ENPM 665 – Final CobraKai Application

Tanishq Javvaji
UID : 119070185

Contents

Introduction:

Advantages of Moving to the cloud:

- **Cost Savings:** Cloud computing can often be more cost-effective than on-premises infrastructure. By moving to the cloud, Cobrakai can reduce or eliminate the need for expensive hardware and IT staff.
- **Scalability:** AWS offers the users to scale resources easily using the AWS load balancer. For the cobrakai application, if the server load exceeds the existing load, we can always use more instances and use Load balancer to balance the load.
- **Reliability:** Using AWS, the website can be hosted in multiple regions, expanding the website's reach.
- **Security:** AWS has several security features to protect its services and ensure smooth and reliable working of the application.
- **Agility**: AWS enables Cobrakai to quickly deploy new applications and patches, which can help them respond to changing market conditions and customer needs.

Migrating to Aws

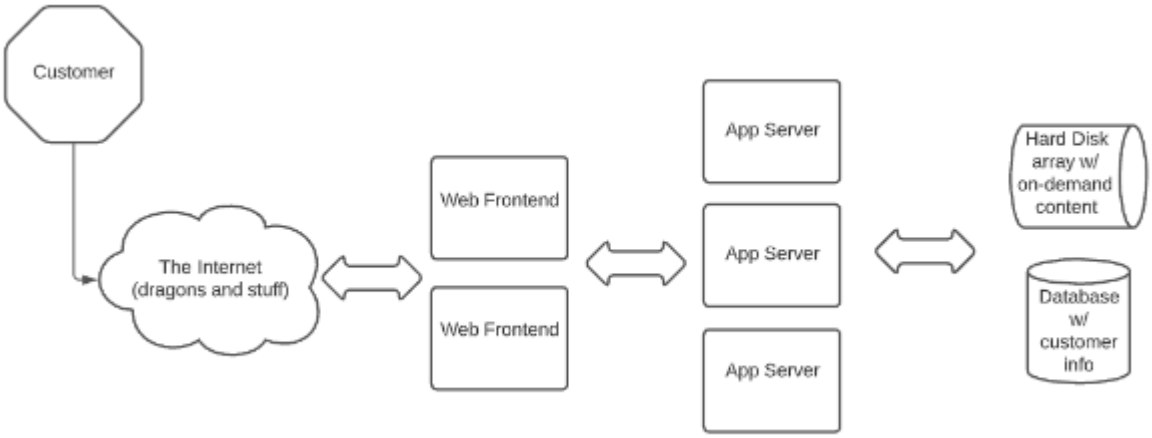
There are several steps that a cobrakai can take when moving to Amazon Web Services (AWS) cloud:

- **Select a migration strategy**: For the Cobrakai application, we will use a method called re-architecture, changing the architecture to a much more advanced and secure application.
- **Plan migration**: Create a detailed plan for migrating your workloads to the cloud. This should include timelines, budgets, and responsibilities for each team member.
- **Set up your AWS account**: For representational purposes, I have used my AWS account to demonstrate various AWS services used in the new architecture.
- **Test and validate**: Once your workloads are in the cloud, test them to ensure they are working as expected. This will help you identify and fix any issues before going live.
- **Go live**: When you are confident that your workload is ready, you can switch to the cloud and decommission your on-premises infrastructure.

Current Issues to Consider

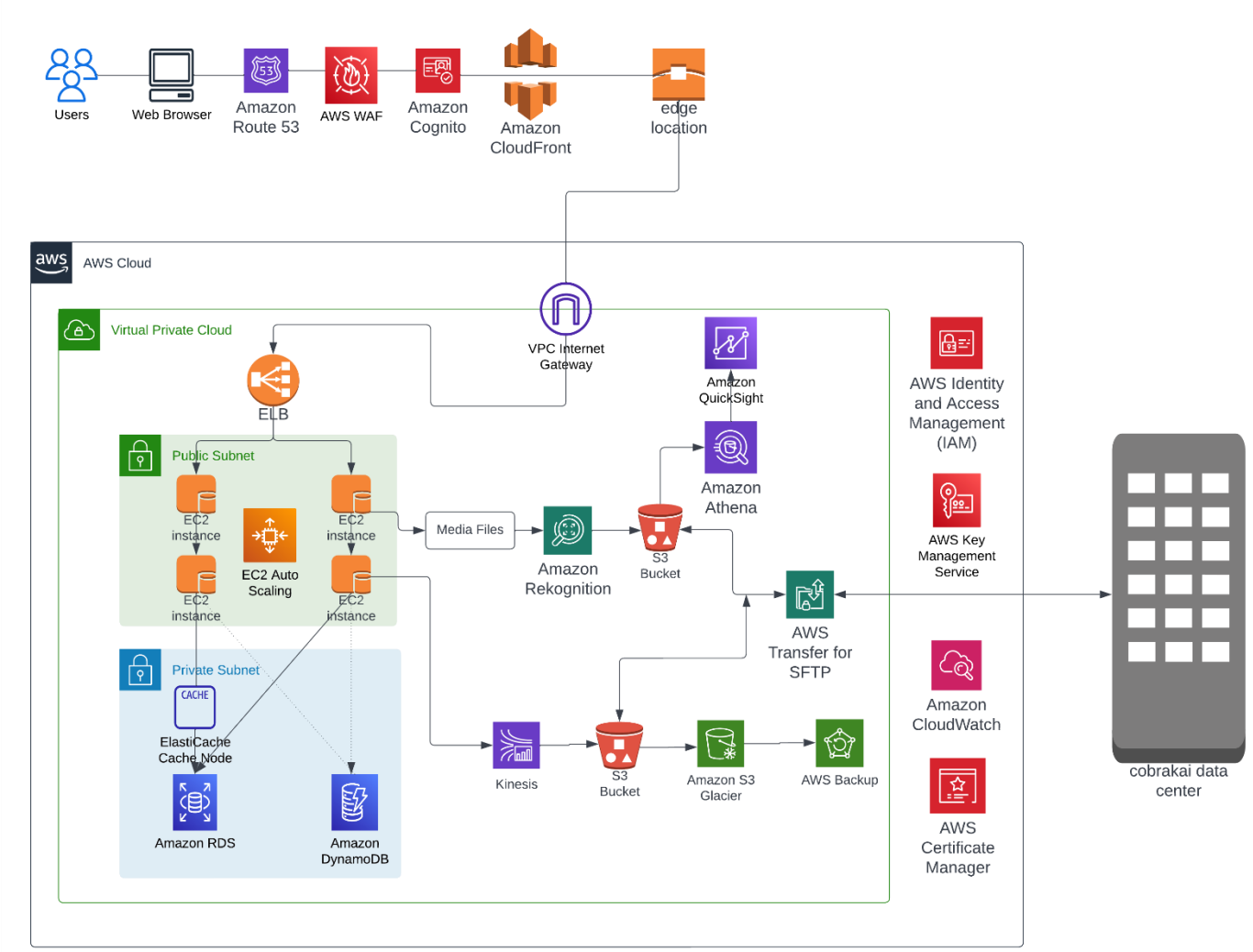
- Cobra Kai does not currently have a patching strategy.
- Cobra Kai does not currently have a backup strategy.
- Cobra Kai does not currently have an account permission strategy, every user has the
- ability to run privileged commands on the web server if they want to.
- Their entire website infrastructure is highly vulnerable to DDoS, hardware failures, and human error. It runs in a closet for crying out loud.
- The website has experienced DDoS attacks and compromise attempts they suspect
- comes from a rival dojo ran by Daniel LaRusso who with his deep pockets has become a persistent threat against Cobra Kai's l IT operations.
- Customers have complained about slow streaming, downloads, and order processing.
- Cobra Kai's platform is processing credit card data and also stores customer PII (name, phone, email, address, and additional details about the customer).
- Cobra Kai's corporate IP range is 129.2.0.0/16 (it's not really but pretend it is).

Current Website Architecture



Customer comes over the Internet and connects to one of the front-end servers via round robin DNS. Data/actions are then sent to one of 3 app servers for processing and the app server will either record data in a master database or sending streaming on-demand content stored on a hard disk array back to the user.

Proposed Architecture using AWS and its services:



In the Proposed architecture, the users connect to the application using the domain name www.cobrakai.com; they are redirected to the website with a secure connection through route 53. AWS CloudFront acts as a content delivery network which can cache static content in edge locations. The edge locations are across the cities, across 100 plus cities across the world. Using CloudFront service, all your data from S3 gets cached in the nearest edge location from where the user is coming. The data is always served from that edge location for all the users in that geography. In AWS, there is a service called ELB, which can distribute the incoming traffic to multiple EC2 machines. We use Amazon RDS for relational databases such as MySQL, and for non-sequential databases, we use Amazon DynamoDB. All media files run through an Amazon Rekognition service that filters videos and images before storing them in S3 buckets. S3 buckets act as external storage. Amazon Simple Storage Service, or S3, provides a variety of storage classes based on Access frequency and time to access the bucket. For the cobra kai application, we will use the S3 standard service for storing the videos and the S3 Glacier deep archive to back up all the data. Amazon Athena service is a

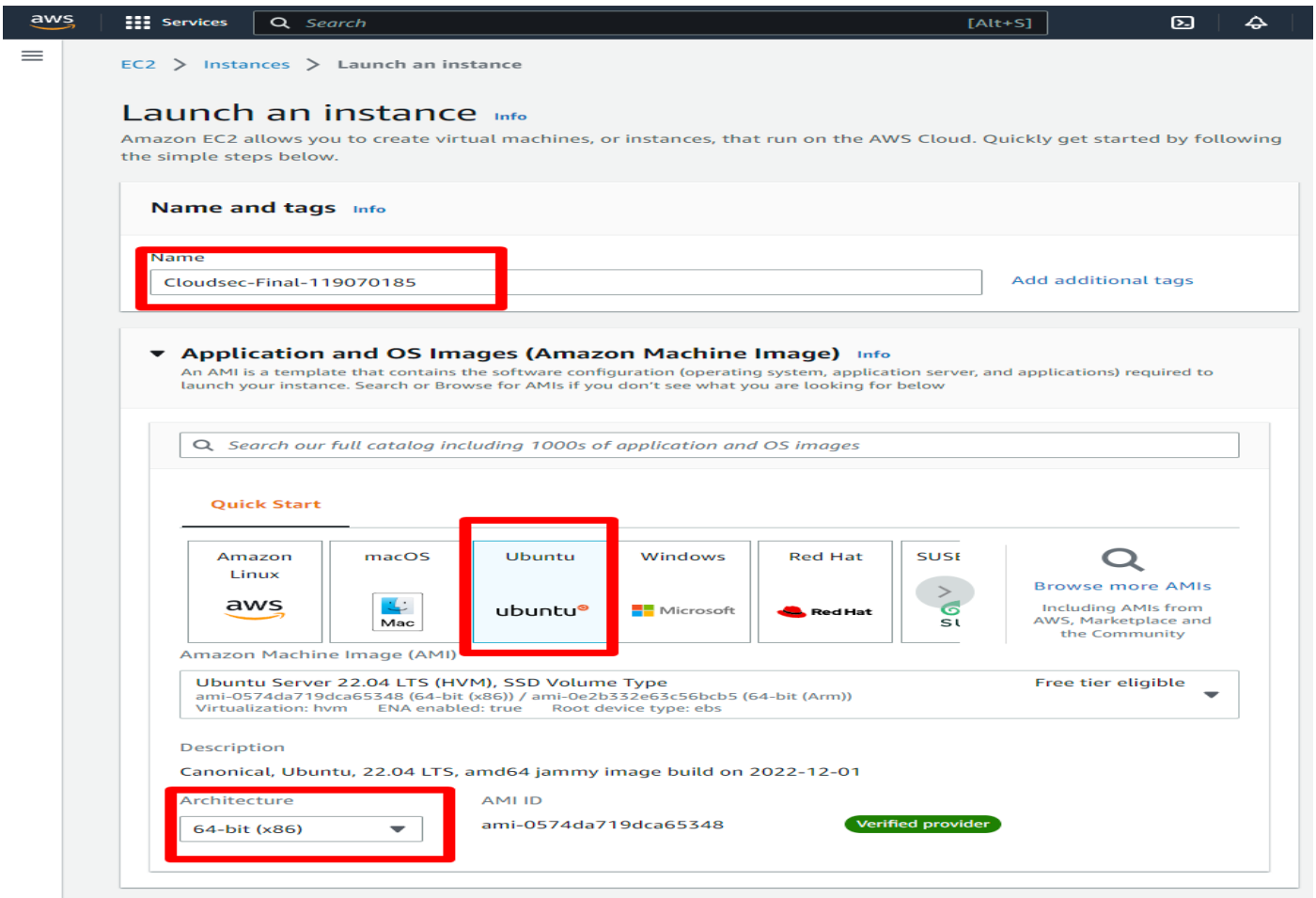
data analytics service which provides analysis of the data present in S3 buckets. We can see all these analytics in quicksite. AWS Transfer for SFTP transfers data from cobrakai's data centre to S3 buckets securely.

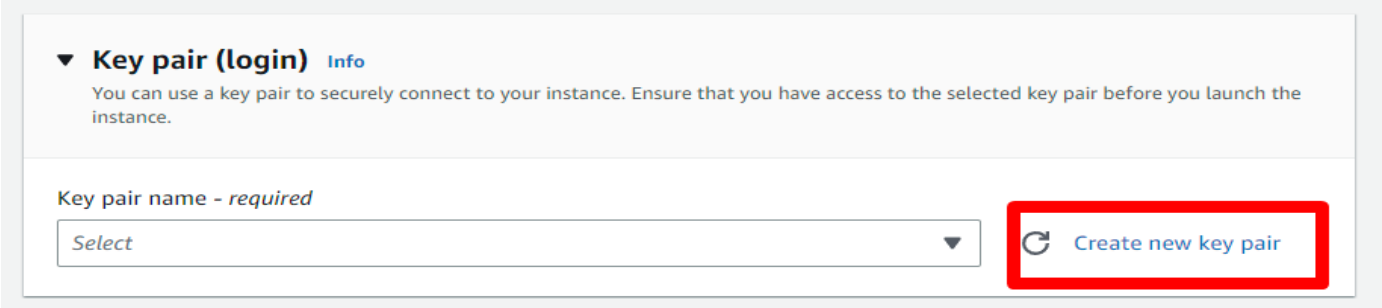
In terms of security, we use various services such as AWS IAM, AWS WAF, AWS Cloud Watch, AWS key management and AWS certificate manager.

