**AWS Cloud Architecting Capstone Project**

**Lab overview and objectives**

This project provides us with an opportunity to demonstrate the solution design skills that we developed throughout this course. The assignment is to design and deploy a solution for the following case.

By the end of this project, we should be able to apply the architectural design principles that we learned in this course to:

* Deploy a PHP application that runs on an Amazon Elastic Compute Cloud (Amazon EC2) instance.
* Create a database instance that the PHP application can query.
* Create a MySQL database from a structured query language (SQL) dump file.
* Update application parameters in an AWS Systems Manager Parameter Store
* Secure the application to prevent public access to backend systems.

**Solution requirements**

* Provide secure hosting of the MySQL database.
* Provide secure access for an administrative user.
* Provide anonymous access to web users.
* Run the website on a t2.micro EC2 instance and provide Secure Shell (SSH) access to administrators.
* Provide high availability to the website through a load balancer.
* Store database connection information in the AWS Systems Manager Parameter Store
* Provide automatic scaling that uses a launch template.

At the end of this lab, the architecture will look like the following example:

A screenshot of a computer

Description automatically generated

**Task 1: Creating a database in RDS**

1. In the **AWS Management Console,** on the **Services** menu, choose **RDS.**
2. Chose **MYSQL**

Template – Dev/Test

Availability and Durability – Multi-AZ DB instance

1. Configure the following parameters -

DB Name : example

Username : admin

Password : lab-password

DB Instance Class – Burstable Classes

Storage – 20GB

VPC : Example VPC

Security group - Example-DB

1. Click on Dashboard and click on **Create Database**

**Task 2: Check Security Groups**

1. Check the available Security Groups present in the lab environment.

**Task 3: Create EC2 Web Instance**

1. In the **AWS Management Console**, on the **Services** menu, choose **EC2.**
2. Click on **Create Instance**
3. Configure the following for the EC2 instance created -

AMI - Amazon Linux 2 AMI

Instance type- t2.Small

VPC - Example VPC

Subnet - Private Subnet 1

Auto- assigned Public IP - Disable

IAM role - Inventory-App-Role

Name – **ExampleApp**

1. Click on "**Select an existing security group**”. After that select the "**Inventory App**" security group.

**Task 4 : Create a Load Balancer**

1. In the **AWS Management Console**, on the **Services** menu, choose **EC2**.
2. Under **Load Balancing**, click on Load Balancer and **Create Load Balancer**
3. Choose Application Load Balancer and click on **Create.**
4. Configure the following parameters for the Load Balancer

Load Balancer Name – myloadbalancer

VPC – Example VPC

Choose us-east-1a – Public Subnet 1

Choose us-east-1b - Public Subnet 2

Security Group – ALBSG

1. In **Listeners and Routing**, **Create Target group.**

Target Group Name -mytargetgroup

VPC -Example VPC

1. Click on **Next** and **Create a target group.**
2. In the load balancer, choose the mytargetgroup. Click on **Create Load Balancer**. Click on **View Load Balancer** and wait until it is ready.

**Task 5 : Create an Autoscaling Group**

1. In Autoscaling Groups, click on **Create Autoscaling group.**
2. Configure the following for the auto scaling group -

Autoscaling Group Name- Myautoscalinggroup

Use the default **Launch template**. Click on **Next**

Under **Network settings**, click the **Example VPC**

Choose **Public Subnet1** and **Public Subnet2** in Availability Zones. Click on **Next**

1. Under Load Balancing, click on Attach to an existing Load balancer and choose the Load balancer created.

Desired Capacity – 1

Minimum Capacity – 1

Maximum Capacity- 2

1. Create an Autoscaling group.

**Task 6. Create Parameter Store in System Manager**

1. Configure the following parameters in the Parameter Store

/example/endpoint - example.cbufnmin3yj3.us-east-1.rds.amazonaws.com

/example/username - admin

/example/password - lab-password

/example/database - example

**Task 7 : Making the Website Available**

1. From the AWS Details Section, Copy the RSA Private Key
2. In EC2 instances ,open Bastion . Click on Connect
3. Paste the Private key in the Terminal and name it labuser.pem
4. Copy the Private IP of the Example APP instance. In the terminal type the following commands

Ssh -i labuser.pem ec2-user@10.0.1.0/24

ls

You should be able to see the Countrydatadump.sql file present.

1. Login to the database by using the following command line :

mysql -u admin -p --host example.cbufnmin3yj3.us-east-1.rds.amazonaws.com

Enter the password - lab-password

show databases;

exit;

1. Import data from the database :

mysql -u admin -p --host example.cbufnmin3yj3.us-east-1.rds.amazonaws.com example <Countrydatadump.sql

Enter the password - lab-password

exit;

**Task 8 : Testing the Web Application**

1. Test the Web Application by using load balancer DNS in new tab. Go to Load balancer you created earlier and copy the DNS address of the load balancer and paste it in a new tab.
2. The **Website is available**, and you can search for new queries in the Website .

**Design Decisions**

Here is a summary of the design decisions I made to complete all deliverables of Capstone project –

1. First, I looked at the resources that were already available when the lab was created.
2. The requirements of the capstone project require an Application Load Balancer and an autoscaling group to help with availability and scalability. I changed the default security group to the existing ALBSG security group and created a new target group. Once this was created, I was able to attach it to the ALB and configure it.
3. After this step, the website is still not available.
4. For the database, I used **MySQL for RDS** and configured it with an attached subnet group in the private subnets of the VPC.
5. I used the EC2 Connect in the bastion host and used the Private Key in the AWS Details tab to create the labuser.pem file. I was able to provide Secure Shell (SSH) access to administrators by changing the inbound rules to configure SSH.
6. After the database was created, I connected to the database and transferred the country data SQL dump file. I ran SQL commands (show tables, select all statement) to check and ensure the data has been transferred.
7. I ensured that the IAM role is attached to the web servers.
8. Finally, I configured all the mentioned system parameters in the parameter store that are used by the PHP application to connect to the database.
9. At last, I checked to see if the website works and can query the database, it worked as expected.