DevOps ENGINEERING ON AWS CLOUD

PROJECT DOCUMENTATION

Cloud Web Application Builder

Prepared by:
Harun Kunovac
Ervin Vladić
Filip Ljoljić

Mentor:
Dženana Dževlan

Date of submission

10.06.2023.

Table of Contents

Phase 1: Planning the design and estimating the cost

Task 1: Creating an architectural diagram

The first task of this phase was to create an architectural diagram. We used the draw.io page to create the AWS infrastructure diagram. Our diagram is on our GitHub account in the docs directory.

Task 2: Developing a cost estimate

The second task of the first phase was to estimate the cost of running the solution. We used the AWS Pricing Calculator for this task. We tried to make our solution efficient and to spend as little money as possible, while at the same time fulfilling requirements such as functional, load-balanced, scalable, highly available, secure, cost-optimized, and high-performing. Just like in the previous task, the PDF file is located in the docs directory on our GitHub repository.

Phase 2: Creating a basic functional web application

Task 1: Creating a virtual network

For the first task in the second phase, we have done everything necessary to create a virtual network. First, it was necessary to create a VPC (virtual private cloud). After that, we created a public subnet, an internet gateway that we attached to the VPC, a public route table, and a public security group where we added HTTP and SSH port 22 as an inbound rule. In order to successfully create an instance, it was necessary to add permissions to the already existing role: AmazonSSMManagedInstanceCore and AmazonEC2FullAccess. Finally, we created an EC2 instance.

Task 2: Creating a virtual machine

In the second task, we had the task of creating a virtual machine on the cloud for hosting a web application. To create a virtual machine, we first connected to the instance using EC2 Instance Connect and used the following commands:

1. sudo nano solutioncodepoc.sh
2. For installing the required web application and database on the virtual machine, we used the JavaScript code from the SolutionCodePOC
3. sudo chmod +x solutioncodepoc.sh
4. sudo ./solutioncodepoc.sh.

We have successfully created a virtual machine.

Task 3: Testing the deployment

For deployment testing, we used the IPv4 address of the virtual machine. We have successfully connected.

Phase 3: Decoupling the application components

Task 1: Changing the VPC configuration

In order to successfully support hosting the database separately from the application, it was necessary to create private subnets. It was necessary to create at least two private subnets in two availability zones.
In the first availability zone (US East (N. Virginia)/us-east-1a), there is one private subnet
In the second availability zone (US East (N. Virginia)/us-east-1b), there is one private subnet
Also, we created a private route table, private security group and NAT gateway

Task 2: Creating and configuring the Amazon RDS database

First, we created a security group for the RDS DB instance and added MySQL/Aurora with port 3306 as an inbound rule. After that, it was necessary to create a DB subnet group. We created a DB subnet group with two previously created private subnets in two availability zones. To finish this task, we created an Amazon RDS instance. An Amazon RDS instance was successfully created.

Task 3: Configuring the development environment

In this task, we created an AWS Cloud9 environment. As it is written in the notes, when creating the AWS Cloud9 environment, we used a t3.micro instance for the AWS Cloud9 environment and Secure Shell (SSH) to connect to the environment.

Task 4: Provisioning Secrets Manager

We created a secret for storing database credentials. We used script to create a secret in Secrets Manager by using the AWS CLI:
1. sudo nano secret.sh
2. Copy the script on which it was necessary to modify: RDS endpoint, password, username, database name
3. sudo chmod +x secret.sh
4. sudo./secret.sh.

We have successfully created a secret in AWS Secret Manager.

Task 5: Provisioning a new instance for the web server

For this task, we created a new instance and a new virtual machine. We used the following commands to install:

1. sudo nano solutioncodepoc2.sh
2. To install the required web application on the virtual machine, we used the JavaScript code provided
3. sudo chmod +x solutioncodepoc2.sh
4. sudo ./solutioncodepoc2.sh

After these commands, we accessed the website, but it was not connected to the RDS database.

Task 6: Migrating the database

After we created a new instance and created a virtual machine, it was necessary to migrate the database to our RDS database instance. We performed the database migration with a script in AWS Cloud9.