1. IAM is used to give roles and attach policies to each role.
2. The HTTP and HTTPS requests are monitored by AWS WAF.
3. Amazon CloudWatch is a monitors AWS services
4. AWS KMS lets you manage encryption keys used to protect data.
5. AWS certification manager deployed on CloudFront and load balancers for secure communication between the application and the user.

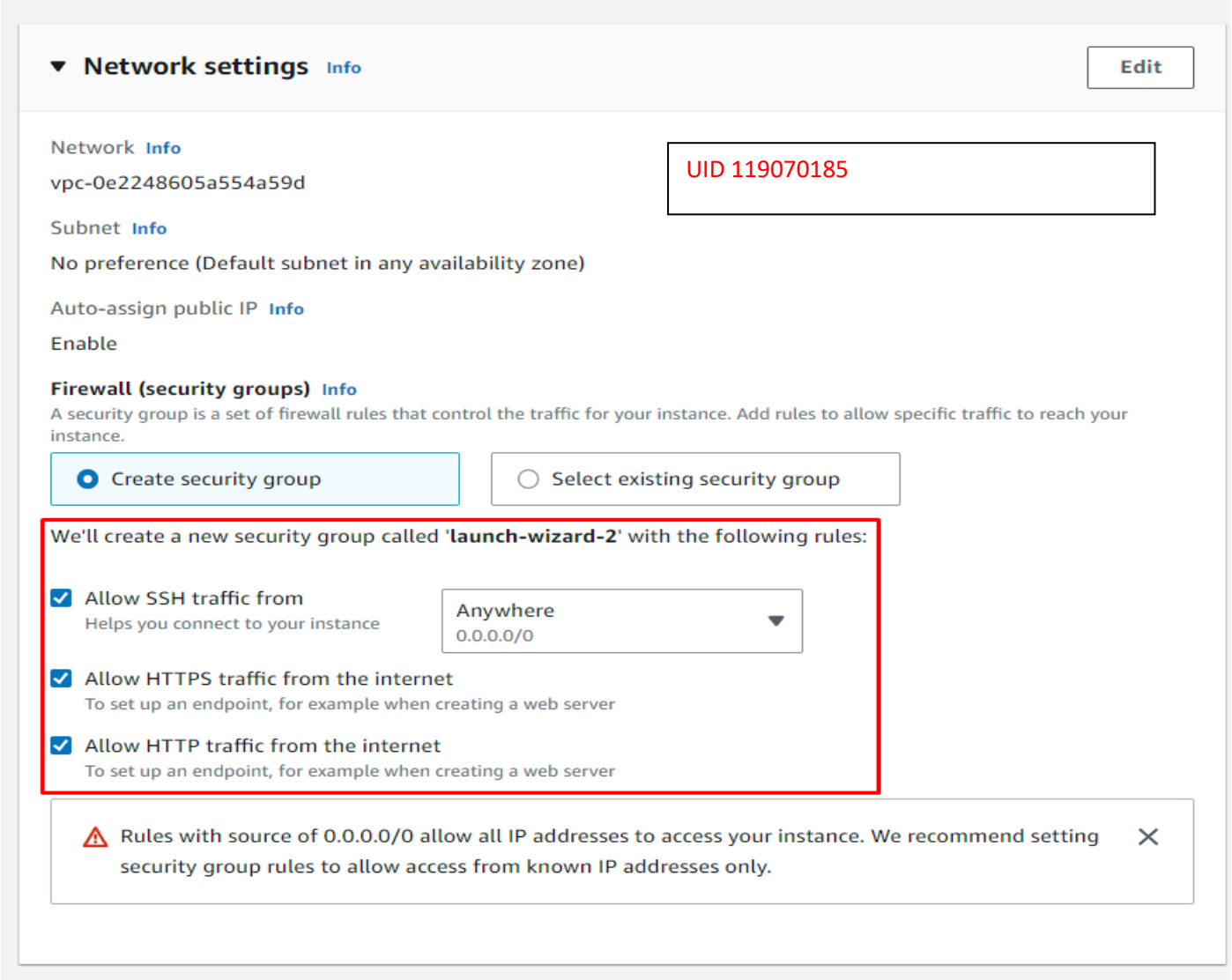
Hosting CobraKai website using EC2:

Amazon EC2 is a virtual machine service that provides resizable computing capacity in the cloud. It enables us to deploy virtual machines. We use EC2 instances to host web applications, run database servers, and process data. In the services tab of the AWS management console, go to services --> EC2 to access the EC2 management console.





Launch an instance with 64-bit Ubuntu and name it Cloudsec-Final-119070185. Create a new key pair for the EC2 instance and save it as a .csv file to access the AccesskeyID and SecretAccessKey.



In the Security groups section, Select Allow SSH traffic from anywhere to connect to the VM using public IP address and allow HTTP and HTTPS traffic to access the website using the domain name.

```
README.md  __pycache__  app  aws-config  enpm809j.py  enpm809j.sh  venv
ubuntu@ip-172-31-91-91:~/enpm809j$ source venv/bin/activate
(venv) ubuntu@ip-172-31-91-91:~/enpm809j$ ./enpm809j.sh
(venv) ubuntu@ip-172-31-91-91:~/enpm809j$  * Serving Flask app 'enpm809j.py'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:5000
 * Running on http://172.31.91.91:5000
Press CTRL+C to quit
^C
(venv) ubuntu@ip-172-31-91-91:~/enpm809j$ gunicorn -b 0.0.0.0:8000 app:app
[2022-12-12 20:17:05 +0000] [1294] [INFO] Starting gunicorn 20.1.0
[2022-12-12 20:17:05 +0000] [1294] [INFO] Listening at: http://0.0.0.0:8000 (1294)
[2022-12-12 20:17:05 +0000] [1294] [INFO] Using worker: sync
[2022-12-12 20:17:05 +0000] [1295] [INFO] Booting worker with pid: 1295
^C[2022-12-12 20:17:08 +0000] [1294] [INFO] Handling signal: int
[2022-12-12 20:17:08 +0000] [1294] [INFO] Shutting down: Master
```

UID 119070185

```
aws  Services  Search [Alt+S]  N. Virginia  tanishq javvaji
ubuntu@ip-172-31-31-69:~$ echo 119070185
119070185
ubuntu@ip-172-31-31-69:~$ sudo apt-get install nginx
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  fontconfig-config fonts-dejavu-core libdeflate0 libfontconfig1 libgd3 libjbig0 libjpeg-turbo8 libjpeg8 libnginx-mod-http-geoip2
  libnginx-mod-http-image-filter libnginx-mod-http-xslt-filter libnginx-mod-mail libnginx-mod-stream libnginx-mod-stream-geoip2 libtiff5
  libwebp7 libxpm4 nginx-common nginx-core
Suggested packages:
  libgd-tools fcgiwrap nginx-doc ssl-cert
The following NEW packages will be installed:
  fontconfig-config fonts-dejavu-core libdeflate0 libfontconfig1 libgd3 libjbig0 libjpeg-turbo8 libjpeg8 libnginx-mod-http-geoip2
  libnginx-mod-http-image-filter libnginx-mod-http-xslt-filter libnginx-mod-mail libnginx-mod-stream libnginx-mod-stream-geoip2 libtiff5
  libwebp7 libxpm4 nginx nginx-common nginx-core
0 upgraded, 20 newly installed, 0 to remove and 17 not upgraded.
Need to get 2688 kB of archives.
After this operation, 8334 kB of additional disk space will be used.
Do you want to continue? [Y/n]
```

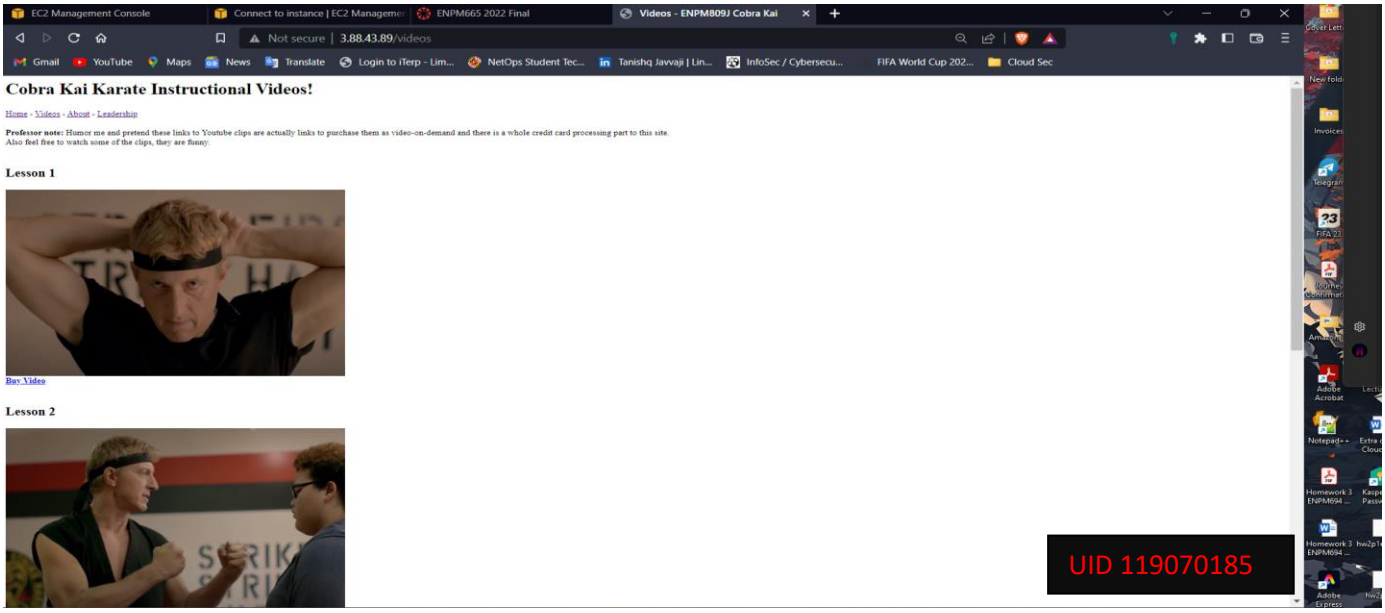
```
(venv) ubuntu@ip-172-31-91-91:~/enpm809j$ sudo nano /etc/systemd/system/helloworld.service
(venv) ubuntu@ip-172-31-91-91:~/enpm809j$ sudo systemctl daemon-reload
sudo systemctl start helloworld
sudo systemctl enable helloworld
Created symlink /etc/systemd/system/multi-user.target.wants/helloworld.service → /etc/systemd/system/helloworld.service.
(venv) ubuntu@ip-172-31-91-91:~/enpm809j$ curl localhost:8000
<html>
<head>
<title>Home - ENPM809J Cobra Kai</title>
</head>
<body>
<center><img src="static/logo.jpeg"></center>
<center>
<a href=/>Home</a> -
<a href=/videos>Videos</a> -
<a href=/about>About</a> -
<a href=/leadership>Leadership</a>
</center>
<br><br>
<center><h2>STRIKE FIRST - STRIKE HARD - NO MERCY</h2></center>
<center><img src="static/cobra-kai-william-zabka-760x380.jpg" width=760 height=380></center>
<br><br>
This the official site of Cobra Kai, a karate dojo in Reseda, Los Angeles, California!  Please check out our <a href=/videos>video</a> collections and purchase as many videos as you think make sense!
<br><br>
Thank you for your patience while we work through these IT issues, I'm not a nerd. -- Sensei Johnny Lawrence
```

