across multiple clouds enhancing the performance of the application. It uses an automated scaling listener to distribute the workload [2]. **TravelMemories** architecture is also performing load-balancing of the traffic using AWS auto-scaling service

The alternative architectures are listed below:

A. Dynamic Data Normalization Architecture
B. Dynamic Scalability Architecture
C. Service Load Balancing Architecture

Comparing architecture to other architectures:

A. **Dynamic Data Normalization Architecture:** Dynamic data normalization architecture helps to normalize the data stored in the cloud storage devices automatically improving the capacity of the cloud storage devices [3]. This improves the performance of the application hosted on the AWS Cloud. However, **TravelMemories** architecture is not normalizing the redundancy of the data stored in the storage devices provided by Amazon Cloud Service. On the other hand, our architecture is auto-scaling the incoming traffic.

B. **Dynamic Scalability Architecture:** Dynamic scalability architecture allows to dynamically scale horizontally and vertically [4]. Automated scaling provides high availability of the websites. Our application is also performing auto-scaling of the resources based on the traffic. If the traffic is low then it will automatically scale down otherwise scale up using the auto-scaling service of Amazon Web Services.
C. **Service Load Balancing Architecture**: Service load balancing architecture provides scaling of the cloud services by the efficient distribution of the network or traffic over multiple servers providing scalability of resources [5]. **TravelMemories** architecture uses AWS Elastic load balancer for handling the load and the traffic of the application.

References:

[1] "Diagrams.net - free flowchart maker and diagrams online," *Flowchart Maker & Online Diagram Software*. [Online]. Available: https://app.diagrams.net/. [Accessed: 07-Nov-2021].

[2] "AWS Auto Scaling - Amazon Web Services (AWS)." [Online]. Available: https://aws.amazon.com/autoscaling/. [Accessed: 07-Nov-2021].

[3] Arcitura patterns. [Online]. Available: https://patterns.arcitura.com/cloud-computing-patterns/design_patterns/dynamic_data_normalization. [Accessed: 07-Nov-2021].

[4] "Dynamic scalability architecture it is a type of architecture where the: Course hero," *Dynamic Scalability Architecture It is a type of architecture where the | Course Hero*. [Online]. Available: https://www.coursehero.com/file/p40srh9a/Dynamic-Scalability-Architecture-It-is-a-type-of-architecture-where-the/. [Accessed: 07-Nov-2021].

[5] "Service load balancing - amazon elastic container service." [Online]. Available: https://docs.aws.amazon.com/AmazonECS/latest/developerguide/service-load-balancing.html. [Accessed: 07-Nov-2021].