Assignment 2 -Developing a highly available Photo Album website

Sanyam Verma, Thursday 08:30AM -10:30 AM, 09-10-2022

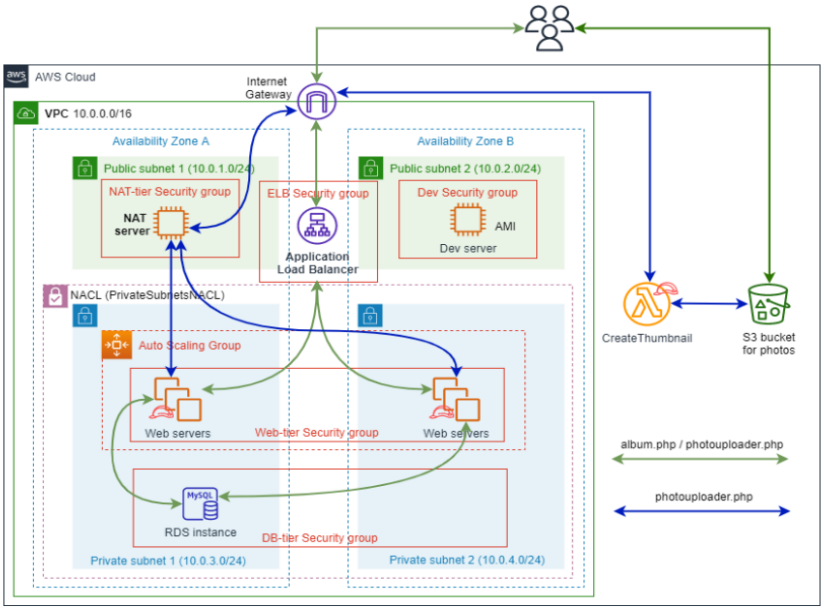
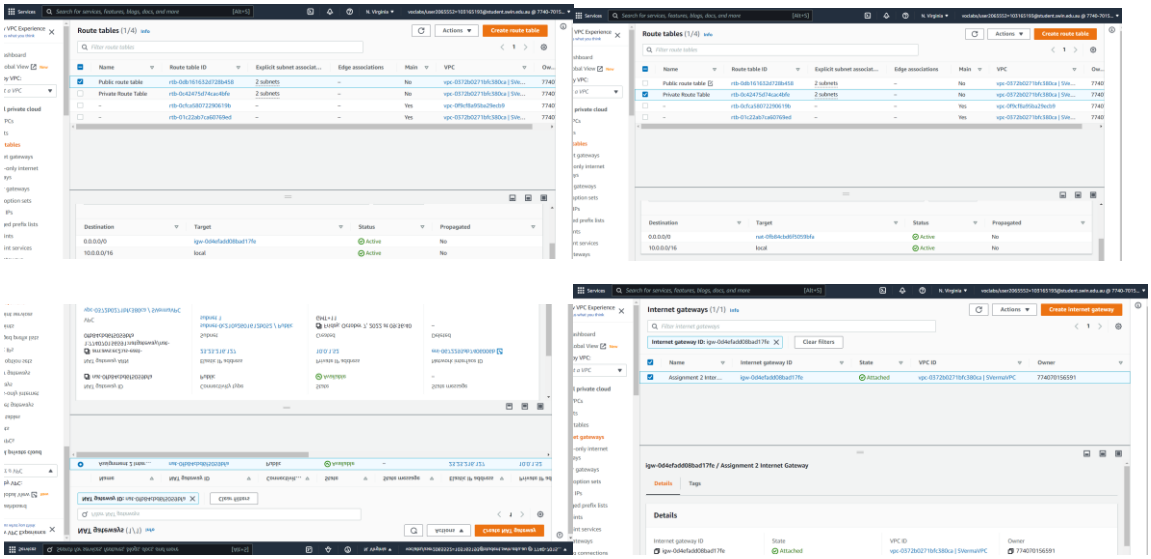
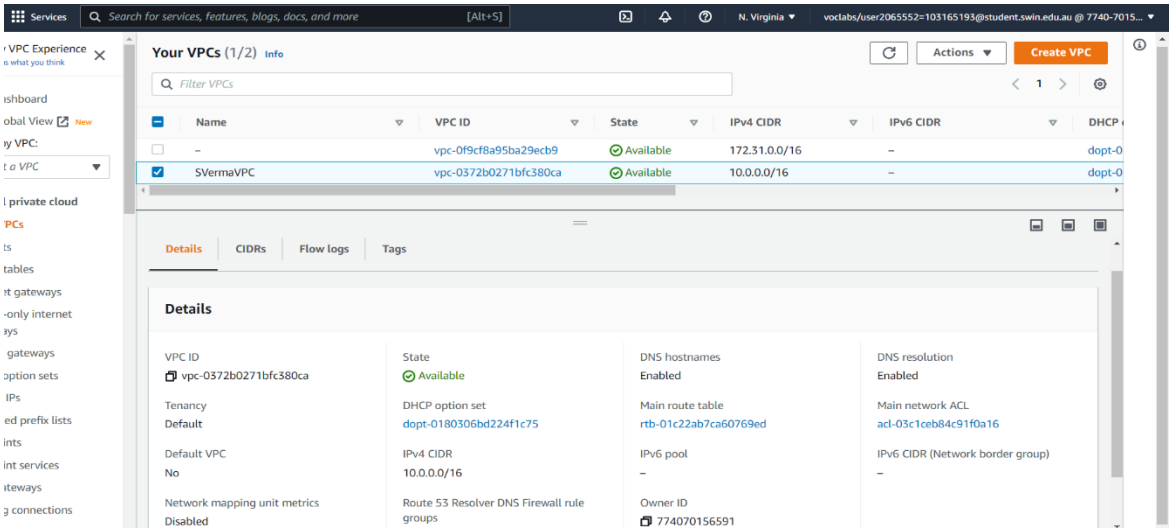