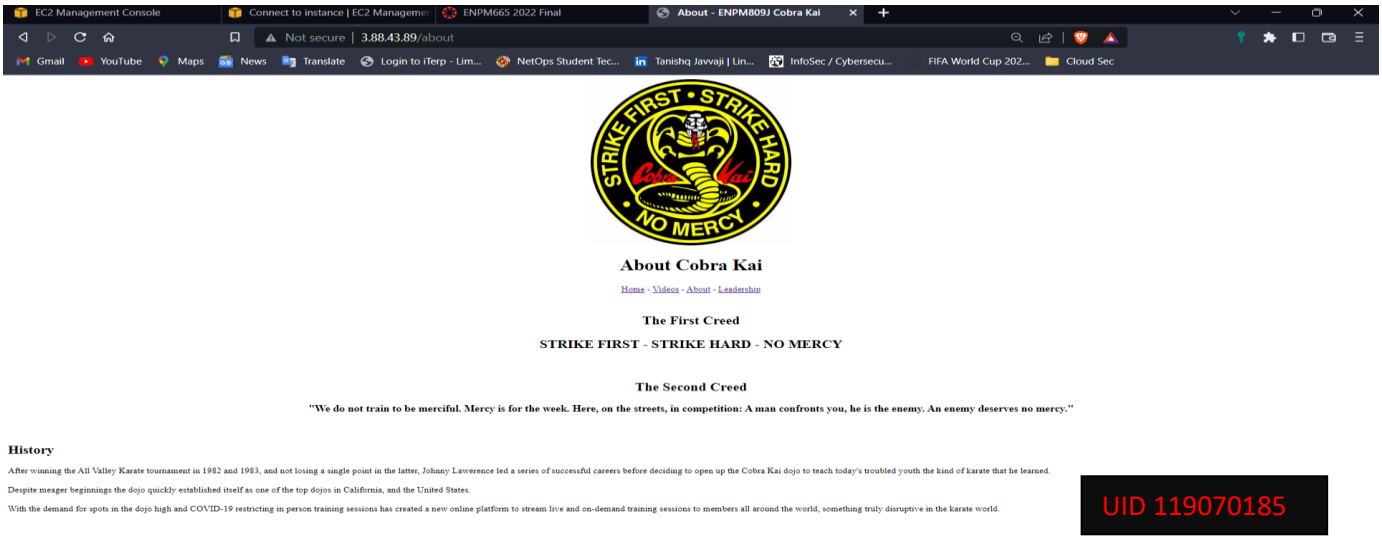
UID 119070185

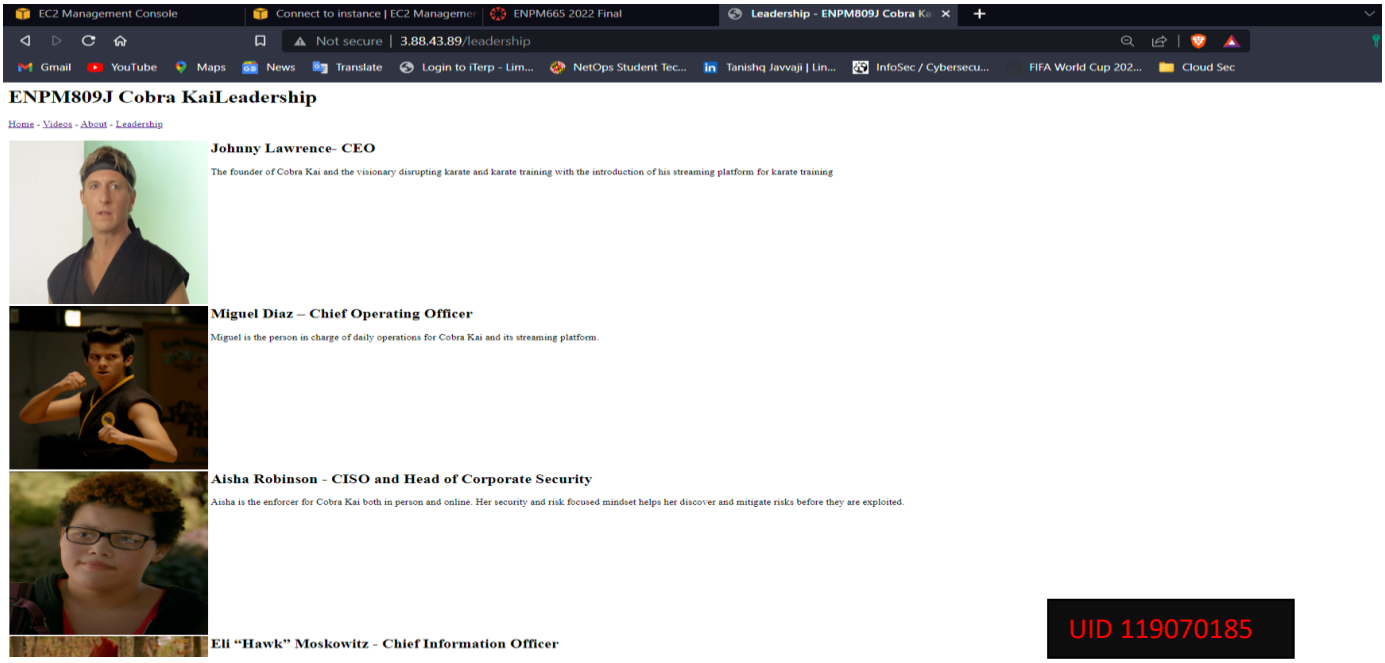
```
EC2 Management Console | Connect to instance | EC2 Manageme | EC2 Instance Connect | ENPM665 2022 Final | Home - ENPM809J Cobra Kai
us-east-1.console.aws.amazon.com/ec2-instance-connect/ssh?region=us-east-1&
Gmail  YouTube  Maps  News  Translate  Login to iTerp - Lim...  NetOps Student Tec...  Tanishq Javvaji | Lin...  InfoSec / Cybersecu...  FIFA World Cup 202...  Cloud Sec
aws  Services  Search [Alt+S]
  GNU nano 6.2                                                    /etc/systemd/syste
[Unit]
Description=Gunicorn instance for a simple hello world app
After=network.target
[Service]
User=ubuntu
Group=www-data
WorkingDirectory=/home/ubuntu/enpm809j
ExecStart=/home/ubuntu/enpm809j/venv/bin/gunicorn -b localhost:8000 app:app
Restart=always
[Install]
WantedBy=multi-user.target
```

UID 119070185

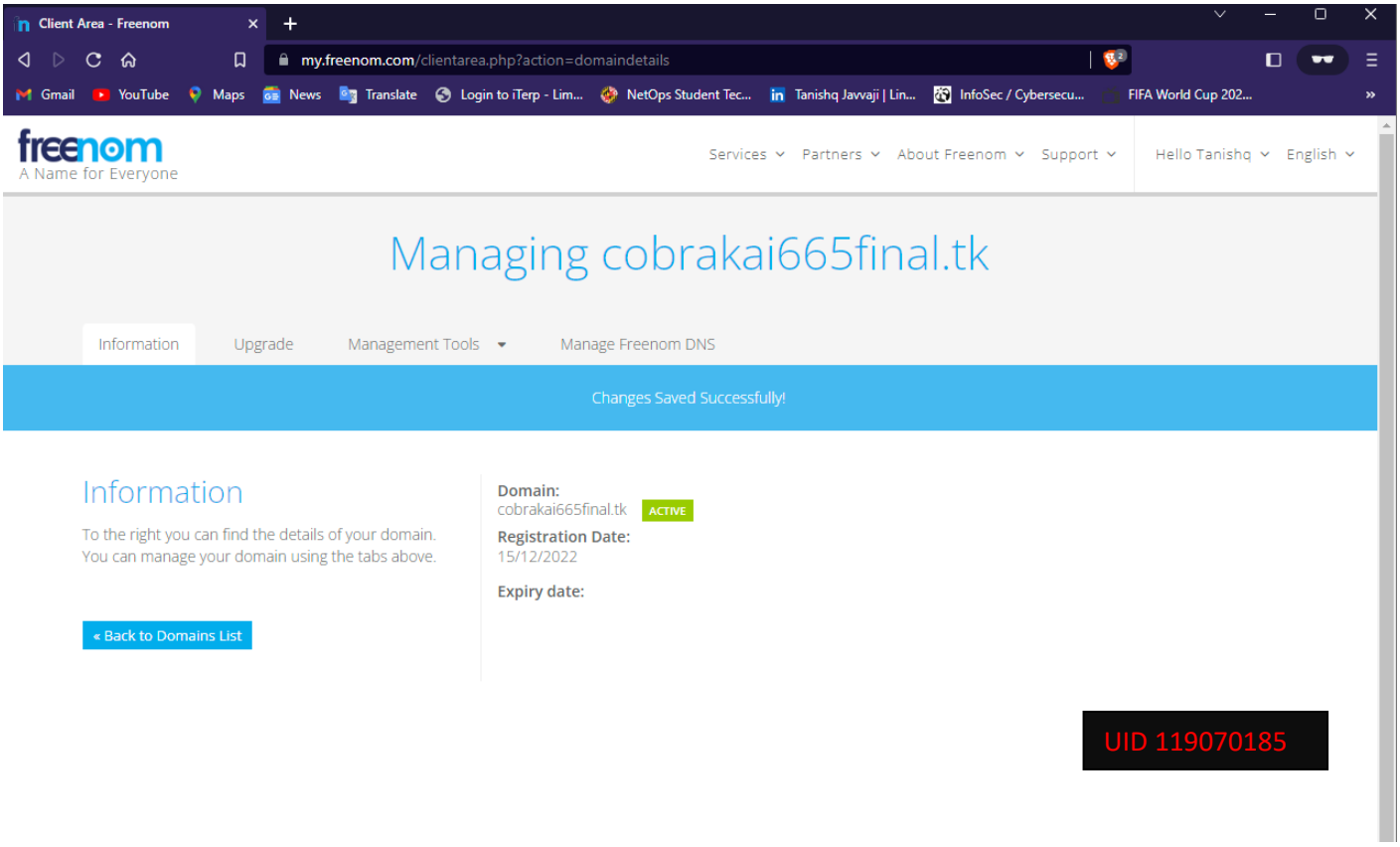
Configure the gunicorn server and use Nginx to point it to our gunicorn server, as shown in the above images, to host the website.


EC2 Management Console
Connect to instance | EC2 Managemer
ENPM665 2022 Final
Videos - ENPM809J Cobra Kai
Not secure | 3.88.43.89/videos
Gmail YouTube Maps News Translate Login to iTerp - Lim... NetOps Student Tec... Tanishq Javvaji | Lin... InfoSec / Cybersecu... FIFA World Cup 202... Cloud Sec
Cobra Kai Karate Instructional Videos!
Home - Videos - About - Leadership
Professor note: Humor me and pretend these links to Youtube clips are actually links to purchase them as video-on-demand and there is a whole credit card processing part to this site.
Also feel free to watch some of the clips, they are funny.
Lesson 1
Buy Video
Lesson 2
UID 119070185


EC2 Management Console
Connect to instance | EC2 Managemer
ENPM665 2022 Final
About - ENPM809J Cobra Kai
Not secure | 3.88.43.89/about
Gmail YouTube Maps News Translate Login to iTerp - Lim... NetOps Student Tec... Tanishq Javvaji | Lin... InfoSec / Cybersecu... FIFA World Cup 202... Cloud Sec
STRIKE FIRST • STRIKE HARD
NO MERCY
About Cobra Kai
Home - Videos - About - Leadership
The First Creed
STRIKE FIRST - STRIKE HARD - NO MERCY
The Second Creed
"We do not train to be merciful. Mercy is for the week. Here, on the streets, in competition: A man confronts you, he is the enemy. An enemy deserves no mercy."
History
After winning the All Valley Karate tournament in 1982 and 1983, and not losing a single point in the latter, Johnny Lawerence led a series of successful careers before deciding to open up the Cobra Kai dojo to teach today's troubled youth the kind of karate that he learned.
Despite meager beginnings the dojo quickly established itself as one of the top dojos in California, and the United States.
With the demand for spots in the dojo high and COVID-19 restricting in person training sessions has created a new online platform to stream live and on-demand training sessions to members all around the world, something truly disruptive in the karate world.
UID 119070185


EC2 Management Console
Connect to instance | EC2 Managemer
ENPM665 2022 Final
Leadership - ENPM809J Cobra Ka
Not secure | 3.88.43.89/leadership
Gmail YouTube Maps News Translate Login to iTerp - Lim... NetOps Student Tec... Tanishq Javvaji | Lin... InfoSec / Cybersecu... FIFA World Cup 202... Cloud Sec
ENPM809J Cobra KaiLeadership
Home - Videos - About - Leadership
Johnny Lawrence- CEO
The founder of Cobra Kai and the visionary disrupting karate and karate training with the introduction of his streaming platform for karate training
Miguel Diaz – Chief Operating Officer
Miguel is the person in charge of daily operations for Cobra Kai and its streaming platform.
Aisha Robinson - CISO and Head of Corporate Security
Aisha is the enforcer for Cobra Kai both in person and online. Her security and risk focused mindset helps her discover and mitigate risks before they are exploited.
Eli "Hawk" Moskowitz - Chief Information Officer
UID 119070185

In the above images, we can access the website using our public IP, which is not secure.
A domain name masks our IP address. Using route 53 we can reroute the name servers to our domain.



Find free domains using **freenom**, or use amazons' built-in service AWS route 53(paid) to get a domain name as shown in the above figure.
Create an account in **freenom** and activate the domain name. After activating the domain name, go to the management tools and nameservers page and use the name servers we created using route 53 to redirect it.

UID 119070185

Records (4) Info

Automatic mode is the current search behavior optimized for best filter results. To change modes go to settings.

Delete record | Import zone file | Create record

Filter records by property or value | Type | Routing policy | Alias

1

Record name	Type	Routin...	Differ...	Value/Route traffic to
cobrakai665fi...	A	Simple	-	52.201.224.69
cobrakai665fi...	NS	Simple	-	ns-1053.awsdns-03.org. ns-1682.awsdns-18.co.uk ns-772.awsdns-32.net. ns-412.awsdns-51.com.
cobrakai665fi...	SOA	Simple	-	ns-1053.awsdns-03.org. aws.
www.cobraka...	CNAME	Simple	-	cobrakai665final.tk

Now the IP address is masked, and the website is hosted using a domain name.

Not secure | cobrakai665final.tk

UID 119070185

Home - Videos - About - Leadership

STRIKE FIRST - STRIKE HARD - NO MERCY

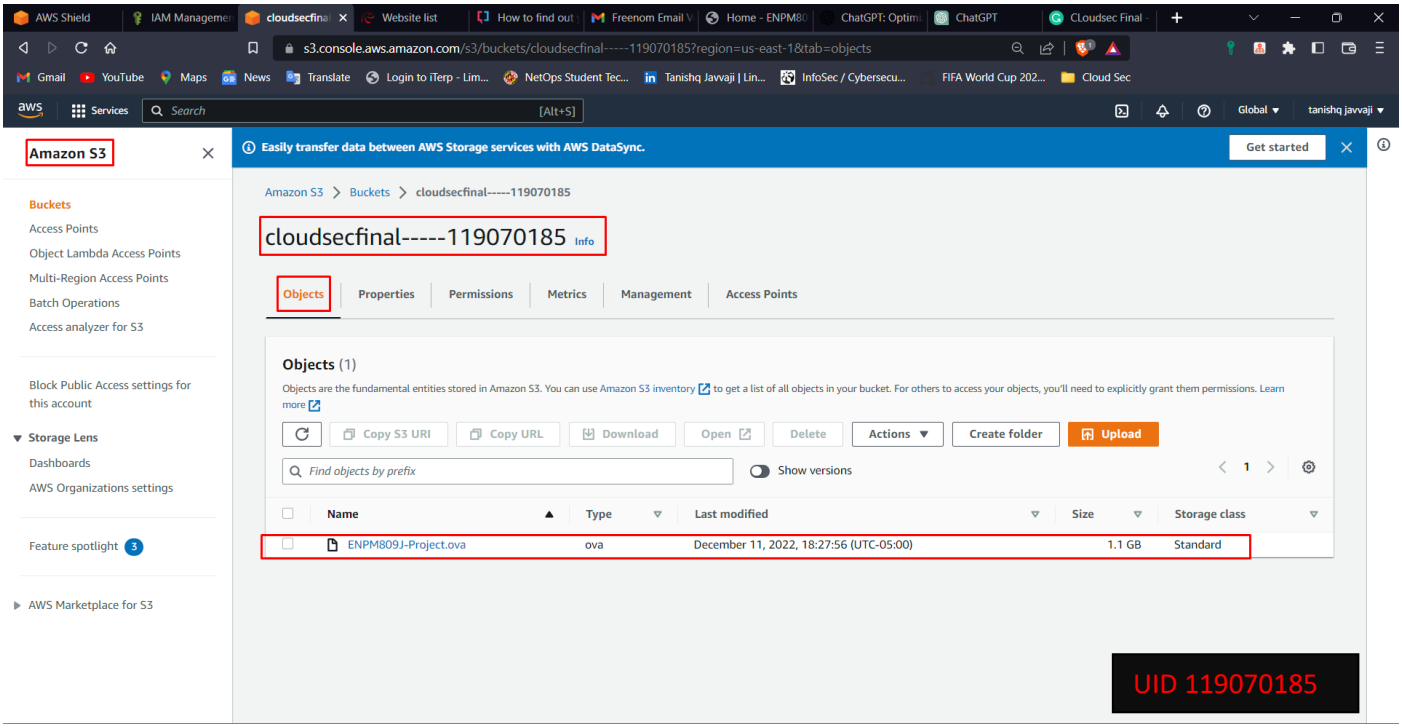
This the official site of Cobra Kai, a karate dojo in Reseda, Los Angeles, California! Please check out our video collections and purchase as many videos as you think make sense!

