MECHANISMS CRITICAL ANALYSIS AND RESPONSE

CSCI 5409: CLOUD COMPUTING

Summer 2022

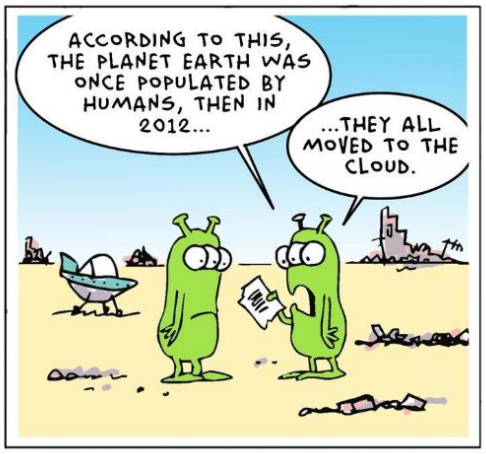

ACCORDING TO THIS, THE PLANET EARTH WAS ONCE POPULATED BY HUMANS, THEN IN 2012...
...THEY ALL MOVED TO THE CLOUD.

Image source: AT&T cybersecurity blogs

Prof. Robert Hawkey,
Instructor & Faculty at Computer Science

Assigned TA:
Kamran Awaisi

GROUP 16: CLOUDVENGERS

Members:

Ruhi Rajnish Tyagi
B00872269, rh679297@dal.ca

Adil Dinmahamad Otha
B00900955, ad842343@dal.ca

Saurabh Jayeshbhai Das
B00911733, sr850847@dal.ca

INDEX

1. <u>MECHANISMS AND SERVICES USED:</u>

CATEGORY	SERVICE	APPLICATION FEATURE
Compute	• AWS Elastic Container Service • AWS Lambda	• AWS ECS will host web application and support microservices. • In our scenario, the students can use email notifications provisioned by Lambda to share their test results.
General	• AWS SNS • Amazon Lex • Amazon Cognito	• For students to send emails to their parents/guardians, we need an email service which is provided by AWS SNS. • AWS Lex will support the teachers in making familiar to the application use. • Cognito will allow signup and login for teachers as well as students.
Network	• AWS API Gateway	• AWS API Gateway enables access to AWS services and data stored in the AWS Cloud.
Storage	• AWS S3 • AWS DynamoDB	• AWS S3 will store static files such as images. • It will act as a persistent database like storing application related data like user details, marks on quiz and so on.

2. <u>DESCRIPTION OF SERVICES</u>:

2.1 AWS Elastic Container Registry (ECR) and Elastic Container Service (ECS):

<u>Role of Service:</u>

Amazon Elastic Container Service (Amazon ECS) is a highly scalable and efficient container management service for managing containers in a cluster [1]. With Docker Containers now becoming a universal software delivery tool, ECS provides a robust platform to deploy containerized applications on the cloud.

ECS provides 2 options or models for running our containers [1]:

Fargate launch type - This is a serverless pay-as-you-go option. You can run containers without needing to manage your infrastructure.

EC2 launch type - Configure and deploy EC2 instances in your cluster to run your containers.

We will be using the ECS's **Fargate** option to deploy our containerized frontend application developed using Angular.

<u>Alternatives:</u>

Elastic Compute Cloud (EC2):

Our application can also be deployed by setting up EC2 instances. We could pull our application's Docker image and configure an EC2 instance to run the container.

Elastic Beanstalk:

Elastic Beanstalk allows us to quickly deploy and manage our application without the need to know about the services that runs this process [2]. To deploy our application, based on our selected platform, Elastic Beanstalk will automatically provision the necessary resources, such as EC2 instances.

Reason for choosing ECS:

ECS internally uses EC2 instances to launch the containers. However, with the serverless Fargate option, the server will only become active (run the code) when it receives the requests. This is known as a "cold start" and may slow the initial response time by a couple of seconds, but the cost savings are quite high compared to an EC2 instance that is running 24/7. Elastic Beanstalk has a similar drawback, as it will internally use EC2 instances to deploy our application which can result in heavy costs.

