REQUIREMENT

- Deploy the application in a fully secure, scalable, and highly available microservice architecture.

SOLUTION OVERVIEW

The solution we have configured Microservices architecture using AWS EKS with AWS managed node and ECR for the Image registry and other services like S3 for the image storage and CloudFront for caching the data, also used ElastiCache Redis for the in-memory store and Created Custom VPC and deployed EKS Managed nodes in Private Subnets, WAF for the Security and rate-limiting of API's

REGION

Every data center in AWS sits in its own region. The region your setup is configured for is AP-SOUTH-1 Mumbai. This means that requests from INDIA will be very fast.

VPC

So for the VPC, we have created Custom VPC with Public and private subnets
VPC name - fantasy
Note: Image used in this doc are for reference only

Name	VPC ID	State	IPv4 CIDR	IPv6 CIDR	DHCP options set	Main route table	Main
–	vpc-dafa2eb1	Available	172.31.0.0/16	–	dopt-0483306f	rtb-0f73f964	acl-4
sportgully	vpc-0379e1a81109e2826	Available	10.0.0.0/16	–	dopt-0483306f	rtb-05b7546ac39076d11	acl-0
SportGullyVPC	vpc-062b37184fc015c44	Available	10.0.0.0/16	–	dopt-0483306f	rtb-0f258e7f712bc9486	acl-0
perf-testing	vpc-0f8682926eb83aa75	Available	10.0.0.0/16	–	dopt-0483306f	rtb-0d4410e59865dedc8	acl-0

CIDR - 10.0.0.0/16

We have also created Public-Private sunsets for each availability zone as shown below.

Subnets

Name	Subnet ID	State	VPC	IPv4 CIDR	IPv6 CIDR
sportgully-Private-az-b	subnet-000654036b5de1cf1	Available	vpc-0379e1a81109e2826 | sportgully	10.0.64.0/18	–
sportgully-Public-az-a	subnet-01cd5d2757dc9a8ab	Available	vpc-0379e1a81109e2826 | sportgully	10.0.255.0/24	–
sportgully-Private-az-a	subnet-0f6818fffc8ce66c9	Available	vpc-0379e1a81109e2826 | sportgully	10.0.0.0/18	–
sportgully-Public-az-c	subnet-02fa615f215e1074f	Available	vpc-0379e1a81109e2826 | sportgully	10.0.253.0/24	–
sportgully-Private-az-c	subnet-02c260840e3bb2b43	Available	vpc-0379e1a81109e2826 | sportgully	10.0.128.0/18	–
sportgully-Public-az-b	subnet-05877687a7375c519	Available	vpc-0379e1a81109e2826 | sportgully	10.0.254.0/24	–

fantasy-Public-az-a - 10.0.0.0/24

fantasy-Private-az-a - 10.0.1.0/24

fantasy-Public-az-b - 10.0.3.0/24

fantasy-Private-az-b - 10.0.2.0/24

fantasy-Public-az-c - 10.0.4.0/24 **fantasy-Private-az-c - 10.0.5.0/24**

- We have created four route tables one for the public subnets and the other three for the private subnets per AZ (availability zone)

Name	Route table ID	Explicit subnet associat...	Edge associations	Main	VPC	Owner ID
sportgully-Private-az-a	rtb-02ad74f9b79b635f2	subnet-0f6818fffc8ce66...	–	No	vpc-0379e1a81109e2826 | sp...	818619633648
sportgully-Public	rtb-0e7031f78910933b1	3 subnets	–	No	vpc-0379e1a81109e2826 | sp...	818619633648
sportgully-Private-az-c	rtb-04b2401cf671659ce	subnet-02c260840e3bb...	–	No	vpc-0379e1a81109e2826 | sp...	818619633648
sportgully-Private-az-b	rtb-077ddffc43da3960d	subnet-000654036b5de...	–	No	vpc-0379e1a81109e2826 | sp...	818619633648

- To access the internet from the private subnets we have created one NAT gateway and updated the Private route tables So instances can have internet access through Nat Gateway.