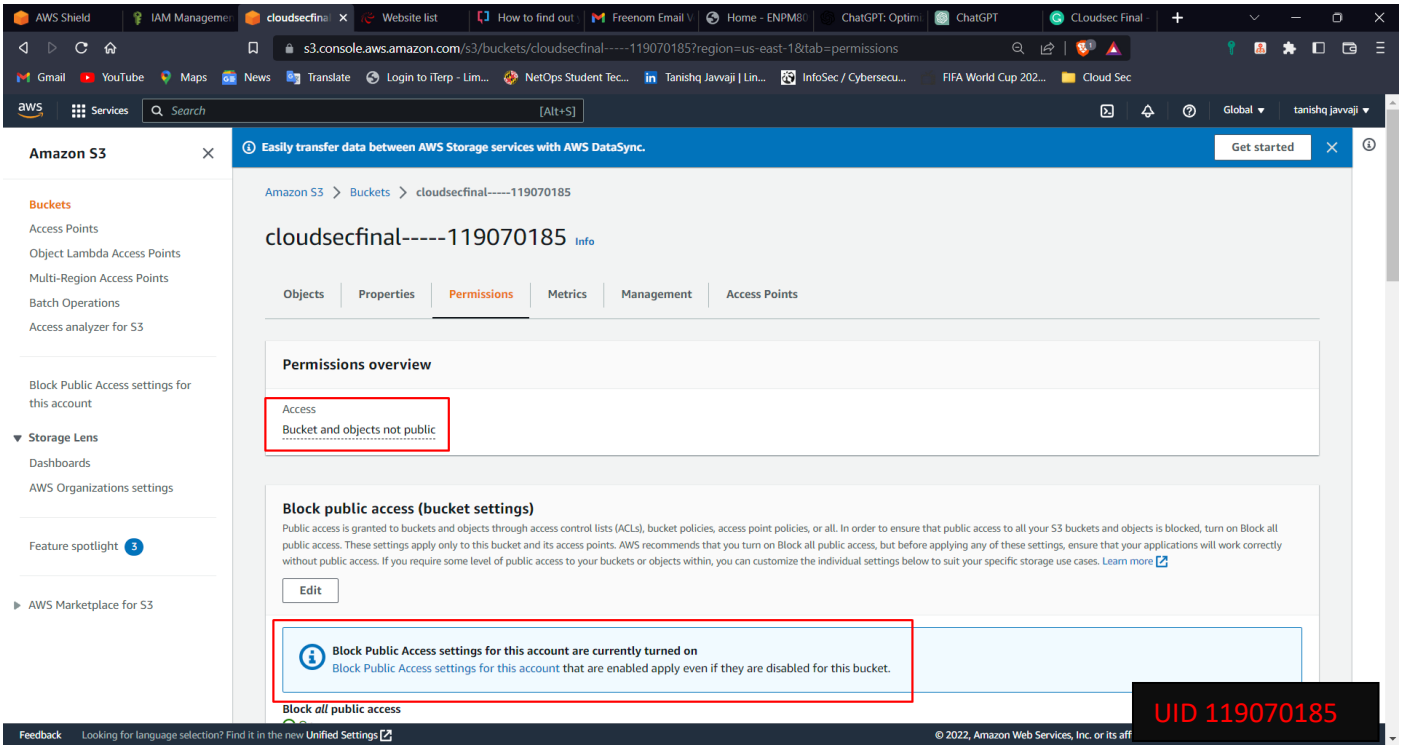
Thank you for your patience while we work through these IT issues, I'm not a nerd. -- Sensei Johnny Lawrence

AWS Cloud Storage: (Backup Strategy)

Amazon S3 or Simple Storage Service is a storage service provided by amazon to store, retrieve, and manage data in the cloud. S3 provides durability, scalability, and security for storing many data types, including photos, videos, documents, and more.
To store data using S3, follow the steps:

1. Create an S3 bucket named as CLoudSecFinal --- 119070185 : To create a bucket, sign into the AWS Management Console → services → S3 dashboard.
2. Upload data: upload the VM containing all the files related to the project or add them directly from the GitHub repository to S3 using the AWS Management Console.
3. Set up access control: S3 provides several options for controlling access to your data, including bucket policies, IAM policies, and pre-signed URLs. Since I am just using the S3 to store the files, I have blocked all public access to the bucket.
4. Configure additional features: S3 offers a variety of features that you can use to enhance the functionality of your storage, such as versioning, object tagging, and lifecycle management.





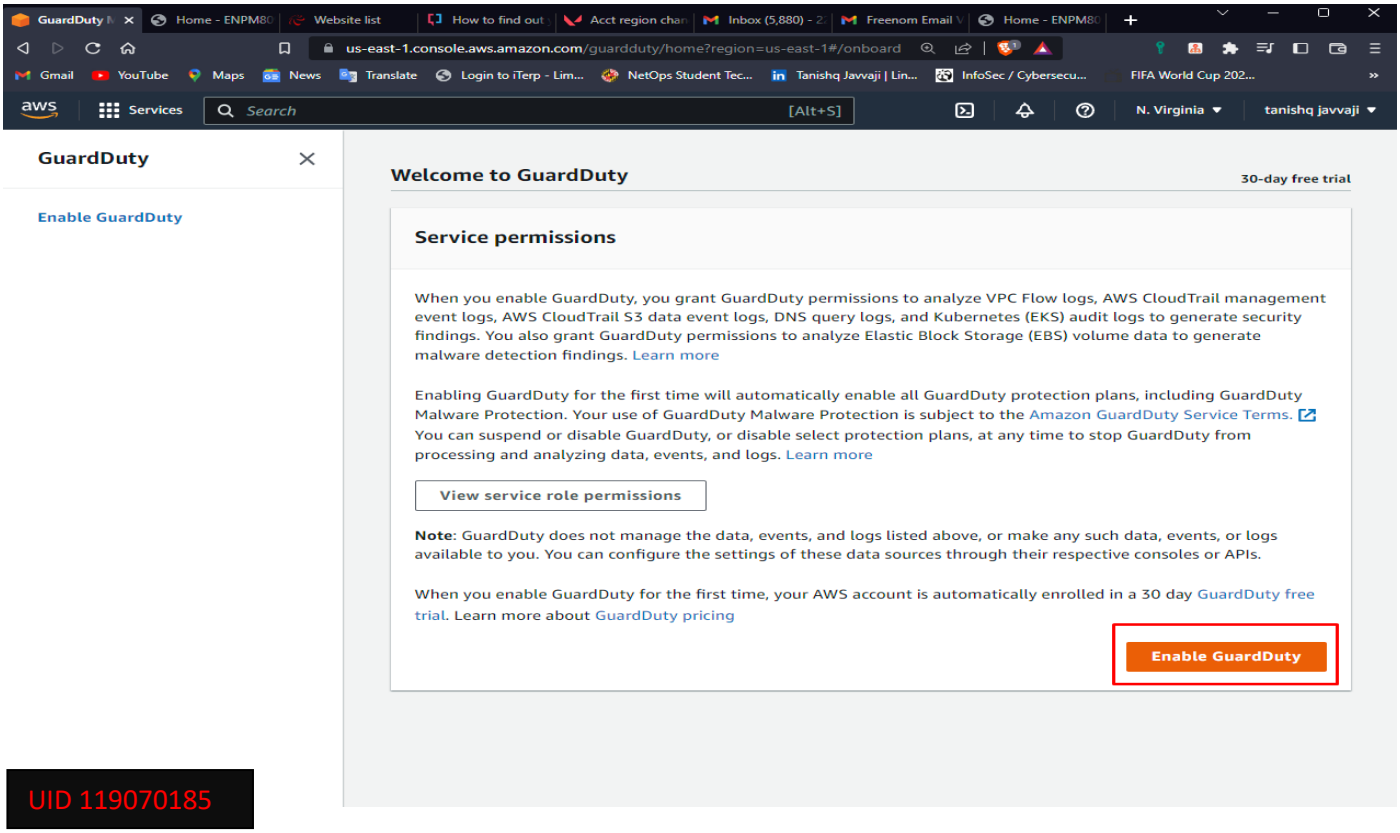
Security

Being a Karate training site that teaches defence strategies against actual physical attacks, Cobra Kai should imply that its online platform is designed to ward against cyberwarriors. Due to the rising number of cybercriminals daily and Daniel LaRusso's current danger, Cobra Kai needs to preserve its name, reputation, and finances to avoid having to shut down its entire website.

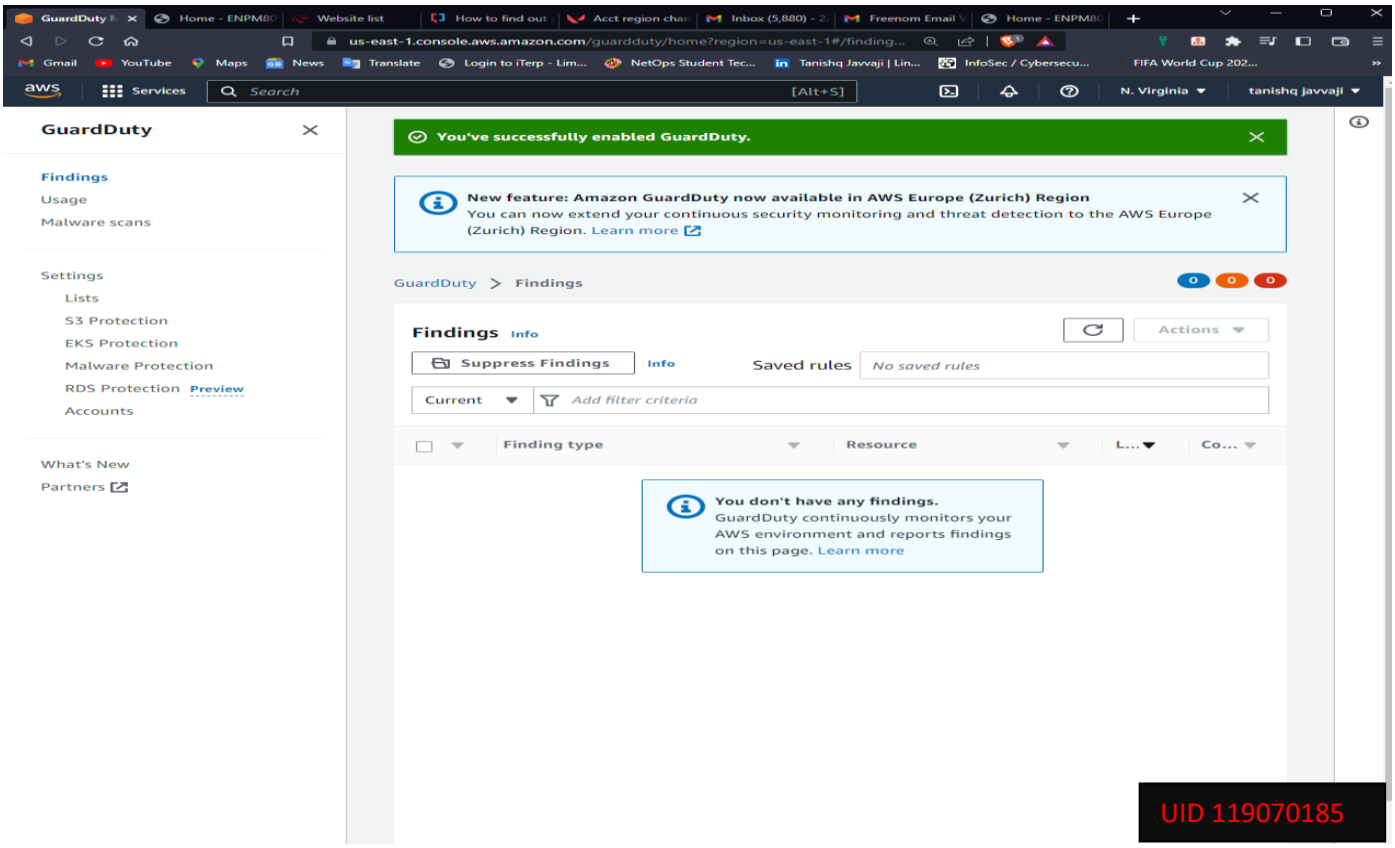
AWS Guardduty

AWS provides a threat detection service called as GuardDuty that uses machine learning and other techniques to identify potential security threats to AWS accounts and resources. The GuardDuty analyses data from AWS VPC Flow Logs, AWS CloudTrail event logs and DNS logs to identify suspicious activity and potential security threats.

Guard Duty categorizes threats into three categories based on the severity of the threats. It can flag malicious IPs from a custom threat list that we can define. Integration with AWS WAF and other security services assures maximum security to the AWS resources and the Web applications. To enable GuardDuty, in the AWS management console → services → GuardDuty and choose “Enable GuardDuty” to enable guard duty to our resources.



