

Project proposal NSSA-712

## **Deploying an End to End Website on AWS**

Group members:

Bashair Algarni - Zaki Bawade

Rochester Institute of Technology

2020

## **INTRODUCTION:**

"AWS or Amazon Web Services is one of the most reliable and well-known cloud computing platforms in the world. It comprises a wide array of storage, database, computation, content delivery, mobile, and other services. It's a great process to host simple websites in addition to the complex Web or mobile applications in the cloud.

AWS helps you save money, workforce, and time compared to building and maintaining more traditional systems as it is very safe and secure. Amazon Web Services provides numerous advantages for IT firms and developers alike, such as cost-effectiveness as we only have to pay for what we use, as we use it. Besides, it is comprehensive, dependable, resilient, and secure. Flexibility enables us to create applications using any programming model and platform you want or need.

Amazon EC2 is a computing web service of AWS that provides instances which are nothing but virtual machines. It is very reliable, scalable, secure, flexible, and easy to use. It is also integrated with other web services of AWS. It allows you to pay for only the computing capacity you use. You are also able to choose the amount of RAM you require and the quantity of CPU you need[3]".

"Amazon Simple Storage Service (Amazon S3) is a scalable storage service offered by AWS. It stores files in the form of objects, and each object can be of size 5 TB maximum[2]. IAM AWS Identity and Access Management (IAM) helps you manage access for resources and users in your AWS ecosystem. It helps you dynamically give granular access to AWS resources[2].

Amazon RDS is a 'managed' relational database service; i.e., it is a service that can manage relational databases for you. It can do various automated tasks, such as doing security patches, backups, etc." [2].

This project's main objective is to get a better understanding of how to deploy end to end website using AWS "and utilizing different cloud computing services to acquire high performance with less cost. Amazon EC2 is used here since it is very popular in providing good cloud computing services. This report shows step by step the procedure of using the AWS Marketplace to provision and produce a new AWS Cloud server.[3]"

## LIT REVIEW:

"Amazon EC2 sets a new perspective and standard for hosting websites. Letting developers scale their numerous machines up or down within minutes presents the capability to innovate distributed and scalable applications that keep running in the cloud. EC2 is secure, reliable, flexible, and most of all, it is cheap. You only pay for the assets you use; you can offer your multi-server application for sale to the public a lot less expensive than at any other time and keep up an incredibly significant value and accessibility level. EC2 is the computing part of the services provided by Amazon. EC2 provides the memory, CPU, transient storage, and operating system. EC2 is the equivalent of a barebones PC. You get the chance to choose from the quantity of CPUs you require (from a string of given options), the quantity of transient storage you require (from a rundown of options), and the measure of RAM you need (from a predefined list of arrangements). You get a choice of different kinds of operating system such as Microsoft Windows, Solaris, or Linux Server. The basis of EC2 is the Amazon Machine Image (AMI). An AMI is a virtual machine with your chosen operating system and applications packaged together. You can make your very own AMIs without any preparation if you need to. Amazon offers several open AMIs with pre-installed applications and many operating systems" [3].

"Amazon S3 (Simple Storage Service) is an online storage service provided by Amazon [2]. The service for storing and retrieving data is available anywhere on the Web, at any

time. The scale continues to grow, and 2 trillion objects have been stored as of April 2013. When using Amazon EC2, S3 can be used for managing virtual machine images, templates, and backups" [4].

"Amazon Route 53 effectively connects user requests to infrastructure running in AWS – such as Amazon EC2 instances, Elastic Load Balancing load balancers, or Amazon S3 buckets – and can also be used to route users to infrastructure outside of AWS. You can use Amazon Route 53 to configure DNS health checks to route traffic to healthy endpoints or independently monitor your application's health and endpoints. Amazon Route 53 also offers Domain Name Registration – you can purchase and manage domain names such as example.com, and Amazon Route 53 will automatically configure DNS settings for your domains.[5]"

