



Building a Highly Available, Scalable Web Application

Presented by/

Mennatallah Abdelkareem Habashi Saleh St-ID:21046685

Supervised by/

DEPI

Eng/Merihan Adel

Overview and objectives:

Throughout various AWS Academy courses, we have completed hands-on labs. We have used different AWS services and features to create compute instances, install operating systems (OSs) and software, deploy code, and secure resources. We practiced how to enable load balancing and automatic scaling, and how to architect for high availability to build simple, lab-specific applications.

In this project, I'm challenged to use familiar AWS services to build a solution. Specific sections of the assignment are meant to challenge me on skills that I have acquired throughout the learning process.

By the end of this project, I should be able to do the following:

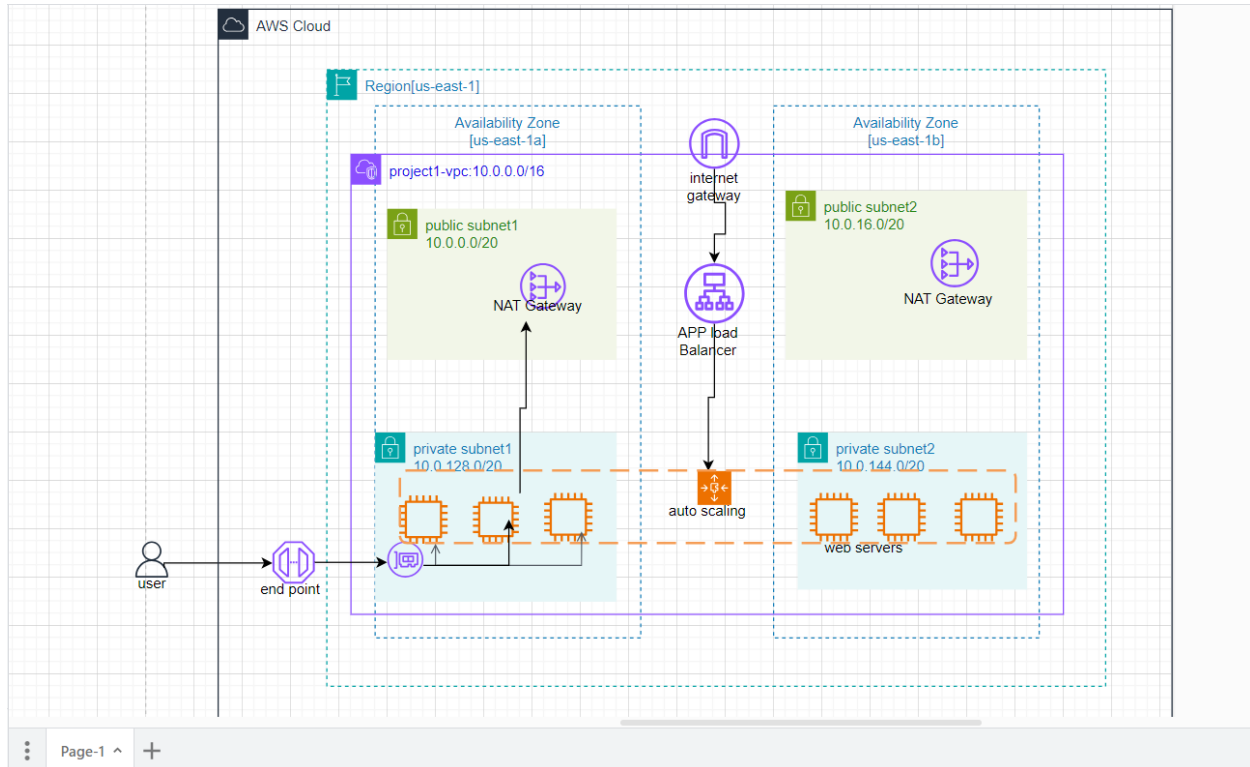
- Create an architectural diagram to depict various AWS services and their interactions with each other.
- Estimate the cost of using services by using the AWS Pricing Calculator.
- Deploy a functional web application that runs on a single virtual machine and is backed by a relational database.
- Architect a web application to separate layers of the application, such as the web server and database.
- Create a virtual network that is configured appropriately to host a web application that is publicly accessible and secure.
- Deploy a web application with the load distributed across multiple web servers.
- Configure the appropriate network security settings for the web servers and database.
- Implement high availability and scalability in the deployed solution.
- Configure access permissions between AWS services.

Phase 1: Planning the design and estimating cost:

In this phase, I will plan the design of my architecture. First, I will create an architecture diagram. Next, I will estimate the cost of the proposed solution, and present the estimate

Task 1: Creating an architectural diagram:

Create an architectural diagram to illustrate what I plan to build



Task 2: Developing a cost estimate:

Develop a cost estimate that shows the cost to run the solution in the us-east-1 Region for 12 months.

aws pricing calculator

FeedbackLanguage: English ▼Contact Sales [\[?\]](#)Create an AWS Account

Estimate summary [Info](#)

Upfront cost
0.00 USD

Monthly cost
79.54 USD

Total 12 months cost
954.48 USD
Includes upfront cost

Getting Started with AWS

Get started for free

Contact Sales

My Estimate

DuplicateDeleteMove toCreate groupAdd supportAdd service

< 1 > ⚙

<input type="checkbox"/>	Service Name	Status	Upfront cost	Monthly cost	Description	Region	Config Summary
<input type="checkbox"/>	Amazon Virtual Private Gateway	-	0.00 USD	7.30 USD	-	US East (N. Virginia)	Number of In-use public IP addresses (2)
<input type="checkbox"/>	Amazon EC2	-	0.00 USD	2.32 USD	-	US East (N. Virginia)	Tenancy (Shared Instances), Operating system (Linux), Workload (Consistent), Number of instances: 2, Advance EC2 instance (t2.micro), Pricing strategy (On-Demand only) (1)
<input type="checkbox"/>	Amazon RDS for MySQL	-	0.00 USD	69.92 USD	-	US East (N. Virginia)	Storage amount (100 GB), Storage for each RDS instance (General Purpose SSD (gp2)), Nodes (1), Instance type (db.t3.micro), Utilization (On-Demand only) (1)

Hi, I can connect you with an AWS representative or answer questions you have on AWS.