UID 119070185



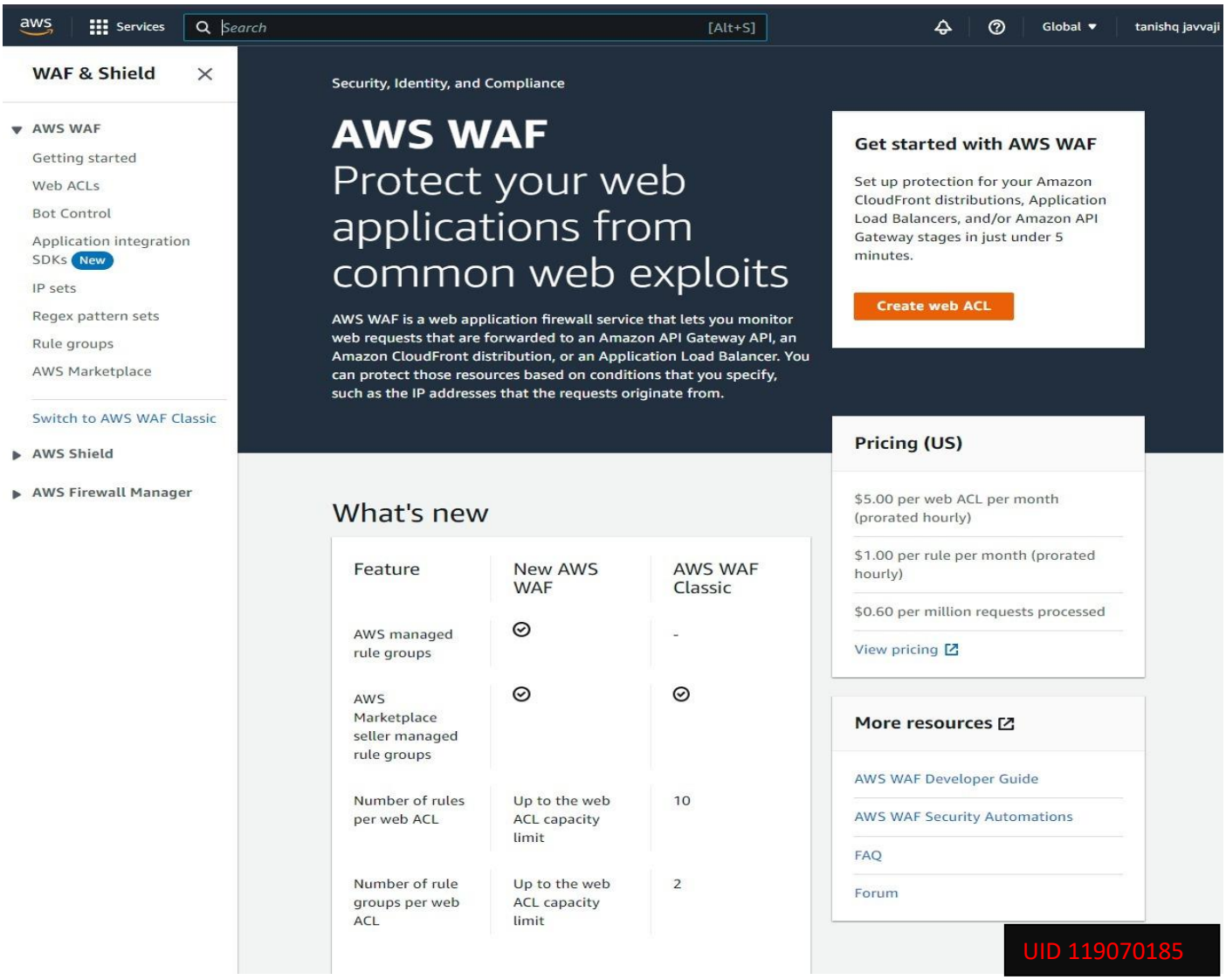
UID 119070185

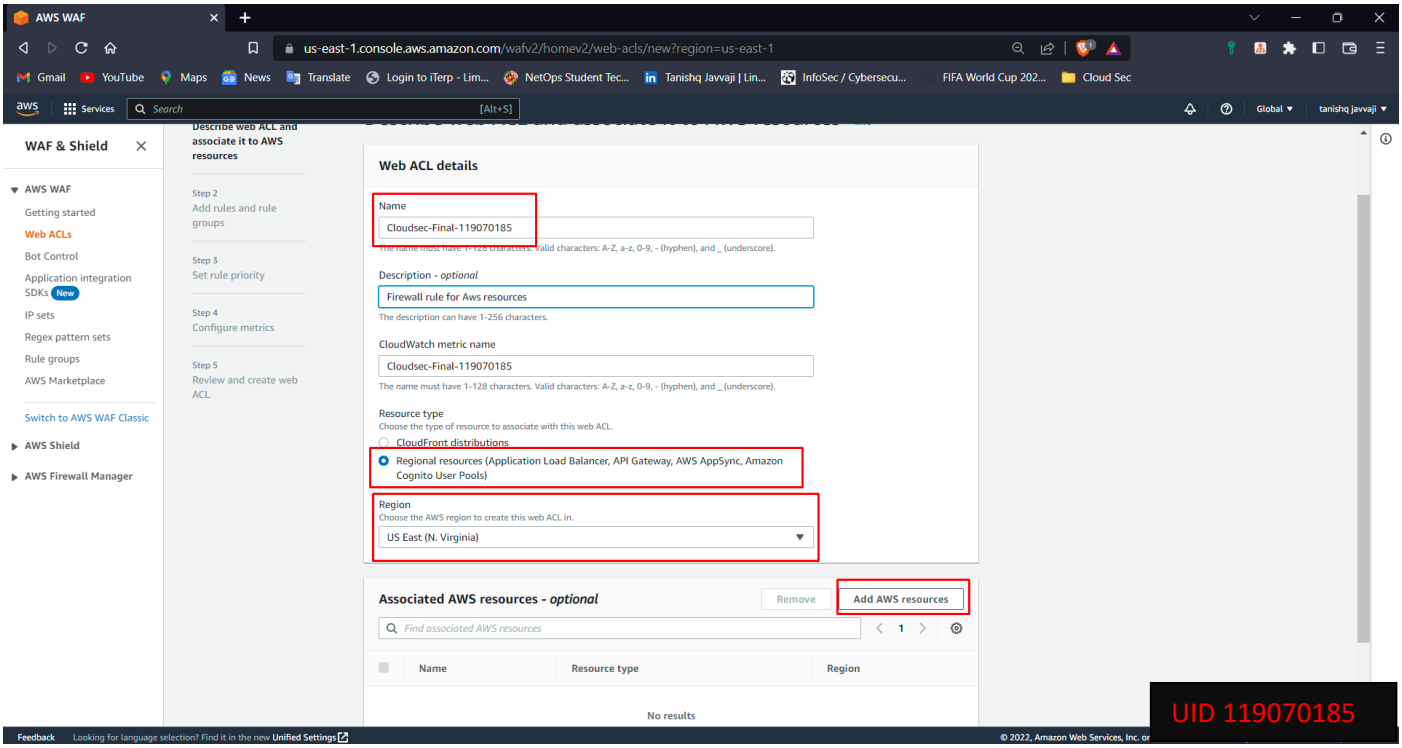
AWS WAF

AWS WAF, also known as web application firewall, is a service provided by Amazon that helps protect web applications from familiar web vulnerabilities that could reduce the availability of an application, jeopardize security, or use excessive resources.

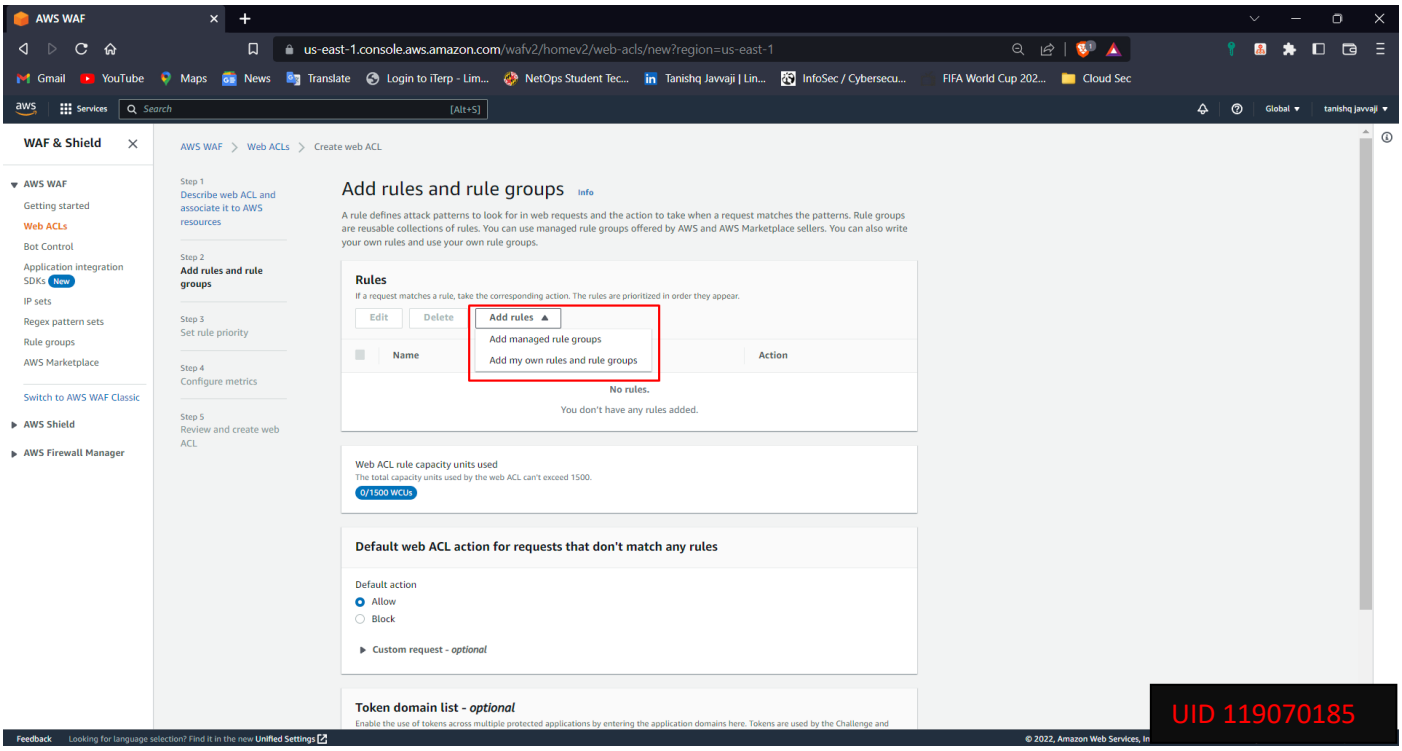
AWS WAF helps create rules that block or allow traffic to your web application based on specified conditions, such as IP address and request header. AWS WAF protects web applications from vulnerabilities, such as, cross-site scripting attacks, SQL injection attacks and other types of malicious traffic.

AWS WAF integrates with various security services, such as Amazon CloudFront, the AWS content delivery network (CDN), and Application Load Balancer, allowing you to apply your rules at the edge of the AWS network. This helps protect the web application from attacks before they reach the origin server. To enable WAF, in the AWS management console, →services →WAF and choose "Create Web ACL" to create firewall rules.





In AWS WAF dashboard, create a Web ACL by giving a name to it, choose resource type as “regional resources” and “Add AWS resources” as shown in the above image to successfully create a WAF rule.



Add your own rules or use pre-defined AWS managed rules, we can also configure CloudWatch metrics with WAF.

AWS Firewall Manager:

You can setup and manage firewall rules for all AWS accounts and apps at one place with the help of Firewall Manager, a security management tool. It is part of the AWS Security Hub service and is designed to provide a centralized way to set and enforce firewall rules for AWS accounts and applications.

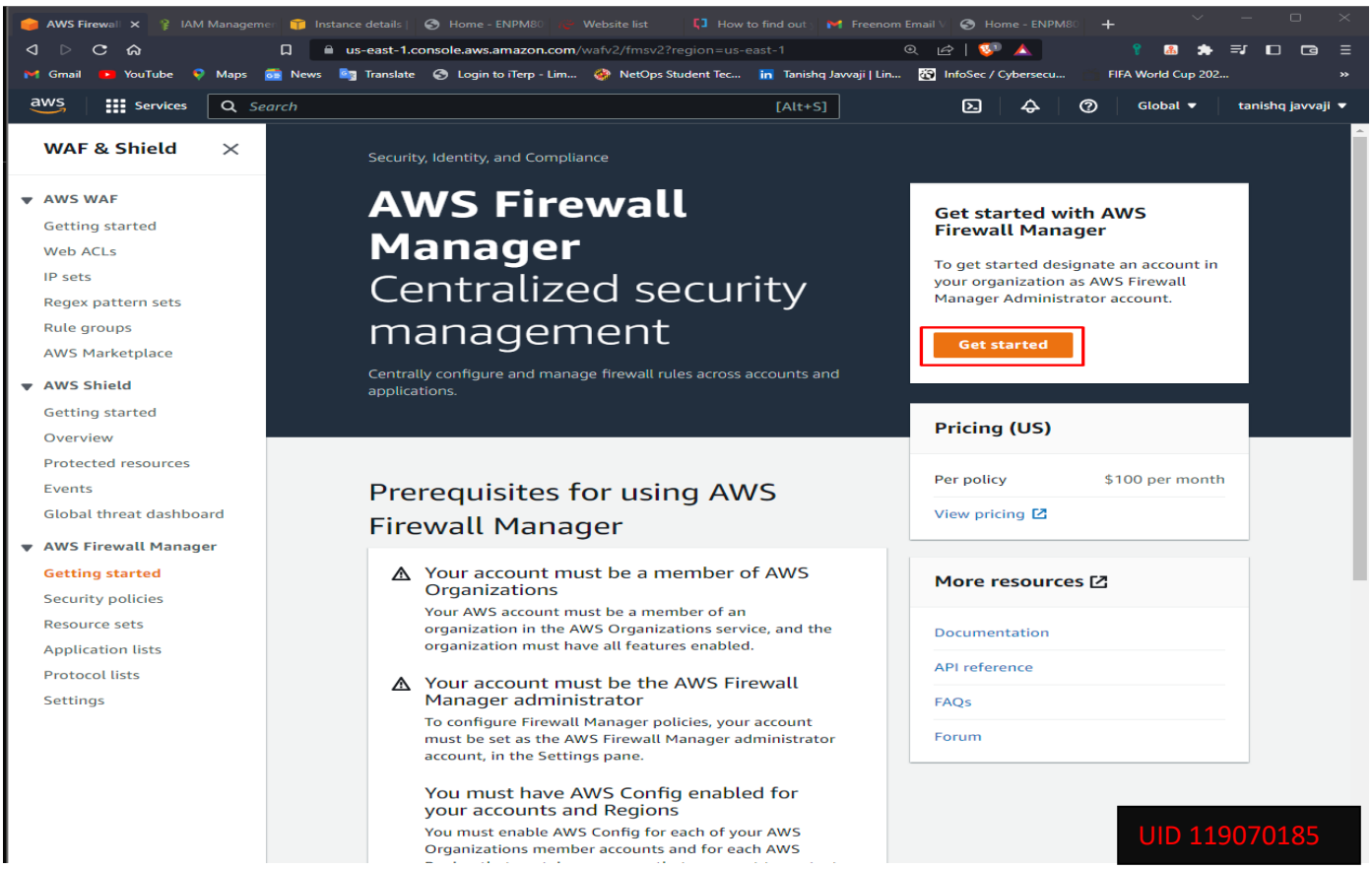
AWS Firewall Manager helps you define security policies for AWS accounts and applications and automatically apply those policies to the accounts and resources. This helps you ensure that the resources are protected and compliant with the Cobrakai's security policies.