Name	NAT gateway ID	Connectivit...	State	State message	Elastic IP address	Private IP address	Network interface ID	VPC
sportgully-az-a	nat-032f48fb9bfd0c26a	Public	Available	–	13.127.19.23	10.0.255.201	eni-02e366bfe436807b8	vpc-0379e

For the **access to MongoDB Atlas**, we have created **VPC Peering** so it will have access to the **MongoDB Atlas privately.**

Name	Peering connection ID	Status	Requester VPC	Accepter VPC	Requester CIDRs	Accepter CIDRs	Requester owner ID	Acc
–	pcx-0093d93088054ada7	Active	vpc-03d7c930e1c0b2537	vpc-0379e1a81109e2826 / sp...	192.168.248.0...	10.0.0.0/16	088492164514	818

RDS

Databases

Group resources | Modify | Actions | Restore from S3 | Create database

Filter by databases

DB identifier	Role	Engine	Region & AZ	Size	Status	CPU	Current activity	Maintenance
sportgully	Instance	MySQL Community	ap-south-1b	db.t2.micro	Available	8.03%	6 Connections	none

We are using RDS for the MySQL use case as of now we are using t2.micro and later on, while moving towards production we can upgrade to a higher version.

RDS configurations (provisioned with terraform)

VPC	fantasy (vpc-0379e1a81109e2826)
Subnet group	fantasy_rds_subnet_group
Subnets	subnet-0f6818fffc8ce66c9
	subnet-000654036b5de1cf1
	subnet-02c260840e3bb2b43
VPC security groups	fantasy_security_group (sg-067101cd463ab8937)
Publicly accessible	No
Option groups	default:mysql-8-0
Engine version	8.0.28
Parameter group	fantasy-rds-parameter-group
Deletion protection	Enabled
Multi-AZ	No
Storage type	General Purpose SSD (gp2)
Storage	20 GiB
Storage autoscaling	Enabled
Maximum storage threshold	1024 GiB
Performance Insights enabled	No
Backup retention period	7 days
Backup window	19:31 UTC (1:01 AM)
Monitoring	Enhanced monitoring enabled Granularity 5 seconds Monitoring role - rds-monitoring-role
Log Exports	Audit log Error log General log Slow query log

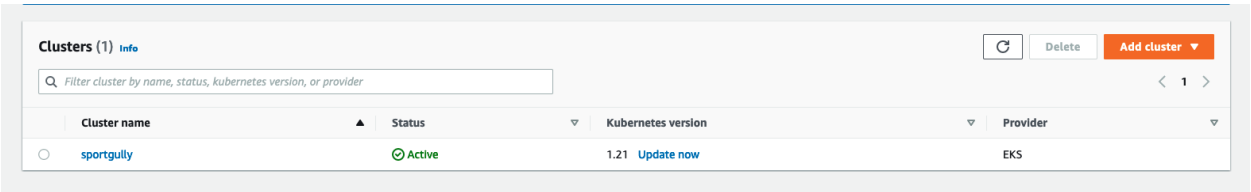
Maintenance	Disabled auto minor version upgrade
DB instance maintenance window	Monday - 10:30 am
Copy tags to snapshots	Enabled
Backup window	12:30 AM - 13:31 PM

Kubernetes

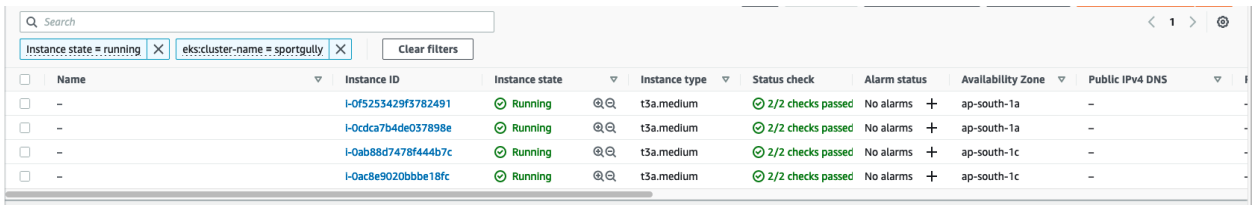
- We have user **EKS** as a Kubernetes service to deploy our entire workload.
- EKS cluster version - **1.21**
- Latest available - **1.22**