Besides this, ECS saves us the effort of manually configuring Docker on an EC2 instance and orchestrating them. ECS will manage the orchestration process along with the support for third party services such as AWS Elastic Container Registry (ECR) for pushing our Docker images. Also, it can automate the container build and deploy process through a CI/CD pipeline.

ECS with EC2 launch type option is also present for organizations that require more control over their container configurations and allows them to manually deploy EC2 instances in a cluster.

2.2 AWS Lambda

Role of Service:

Lambda is a compute service that lets us run code without provisioning or managing servers [3].

We will be using AWS Lambda to deploy our Backend application functions/APIs developed using NodeJS. These functions will be triggered by our Frontend application.

Alternatives:

Elastic Compute Cloud (EC2):

Our Backend Application can also be configured and deployed by configuring an EC2 instance and installing the necessary dependencies such as Node and Express. The server will run our code 24/7 to handle incoming requests.

Elastic Beanstalk:

As mentioned previously, Elastic Beanstalk could also serve as an alternative to deploy our application without the need to choose the internal AWS Services to run this process.

Reason for choosing AWS Lambda:

AWS Lambda runs your code on a high-availability compute infrastructure and performs all the administration of the compute resources, including server and operating system maintenance, capacity provisioning and automatic scaling, code monitoring and logging [3]. All we need is a language that is supported by AWS Lambda (such as NodeJS).

The main benefit of this service is that the server will only be active (run our code) when it receives the request and will scale automatically depending upon the number of requests. Therefore, its pay-as-you-go option facilitates high-cost savings as we are only charged when the code is running.

As mentioned earlier, EC2 will enable us to manually set up a server by installing the necessary dependencies which runs 24/7. Elastic Beanstalk serves the same purpose but without the need to worry about the internal infrastructure. However, 24/7 runtime of these servers may incur much higher costs.

2.3 AWS CloudFormation:

Role of Service:

AWS CloudFormation is a service to model our AWS infrastructure using a code template that describes the AWS resources our system needs [4]. It will take care of provisioning all these resources based on the template. Hence, it allows us to set up our entire infrastructure using code.

Alternatives:

Terraform by HashiCorp:

This is a similar service that enables infrastructure automation for provisioning of resources of any cloud. It supports multiple cloud providers such as AWS, GCP and Azure.

Manual Infrastructure set up using AWS Console:

We could use the AWS Console UI for setting up the necessary resources for our system and then configuring them to work together.

<u>Reason for choosing AWS CloudFormation:</u>

Normally, for an application, we configure individual AWS services using AWS Console UI and then configure them to work together. Instead of this approach, to reduce the complexity and time required, we could create a CloudFormation template or use an existing one that describes all the services along with their properties.

Besides this, there may also be a scenario where we need to upgrade some of our resources to a high performing one. So, normally, we upgrade the necessary resources and in case of failure, we roll them back. However, CloudFormation allows us to write text files that contain our template and track them as we track files in a version control system (VCS). This allows us to track the exact changes, roll back to a previous version, who and when these changes were made.

Terraform, even though it supports multiple cloud providers, has a steep learning curve [5]. Also, in the event that a new AWS Service is launched, it may take some time for Terraform to support it. Besides this, since our academic project only uses AWS Services, AWS CloudFormation will be a good option to maintain consistency and ensure that adequate support is provided for setting up an Infrastructure as a Code.

2.4 AWS S3 [6]:

<u>Role of Service:</u>

The app will be interactive to engage learners. To foster better design, we would be using images, videos, and animations. A part of the app will also host a video explaining how teachers can make quizzes. To serve these static assets well and good, we'll be using the S3 service offered by AWS.

Alternatives:

Wasabi:

Wasabi can be considered as an alternative given how cheap the pricing for the storage service that they offer is. For under a TB, they charge just $ 5.99 / month (Prices are in USD). They also mention that there's no extra cost for data egress or API requests. They compete directly with S3 on their promotional site. They also claim a durability of 11 9s which is phenomenal at least on paper unless experienced using it [7].