AWS Firewall Manager supports both network-level and host-level firewalls, it can be used to in Amazon Elastic Container Service (ECS), Amazon Virtual Private Cloud (VPC), and Amazon Elastic Compute Cloud (EC2) resources to manage firewall rules.

You can use AWS Firewall Manager to:

- Define and enforce firewall rules across all AWS accounts and applications.
- Monitor the compliance of your firewall rules with your security policies.
- Receive alerts when firewall rules are not compliant with your security policies.
- Automatically apply security updates and patches to your firewall rules.

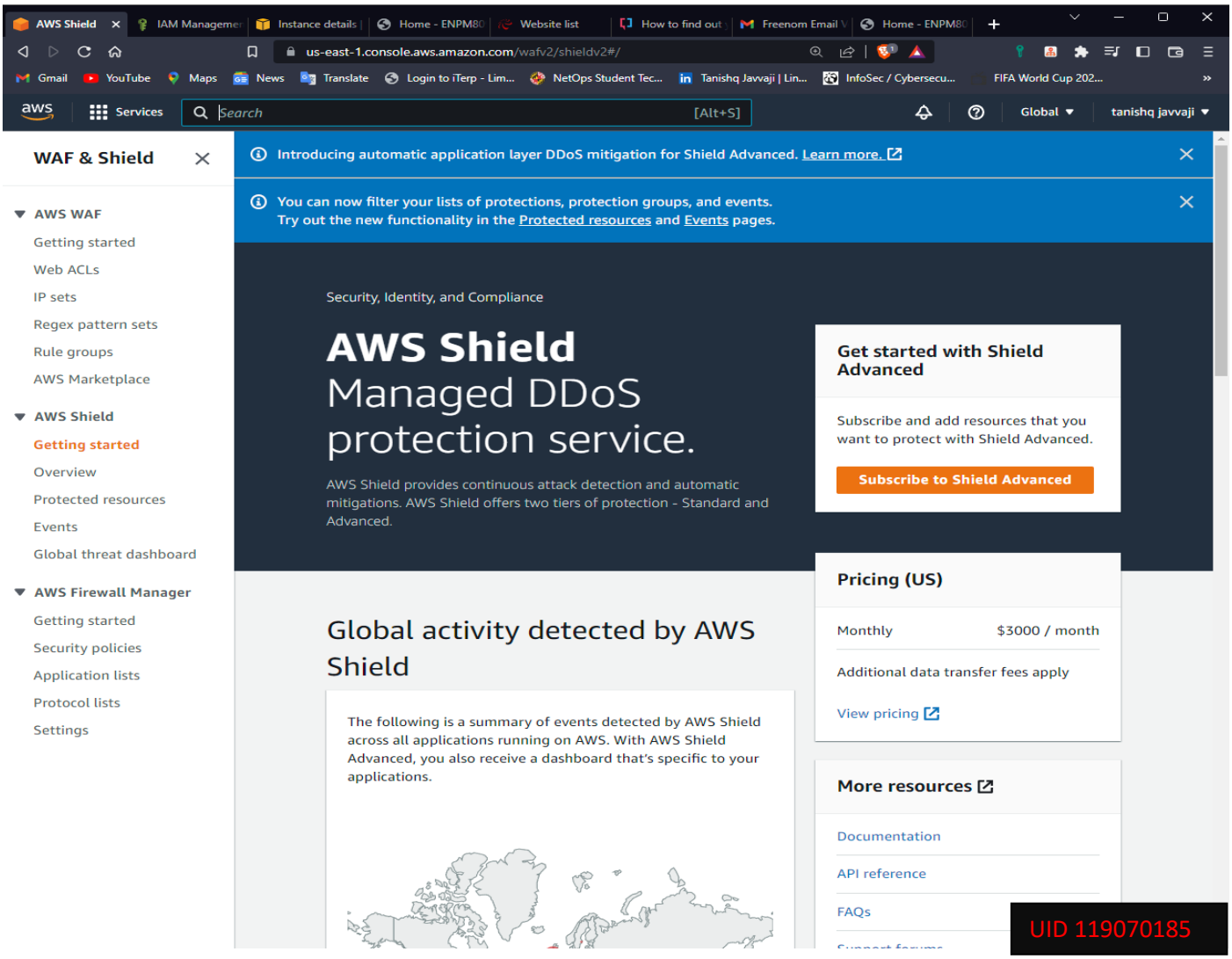
AWS Firewall Manager is a powerful tool for managing and securing your cloud resources, and it can help you ensure that your accounts and applications are protected and compliant with your security policies.



AWS Shield:

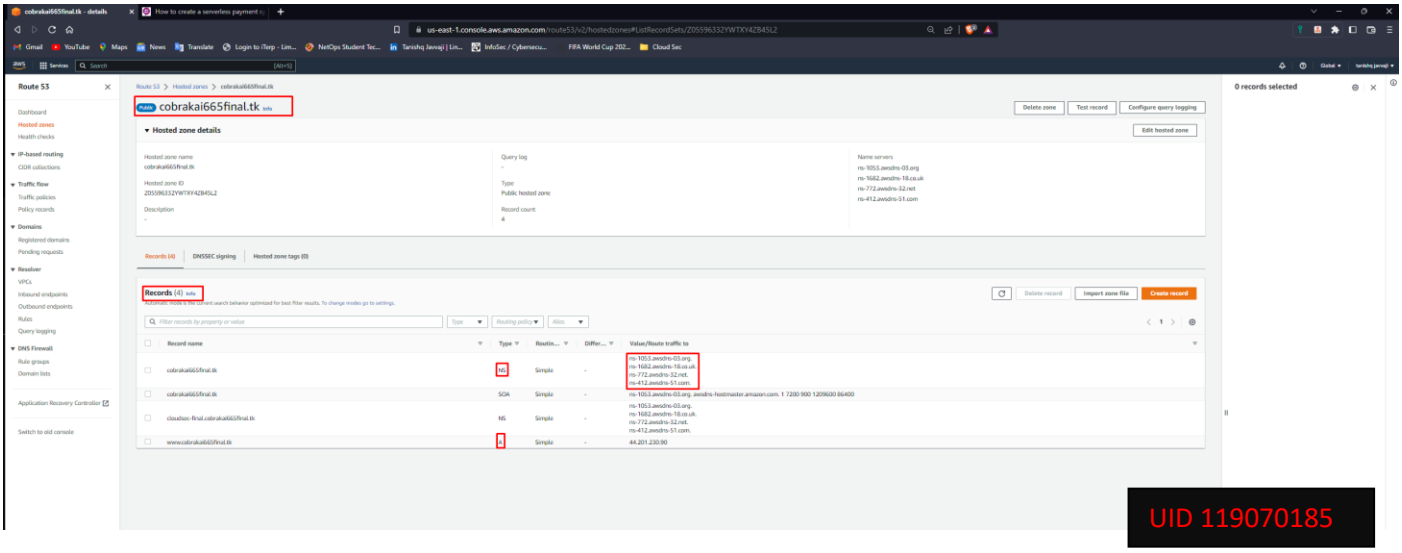
AWS Shield is a default DDoS protection service provided by amazon to protect our instance from common DDoS attacks. For extensive security we can subscribe to Shield Advanced which gives us features like automatic application layer monitoring, custom detection and CloudWatch metrics. Standard version of AWS shield protects from DDoS attacks.

s



Route 53:

Amazon provides a DNS service called as Route 53 to route end users to the internet applications. It translates domain names (cobrakai665final.tk) into the IP addresses computers use to connect (e.g., 192.168.175.1). It integrates with other AWS services, such as Amazon CloudFront, Amazon S3 and Amazon EC2, making it easy to route traffic to AWS resources in this case Amazon EC2. Route 53 offers a variety of features, including support for domain registration, DNS record management, health checks, and traffic routing policies. For Cobrakai application, I rerouted the domain name to the public IP of the website.



Here type NS gives the name servers and type A reroutes all the IPv4 address to the public IP address.

Identity and Access Management

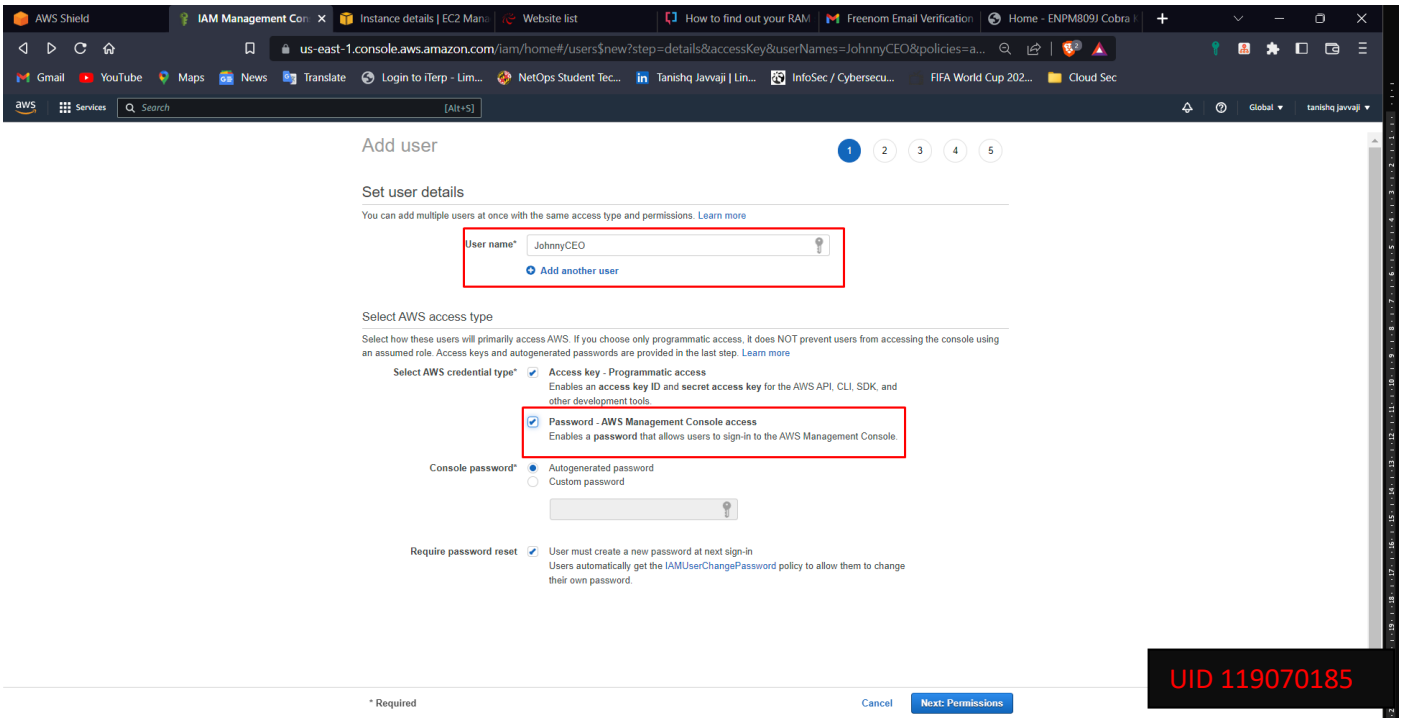
AWS Identity and Access Management helps controlled access to users for AWS resources. We can create and manage users and groups which allows or denies access to AWS resources in accordance with the policies we attach to them. IAM makes sure that the resources are secure and only authorized users can access them.

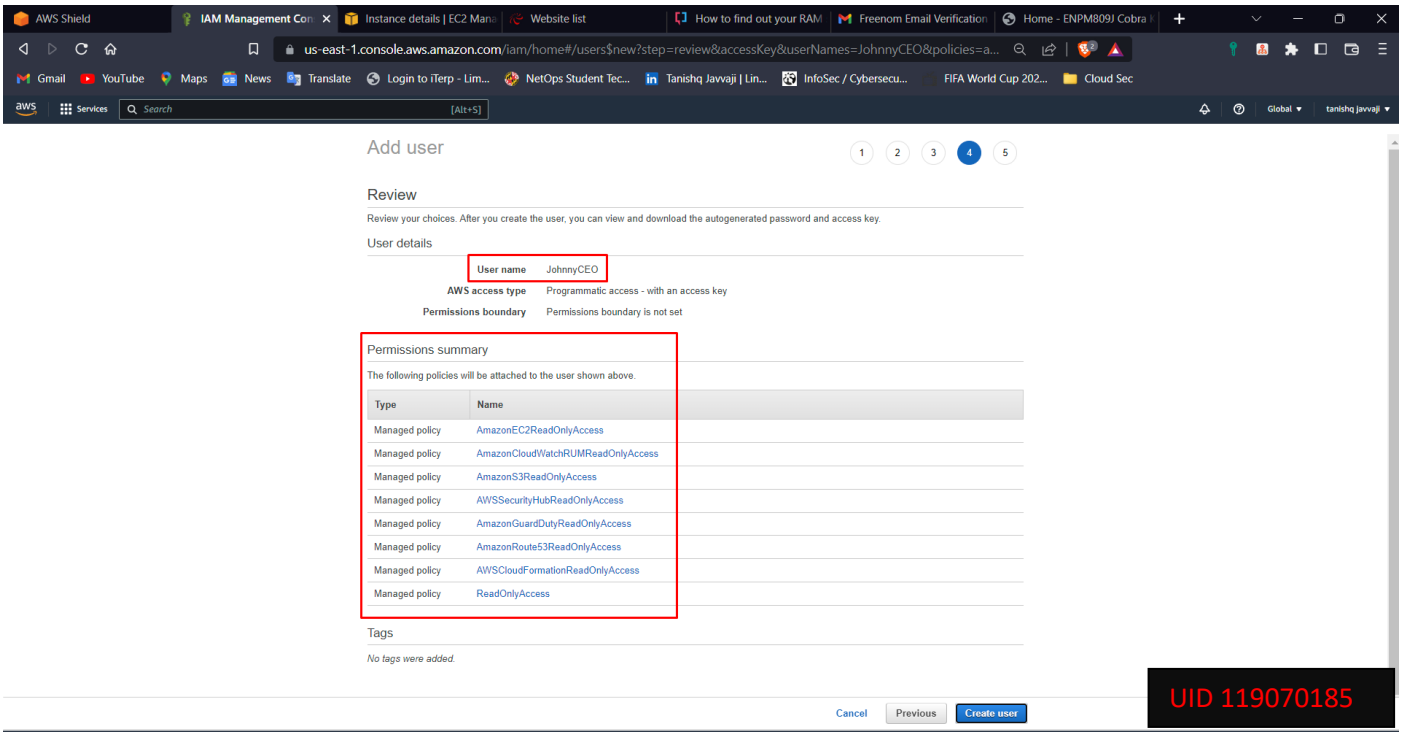
Some key features of IAM include:

- Centralized control of your AWS resources.
- Multiple users can access your AWS resources.
- Identity federation (including support for Microsoft Active Directory).
- Multifactor authentication.
- Identity and Access Management APIs.

