

# Technical Report

## Cloud Computing (CSC-40039)

(2125 Words)

### 1 Design of Cloud Application

We are going to design a native cloud application for an image web service. We will use a car store website as an example. Before we get into the specifics of the design, it is important to understand the three basic cloud service models. Each model reflects a distinct aspect of the cloud computing stack and provides the user with a different amount of control over his IT assets.

1. **Infrastructure as a service (IaaS)** refers to cloud IT services that include access to network configuration, computers (virtual or dedicated hardware), and data storage space. Cloud services that are classified as IaaS give customers the most flexibility and management control over their IT resources. IaaS services resemble existing on-premises computing capabilities that many IT teams are already familiar with. AWS (Amazon Web Services) services, such as Amazon EC2, are classified as IaaS, which means the customer is responsible for all security configuration and administration. Customers who use EC2 instances are responsible for administering the guest operating system (including updates and security patches), any application software deployed on the instances, and the AWS-provided security groups [1].
2. **Platform as a service (PaaS)** refers to services that take care of the underlying infrastructure for the customer (hardware, operating systems, etc.). PaaS services allow customers to concentrate solely on the deployment and management of applications. Customers do not have to worry about acquiring resources, planning capacity, maintaining software, or patching. Because AWS manages the infrastructure layer, operating system, and platforms, PaaS services like AWS Lambda and Amazon RDS can be classified as PaaS. To store and retrieve data, customers merely need to access the endpoints. Customers are responsible for managing their data, identifying their assets, and applying the proper rights while using PaaS services. These services, on the other hand, are more like managed services, with AWS taking care of a bigger amount of the security requirements. AWS handles basic security responsibilities including operating system and database patching, firewall setting, and disaster recovery for these services [1].
3. **Software as a service (SaaS)** refers to cloud-based software that is often accessed through a web browser, mobile app, or application programming interface (API). SaaS products are often licensed on a subscription or pay-as-you-go basis. Customers who use SaaS services do not have to worry about maintaining the infrastructure that supports the service. Given their qualities, some AWS services, such as AWS Trusted Advisor, AWS Shield, and Amazon Chime, could be classified as SaaS solutions. Web-based email is a common example of a SaaS service, in which you can send and receive email without having to manage feature changes to the email service or maintaining the servers and operating systems on which the email program runs. [1].

Figure 1 represents the design Architecture of our Cloud Application. To serve pictures for regionally optimal content delivery, we use the Amazon CloudFront content delivery network (CDN). Original photos are kept in Amazon Simple Storage Service (Amazon S3) buckets that are set up as CloudFront Distribution Origins in our example. A CDN is an essential component of any modern web application. CDNs are used to simply improve content delivery by replicating frequently requested files (static content) across a globally distributed network of cache servers.

CDNs, on the other hand, have grown in importance over time. By sending a local copy of the content from a nearby cache edge, a CDN can minimize the burden on an application origin and improve the requestor's experience. Because the CDN handles heavy lifting, the application origin is free to open the connection and send the content immediately. As a result, the application origins do not need to scale to fulfil static content demand [1].