Privacy | [Site terms](#) | [Cookie preferences](#) | © 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

POSSIBLE DATA LOSS Some features might be lost if you save this workbook in the comma-delimited (.csv) format. To preserve these features, save it in an Excel file format.

Don't show againSave As...

A1

Estimate summary

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
1	Estimate summary																						
2	Upfront cost	Monthly cost	Total 12 m	Currency																			
3	0	79.54	954.48	USD																			
4				* Includes upfront cost																			
5																							
6																							
7	Detailed Estimate																						
8	Group hier	Region	Description	Service	Upfront	Monthly	First 12 m	Currency	Status														
9	My Estima	US East (N. Virginia)	Public IPv4		0	7.3	87.6	USD															
10	My Estima	US East (N. Virginia)	Amazon El		0	2.32	27.84	USD															
11	My Estima	US East (N. Virginia)	Amazon R		0	69.92	839.04	USD															
12																							
13																							
14																							
15	Acknowledgement																						
16																							
17																							
18																							
19																							
20																							
21																							
22																							
23																							
24																							
25																							
26																							
27																							
28																							

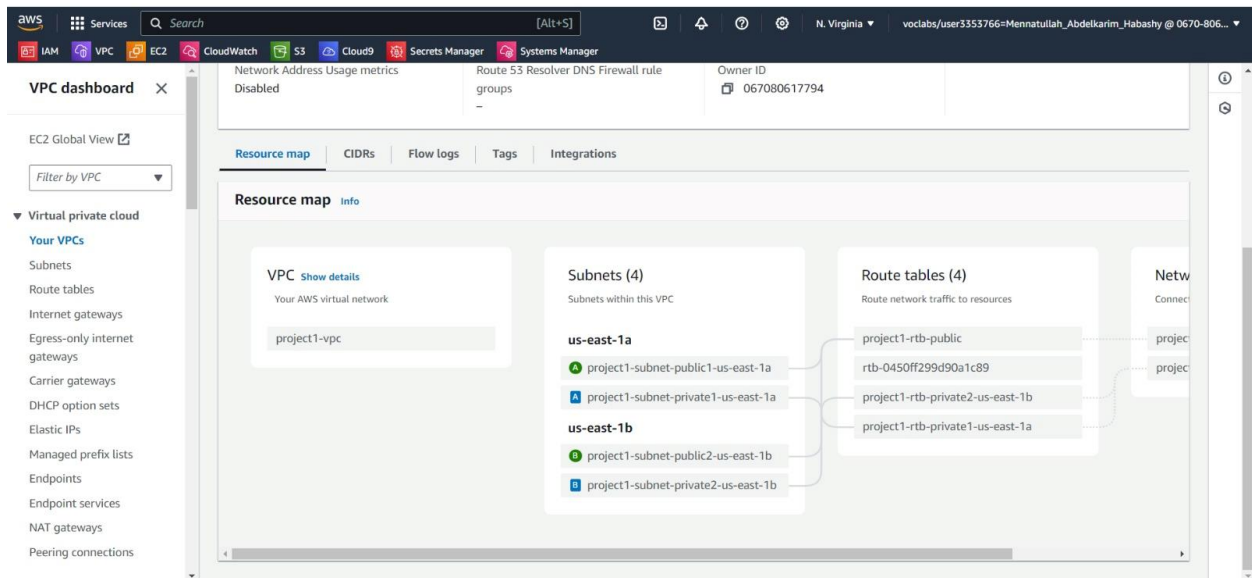
My Estimate

Phase 2: Creating a basic functional web application:

In this phase, I will start to build the solution. The objective of this phase is to have a functional web application that works on a single virtual machine in a virtual network that I create

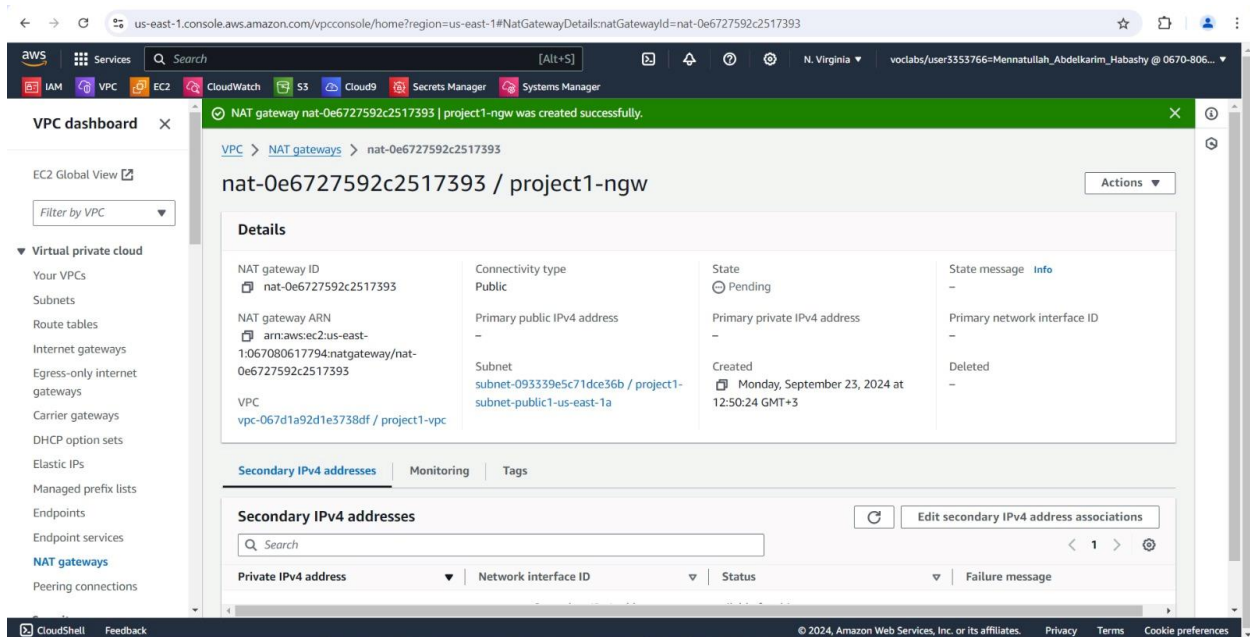
Task 1: Creating a virtual network:

Create a virtual network to host the web application(Create networking resources such as a virtual private cloud (VPC) and subnets.)



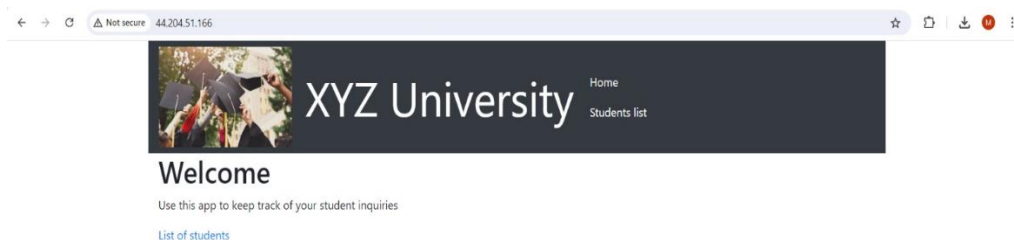
Task 2: Creating a virtual machine:

Create a virtual machine in the cloud to host the web application.



Task 3: Testing the deployment:

Test the deployment of the web application to ensure it is accessible from the internet and functional. (Perform a few tasks, such as viewing, adding, deleting, or modifying records).

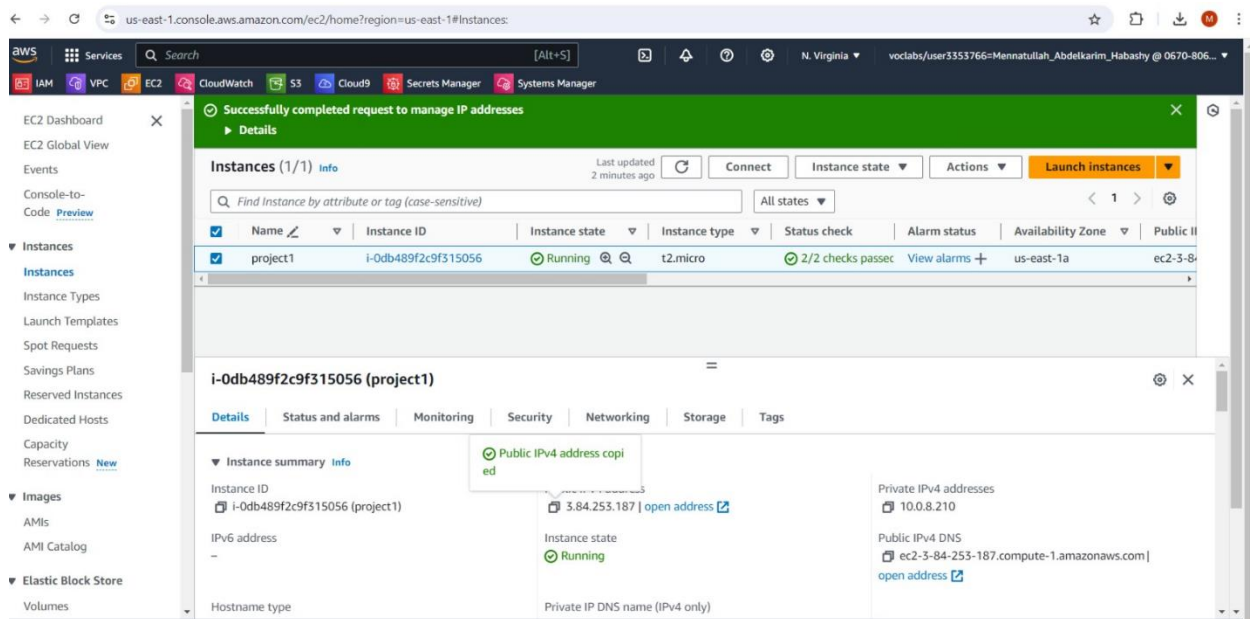


Phase 3: Decoupling the application components:

In this phase, I will continue building. The objective is to separate the database and the web server infrastructure so that they run independently. The web application should run on a separate virtual machine, and the database should run on the managed service infrastructure.