Azure Blob Storage

Microsoft's Azure offers the Blob storage mechanism using which we ca store the static files needed for our app. The offering's price is at par with the AWS's S3. Moreover, the offerings have many tiers among which we can select as per the choices [8].

Reason for choosing AWS S3:

Wasabi's offering is the best in price as compared to any major cloud providers out there in the market. However, sticking to AWS means that our project will have a homogenous development environment. Switching to Wasabi would mean we would have to hit their APIs using the API offered. More packages and dependencies mean handling more load on the server. Azure offering is at par with AWS and gives no better edge on AWS to think of switching over.

2.5 AWS Cognito:

Role of Service:

All the users in the system be it teachers or the students would register their accounts with the system using the Cognito service. It will store all the required data attributes of both the user types [9].

Alternatives:

Auth0:

Auth0 is a leader in authentication. The service offers a secure way to handle user logins and access the customer information [10].

Google Firebase Authentication:

Firebase' ease of access is what makes it a competitor for authentication needs. The UI is lucid and functionality is just enough to get the job done [11].

Google Cloud Identity Platform:

It takes the authentication via multi-factor levels. Comes with enterprise-grade support. The pricing is similar to Cognito [12].

Reason for choosing AWS Cognito:

Firebase's cost is more than both Cognito and GCP Identity platform. Auth)'s pricing is way too costlier than all the CIAMs (Customer Identity and Access Management) offerings. Cognito offers custom UI for the login screen thereby adding in a layer of design to the use case. There's no much cost difference in others and Cognito hence we thought of going ahead with Cognito for easier integration with our other AWS modules.

2.6 AWS SNS:

Role of Service:

SNS will act as the notification service in our app. With its heavy interaction, SNS will send notification to students when a new quiz is uploaded. Also, just like Dal's brightspace marks release mechanism, it will shoot up a notification when marks for a quiz are up (instantaneously). It will also be used to inform the parents / guardians of the students when the teacher wants to individually send the progress report of each student [13].

Alternatives:

Firebase Cloud Messaging:

This offering from GCP is used to send in-app notifications to the users of the app. It can transfer a payload of upto 4000 bytes to the client app [14].

Azure Notification Hubs:

It offers us to define customer segments to custom-tailor our needs of individual user bases [15].

Reason for choosing AWS SNS:

Firebase's Cloud messaging is free, and Google is very generous to offer this at no extra-cost. However, SNS is used to trigger flows even within the system, i.e. with SNS we can trigger a Lambda. However, for doing the same thing with GCP and Azure, there are different offerings. Both the offerings listed above are just for in-app notification services. Also, the bottom-most tier of the SNS offering is more than sufficient given the scope of our project.

2.7 API Gateway:

Role of Service:

It acts as a doorway to the backend of the app. All incoming requests will be routed through this service. For our app, teachers and students will be constantly triggering events in the UI such as creating quizzes, attempting them, and as also updating their profiles. These operations will then trigger REST API's that will serve the functionalities and mostly persist data in the database [16].

Alternatives:

GCP API Gateway:

It is very much similar to AWS' offering and is a fully managed service for APIs. It costs nothing up to 2 million requests and $3 per million calls [17].

Azure API Management pricing:

The offering is close to GCP and AWS, but the pricing is made available in different tiers rather than just one listing [18].

Reason for choosing API Gateway:
There's a subtle difference in the price of all the three offerings mentioned above. However, to keep things under the same environment and better control the code through just one unified API, we thought of using the API Gateway from AWS.

2.8 AWS Lex:

Role of Service:

Amazon Lex is a fully managed artificial intelligence (AI) service that lets you design, build, test, and deploy conversational interfaces in apps using advanced natural language models [19].