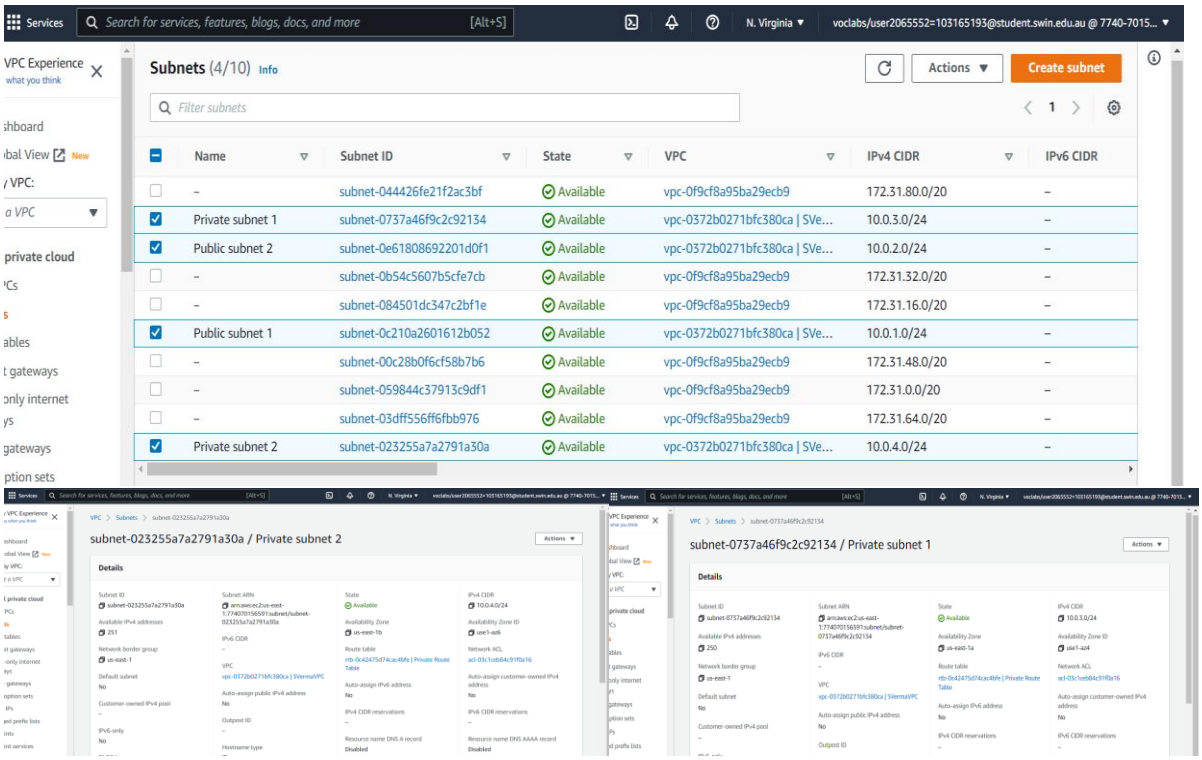
Figure 1 - Architecture diagram

Fig1

1. **VPC configured with 2AZs both with public and private subnets. Public and private route tables route to IGW and NAT-** In fig 2 a VPC is created with 4 subnets (2 public and 2 private) with different CIDR ranges connected to it. VPC has 2 availability zones with 1 public and 1 private in one availability zones and 1 public, 1 private in other availability zones. Also, public subnets are connected to public route table and private route tables are connected to private subnets. Also, public route table is connected to internet gateway and private route table is connected to NAT gateway for the traffic allowance as shown in fig 1.




Services
Search for services, features, blogs, docs, and more
[Alt+S]
N. Virginia
voclabs/user2065552=103165193@student.swin.edu.au @ 7740-7015...
Your VPCs (1/2) Info
Actions
Create VPC
Filter VPCs
Name
VPC ID
State
IPv4 CIDR
IPv6 CIDR
DHCP
vpc-0f9cf8a95ba29ecb9
Available
172.31.0.0/16
SVermaVPC
vpc-0372b0271bfc380ca
Available
10.0.0.0/16
Details
CIDRs
Flow logs
Tags
Details
VPC ID
vpc-0372b0271bfc380ca
State
Available
DNS hostnames
Enabled
DNS resolution
Enabled
Tenancy
Default
DHCP option set
dopt-0180306bd224f1c75
Main route table
rtb-01c22ab7ca60769ed
Main network ACL
acl-03c1ceb84c91f0a16
Default VPC
No
IPv4 CIDR
10.0.0.0/16
IPv6 pool
IPv6 CIDR (Network border group)
Network mapping unit metrics
Disabled
Route 53 Resolver DNS Firewall rule groups
Owner ID
774070156591


Services
Search for services, features, blogs, docs, and more
[Alt+S]
N. Virginia
voclabs/user2065552=103165193@student.swin.edu.au @ 7740-7015...
Subnets (4/10) Info
Actions
Create subnet
Filter subnets
Name
Subnet ID
State
VPC
IPv4 CIDR
IPv6 CIDR
subnet-044426fe21f2ac3bf
Available
vpc-0f9cf8a95ba29ecb9
172.31.80.0/20
Private subnet 1
subnet-0737a46f9c2c92134
vpc-0372b0271bfc380ca | SVe...
10.0.3.0/24
Public subnet 2
subnet-0e61808692201d0f1
10.0.2.0/24
subnet-0b54c5607b5cfe7cb
172.31.32.0/20
subnet-084501dc347c2bf1e
172.31.16.0/20
Public subnet 1
subnet-0c210a2601612b052
10.0.1.0/24
subnet-00c28b0f6cf58b7b6
172.31.48.0/20
subnet-059844c37913c9df1
172.31.0.0/20
subnet-03dff556ff6fbb976
172.31.64.0/20
Private subnet 2
subnet-023255a7a2791a30a
10.0.4.0/24

subnet-023255a7a2791a30a / Private subnet 2
subnet-0737a46f9c2c92134 / Private subnet 1