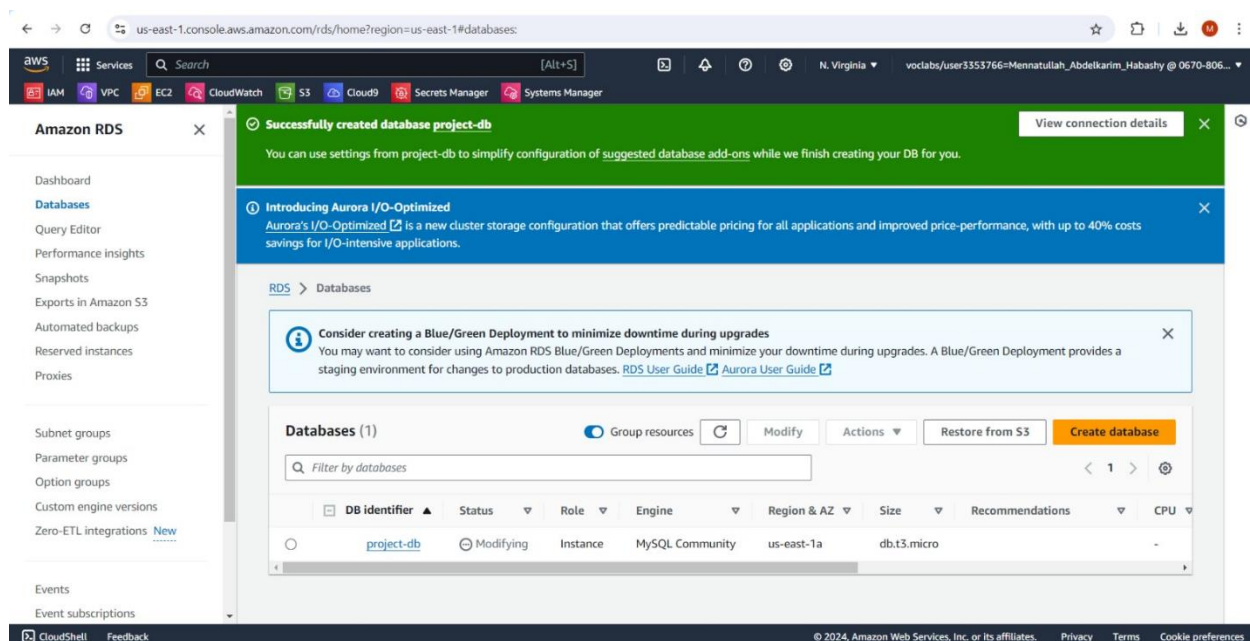
Task 1: Changing the VPC configuration:

Update or re-create the virtual network components that are necessary to support hosting the database separately from the application.



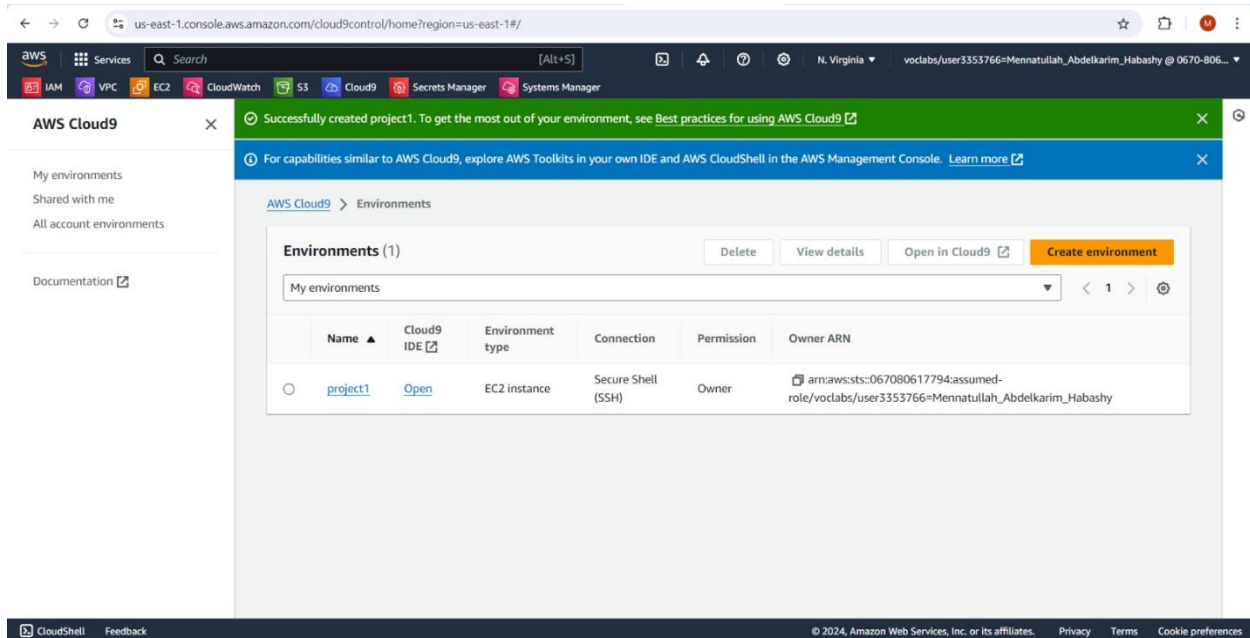
Task 2: Creating and configuring the Amazon RDS database:

Create an Amazon Relational Database Service (Amazon RDS) database that runs a MySQL engine.