"Amazon Relational Database Service (Amazon RDS) makes it easy to set up, operate, and scale a relational database in the cloud. It provides cost-efficient and resizable capacity while automating time-consuming administration tasks such as hardware provisioning, database setup, patching, and backups. It frees you to focus on your applications so you can give them the fast performance, high availability, security, and compatibility they need." [6]

## METHODOLOGY:

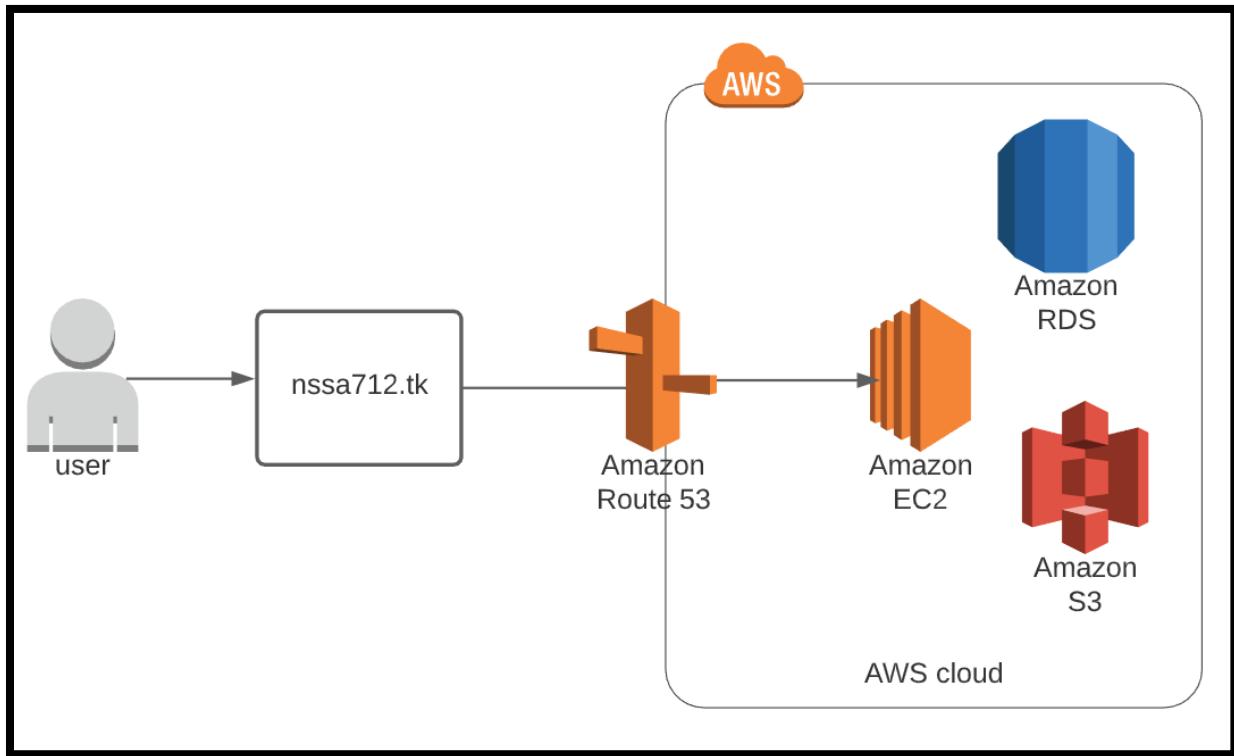
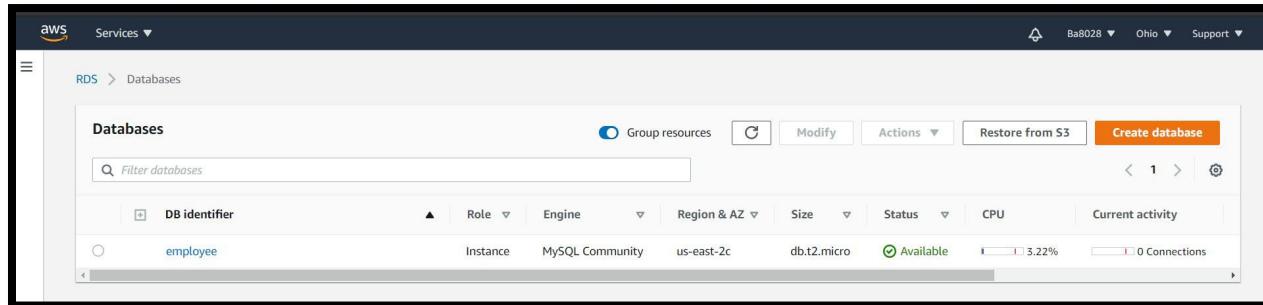