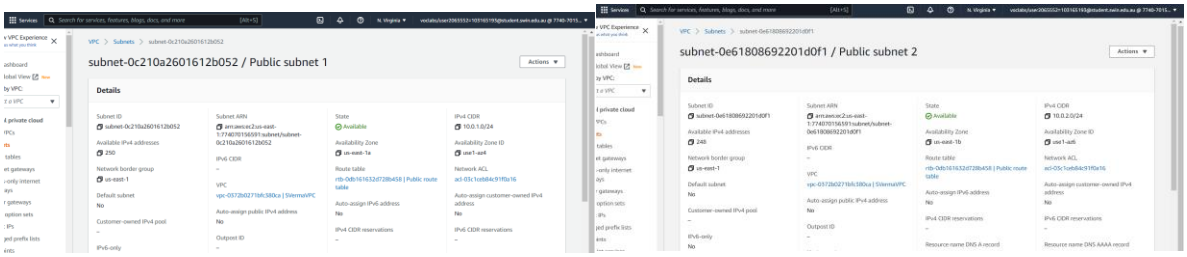

subnet-0c210a2601612b052 / Public subnet 1
subnet-0e61808692201d0f1 / Public subnet 2

Fig 2

2. **Security groups correctly configured** – For the DBserverSG outbound traffic is from web servers via via MYSQL/Aurora. For the WebserverSG inbound rules from ELB security group is allowed via HTTP. DevserverSG is all traffic allowed according to specifications (it does not follow least privilege principle. The ELB security group inbound rules is all HTTP, HTTPS(it accepts from all servers that sends HTTP,HTTPS servers) as shown in fig 3.

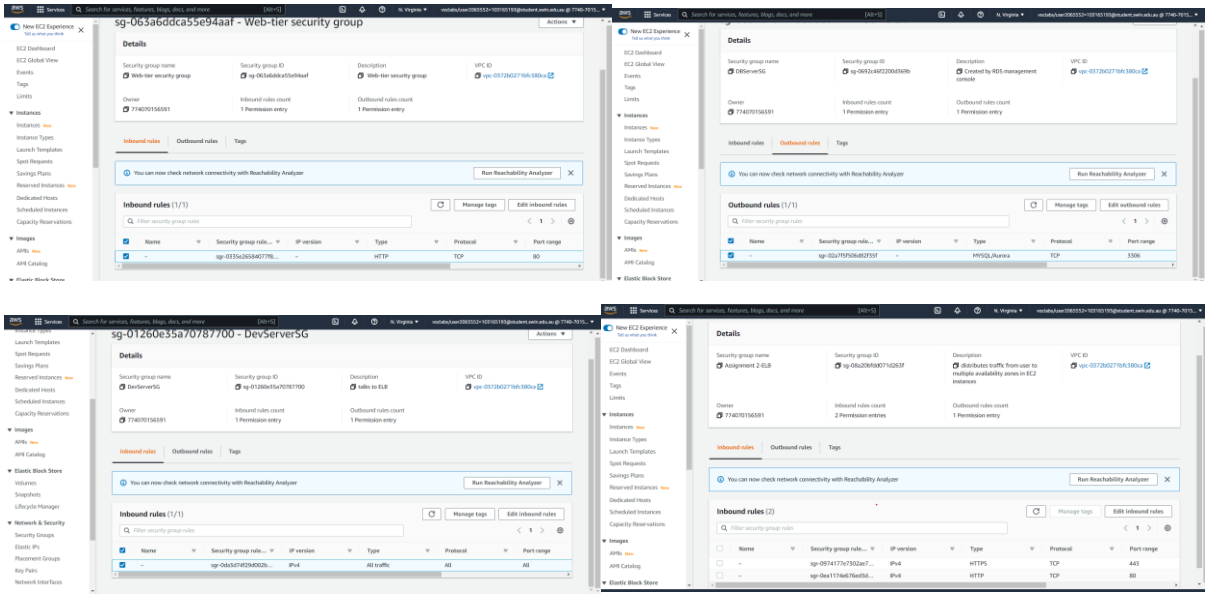
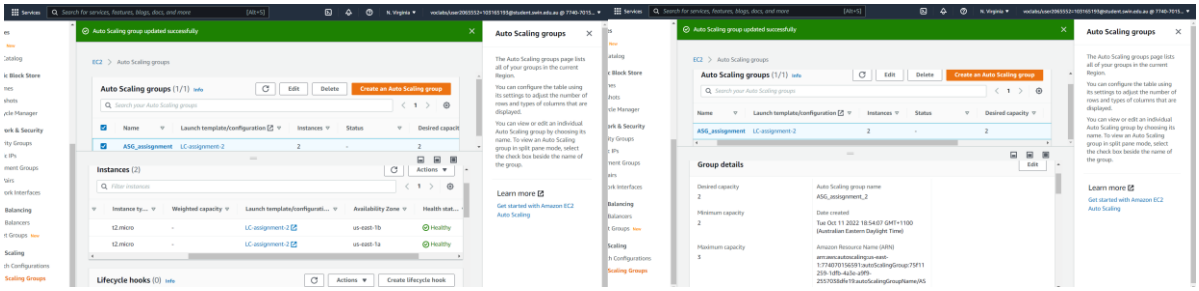


Fig 3

3. **ASG configured and working correctly** - The minimum number of servers is 2. The maximum number of servers is 3. Also, there is a target tracking scaling policy to keep the request count per target of your ELB target group at 30 for Auto Scaling group. Also, it launches instances in private subnets as shown in fig 4.