1. sudo nano migrate.sh
2. It was necessary to change the EC2instancePrivateip instance that we created in Phase 2 and the RDS endpoint.
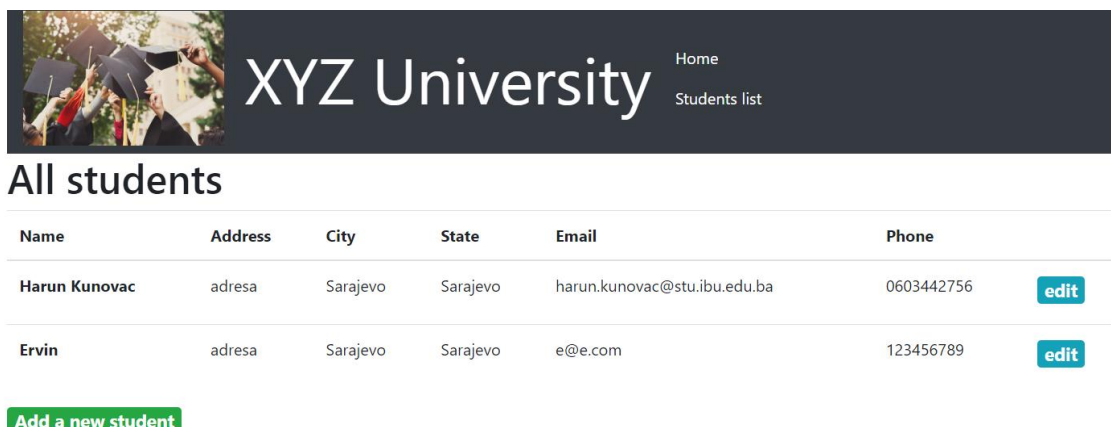3. sudo chmod +x migrate.sh
4. sudo ./migrate.sh

After that, it was necessary to connect the RDS database to the new instance. We have successfully connected RDS and the new EC2 instance.

Task 7: Testing the application

After we connected the EC2 instance with RDS, it was necessary to access the page again. We successfully migrated the database, and insight into students was possible as well as adding, deleting, and editing student records.

We accessed the database through Session Manager with the command:
mysql -h students.ckwb5jxkvhlp.us-east-1.rds.amazonaws.com -u admin -p and with the entered password

Below are the pictures:

Name	Address	City	State	Email	Phone	
Harun Kunovac	adresa	Sarajevo	Sarajevo	harun.kunovac@stu.ibu.edu.ba	0603442756	edit
Ervin	adresa	Sarajevo	Sarajevo	e@e.com	123456789	edit

Add a new student

```
0c204fad702a2b18c
$ bash
ssm-user@ip-10-0-0-81:/var/snap/amazon-ssm-agent/6563$ sudo su
root@ip-10-0-0-81:/var/snap/amazon-ssm-agent/6563# mysql -h students.ckwb5jxkvhlp.us-east-1.rds.amazonaws.com -u admin -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 320
Server version: 8.0.32 Source distribution

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use STUDENTS;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> select * from students;
+----+---------------+---------+----------+----------+----------------------------+------------+
| id | name          | address | city     | state    | email                      | phone      |
+----+---------------+---------+----------+----------+----------------------------+------------+
|  1 | Harun Kunovac | adresa  | Sarajevo | Sarajevo | harun.kunovac@stu.ibu.edu.ba | 0603442756 |
|  2 | Ervin         | adresa  | Sarajevo | Sarajevo | e@e.com                    | 123456789  |
+----+---------------+---------+----------+----------+----------------------------+------------+
2 rows in set (0.00 sec)
```

Phase 4: Implementing high availability and scalability

Task 1: Creating an Application Load Balancer

In this phase, our first task was to create an application load balancer. When creating the load balancer, we chose two availability zones (maybe it was necessary to create another public subnet). us-east-1a (use1-az1), us-east-1b (use1-az2). We have successfully created an application load balancer.

Task 2: Implementing Amazon EC2 Auto Scaling

In this task, it was necessary to launch a template and use an Auto Scaling group to launch the EC2 instances that host the web application. We created an AMI from an already-running instance and finally configured the Auto Scaling group to use a load balancer.

Task 3: Accessing the application

We accessed the application and successfully saw added, edited, and modified student records.

Task 4: Load testing the application

In order to perform a load test on the application to monitor scaling, it was necessary to run a script in AWS Cloud9.

1. sudo nano load.sh
2. It was necessary to change a load balancer
3. sudo chmod +x load.sh
4. sudo ./load.sh