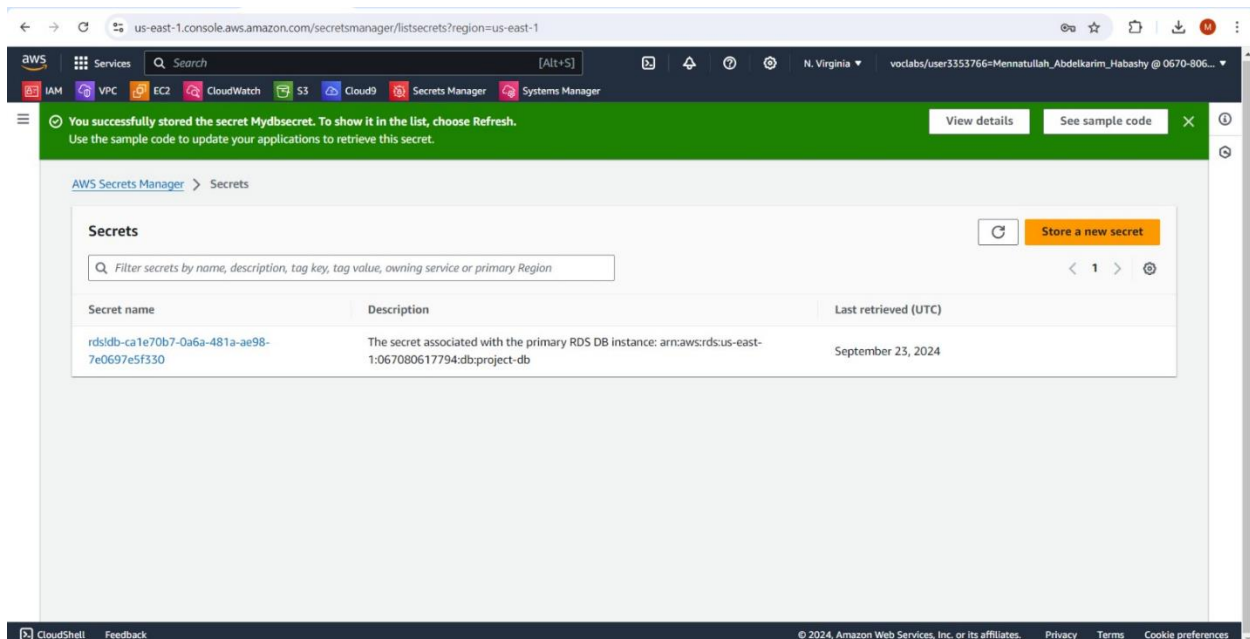
Task 3: Configuring the development environment:

Provision an AWS Cloud9 environment to run AWS Command Line Interface (AWS CLI) commands in later tasks.



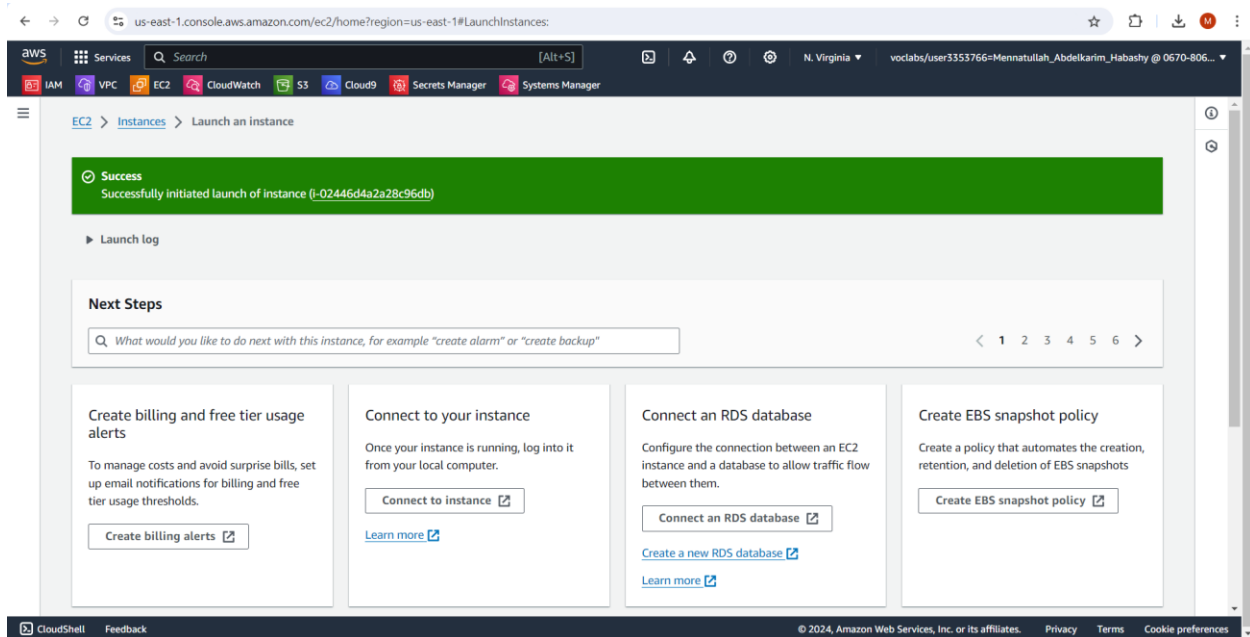
Task 4: Provisioning Secrets Manager:

Use AWS Secrets Manager to create a secret to store the database credentials, and configure the web application to use Secrets Manager.