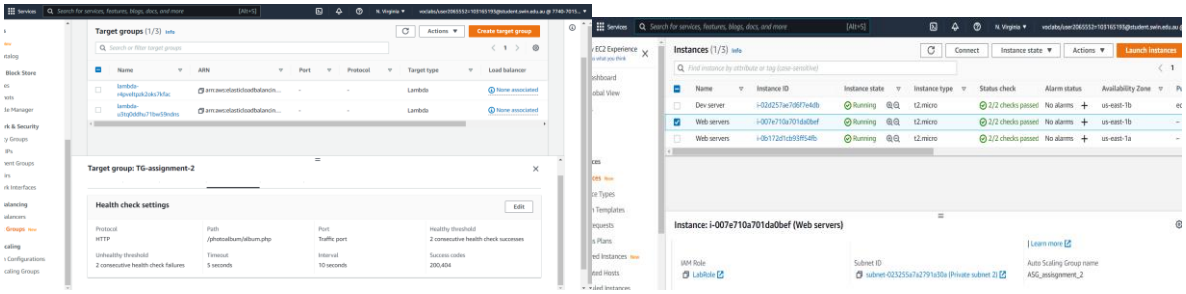


Fig 4

4. **ELB configured correctly with elastic IP public address** – It launches web servers by auto scaling group in the private subnets, and all the health checks of these instances are healthy which is done by target groups for the instances launched by auto scaling group in private subnets as shown in fig 5.

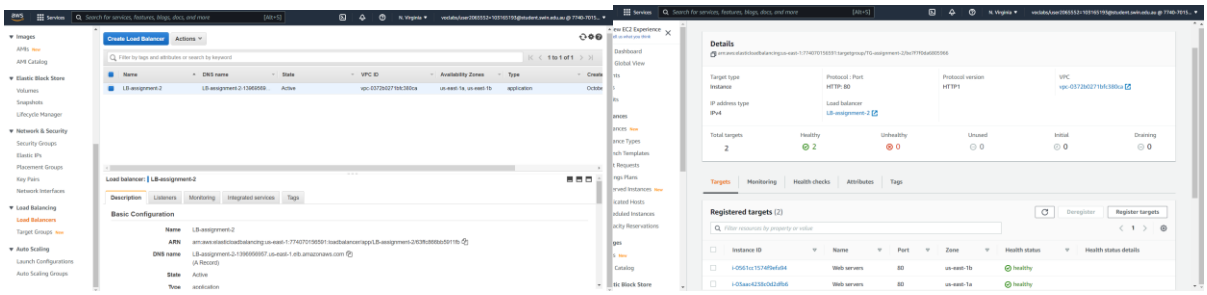
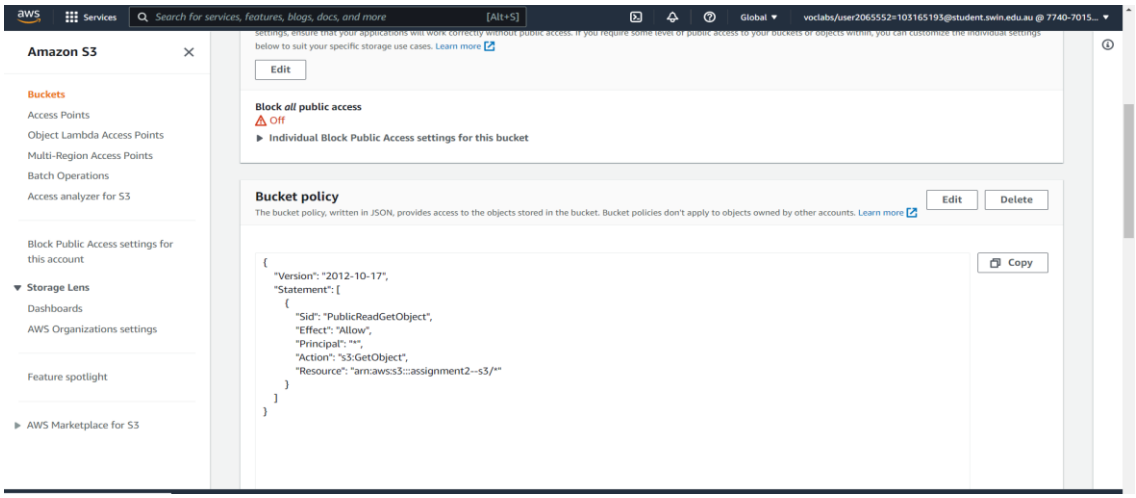


Fig 5

5. **Photos stored in S3 are correctly accessible and bucket policy is correct –** It is an example as shown image 4.3, if we copy the object URL and paste it in on browser, it is accessible on the browser tab. All objects (photos) in this S3 bucket are publicly available. An appropriate access policy to enable public access to all available objects in this S3 bucket. The S3 bucket policy is configured correctly that restricts access to a specific HTTP referrer as shown in fig 3 as shown in fig 6.