Alternatives:

Amazon ChatBot:

AWS Chatbot is a tool that allows your team to swiftly respond to operational concerns, security events, deployment procedures, and any other notifications for applications running on your AWS accounts. It enables teams to collaborate in real time to monitor and resolve difficulties [20].

Reason for choosing Amazon Lex:

Since from our application perspective, AWS Lex will support the teachers in making familiar to the application use. We are not using any application services where we must retrieve any data or notifications from them. Our chatbot is simply there to guide the teachers in how to make quizzes, make them live and see the progress reports of attended exams.

While on other hand Amazon Chatbot is more concentrated in resolving team difficulties by getting notifications from other services like Slack and Chime Chat. And above all, since we are not using Slack or Chime chat, we cannot use this service.

2.9 AWS DynamoDB:

Role of Service:

Web apps generate huge amount of unstructured data. This is because of the increased interaction of user with the system and thus the data is so variable that storing it using NoSQL DBs makes more rational than SQL DBs. DynamoDB is a key-value and document type database that will be used to store all the app data for the project [19].

Alternatives:

Amazon DocumentDB:

Web apps generate huge amount of unstructured data. This is because of the increased interaction of user with the system and thus the data is so variable that storing it using NoSQL DBs makes more rational than SQL DBs. DynamoDB is a key-value and document type database that will be used to store all the app data for the project [21].

AWS RDS:

AWS RDS is a relational alternative for DynamoDB. Amazon Relational Database Service (Amazon RDS) is a collection of managed services that makes it simple to set up, operate, and scale databases in the cloud. Choose from seven popular engines — Amazon Aurora with MySQL compatibility, Amazon Aurora with PostgreSQL compatibility, MySQL, MariaDB, PostgreSQL, Oracle, and SQL Server — and deploy on-premises with Amazon RDS on AWS Outposts. [22]

Reason for choosing Amazon DocumentDB:

DynamoDB will act as a persistent database like storing application related data like user details, marks on quiz and so on. As our application is not going to handle complex data that have high workload, we ruled out to use DocumentDB. DynamoDB is a key-value store-based NoSQL database. It is less expensive than DocumentDB, but because most of the data is unique key-value stores, it performs better than DocumentDB, which allows indexes to be added to the tables. DynamoDB also outperforms DocumentDB in terms of performance [19].

About RDS, when compared to the cost of using RDS, DynamoDB has proven to be a more cost-effective solution. In addition, the schema flexibility of a NoSQL database is what we desire.

2.10 Amazon Secrets manager:

Role of Service:

All database credentials and API keys can be securely saved using this service. AWS secrets managers assist you in safeguarding the secrets required to access an application. You can use this service to get access to your credentials whenever and wherever you need them. This makes use of the "Secrets Manager API," which eliminates the requirement for sensitive data to be hardcoded in code [19].

Alternatives:

Amazon Key Management Services:

AWS Key Management Service (KMS) is a managed service that makes it easy for you to create and manage keys and control the use of encryption across a wide range of AWS services. KMS is a secure and resilient service that uses FIPS 140-2 validated hardware security modules to isolate and protect your keys [23].

Reason for choosing Amazon Secrets manager:

One of the reasons for using AWS secrets manager is that it simplifies the process of encrypting keys and allows users to manage their credentials in a transparent manner. Another reason is that AWS Secrets Manager has a decent set of APIs that are compatible with languages like Java, Python and several other.

3. PRICING:

3.1 AWS Elastic Container Registry (ECR) and Elastic Container Service (ECS):

ECS does not have additional charges, but we only pay for the resources we use based on the launch type we select: **Fargate** or **EC2 [24]**.

We will be using the Fargate option for our system. AWS Fargate pricing is based on the vCPU, memory, OS, CPU architecture and storage resources consumed from the time the container image is downloaded (Docker pull) until the ECS task terminates.