Fig1.The Project Architectures

Below we have described the complete deployment of an end-to-end website on AWS using AWS S3, RDS, EC2, and Route 53 DNS service. The complete project uses AWS free tier and a domain naming website. From the above diagram, nssa712.tk is the name of the deployed website. Amazon Route S3 has the necessary zone file and provides reverse and forward lookups. EC2 instance hosts python code based website. Amazon RDS is the MYSQL database, and Amazon S3 is the storage where the files uploaded on the website are stored.

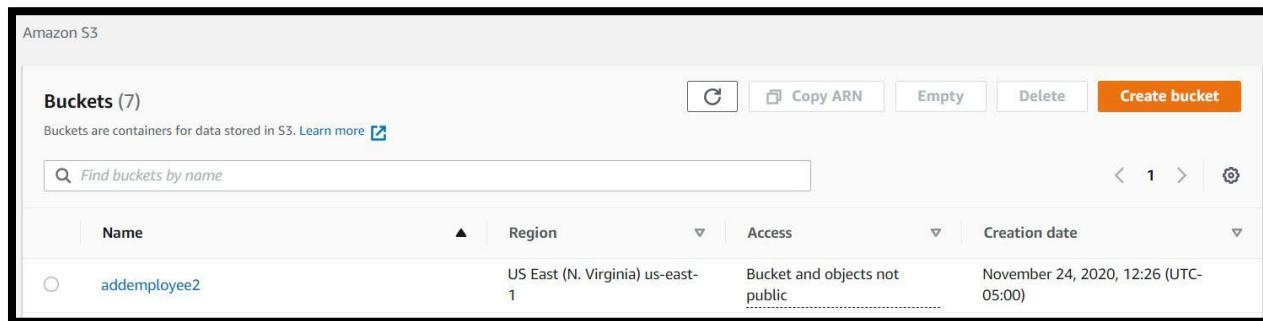
## Steps to accomplish this project :

### Step1: Deploy MySQL database on AWS using RDS



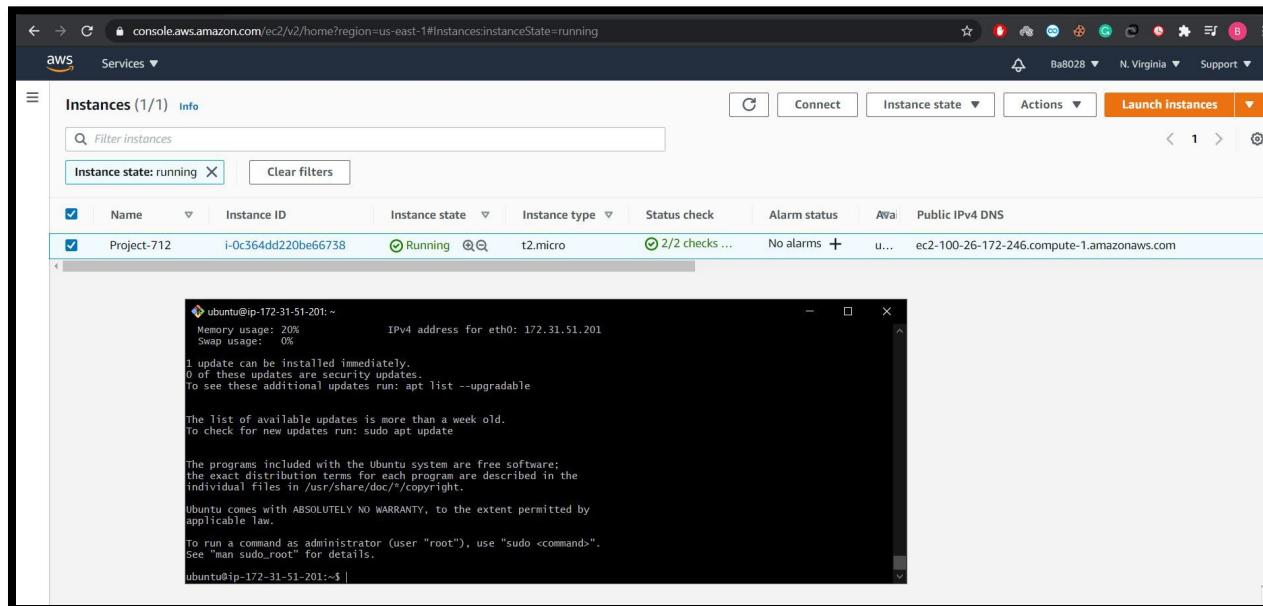
The screenshot shows the AWS RDS Databases console. At the top, there are buttons for 'Group resources', 'Actions', 'Restore from S3', and a prominent orange 'Create database' button. Below this is a search bar labeled 'Filter databases'. A table lists the database 'employee', which is an instance of MySQL Community, located in the us-east-2c region, using the db.t2.micro engine. The status is 'Available' with 3.22% CPU usage and 0 connections.