Configuration

VPC	fantasy (vpc-0379e1a81109e2826)
Subnets	subnet-05877687a7375c519 subnet-02fa615f215e1074f subnet-000654036b5de1cf1 subnet-0f6818fffc8ce66c9 subnet-01cd5d2757dc9a8ab subnet-02c260840e3bb2b43
Cluster security group	eks-cluster-sg-fantasy-1333161(sg-06894552ad3d3743f)
API server endpoint access	Public and private
Add-ons	coredns, kube-proxy, vpc-cni



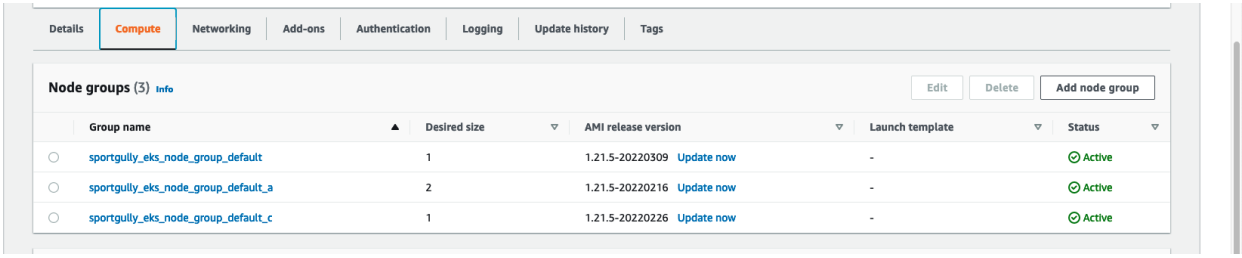
- We have added worker nodes as EC2 in private subnets.



Node Groups

- We have created the Node groups as per the AZ (availability zones) for different purposes like some of the Pods are deployed into the Node which lies in AZ-B so we have to create the Node Group accordingly to the particular subnets.
- Also, we have created one default node group which targets all the AZ (availability zones).

Node IAM role ARN	arn:aws:iam::818619633648:role/fantasy_eks_node_group_role
Subnets	subnet-000654036b5de1cf1 subnet-0f6818fffc8ce66c9 subnet-02c260840e3bb2b43
Configure SSH access to nodes	Disabled
Disk size	50GB



Give Cluster Access to other Users

Now, to use the clusters and give other users access to the cluster follow the below details.

- Follow this document to install kubectl in your system to interact with the EKS cluster and also install aws cli and configure first
- https://docs.aws.amazon.com/eks/latest/userguide/install-kubectl.html
- **Note**: install the kubectl version as per the version of your cluster
- After installation of the kubectl uses the below command to update the kube config in your system.
- `aws eks update-kubeconfig --name fantasy --region ap-south-1`
- Follow this document to add your IAM user to get access to the cluster
- https://aws.amazon.com/premiumsupport/knowledge-center/amazon-eks-cluster-access/

Note: you need to update `aws-auth` from the other user account because you don't have access to update the aws-auth config map now