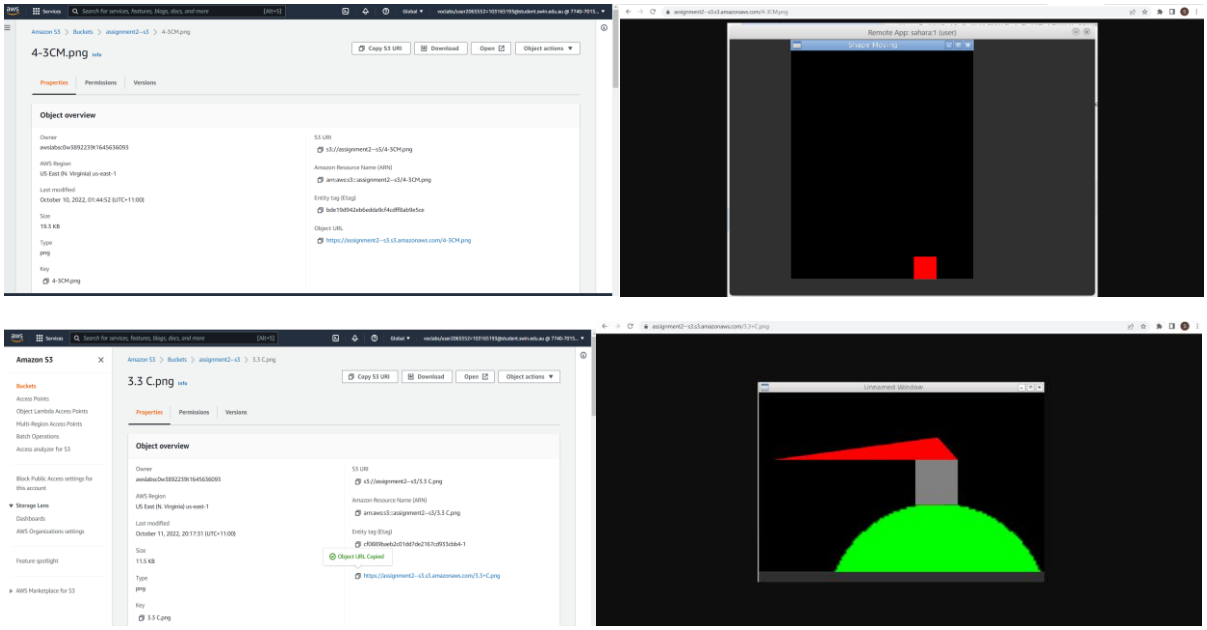


Fig 6

6. **Lambda configured and working correctly –** It follows the specifications of runtime python 3.7 and an IAM role is configured with policies that allow this Lambda function to get objects from and put objects into the S3 bucket and it resizes the size of the image in S3 bucket. Also, the test the function with our bucket and image details in the event JSON tab and it was successful as shown in fig 3 as shown in fig 7.

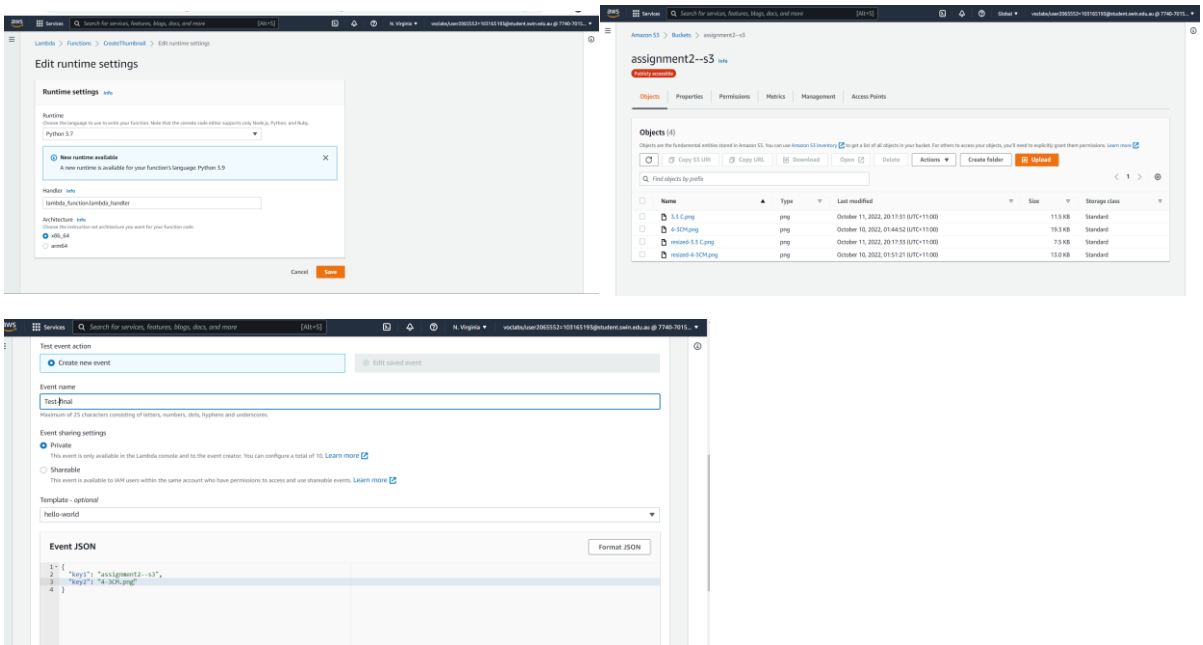


Fig 7

7. RDS configured and working correctly – website is accessible through database and it resides in private subnets (1,2) with the VPC attached to it, it also has webserverSG security rule which sends traffic to web servers as shown in fig 8. PhpMyAdmin was accessible via Dev server, it is only possible if it's configuration is correct.