### Step 2: Deploy S3 storage bucket



The screenshot shows the AWS S3 Buckets console. It displays a list of buckets with 7 items. One bucket is selected and shown in detail: 'addemployee2', located in the US East (N. Virginia) region (us-east-1). The access level is set to 'Bucket and objects not public', and it was created on November 24, 2020, at 12:26 (UTC-05:00).

### Step 3: Deploy EC2 instance



The screenshot shows the AWS EC2 Instances console. It lists one instance, 'Project-712', which is running and has the instance ID i-0c364dd220be66738. The instance type is t2.micro. A terminal window is open on the instance, displaying the Ubuntu welcome screen and system status information.

```
ubuntu@ip-172-31-51-201:~$ Memory usage: 20%          IPV4 address for eth0: 172.31.51.201
Swap usage:  0%
1 update can be installed immediately.
0 of these updates are security updates.
To see these additional updates run: apt list --upgradable

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-51-201:~$ |
```

## Step 4: Created Database Entries

```
mysql> show databases;
+-----+
| Database |
+-----+
| employee |
| information_schema |
| mysql |
| performance_schema |
+-----+
4 rows in set (0.01 sec)

mysql> create table employee(
-> empid varchar(20),
-> fname varchar(20),
-> lname varchar(20),
-> pri_skill varchar(20),
-> location varchar(20));
ERROR 1046 (30000): No database selected
mysql> create table employee ( empid varchar(20), fname varchar(20), lname varchar(20), pri_skill varchar(20), location varchar(20));
ERROR 1046 (30000): No database selected
mysql> create table employee( empid varchar(20), fname varchar(20), lname varchar(20), pri_skill varchar(20), location varchar(20));
ERROR 1046 (30000): No database selected
mysql> use employee
Database changed
mysql> Create Table employee( empid varchar(20), fname varchar(20), lname varchar(20), pri_skill varchar(20), location varchar(20));
Query OK, 0 rows affected (0.03 sec)

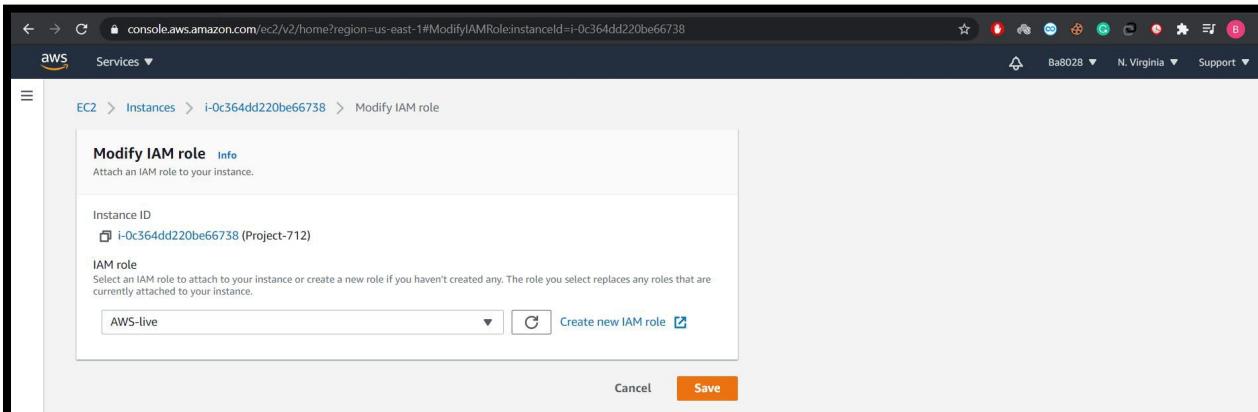
mysql> show databases;
+-----+
| Database |
+-----+
| employee |
| information_schema |
| mysql |
| performance_schema |
+-----+
4 rows in set (0.00 sec)

mysql> show employee;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that
corresponds to your MySQL server version for the right syntax to use near 'emplo
ye' at line 1
mysql> use employee
Database changed
mysql> show tables
-> ;
+-----+
| Tables_in_employee |
| employee |
+-----+
1 row in set (0.00 sec)
```