- Now try kubectl get pod you will get below result

```
→  Downloads kubectl get pods
NAME                                    READY   STATUS      RESTARTS   AGE
bonus-expire-1635120000-r6xz9           0/1     Completed   0          2d13h
bonus-expire-1635206400-7c8mb           0/1     Completed   0          37h
bonus-expire-1635292800-hzs6f           0/1     Completed   0          13h
docker-registry-lb-5b9f8c88f9-bf789     1/1     Running     0          49d
leaderboard-1635340680-c695c            0/1     Completed   0          12m
leaderboard-1635340740-gdktg            0/1     Completed   0          11m
leaderboard-1635340800-7nfd6            0/1     Completed   0          10m
match-live-1635340680-cwrlr             0/1     Completed   0          12m
match-live-1635340740-6wzbh             0/1     Completed   0          11m
match-live-1635340800-jswpl             0/1     Completed   0          10m
set-selectby-1635340680-ddnpp           0/1     Completed   0          12m
set-selectby-1635340740-m4zcf           0/1     Completed   0          11m
set-selectby-1635340800-w2tsj           0/1     Completed   0          10m
```

- Please note that we have configured three environments production, staging, dev, GitLab(for CI/CD in GitLab runner), and in the default namespace.

```
→  Downloads kubectl get ns
NAME              STATUS   AGE
crictracker       Active   7d17h
default           Active   88d
dev               Active   65d
gitlab            Active   78d
kube-node-lease   Active   88d
kube-public       Active   88d
kube-system       Active   88d
lens-platform     Active   62d
loki-grafana      Active   15d
production        Active   88d
redis             Active   44h
staging           Active   88d
```

ECR

Visibility settings	Private
Tag immutability	Disabled
Scan on push	Enabled

Repository name	URI	Created at	Tag immutability	Scan frequency	Encryption type	Pull through cache
mongo-backup	818619633648.dkr.ecr.ap-south-1.amazonaws.com/mongo-backup	March 25, 2022, 10:47:15 (UTC+05.5)	Disabled	Scan on push	AES-256	Inactive
mongo-backup-staging	818619633648.dkr.ecr.ap-south-1.amazonaws.com/mongo-backup-staging	March 25, 2022, 10:47:15 (UTC+05.5)	Disabled	Scan on push	AES-256	Inactive
mysql-backup	818619633648.dkr.ecr.ap-south-1.amazonaws.com/mysql-backup	March 25, 2022, 10:47:15 (UTC+05.5)	Disabled	Scan on push	AES-256	Inactive
mysql-backup-staging	818619633648.dkr.ecr.ap-south-1.amazonaws.com/mysql-backup-staging	March 25, 2022, 10:47:15 (UTC+05.5)	Disabled	Scan on push	AES-256	Inactive
sportgully-admin-panel	818619633648.dkr.ecr.ap-south-1.amazonaws.com/sportgully-admin-panel	February 25, 2022, 10:33:24 (UTC+05.5)	Disabled	Scan on push	AES-256	Inactive
sportgully-admin-panel-stag	818619633648.dkr.ecr.ap-south-1.amazonaws.com/sportgully-admin-panel-stag	February 25, 2022, 10:33:22 (UTC+05.5)	Disabled	Scan on push	AES-256	Inactive
sportgully-app	818619633648.dkr.ecr.ap-south-1.amazonaws.com/sportgully-app	February 25, 2022, 10:33:22 (UTC+05.5)	Disabled	Scan on push	AES-256	Inactive
sportgully-app-stag	818619633648.dkr.ecr.ap-south-1.amazonaws.com/sportgully-app-stag	February 25, 2022, 10:33:25 (UTC+05.5)	Disabled	Scan on push	AES-256	Inactive
sportgully-node	818619633648.dkr.ecr.ap-south-1.amazonaws.com/sportgully-node	February 25, 2022, 10:33:26 (UTC+05.5)	Disabled	Scan on push	AES-256	Inactive
sportgully-node-backend	818619633648.dkr.ecr.ap-south-1.amazonaws.com/sportgully-node-backend	February 25, 2022, 10:33:26 (UTC+05.5)	Disabled	Scan on push	AES-256	Inactive
sportgully-node-backend-stag	818619633648.dkr.ecr.ap-south-1.amazonaws.com/sportgully-node-backend-stag	February 25, 2022, 10:33:27 (UTC+05.5)	Disabled	Scan on push	AES-256	Inactive
sportgully-node-stag	818619633648.dkr.ecr.ap-south-1.amazonaws.com/sportgully-node-stag	February 25, 2022, 10:33:22 (UTC+05.5)	Disabled	Scan on push	AES-256	Inactive
sportgully-prize-distribution	818619633648.dkr.ecr.ap-south-1.amazonaws.com/sportgully-prize-distribution	May 05, 2022, 14:48:59 (UTC+05.5)	Disabled	Scan on push	AES-256	Inactive
sportgully-prize-distribution-stag	818619633648.dkr.ecr.ap-south-1.amazonaws.com/sportgully-prize-distribution-stag	May 05, 2022, 15:36:29 (UTC+05.5)	Disabled	Scan on push	AES-256	Inactive