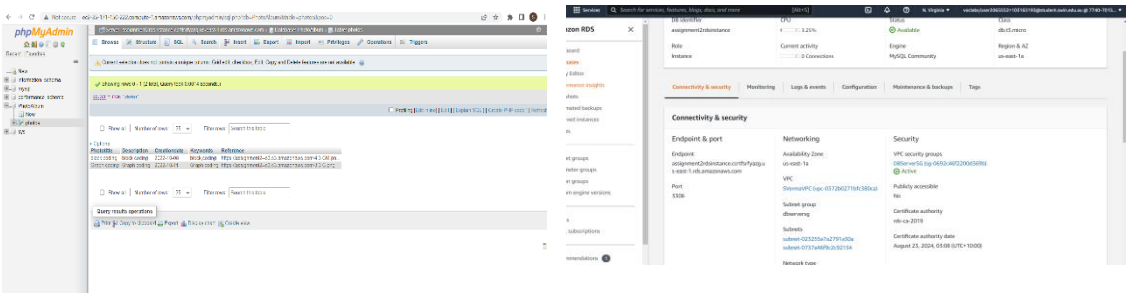


Fig 8

8. **Photos stored in S3 are correctly accessible. S3 bucket policy is** correct – In S3 bucket with pasting object URL in the browser we get the image as shown in fig 9. Also, the policy given is configured correctly with putting our bucket name in the resource and in principal putting * in between.

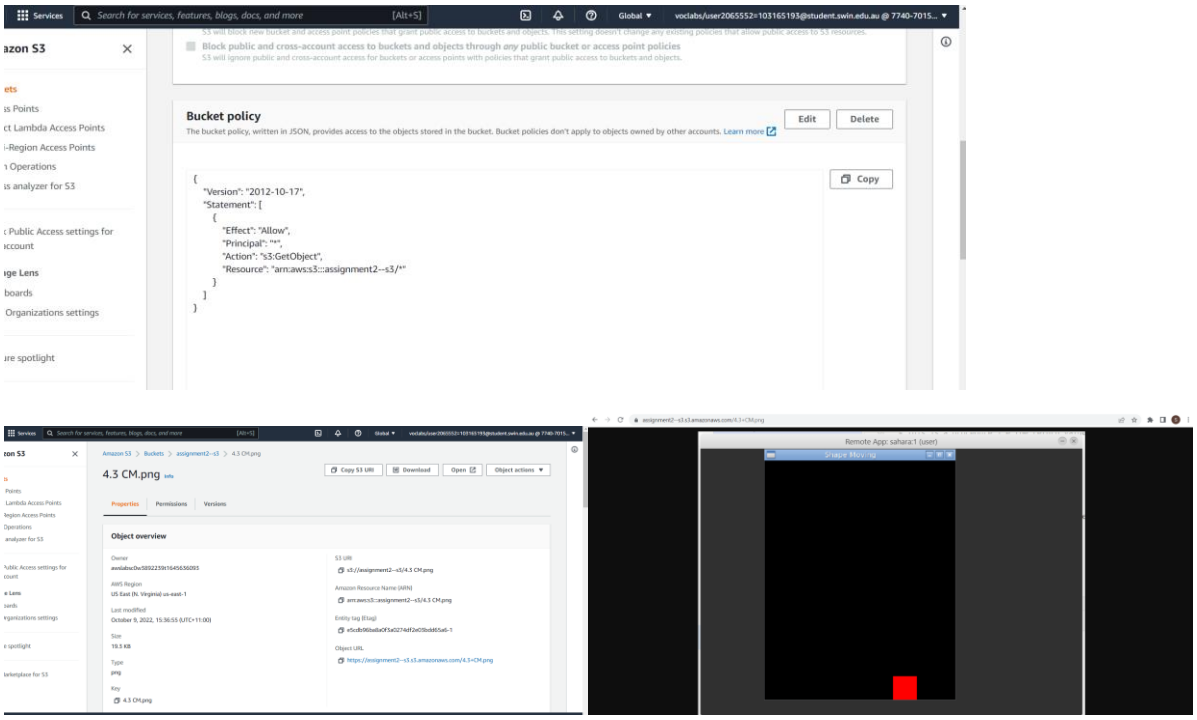


Fig 9

9. **Website accessible via ELB** - The website is accessible through http://[your.elb.dns]/photoalbum/album.php as shown in figure 10.