Pricing Details:

Region: US-east (Ohio)

Operating System: Linux x86

Per CPU per hour: 0.04048 USD

Per GB per hour: 0.004445 USD

20 GB of ephemeral storage (volatile temporary storage) is available for all Fargate Tasks by default. The price for the additional storage is:

Per storage GB per hour: 0.000111 USD

3.2 AWS Lambda

The pricing is based on the number of requests for your functions and the duration it takes for your code to execute. The duration is measured in GB-seconds: the number of seconds your function runs for, multiplied by the amount of RAM memory consumed [25].

The AWS Lambda free tier includes one million free requests per month and 400,000 GB-seconds of compute time per month.

<u>Pricing Details:</u>

Architecture: x86

Duration: $0.0000166667 for every GB-second

Requests: $0.20 per 1M requests

Ephemeral Storage: $0.0000000309 for every GB-second

3.3 AWS CloudFormation [26]

This service is free to use, and we only pay for the services we mention in our CloudFormation template.

3.4 AWS S3:

The pricing for S3 depends on the amount of storage capacity stored. It is also dependent on the region. For us-east-1 here are the details of the pricing:

On the standard tier, the first 50TB / month cost $0.023 / GB. The standard tier is the one that is mostly used in apps unless specifically some requires an even more granular control on the data stores like monitoring and fast access. The cost per GB decreases as the capacity is more. For 450 TB / month, the cost is $0.022 / GB whereas for 500 TB / month its $0.021 / GB [27].

3.5 AWS Cognito:

Cognito offers a very generous tier. The unit for cost is MAU (Monthly Active Users) The first 50,000 users are free and then after that it costs $0.0055 for each MAU up to 100,000 users. Similar to S3, the cost per MAU decreases with the number of users increasing. The next 900,000 users cost $0.0046 and then the next 9,000,000 cost $0.00325. The cap is if the MAU are more than 10,000,000 it costs $0.0025 [28].

3.6 AWS SNS:

SNS's unit of calculating the cost is based on the type of end point that it send the notification too. For mobile push notifications, it costs $0.50 per million notifications. Whereas for emails each 100,000 notifications cost $2.00. For sending notifications over HTTP it costs $0.60 per million notifications. For sending notifications to Simple Queue Service (SQS) and AWS Lambda doesn't cost anything. However for sending notification to Amazon Kinesis Data Firehose costs $0.19 per million notifications [29].

3.7 AWS API Gateway:

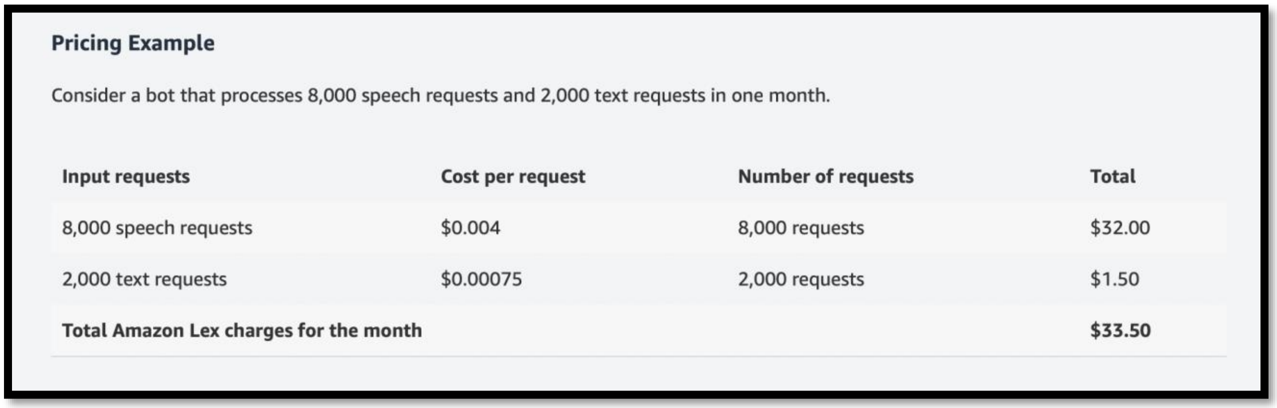
Cost of the API gateway service is dependent on the number of requests made. For the us-east-1 region, it costs $1.00 per million requests for the first 300 million. For 300+ million requests it costs $0.90 per million.