## Step 5: Deployed website on EC2 instance

```
ubuntu@ip-1-2-31-51-201:~/aws-live$ sudo python3 EmpApp.py
 * Serving Flask app "EmpApp" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Running on http://0.0.0.0:80/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 748-116-005
```

## Step 6: Provide an IAM role to the EC2 instance



## Step 7: Create an IAM role to access S3 from EC2

The screenshot shows the AWS IAM Roles Summary page. The left sidebar is titled 'Identity and Access Management (IAM)' and includes sections for Dashboard, Access management (Groups, Users, Roles), Policies, Identity providers, Account settings, Access reports (Archive rules, Analyzers, Settings), Credential report, Organization activity, and Service control policies (SCPs). The main area is titled 'Summary' for the role 'AWS-live'. It displays the following details:

- Role ARN:** arn:aws:iam::262488438002:role/AWS-live
- Role description:** Allows EC2 Instances to call AWS services on your behalf. | Edit
- Instance Profile ARNs:** arn:aws:iam::262488438002:instance-profile/AWS-live
- Path:** /
- Creation time:** 2020-11-24 16:30 EST
- Last activity:** Not accessed in the tracking period
- Maximum session duration:** 1 hour | Edit

The 'Permissions' tab is selected, showing one policy applied: 'AdministratorAccess' (AWS managed policy). Other tabs include Trust relationships, Tags, Access Advisor, and Revoke sessions.

## Step 8: Create a Domain Name

The screenshot shows the 'Managing nssa712.tk' page. The top navigation bar includes links for Information, Upgrade, Management Tools (dropdown), and Manage Freenom DNS. A blue banner at the top right says 'Changes Saved Successfully!'. The main content area is titled 'Nameservers' and contains the following information:

You can change where your domain points to here. Please be aware changes can take up to 24 hours to propagate.

Two radio buttons are present:

- Use default nameservers (Freenom Nameservers)
- Use custom nameservers (enter below)

Below the radio buttons, there are four input fields for custom nameservers:

- Nameserver 1: NS-1494.AWSDNS-58.ORG
- Nameserver 2: NS-1694.AWSDNS-19.CO.UK
- Nameserver 3: NS-237.AWSDNS-29.COM
- Nameserver 4: NS-518.AWSDNS-00.NET

## Step 9: Configure Route53

The screenshot shows the AWS Route 53 console with the hosted zone 'nssa712.tk'. The left sidebar includes options like Dashboard, Hosted zones (selected), Health checks, Traffic flow, Domains, and Resolver. The main area shows 'Hosted zone details' with 'Records (4)'. The table lists the following records:

Record name	Type	Routing policy	Difference Initiator	Alias	Value/Route traffic to	TTL (seconds)	Health check	Evaluate target health	Record ID
ns-518.awsdns-00.net.	A	Simple	-	No	100.26.172.246	300	-	-	-
ns-1494.awsdns-58.org.	NS	Simple	-	No	ns-518.awsdns-00.net. awsdns-hostmaster.amazon.com. 1 7200 900 1209600 86400	172800	-	-	-
ns-1694.awsdns-19.co.uk.	SOA	Simple	-	No	ns-518.awsdns-00.net. awsdns-hostmaster.amazon.com. 1 7200 900 1209600 86400	900	-	-	-
ns-237.awsdns-29.com.	A	Simple	-	No	100.26.172.246	300	-	-	-

## Step 10: Successfully deployed End to End Website on EC2

The screenshot shows a web browser window with the URL 'Not secure | nssa712.tk'. The page title is 'Employee Database'. It features a form with the following fields:

- Employee ID:
- First Name:
- Last Name:
- Primary Skills:
- Location:
- Image:  Choose file No file chosen

Buttons on the page include 'GET EMPLOYEE INFORMATION', 'UPDATE DATABASE', and a link 'About Us'.

### Step 11: Show the Form Entry :

The screenshot shows a web browser window titled "Employee Database". The page contains a form with the following fields:

- Employee ID: 2
- First Name: Harry
- Last Name: Maguire
- Primary Skills: DevOps
- Location: London

Below the form is a file upload field labeled "Image: Choose file" with the value "4E0744CD-7...F2C4406.jpg". At the bottom of the page are two buttons: "UPDATE DATABASE" and a link "About Us".

### Step 12: Save Successful

The screenshot shows a web browser window titled "SAVE SUCCESSFUL". The page displays the following message:

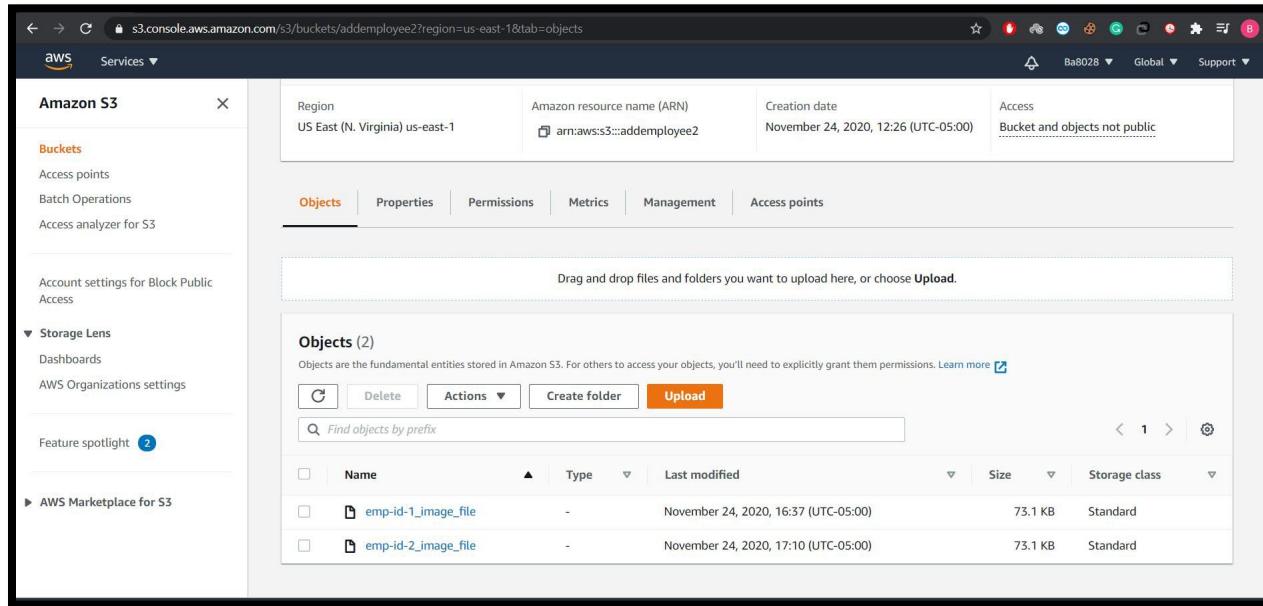
Following Employee has been added to the database  
Harry Maguire

At the bottom of the page is a "GO BACK" button.