Depending on the operations the user performs different policies have been attached for appropriate access to AWS resources.

Johnny Lawrence Chief Executive Officer

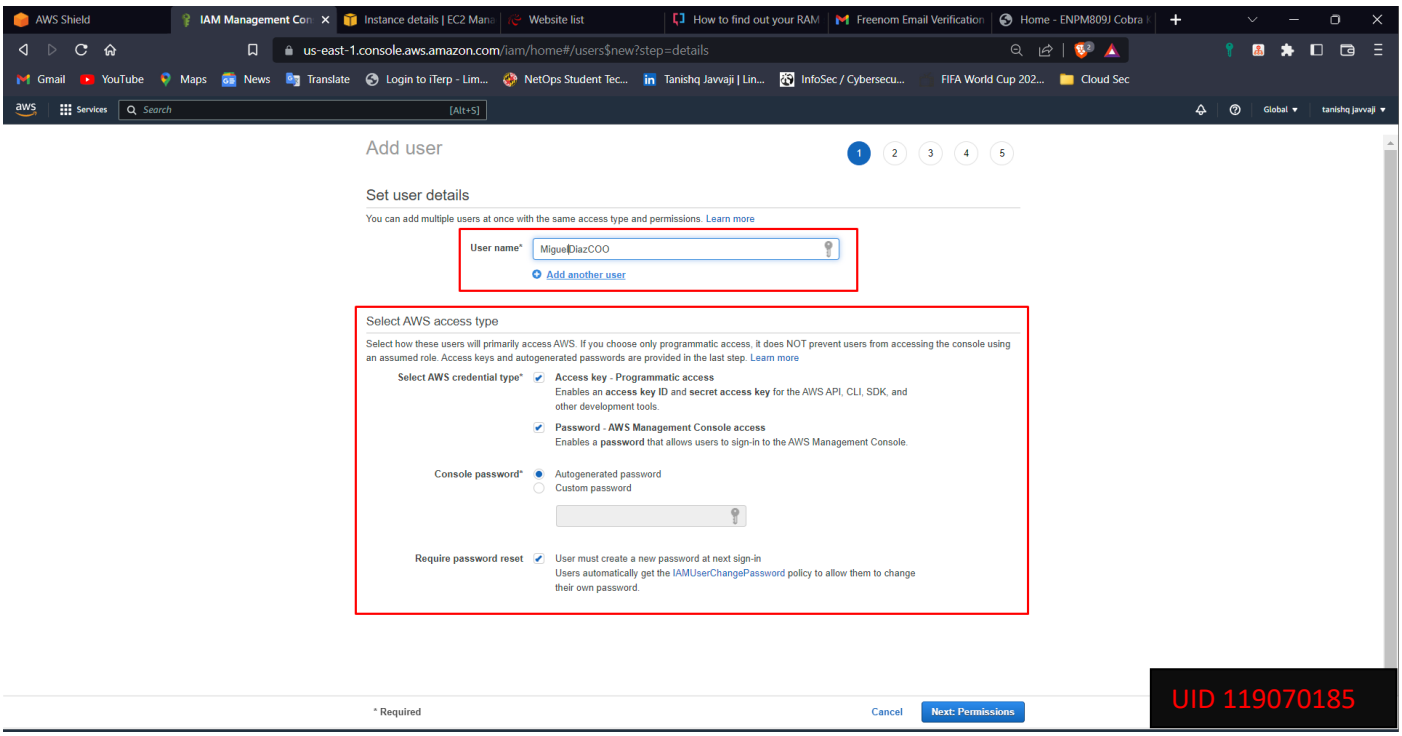


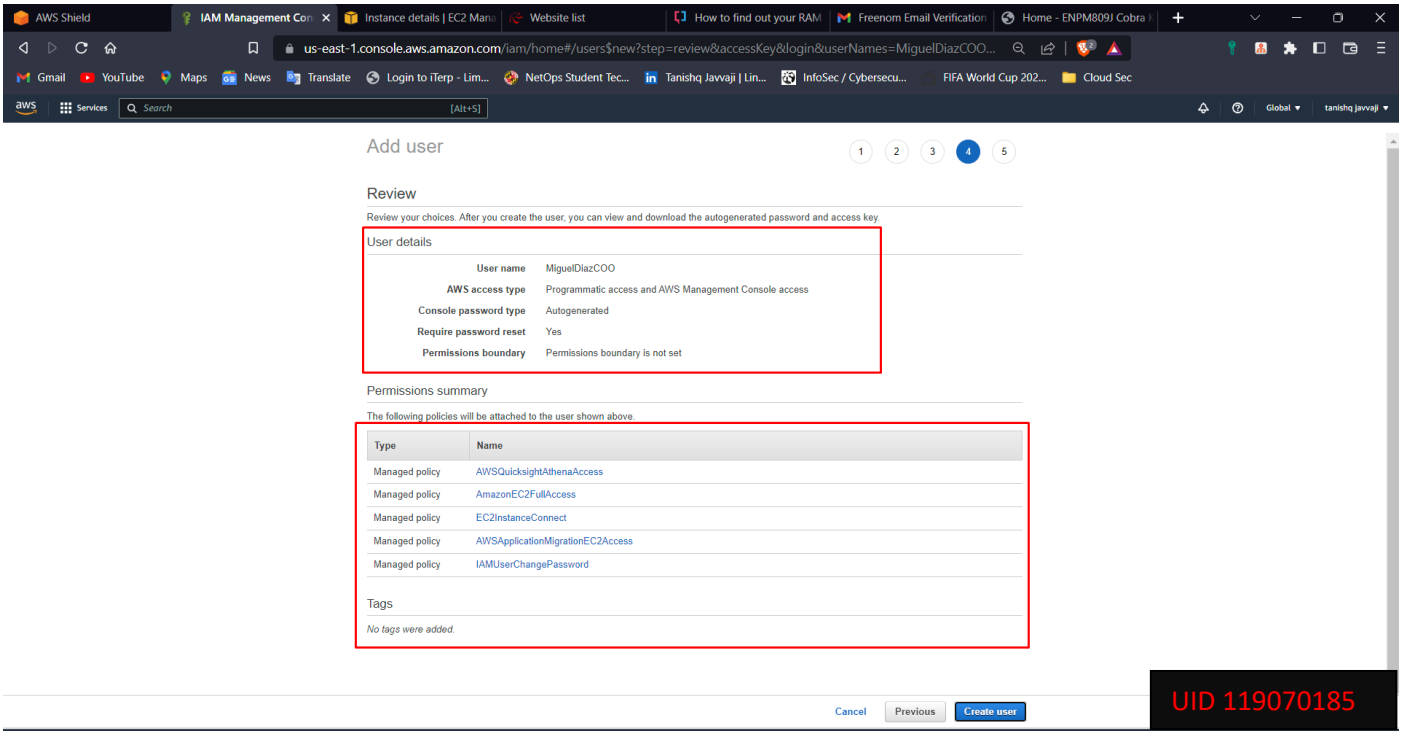


Since Johnny Lawrence is CEO and does not perform any technical access, he has been provided with read-only access to all the AWS resources used for the web application. Policies

attached to the CEO's role include read-only access to Amazon EC2, Amazon CloudWatch, Amazon S3, and Amazon CloudFormation.

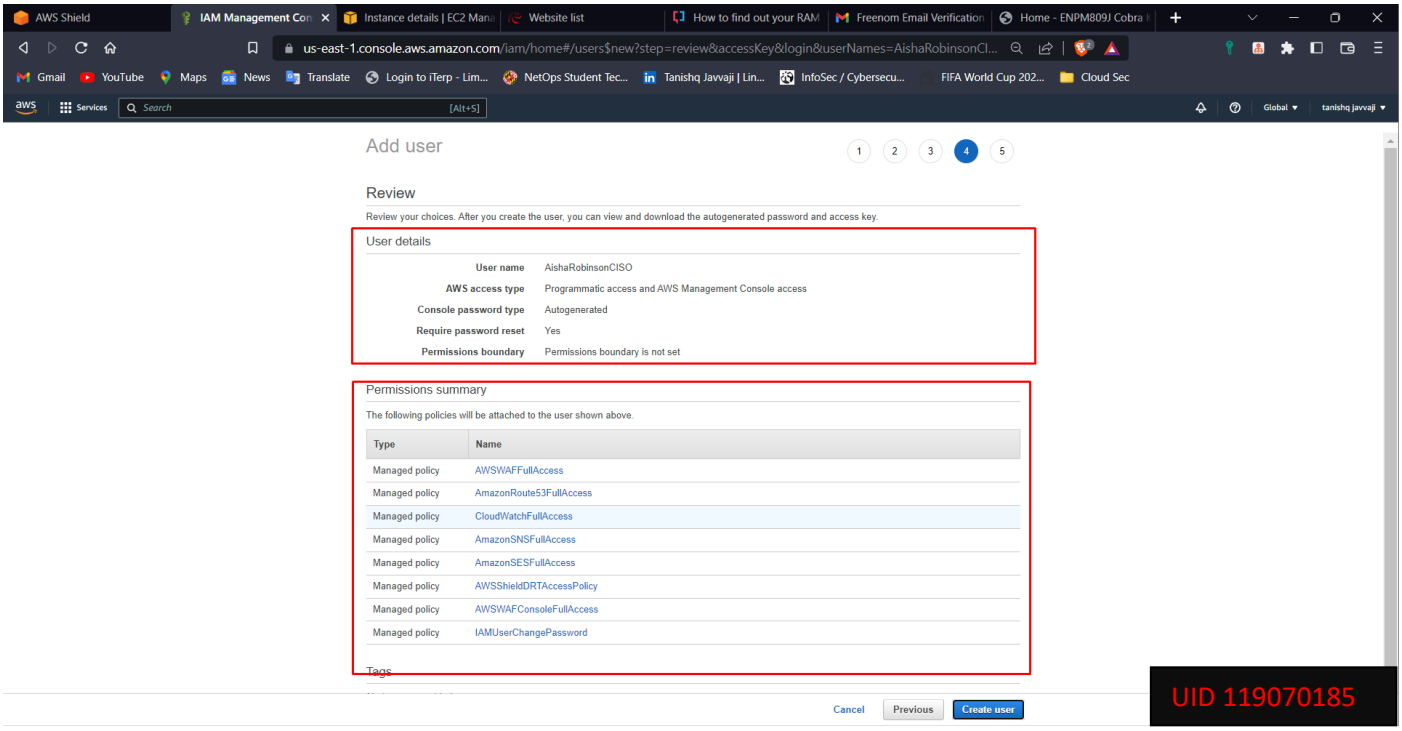
Miguel Diaz Chief Operating Officer





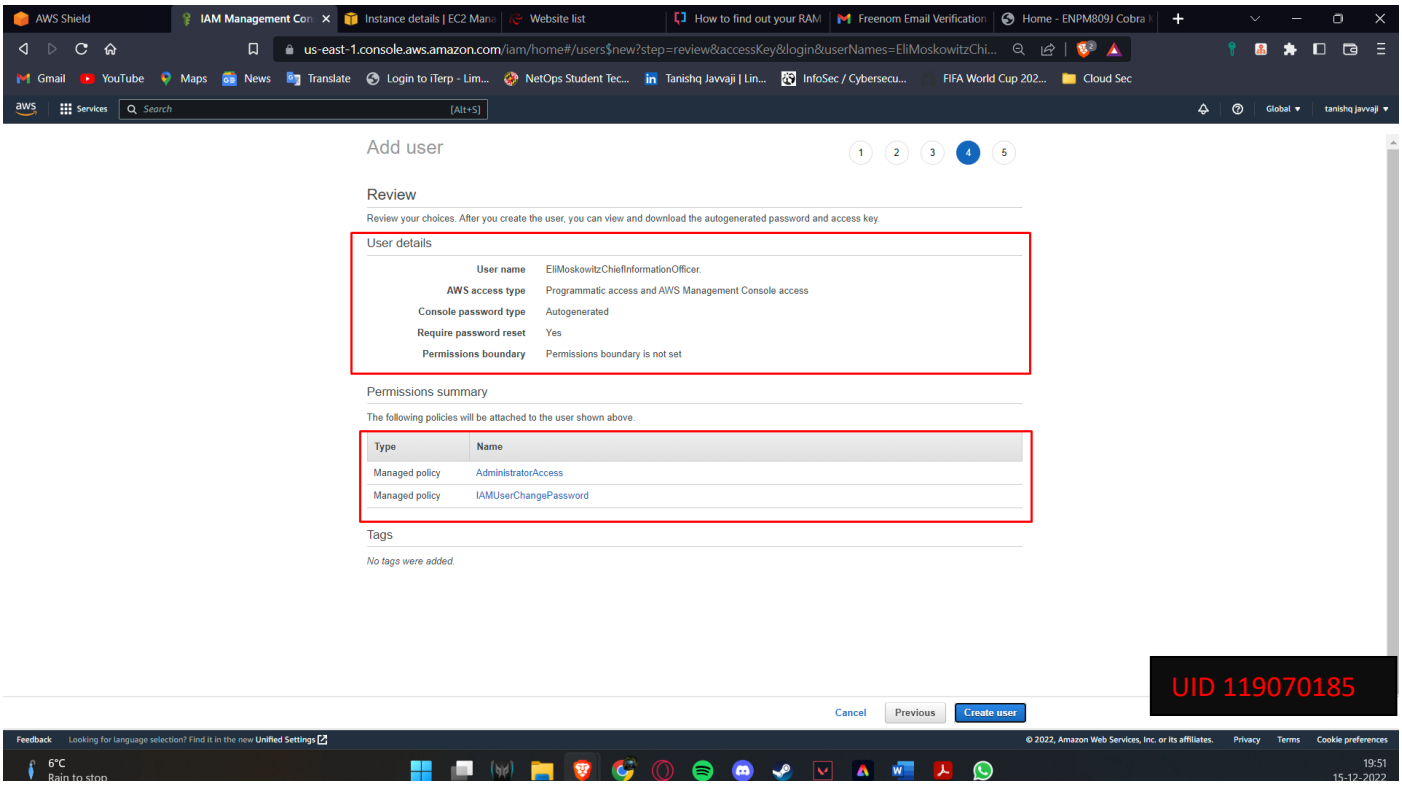
Miguel Diaz has been assigned policies which lets him full access AWS EC2 and AWS Quicksight Athena as he is the head of operations for Cobrakai.