For REST API calls, we only pay for the amount of data transferred out. For the first 333 million requests, it is $3.50 per million.

We can even cache our responses in case we need to speed up our responses for the API calls. Caching is charged as per the memory used and the time for which the caching was used. For 0.5 GB of cache, it costs $0.02 per hour. This goes all the way upto $3.80 per hour for 237.0 GB [30].

3.8 Amazon Lex:

Region: US East (Virginia)

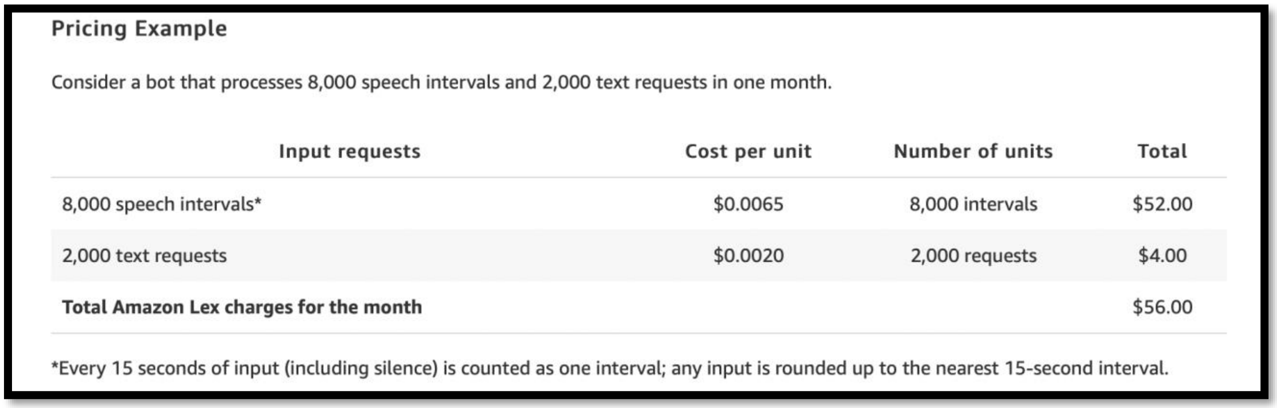


Pricing Example

Consider a bot that processes 8,000 speech requests and 2,000 text requests in one month.

Input requests	Cost per request	Number of requests	Total
8,000 speech requests	$0.004	8,000 requests	$32.00
2,000 text requests	$0.00075	2,000 requests	$1.50
Total Amazon Lex charges for the month			**$33.50**

Figure 1. *Pricing of lex based on speech/text requests for one month-1 [31].*



Pricing Example

Consider a bot that processes 8,000 speech intervals and 2,000 text requests in one month.

Input requests	Cost per unit	Number of units	Total
8,000 speech intervals*	$0.0065	8,000 intervals	$52.00
2,000 text requests	$0.0020	2,000 requests	$4.00
Total Amazon Lex charges for the month			$56.00

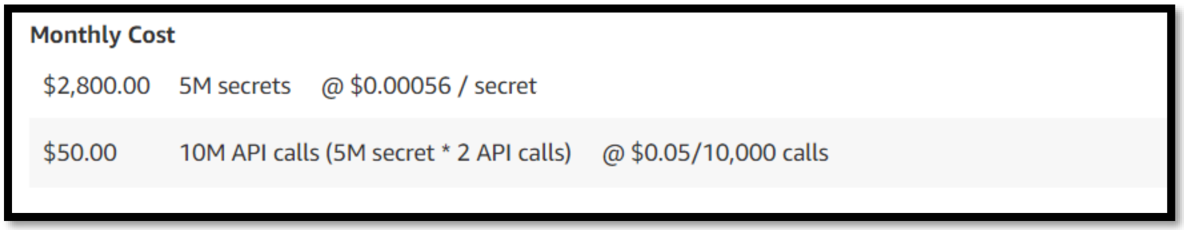
*Every 15 seconds of input (including silence) is counted as one interval; any input is rounded up to the nearest 15-second interval.