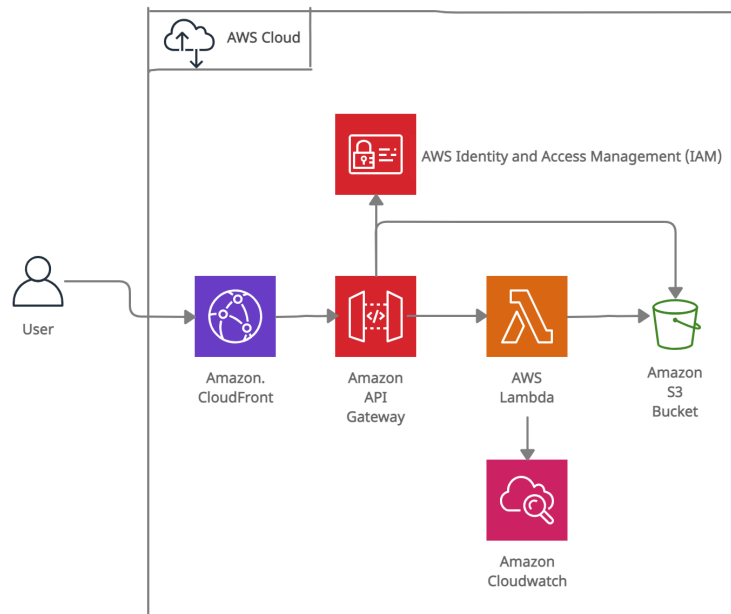


Figure 1: Design Architecture for our Cloud Application

For our Car Store website, we can utilize the Amazon API Gateway console to construct and test a simple REST API with HTTP integration. The Amazon API Gateway service is a fully managed service that makes it simple to create, publish, maintain, monitor, and secure REST, HTTP, and WebSocket APIs at any size. API Gateway generates HTTP-based RESTful APIs. It implements typical HTTP methods including GET, POST, PUT, PATCH, and DELETE and allows for stateless client-server communication [2]. In this case, we are using the sample REST URL.

REST URL: `http://www.carstoreimageservice.com/company/model/2022/car_sample.jpg`

To retrieve the images from Amazon S3, we must have READ access to the objects. So, to allow our API to access our S3 bucket and interact with AWS Lambda, we should create an AWS Identity and Access Management (IAM) role. IAM can be used to manage authentication as well as define and enforce authorization policies, allowing us to control which users can have access to services.

AWS S3 is one of the best options for storing static files like photos. We chose S3 because it is one of the most cost-effective cloud storage options; we will not be charged for the number of times it is read, but simply for the quantity of outbound traffic. Although Amazon S3 buckets lack a directory hierarchy, they do have a logical hierarchy. To get an object, we must use the GET operation to specify the object's entire key name [3]. The following is an example of a GET request response:

Sample Request:

```
.  
GET /company/model/2022/car_sample.jpg HTTP/1.1  
Host: bucket.s3.<Region>.amazonaws.com  
Date: Mon, 25 April 2022 10:30:00 GMT  
Authorization: authorization string
```

Sample Response:

```
HTTP/1.1 200 OK
x-amz-id-2: eftixk72aD6Ap51TnqcoF8eFidJG9Z/2mkiDFu8yU9AS1ed40pIszj7
x-amz-request-id: 318BC8BC148832E5
Date: Mon, 25 April 2022 10:30:01 GMT
Last-Modified: Wed, 20 April 2022 09:10:00 GMT
ETag: "fba9dede5f27731c9771645a39863328"
Content-Length: 434234
```

[434234 bytes of object data]

We are utilizing Lambda, a serverless computing service that responds to events by executing code. The events in our scenario are the API requests from API Gateway and object metadata from Amazon S3. We can use AWS Lambda to run code without having to provision or manage servers. The main advantage of Lambda is that we only need to pay for compute time used. When our code is not in use, there is no pricing at all. After deploying our API, we can use Amazon CloudWatch to monitor Lambda by accessing all the GET and POST request logs. Amazon CloudWatch is a service that gives data and actionable insights into our apps so we can monitor them, respond to system-wide performance changes, optimize resource use, and receive a unified view of our system's operational health.

Most of the AWS services in our design such as Amazon CloudFront, Amazon API Gateway, and Amazon CloudWatch are all SaaS components. AWS Lambda is a PaaS component. Amazon S3 is an AWS IaaS solution. Amazon S3 is a cloud-based data storage service that is extremely scalable, secure, and has low-latency.

## 2 Evaluation on Bioinformatics Application

A biopharmaceutical company requires a computational analysis platform to compare various bacterial organisms based on their features. The GO:0030420 term and its sub terms (Figure 2) are used to indicate whether a bacteria can have the competence feature, which is about being transformed to take up exogenous genetic materials. Our goal is to identify which organisms (using the provided seven input files) may exhibit "competence" as a physiological state.




Child Term	Relationship to GO:0030420
GO:0045809  positive regulation of establishment of competence for transformation	positively_regulates
GO:0045304  regulation of establishment of competence for transformation	regulates
GO:0045808  negative regulation of establishment of competence for transformation	negatively_regulates

Figure 2: Sub terms for the GO:0030420 term [4]

We developed an application using Hadoop and MapReduce to solve this problem. The existence of the GO:0030420 term, and its sub terms (Child terms) in a file suggests that the corresponding organism may exhibit the competence feature. Therefore, we count the number of GO assignments for the GO:0030420 term and related sub terms for each file to identify which organisms exhibit "competence" as a physiological state.