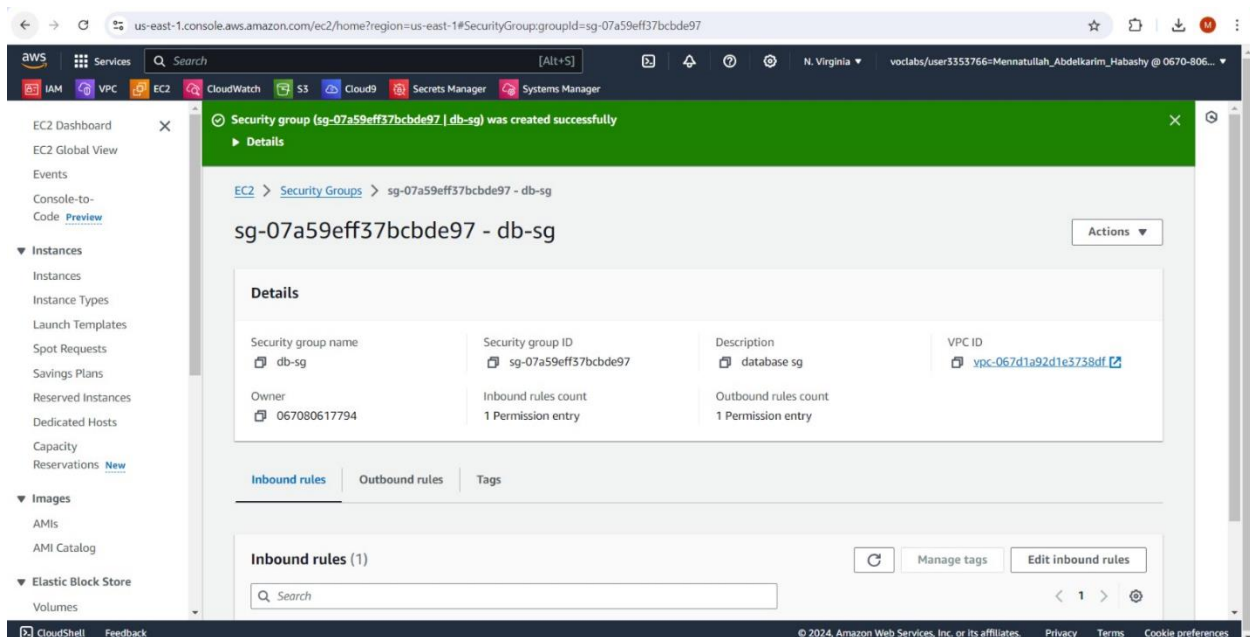
Task 5: Provisioning a new instance for the web server:

Create a new virtual machine to host the web application.



Task 6: Migrating the database:

Migrate the data from the original database, which is on an EC2 instance, to the new Amazon RDS database.




```
us-east-1.console.aws.amazon.com/ec2-instance-connect/ssh?addressFamily=ipv4&connType=standard&instanceId=i-015590991337b570d&osUser=ubuntu&region=us-east-1&sshPort=22#/  
AWS IAM VPC EC2 CloudWatch S3 Cloud9 Secrets Manager Systems Manager  
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 'show da  
tabases' at line 3  
mysql> show databases;  
+-----+  
| Database |  
+-----+  
| STUDENTS |  
| information_schema |  
| mysql |  
| performance_schema |  
| sys |  
+-----+  
5 rows in set (0.01 sec)  
  
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 | menna | k.d | k.d | k.d | k.d@gmail.com | 0123456789 |  
| 2 | me | test | test | test | test@gmail.com | 0123456789 |  
+----+-----+-----+-----+-----+-----+-----+  
2 rows in set (0.00 sec)  
  
mysql>  
  
i-015590991337b570d (project1)  
PublicIPs: 44.212.43.127 PrivateIPs: 10.0.8.35  
CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences
```

Task 7: Testing the application:

Access the application and perform a few tasks to test it. For example, view, add, delete, and modify student records.

Not secure 54.174.177.130/students



XYZ University

Home
Students list

All students

Name	Address	City	State	Email	Phone	
menna	k.d	k.d	k.d	k.d@gmail.com	0123456789	edit
me	test	test	test	test@gmail.com	0123456789	edit

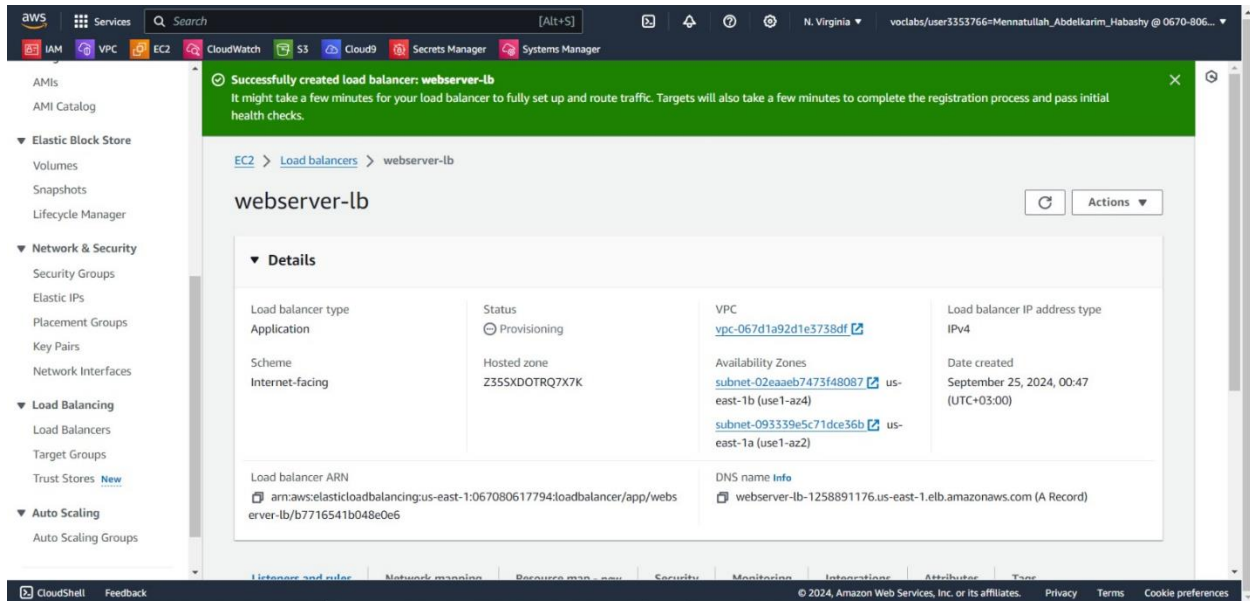
[Add a new student](#)

Phase 4: Implementing high availability and scalability

In this phase, I will complete the design and fulfill the remaining solution requirements. The objective is to use the key components that I created in earlier phases to build a scalable and highly available architecture.