Figure 2. *Pricing of lex based on speech/text requests for one month-2 [31].*

3.9 Amazon DynamoDB:

Type: DynamoDB provisioned capacity
Data Storage Size: 1 GB
Average Size: 375 Bytes
Reserved Capacity for read and write: 0%
Cost (per month): **11.41 USD** [31]

3.10 Amazon Secrets Manager:



Monthly Cost

$2,800.00	5M secrets	@ $0.00056 / secret
$50.00	10M API calls (5M secret * 2 API calls)	@ $0.05/10,000 calls

Figure 3. *Pricing of Secrets Manager based on requests for one month [31].*

4. REFERENCES:

[1] "What is Amazon Elastic Container Service?" [Online]. Available: https://docs.aws.amazon.com/AmazonECS/latest/developerguide/Welcome.html. [Accessed: 11-Jun-2022].

[2] J. van. Vliet, "Elastic beanstalk," Amazon. [Online]. Available: https://docs.aws.amazon.com/elastic-beanstalk/index.html. [Accessed: 11-Jun-2022].

[3] R. W. Hendrix, "Lambda," Amazon. [Online]. Available: https://docs.aws.amazon.com/lambda/latest/dg/welcome.html. [Accessed: 11-Jun-2022].

[4] "What is AWS cloudformation? - docs.aws.amazon.com." [Online]. Available: https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/Welcome.html. [Accessed: 11-Jun-2022].

[5] A. Valdes, "Terraform vs CloudFormation: The final battle," ClickIT, 07-Mar-2022. [Online]. Available: https://www.clickittech.com/devops/terraform-vs-cloudformation/#:~:text=While%20CloudFormation%20is%20confined%20to,most%20of%20the%20AWS%20resources. [Accessed: 11-Jun-2022].

[6] "Cloud Object Storage – Amazon S3 – Amazon Web Services," Amazon Web Services, Inc., 2022. [Online]. Available: https://aws.amazon.com/s3/. [Accessed: Jun. 13, 2022].

[7] "Wasabi Cloud Storage Pricing vs. S3 vs. Azure vs. Google Cloud," Wasabi, May 10, 2022. [Online]. Available: https://wasabi.com/cloud-storage-pricing/#cost-estimates. [Accessed: Jun. 12, 2022].

[8] "Azure Blob storage | Microsoft Azure," Microsoft.com, 2022. [Online]. Available: https://azure.microsoft.com/en-ca/services/storage/blobs/#overview. [Accessed: Jun. 12, 2022].

[9] "Amazon Cognito - Simple and Secure User Sign Up & Sign In | Amazon Web Services (AWS)," Amazon Web Services, Inc., 2022. [Online]. Available: https://aws.amazon.com/cognito/. [Accessed: Jun. 12, 2022].

[10] "Auth0: Secure access for everyone. But not just anyone.," Auth0, 2013. [Online]. Available: https://auth0.com/. [Accessed: Jun. 12, 2022].

[11] "Firebase Authentication | Simple, no-cost multi-platform sign-in," Firebase, 2022. [Online]. Available: https://firebase.google.com/products/auth. [Accessed: Jun. 12, 2022].

[12] "Identity Platform | Google Cloud," Google Cloud, 2022. [Online]. Available: https://cloud.google.com/identity-platform. [Accessed: Jun. 12, 2022].

[13] "Amazon Simple Notification Service (SNS) | Messaging Service | AWS," Amazon Web Services, Inc., 2021. [Online]. Available: https://aws.amazon.com/sns/. [Accessed: Jun. 12, 2022].

[14] "Firebase Cloud Messaging | Send notifications across platforms at no-cost," Firebase, 2022. [Online]. Available: https://firebase.google.com/products/cloud-messaging. [Accessed: Jun. 12, 2022].