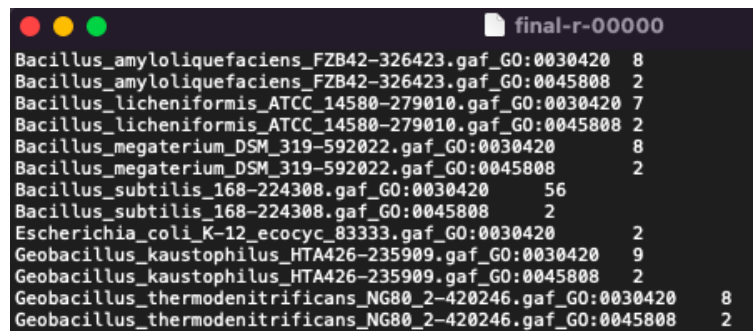
Our source files are seven input files, representing 7 organisms. Input files are available in the Gene Annotation File (GAF) format, which defines 17 tab-delimited columns and keeping it in input folder. We created a **Bioinformatic\_App.java** application using Hadoop and MapReduce. By using mapper function we were able to tokenize the words. Then we searched the GO:0030420 term and its sub terms using simple if condition. By using the reducer function we summed the occurrence of terms resulting in the counts. To display the 7 types of organisms and their counts in the output, we concatenated the filenames with the searched string. Sample output generated is depicted in Figure 3.

Later we created the same in Maven for build automation. Run the application from the Hadoop folder. First, remove the output folder in the directory if it exists. Then execute the

below command.

```
bin/hadoop jar Bioinformatic_App/target/  
Bioinformatic_App-1.0-SNAPSHOT.jar Bioinformatic_App input output
```

**Analysis on the output:** We could see all seven files or organisms have this GO:0030420 term. Suggesting these 7 organisms will exhibit “competence” as a physiological state. Out of these 7, *Bacillus subtilis* 168 has the highest occurrences of this GO term, i.e., 56. Therefore *Bacillus subtilis* 168 has a higher probability of transforming to take up exogenous genetic materials. If we consider child terms, most of the organisms have positively regulates associated to it i.e., GO:0045809 and its occurrences are always 2. We could also observe that the *Escherichia coli* K-12 does not have any occurrence of its child terms.



<i>Bacillus_amyloliquefaciens_FZB42-326423.gaf</i>	GO:0030420	8
<i>Bacillus_amyloliquefaciens_FZB42-326423.gaf</i>	GO:0045808	2
<i>Bacillus_licheniformis_ATCC_14580-279010.gaf</i>	GO:0030420	7
<i>Bacillus_licheniformis_ATCC_14580-279010.gaf</i>	GO:0045808	2
<i>Bacillus_megaterium_DSM_319-592022.gaf</i>	GO:0030420	8
<i>Bacillus_megaterium_DSM_319-592022.gaf</i>	GO:0045808	2
<i>Bacillus_subtilis_168-224308.gaf</i>	GO:0030420	56
<i>Bacillus_subtilis_168-224308.gaf</i>	GO:0045808	2
<i>Escherichia_coli_K-12_ecocyc_83333.gaf</i>	GO:0030420	2
<i>Geobacillus_kaustophilus_HTA426-235909.gaf</i>	GO:0030420	9
<i>Geobacillus_kaustophilus_HTA426-235909.gaf</i>	GO:0045808	2
<i>Geobacillus_thermodenitrificans_NG80_2-420246.gaf</i>	GO:0030420	8
<i>Geobacillus_thermodenitrificans_NG80_2-420246.gaf</i>	GO:0045808	2