Task 1: Creating an Application Load Balancer:

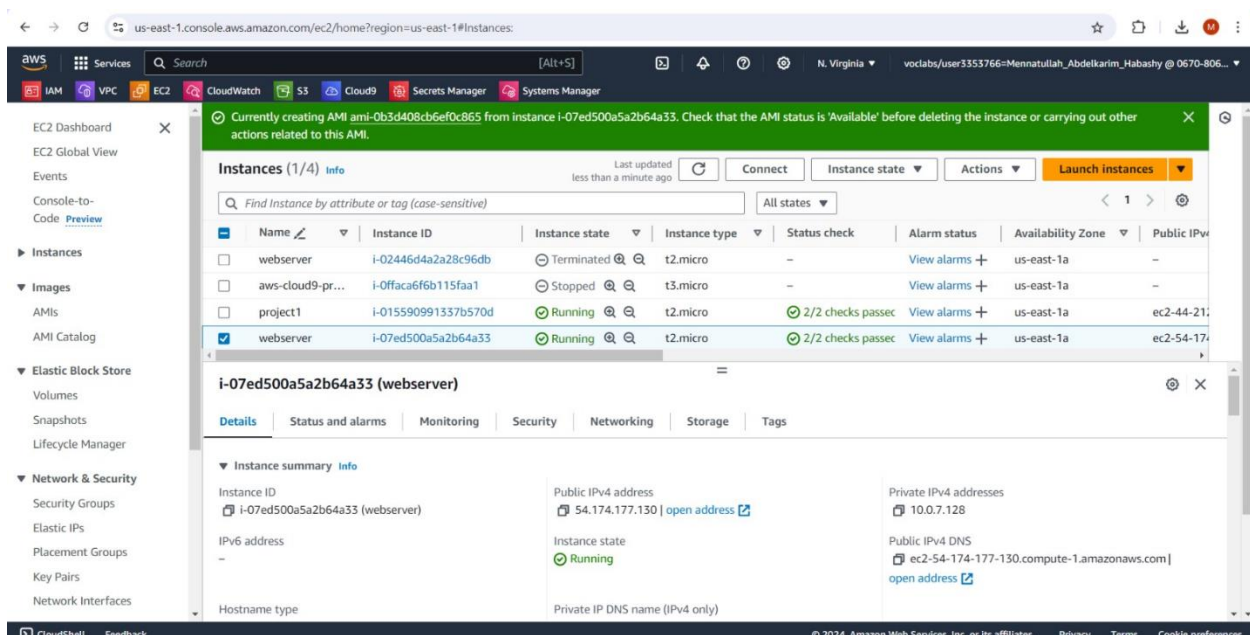
Launch a load balancer. The endpoint will be used to access The web application.



Task 2: Implementing Amazon EC2 Auto Scaling:

In this task I will Create a new launch template, and use an Auto Scaling group to launch the EC2 instances that host the web application.

-Create an image for Auto Scaling:



-create launch template:

Success
Successfully created `webserver[lt-0f5573b94f73795ee]`.

► Actions log

Next Steps

Launch an instance
With On-Demand Instances, you pay for compute capacity by the second (for Linux, with a minimum of 60 seconds) or by the hour (for all other operating systems) with no long-term commitments or upfront payments. Launch an On-Demand Instance from your launch template.
[Launch instance from this template](#)

Create an Auto Scaling group from your template
Amazon EC2 Auto Scaling helps you maintain application availability and allows you to scale your Amazon EC2 capacity up or down automatically according to conditions you define. You can use Auto Scaling to help ensure that you are running your desired number of Amazon EC2 instances during demand spikes to maintain performance and decrease capacity during lulls to reduce costs.
[Create Auto Scaling group](#)

Create Spot Fleet
A Spot Instance is an unused EC2 instance that is available for less than the On-Demand price. Because Spot Instances enable you to request unused EC2 instances at steep discounts, you can lower your Amazon EC2 costs significantly. The hourly price for a Spot Instance (of each instance type in each Availability Zone) is set by Amazon EC2, and adjusted gradually based on the long-term supply of and demand for Spot Instances. Spot instances are well-suited for data-analysis, batch jobs, background processing, and optional tasks.

-Auto Scaling

Auto Scaling groups (1) Info

Search your Auto Scaling groups

	Name	Launch template/configuration	Instances	Status	Desired capacity	Min	Max	Availabil...
<input type="checkbox"/>	webserver	webserver Version Default	0	Updating capacity...	2	1	3	us-east-1a, ...

-create target group

Successfully created the target group: webserver. Anomaly detection is automatically applied to all registered targets. Results can be viewed in the Targets tab.

EC2 > Target groups > webserver

webserver

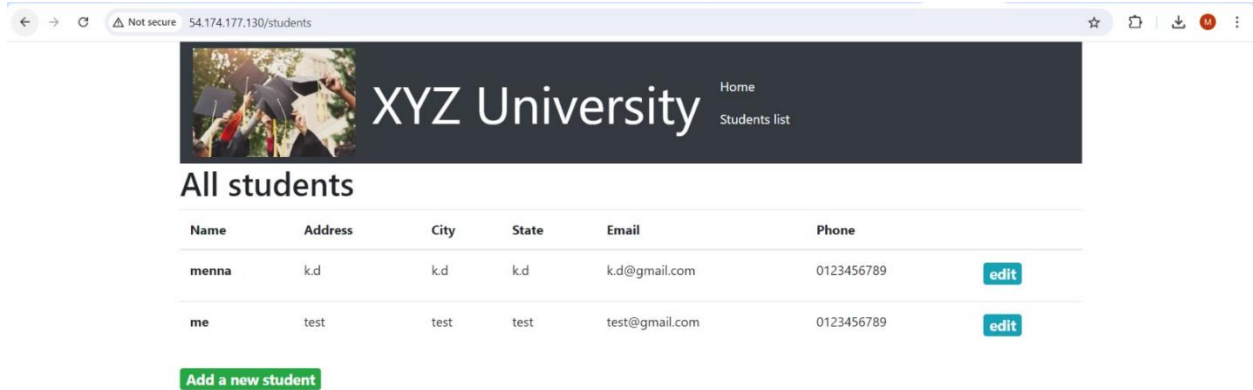
Details
arn:aws:elasticloadbalancing:us-east-1:067080617794:targetgroup/webserver/4f77ba8ad793b56