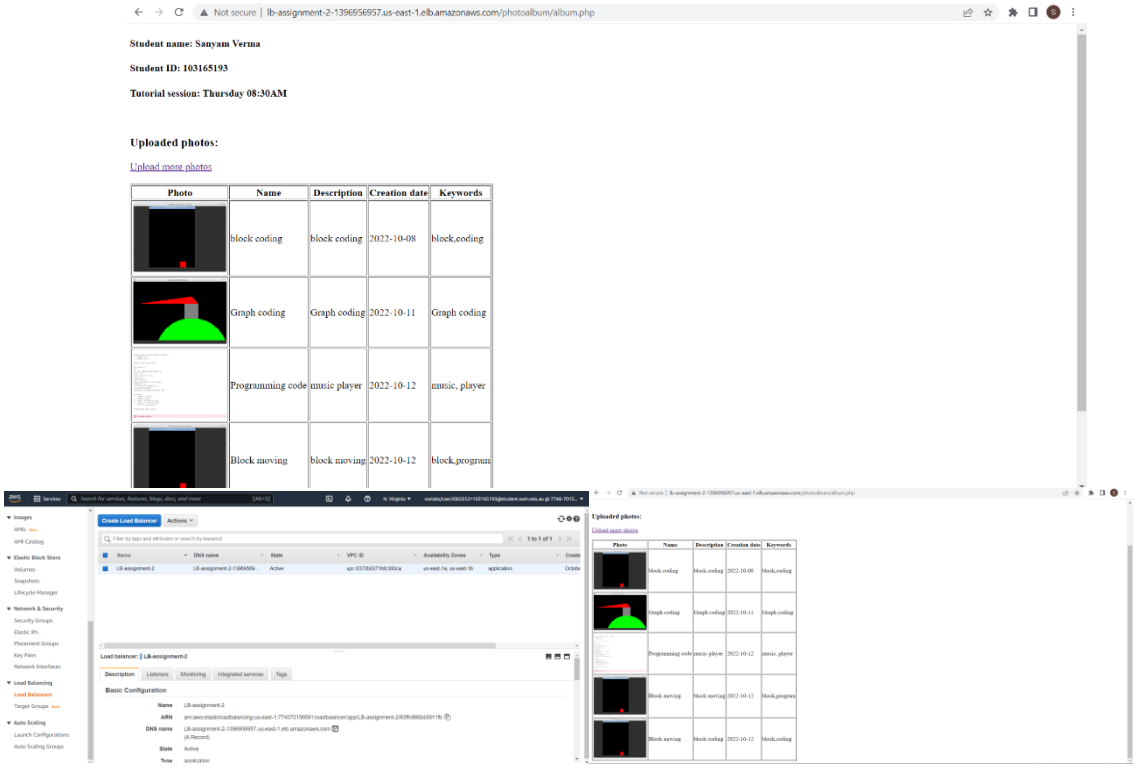


Fig 10

10. **Photos and meta data displayed on albums.php page** – In below figure Photos uploaded are shown under photos tab, and the meta-data is shown in next rows and columns like Name(block coding), Description(block coding), Creation date(22-10-08), Keywords(block, coding) as shown in fig 11.

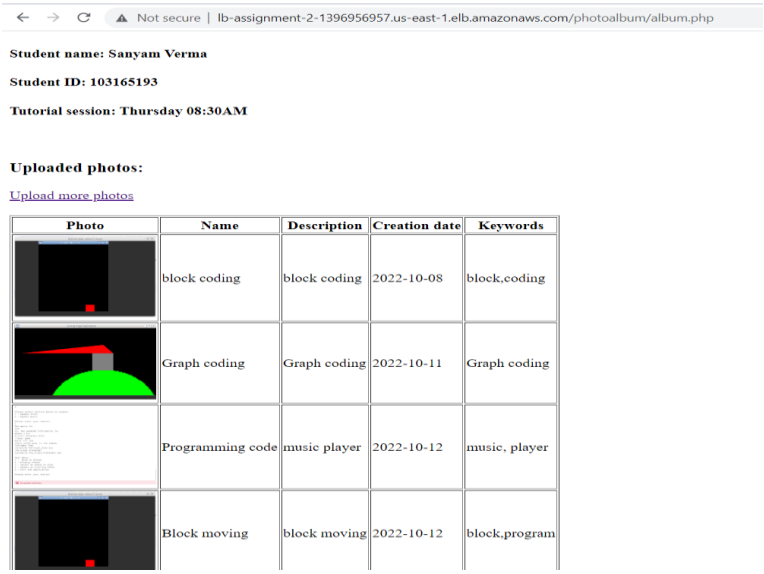
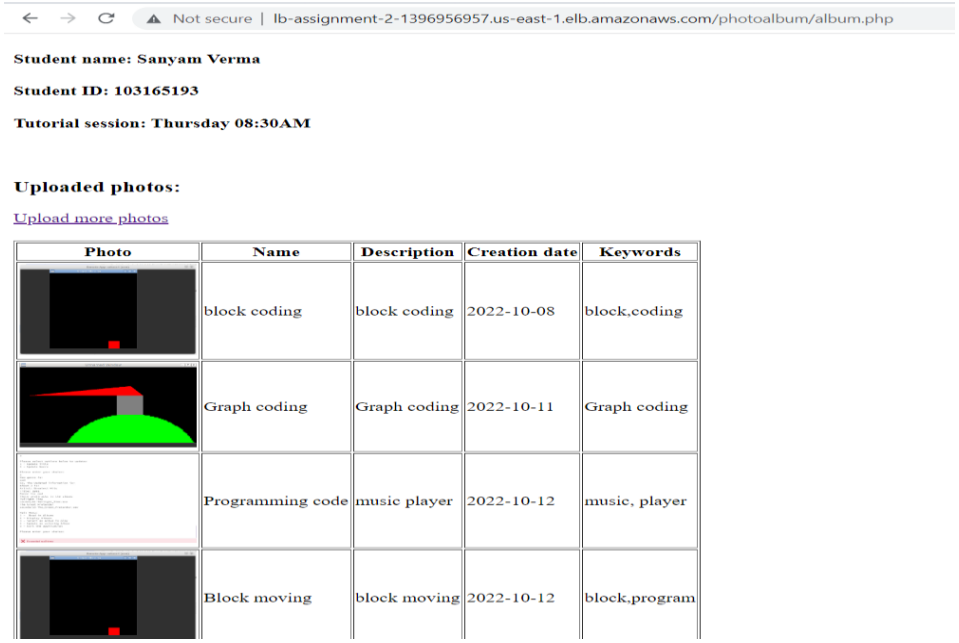


Fig 11

11. **Photos and their meta-data can be uploaded to the S3 bucket and RDS database, respectively -** I uploaded the photos on web page as shown in fig 9. The photos uploaded in the web-site, their meta-data is shown in S3 bucket and RDS database respectively as shown in fig 12.