Figure 3: Final Output

### 3 Future Cloud Integration of Bioinformatics Application

A Distributed File System (DFS) is a file system that is distributed among numerous file servers or locations. It enables programs to access and store isolated data in the same way that they access and save local files, allowing programmers to access files from any network or computer. Hadoop applications use the Hadoop Distributed File System (HDFS) as their primary data storage system. HDFS is a distributed file system that uses a NameNode and DataNode design to allow high-performance access. Whereas Cloud file storage is a type of cloud storage that allows servers and applications to access data via shared file systems. A cloud file system is a hierarchical storage system that allows users to share file data. Users could create, delete, change, read, and write files, as well as organize them logically in directory trees for easy access. Cloud utilizes Object storage mechanism, and it has huge scalability and metadata properties, that are frequently used in cloud-based applications. Object storage systems, such as Amazon Simple Storage Service (Amazon S3), are suitable for developing modern applications from the ground up that require scale and flexibility, as well as importing existing data stores for analytics and backup.

We will be able to integrate our bioinformatics program to the cloud-based system in the future. Data can be moved between Hadoop Distributed File Systems (HDFS) and AWS Storage services using AWS DataSync. We can move files and folders from their Hadoop clusters to AWS Storage, Amazon S3. We can use Amazon EMR (Amazon Elastic MapReduce). It is a managed cluster platform that makes it easier to run big data frameworks on AWS, such as Apache Hadoop, and Apache Spark, to process and analyze massive volumes of data [5]. So, our application can process and analyze vast amounts of data.

#### Advantages of migrating our application to the cloud :

- **Lower Cost:** Instead of overbuying the storage resources up front and guessing how much data storage you will need in the future, Cloud Storage offers pay as what you need.
- **Compute and storage are separated:** You can access data directly from various clusters when you store it in Cloud Storage rather than HDFS. This makes it simpler to deconstruct and rebuild clusters without having to move data. It also allows users to benefit from job-scoped clusters.

- **Interoperability:** Storing data in Cloud Storage allows Spark and Hadoop instances to work together seamlessly.
- **Data availability:** Cloud Storage data is highly available and may be globally copied without sacrificing performance. NameNode single point of failure or cluster failure are not a problem for Cloud Storage.






#### Disadvantages of migrating our application to the cloud:

- I/O variance can increase by using cloud storage.
- Files append, and transactions are not supported in Cloud Storage. Objects are immutable, which means that they will not change over their storage lifespan.
- Request latency may be higher with cloud storage. When compared to HDFS, cloud storage may have a longer request round-trip latency.

## 4 AWS Academy Cloud Foundations course

Enlisting the five AWS modules that I completed from the online AWS Academy Cloud Foundations course, together with my score for each module.

1. Module 1 - Cloud Concepts Overview
2. Module 2 - Cloud Economics and Billing
3. Module 3 - AWS Global Infrastructure Overview
4. Module 4 - AWS Cloud Security
5. Module 5 - Networking and Content Delivery

Name	Due	Status	Score	Out of	
<a href="#">Module 1 Knowledge Check</a> Knowledge Checks			90	100	
<a href="#">Module 2 Knowledge Check</a> Knowledge Checks			100	100	
<a href="#">Module 3 Knowledge Check</a> Knowledge Checks			100	100	
<a href="#">Module 4 Knowledge Check</a> Knowledge Checks			100	100	
<a href="#">Module 5 Knowledge Check</a> Knowledge Checks			100	100	

## References

- [1] Anon., n.d. Cloud Concepts Overview. [Online] Available at: <https://awsacademy.instructure.com/courses/12799/modules/items/1104478> [Accessed 18 04 2022].
- [2] Anon., n.d. Amazon API Gateway. [Online] Available at: <https://aws.amazon.com/api-gateway/> [Accessed 20 04 2022].
- [3] Amazon, n.d. GetObject. [Online] Available at: [https://docs.aws.amazon.com/AmazonS3/latest/API/API\\_GetObject.html](https://docs.aws.amazon.com/AmazonS3/latest/API/API_GetObject.html) [Accessed 20 04 2022].
- [4] Anon., n.d. GO:0030420. [Online] Available at: <https://www.ebi.ac.uk/QuickGO/term/GO:0030420> [Accessed 20 04 2022].
- [5] Anon., n.d. What is Amazon EMR?. [Online] Available at: <https://docs.aws.amazon.com/emr/latest/ManagementGuide/emr-what-is-emr.html>