[15] "Notification Hubs - Mobile push notifications | Microsoft Azure," Microsoft.com, 2022. [Online]. Available: https://azure.microsoft.com/en-ca/services/notification-hubs/#overview. [Accessed: Jun. 12, 2022].

[16] "Amazon API Gateway | API Management | Amazon Web Services," Amazon Web Services, Inc., 2022. [Online]. Available: https://aws.amazon.com/api-gateway/. [Accessed: Jun. 12, 2022].

[17] "API Gateway | Google Cloud," *Google Cloud,* 2022. [Online]. Available: https://cloud.google.com/api-gateway. [Accessed: Jun. 12, 2022].

[18] "API Management – Manage APIs | Microsoft Azure," *Microsoft.com,* 2021. [Online]. Available: https://azure.microsoft.com/en-us/services/api-management/. [Accessed: Jun. 12, 2022].

[19] "Project_Proposal_Group16_S22", Project proposal for group 16 – summer 22 – Cloudvengers [Online]. Available: https://dal.brightspace.com/d2l/common/viewFile.d2lfile/Database/MTI0OTM4NTU/Project_Proposal_Group16_S22.pdf?ou=222417 [Accessed: Jun. 11, 2022].

[20] "The Main Difference Between AWS Chatbot vs Amazon Lex - Explained", The Chatbot Business Framework, 2022. [Online]. Available: https://chatbotbusinessframework.com/difference-between-aws-chatbot-vs-amazon-lex/. [Accessed: 10- Jun- 2022].

[21] "Amazon DocumentDB", Amazon Web Services, Inc., 2022. [Online]. Available: https://aws.amazon.com/documentdb/. [Accessed: 11- Jun- 2022].

[22] "Fully Managed Relational Database - Amazon RDS - Amazon Web Services", Amazon Web Services, Inc., 2022. [Online]. Available: https://aws.amazon.com/rds/. [Accessed: 11- Jun- 2022].

[23] "Encryption cryptography signing - AWS Key Management Service - Amazon Web Services", Amazon Web Services, Inc., 2022. [Online]. Available: https://aws.amazon.com/kms/. [Accessed: 12- Jun- 2022].

[24] “AWS Fargate,” Amazon. [Online]. Available: https://aws.amazon.com/fargate/pricing/. [Accessed: 11-Jun-2022].

[25] “AWS Lambda,” Amazon. [Online]. Available: https://aws.amazon.com/lambda/pricing/. [Accessed: 11-Jun-2022].

[26] “Provision Infrastructure As Code – AWS CloudFormation – Amazon Web Services,” *Amazon Web Services, Inc.*, 2022. [Online]. Available: https://aws.amazon.com/cloudformation/. [Accessed: Jun. 12, 2022].

[27] “Amazon S3 Simple Storage Service Pricing - Amazon Web Services,” Amazon Web Services, Inc., 2022. [Online]. Available: https://aws.amazon.com/s3/pricing/. [Accessed: Jun. 12, 2022].

[28] “Pricing | Amazon Cognito | Amazon Web Services (AWS),” Amazon Web Services, Inc., 2022. [Online]. Available: https://aws.amazon.com/cognito/pricing/. [Accessed: Jun. 12, 2022].

[29] “Amazon Simple Notification Service (SNS) Pricing | Messaging Service | AWS,” Amazon Web Services, Inc., 2022. [Online]. Available: https://aws.amazon.com/sns/pricing/. [Accessed: Jun. 12, 2022].

[30] “Amazon API Gateway Pricing | API Management | Amazon Web Services,” Amazon Web Services, Inc., 2022. [Online]. Available: https://aws.amazon.com/api-gateway/pricing/. [Accessed: Jun. 12, 2022].

[31] "Pricing", Amazon Web Services, Inc., 2022. [Online]. Available: https://aws.amazon.com/pricing/. [Accessed: 12- Jun- 2022].