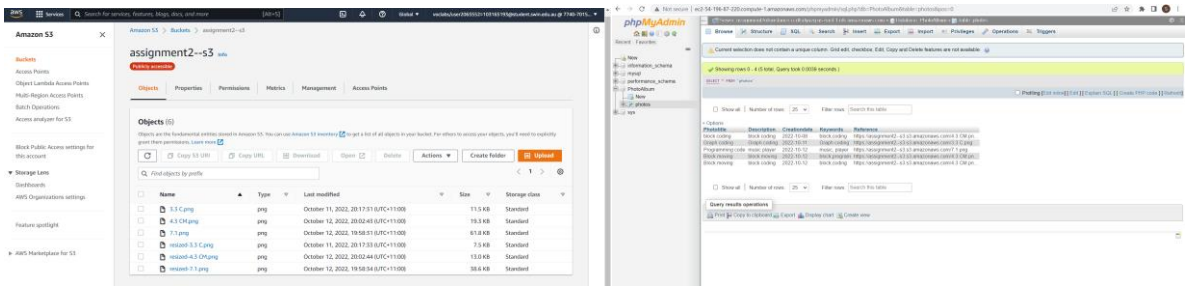


Fig 12

12. **Photos are resized by lambda function –** Photos uploaded from the web-site is resized into smaller size by lambda function. The lambda contains package provided by the unit convener which contains the library and full source code to resize images and download/upload images to S3 (for best result, please use PNG images). The package is ready to work without any modification as shown in fig 13.

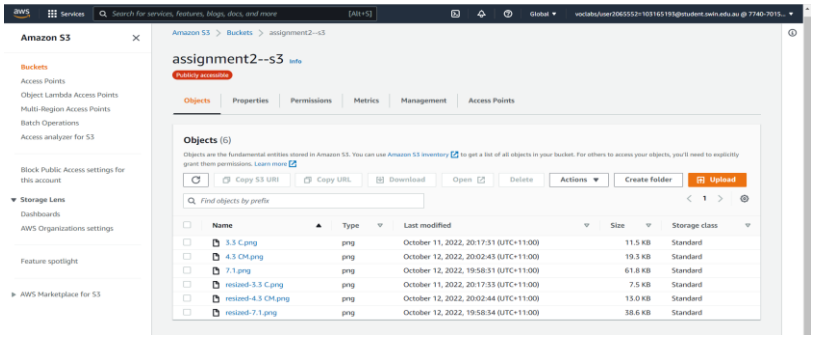
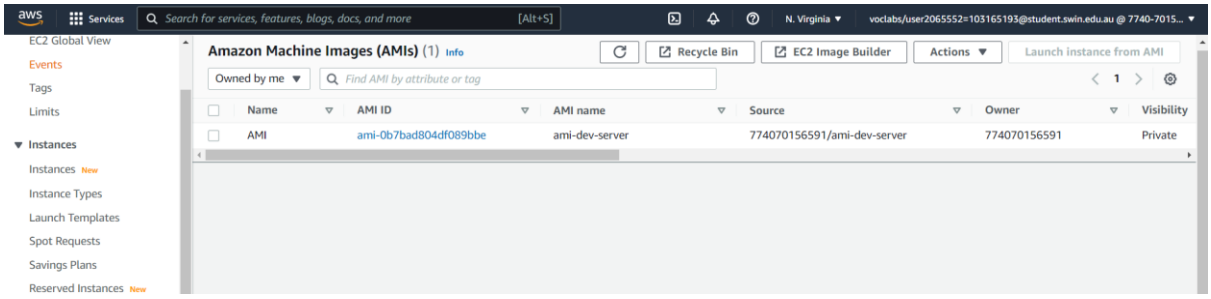


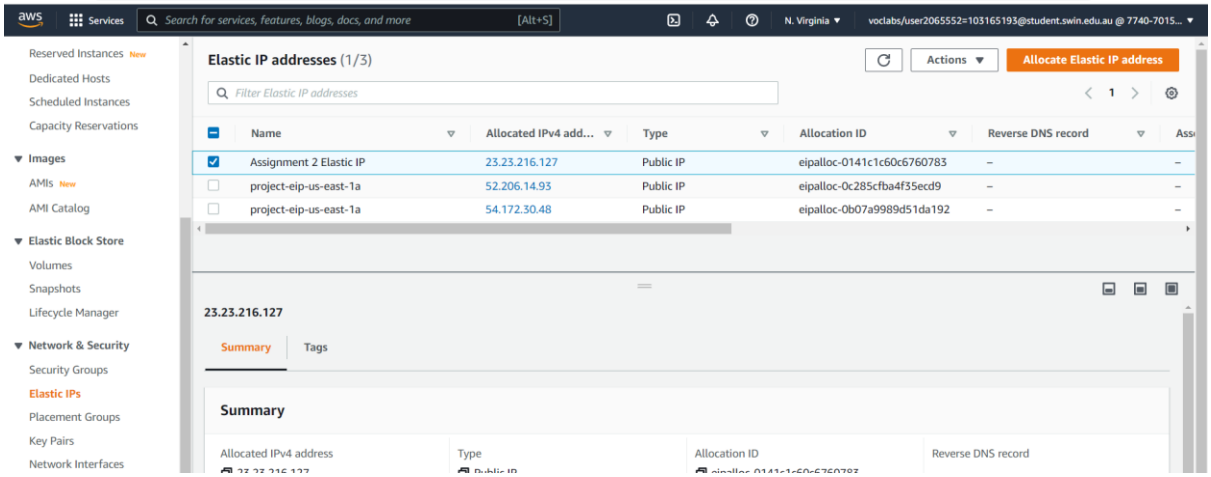
Fig 13

Errors faced

First was the health check, my web server instances were not showing healthy, so I configured it's health path and the source code I changed to 202,404 due to SSL error as shown in figure 4. Also, my website was not accessible via ELBdns (elastic public address of Load balancer), I selected default AMI for the Dev server and the AMI used in launch configuration (LC) was different. So, through AMI, I launched a new instance which was similar to AMI used by LC.



(AMI used)



(Elastic IP used)