### Step 13: Check the Database Entries

```
mysql> select * from employee;
+-----+-----+-----+-----+-----+
| empid | fname | lname | pri_skill | location |
+-----+-----+-----+-----+-----+
| 1     | Eric  | Dier  | aws      | London   |
| 2     | Harry | Maguire | DevOps    | London   |
+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)
```

## Step 14: Check that image files saved in S3 storage



The screenshot shows the AWS S3 console interface. On the left, there's a sidebar with options like 'Buckets', 'Storage Lens', and 'AWS Marketplace for S3'. The main area displays details for a bucket named 'addemployee2': Region (US East (N. Virginia) us-east-1), Amazon resource name (ARN) (arn:aws:s3:::addemployee2), Creation date (November 24, 2020, 12:26 (UTC-05:00)), and Access (Bucket and objects not public). Below this, tabs for 'Objects', 'Properties', 'Permissions', 'Metrics', 'Management', and 'Access points' are visible, with 'Objects' being the active tab. A central panel has a message to 'Drag and drop files and folders you want to upload here, or choose Upload.' followed by a table titled 'Objects (2)'. The table lists two items:

Name	Type	Last modified	Size	Storage class
emp-id-1_image_file	-	November 24, 2020, 16:37 (UTC-05:00)	73.1 KB	Standard
emp-id-2_image_file	-	November 24, 2020, 17:10 (UTC-05:00)	73.1 KB	Standard

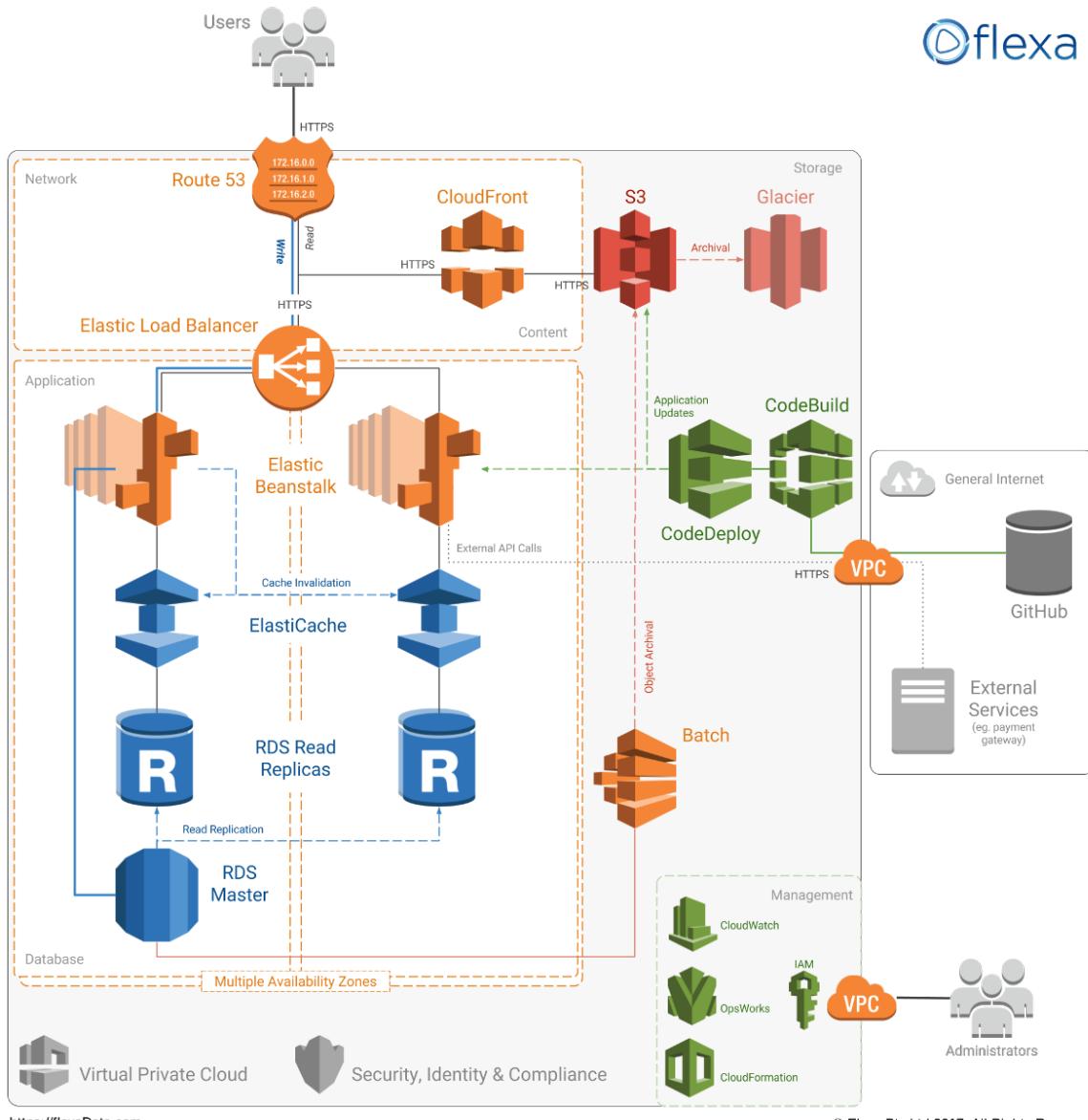
## RESULTS:

This project assesses the Amazon Web Services (AWS), the world's leading cloud provider. AWS provides a very resilient, safe, and secure platform for storing computing, database, and other services[3]. Many AWS services are used to deploy an end to end website, including EC2, S3, RDS, and Route 53. In step 11, we test the created website by entering employee details; in step 12, the details are successfully submitted. In step 13, we see that the entries are written to the database, and in step 14, our S3 storage reflects the details of the image uploaded.

## FUTURE WORK:

Our goal was to understand how we can use all these AWS services to deploy an end to end website. Our next improvement for future implementations would be to improve the website design and scale it up using other AWS services like AWS Elastic Beanstalk. Additionally, we can improve the resiliency and scalability by using a load balancer such as HAProxy. Services such as firewall, CDN (content delivery network), pipelining, and scaling

features can be added to the website to improve user experience, as seen in the architecture given below.



## **REFERENCES:**

- [1] "AWS Projects for beginners | Deploying End to End Website on AWS | Intellipaat" 2020.. Available:<https://www.youtube.com/watch?v=7Gym2XVcA5A> [Accessed: 11 24, 2020].
- [2] M. S. says: D. A. Says: L. says: E. C. says: A. says: and A. S. Says: "AWS Tutorial – Learn Amazon Web Services from Experts - Intellipaat," Intellipaat Blog. [Online]. Available: <https://intellipaat.com/blog/tutorial/amazon-web-services-aws-tutorial/>. [Accessed: 25-Nov-2020].
- [3] S. Shokeen and A. Singh, "Deploying an e-commerce website using Amazon Web Services," 2019 International Conference on Contemporary Computing and Informatics (IC3I), Singapore, Singapore, 2019, pp. 94-100, DOI: 10.1109/IC3I46837.2019.9055586.
- [4] Y. Tanimura, S. Yanagita and T. Hamanishi, "A High Performance, QoS-Enabled, S3-Based Object Store," 2014 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, Chicago, IL, 2014, pp. 784-791, DOI: 10.1109/CCGrid.2014.76.
- [5] D. Kopitz and B. Marks, "RDS: the radio data system," Amazon, 1999. [Online]. Available: <https://aws.amazon.com/rds/>. [Accessed: 25-Nov-2020].
- [6] "Amazon Route 53," Amazon. [Online]. Available: <https://aws.amazon.com/route53/>. [Accessed: 25-Nov-2020].