Aisha Robinson - CISO and Head of Corporate Security



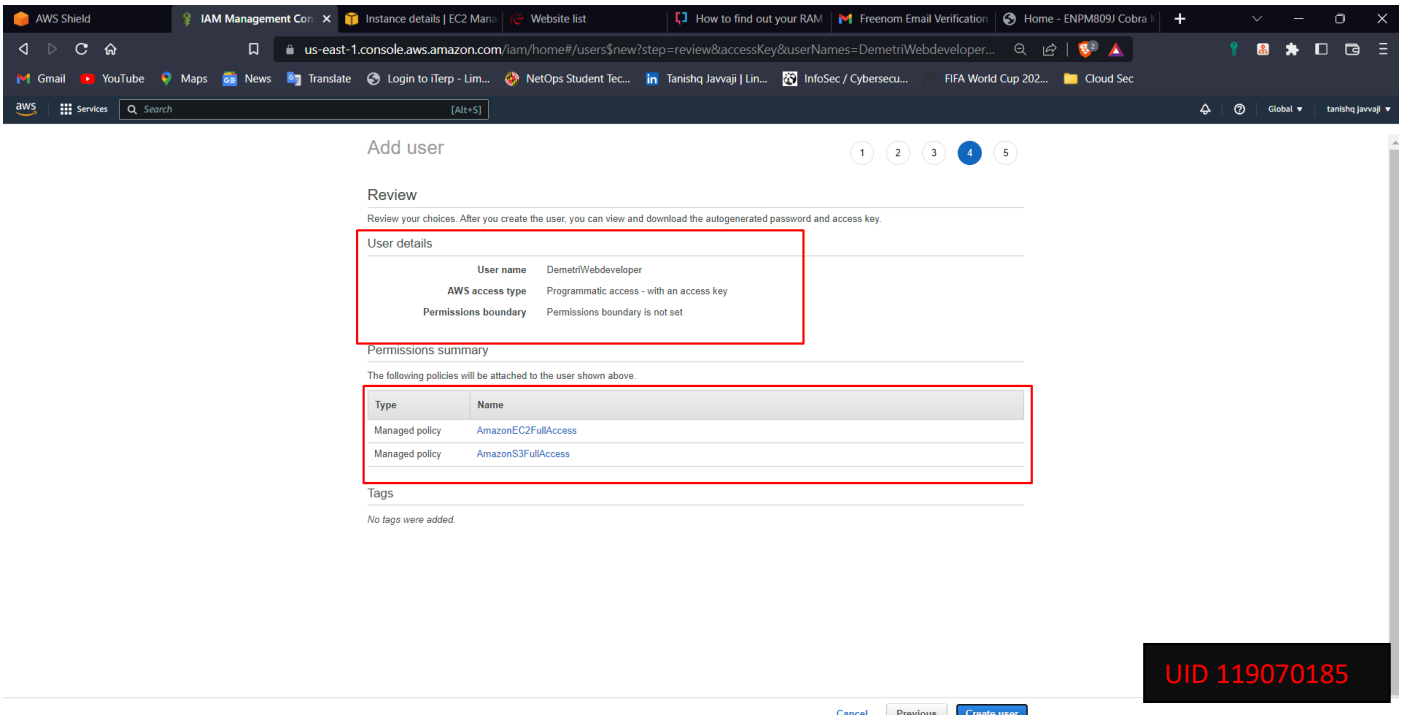
As Aisha is a Chief Information Security Officer, she has been given access to all AWS security services such AWS WAF, CloudWatch, SES, SNS, Route 53 and Aws Shield. Attaching the above policies gives her full access to the AWS security services.

Eli "Hawk" Moskowitz - Chief Information Officer



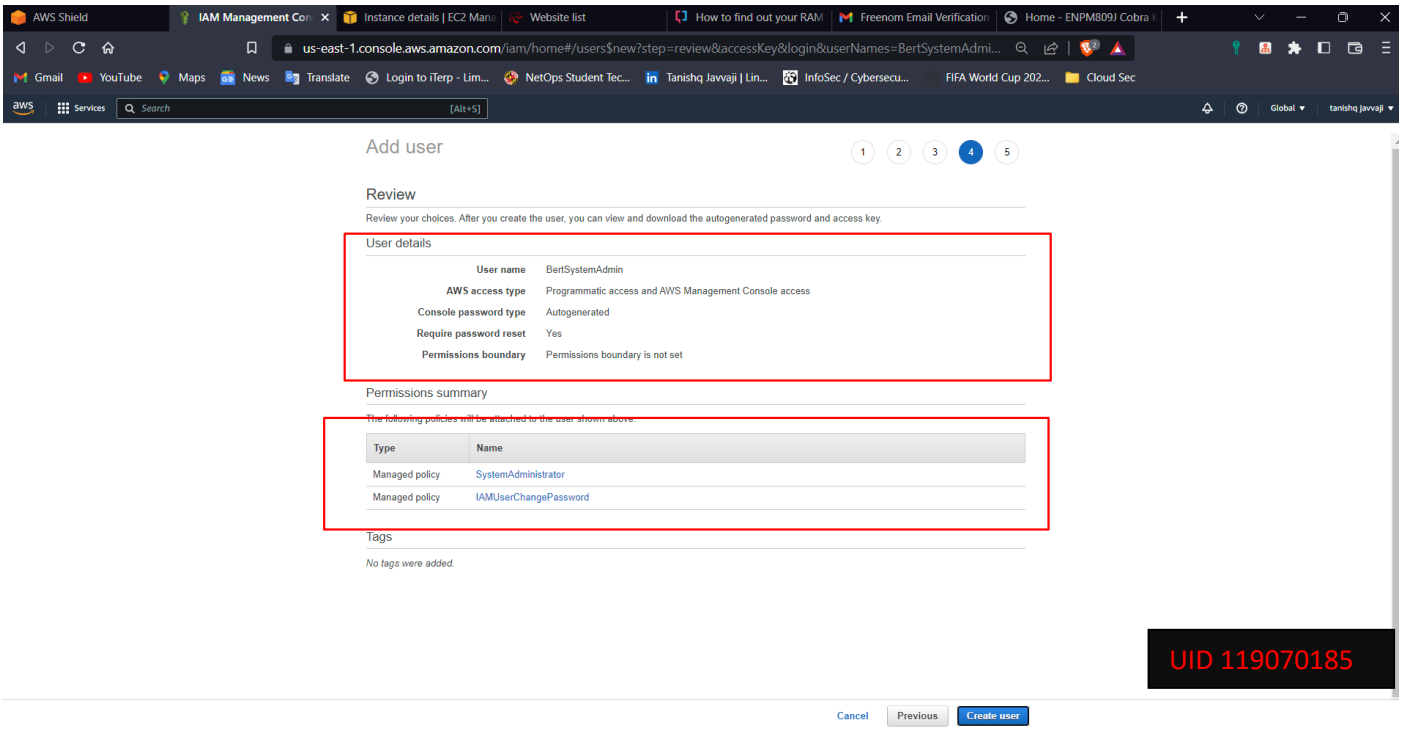
Eli “Hawk” Moskowitz, the chief operating officer has been given complete administrative access.

Demetri - Web Developer



Demetri, the web developer for the application has been given access to EC2 instances and s3 to bucket to develop, update and backup the data in s3.

Bert-system administrator



As a system administrator Bert has access to the group called “System Administrator” which gives him access to all resources except IAM. The policy IAMUserchangePassword lets the user change password for their account after the first initial login.

Payment Processing:

AWS Lambda is a service that helps automate stuff in AWS as well as enables you to run response to specific events, For example, updating contents of Amazon S3 bucket. Lambda can be used to build various applications, including payment processing systems.

To use AWS Lambda for payment processing, follow these steps:

1. Design your payment processing system: Determine the specific steps that need to be taken to process payments and how you want to trigger those steps. For example, you can process payments whenever a customer submits an order on your website or whenever a payment is made through a mobile app.
2. Set up your Lambda function: Use the AWS Management Console or one of the AWS APIs to create a new Lambda function. Choose the programming language you want to use and specify the trigger to invoke your function.
3. Write your code: Write the code for your Lambda function using the programming language you selected. This code should include the logic for processing payments and any other tasks you want to perform in response to the trigger you specified.
4. Test your function: Use the AWS Management Console or one of the AWS APIs to test your Lambda function and ensure it works as expected.
5. Deploy your function: When ready to go live, use the AWS Management Console or one of the AWS APIs to deploy your Lambda function.

By following these steps, We build a payment processing system using Lambda Function that is scalable, cost-effective, and easy to maintain.

Patching Strategy

The teams responsible for application development and operations are the primary consumers of the patching solution. Usually, various environments, including development, test integration, user acceptability, and production, are used to deploy each program. The application teams in each environment plan the patching schedules so that before a patch is applied to the production environment, ensure it's been tested and found to have no negative impact on the program. Given below is a table of how to have a planned patching strategy.

Patch Baseline	Non Production environment		Production environment
	Development environment	Test environment	
Step 1: Schedule the patch groups and maintenance windows for each month.			
1st week of month New patches become available	Maintenance window: 2nd week of month	Maintenance window: 3rd week of month	Maintenance window: 4th week of month
Step 2	Step 3	Step 4	Step 5

Table 1: Patching strategy

Step 1:

The maintenance windows for each application team's servers are scheduled in a range of environments, and the tags which represent the servers' patch groups and maintenance windows are set up accordingly:

The servers in the application environment, targets of a specific patch baseline, are represented by the Patch Group tag. Patch groups assist in ensuring that the appropriate patch baselines are applied to the appropriate collection of instances. Patch groups help prevent patches' deployment into the production environment before testing.

The Maintenance Window tag represents the patching schedule for the servers. A standard maintenance window should exist for all servers in a patch group.

Step 2:

Systems Manager - Patch Manager regularly makes new patches available through operating system-specific patch baselines based on defined configurations.

Step 3:

Based on Patch groups and maintenance window tags, write an automation code to configure the Patch manager to apply patches to the development environment.

- The application development teams test the program when patching is finished to ensure that everything functions as intended.
- Application teams ask the cloud services team to stop patching to other patch groups and other environments if the new patch causes any issues with their application, either by stopping the maintenance windows or deregistering the patch task execution.

Step 4:

Any additional non-production environments needing patching are done when the development environment has been successfully patched. The program is tested and confirmed to function properly in all non-production contexts, just like in the development environment. Application teams ask the cloud services team to stop patching to the production environment if there are any issues.

Step 5

The production environment is patched once all the non-production environments have been successfully updated.

Conclusion:

In Conclusion, the document outlines the various features of AWS services and how to use configure them. By following the above mentioned recommendations, Cobrakai application is completely migrated to the cloud in a secure manner, moving the entire application to the cloud enhanced the security of the application, different roles have been given to users according to their tasks, a step by step patching strategy has been provided along with setting a Lambda Function to process payments.

References

Amazon . (n.d.). *Amazon CloudWatch.* https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/WhatIsCloudWatch.html.

Amazon. (n.d.). *Amazon GuardDuty.* https://docs.aws.amazon.com/guardduty/latest/ug/guardduty_finding-types-ec2.html.

Amazon. (n.d.). *Amazon Rekognition.* https://docs.aws.amazon.com/rekognition/latest/dg/what-is.html.

Amazon. (n.d.). *Amazon Route 53.* https://aws.amazon.com/blogs/security/how-to-protect-your-web-application-against-ddos-attacks-by-using-amazon-route-53-and-a-content-delivery-network/.

Amazon. (n.d.). *Amazon S3 glacier documetation.* https://docs.aws.amazon.com/amazonglacier/latest/dev/introduction.html.

Amazon. (n.d.). *Amazon Simple Storage Service.* https://docs.aws.amazon.com/AmazonS3/latest/userguide/Welcome.html#S3Features.

Amazon. (n.d.). *AWS Best Practices for DDoS.* https://d0.awsstatic.com/whitepapers/Security/DDoS_White_Paper.pdf.

Amazon. (n.d.). *AWS Identity and Access Management.* https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/IAMRoute53.html.

Amazon. (n.d.). *AWS Key Management Services.* https://aws.amazon.com/kms/features/.

Amazon. (n.d.). *AWS Lambda Developer Guide.* https://docs.aws.amazon.com/lambda/latest/dg/gettingstarted-awscli.html.

Amazon. (n.d.). *AWS Shield.* https://docs.aws.amazon.com/waf/latest/developerguide/shield-chapter.html.

Amazon. (n.d.). *AWS Systems Manager Patch Manager.* https://docs.aws.amazon.com/systems-manager/latest/userguide/systems-manager-patch.html.

Amazon. (n.d.). *AWS WAF.* https://docs.aws.amazon.com/waf/latest/developerguide/ddos-responding.html.

Documentation, R. (n.d.). *Nginx and Gunicorn.* https://rdmo.readthedocs.io/en/latest/deployment/nginx.html.

Freecodecamp. (n.d.). *How to build a web application using Flask and deploy it to the cloud.* https://www.freecodecamp.org/news/how-to-build-a-web-application-using-flask-and-deploy-it-to-the-cloud-3551c985e492/.

Github. (n.d.). *Free domain with Freenom.* https://minnojs.github.io/docs/start/server/installation/domain/freedomain/.

Javvaji, T. (2022). *Cobrakai Application.* Maryland Applied Graduate Engineering. College Park: University of Maryland College Park.

Maryland, U. (2022). *COBRA KAI STRIKE FIRST - STRIKE HARD - NO MERCY.* https://umd.instructure.com/courses/1336126/assignments/6164203.

Maryland, U. o. (n.d.). *ENPM665 – Final.* https://umd.instructure.com/courses/1336126/assignments/6193427.