Target type Instance	Protocol : Port HTTP: 80	Protocol version HTTP1	VPC vpc-067d1a92d1e3738df
IP address type IPv4	Load balancer None associated		
2 Total targets	0 Healthy 0 Anomalous	0 Unhealthy	2 Unused
			0 Initial
			0 Draining

► Distribution of targets by Availability Zone (AZ)
Select values in this table to see corresponding filters applied to the Registered targets table below.

Task 3: Accessing the application:

Access the application and perform a few tasks to test it. For example, view, add, delete, and modify student records.



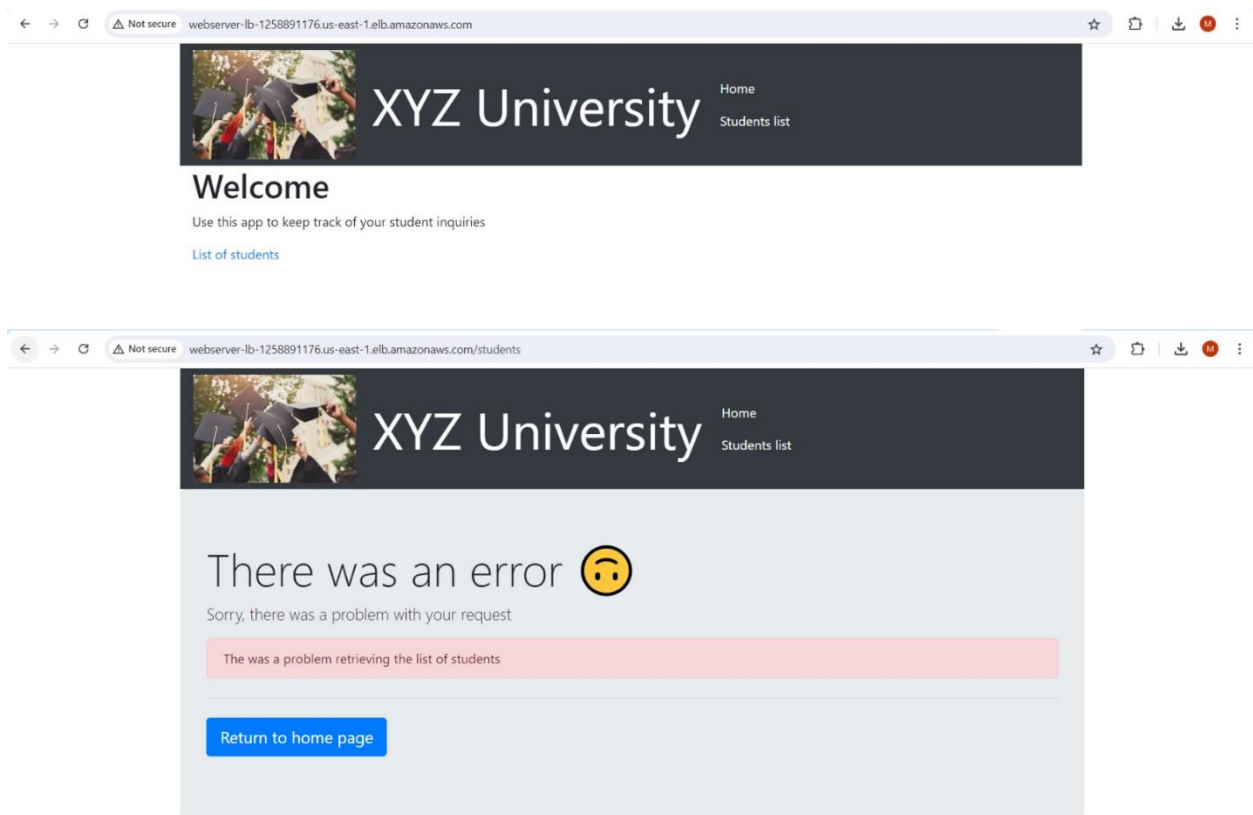
The screenshot shows a web browser at the URL 54.174.177.130/students. The page features a dark header with the XYZ University logo and navigation links for 'Home' and 'Students list'. Below the header, the title 'All students' is displayed. A table lists two students: 'menna' and 'me'. Each row includes columns for Name, Address, City, State, Email, and Phone, along with an 'edit' button. At the bottom, there is a green button labeled 'Add a new student'.

Name	Address	City	State	Email	Phone	
menna	k.d	k.d	k.d	k.d@gmail.com	0123456789	edit
me	test	test	test	test@gmail.com	0123456789	edit

[Add a new student](#)

Task 4: Load testing the application:

Perform a load test on the application to monitor scaling.



The first screenshot shows the 'Welcome' page of the XYZ University application. The URL is webserver-lb-1258891176.us-east-1.elb.amazonaws.com. The page includes the university logo, navigation links, and a message: 'Use this app to keep track of your student inquiries'. A link for 'List of students' is provided.

The second screenshot shows an error page with the title 'There was an error' and a sad face emoji. The message states: 'Sorry, there was a problem with your request'. A red box contains the text: 'The was a problem retrieving the list of students'. A blue button labeled 'Return to home page' is at the bottom.

And Finally this is My badge from AWS Academy ,

<https://www.credly.com/go/D2WGJS12>