S3- Images - KYC bucket with restriction

Aim: To ensure the user's data security we have to restrict the access of the S3 bucket to the only Root user, dedicated s3 user(kyc_user), and The credentials used to provision the infrastructure form terraform.

Our Solution: Bucket policy for KYC restrictions - this attached document contains the details regarding the IAM user policy and s3 bucket policy.

Route 53

- We have created one Route53 hosted zone.
- On is pubic in this we have added our all the record for the application.
- For the internal private hosted zone internal routes like RDS, Elasticache, and RDS Replicas.

- **Note**: as we have not moved towards final production so we have created beta. subdomain.

Hosted zones (3)

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

View details | Edit | Delete | Create hosted zone

Filter hosted zones by property or value

Domain name	Type	Created by	Record count	Description	Hosted zone ID
sportgully.com	Public	Route 53	28	-	Z0112560W6HNLXCIPWEC
beta.sportgully.com	Public	Route 53	13	sportgully public zone	Z10373691VQO4I85TRAAC
sportgully.com.private	Private	Route 53	6	sportgully Private zone	Z0847996185HI5GW1G575

Public Hosted zone records

ecord name	Type	Routin...	Differ...	Value/Route traffic to
eta.sportgully.com	NS	Simple	-	ns-1810.awsdns-34.co.uk. ns-1423.awsdns-49.org. ns-515.awsdns-00.net. ns-425.awsdns-53.com.
eta.sportgully.com	SOA	Simple	-	ns-1810.awsdns-34.co.uk. awsdns-hostmaster.amazon.com. 1 7200 900 1209600 86400
276d9b79a2b8decc41bbf02326abc7cd.beta.sportgully.com	CNAME	Simple	-	_a5dac799b845c42946a92cb199bdef7f.gskhnxswdw.acm-validations.aws.
dmin-stag.beta.sportgully.com	A	Simple	-	k8s-sportsgullydevsta-a318187ecd-297556309.ap-south-1.elb.amazonaws.com.
dmin-stag.beta.sportgully.com	TXT	Simple	-	"heritage=external-dns,external-dns/owner=my-identifier,external-dns/resource=ingress/staging/sportgully-ingress-stag"
dminv2.beta.sportgully.com	A	Simple	-	k8s-sportsgullydevsta-a318187ecd-297556309.ap-south-1.elb.amazonaws.com.
dminv2.beta.sportgully.com	TXT	Simple	-	"heritage=external-dns,external-dns/owner=my-identifier,external-dns/resource=ingress/staging/sportgully-ingress-stag"
piv2.beta.sportgully.com	A	Simple	-	k8s-sportsgullydevsta-a318187ecd-297556309.ap-south-1.elb.amazonaws.com.
piv2.beta.sportgully.com	TXT	Simple	-	"heritage=external-dns,external-dns/owner=my-identifier,external-dns/resource=ingress/staging/sportgully-ingress-stag"
ame.beta.sportgully.com	A	Simple	-	k8s-sportsgullydevsta-a318187ecd-297556309.ap-south-1.elb.amazonaws.com.
ame.beta.sportgully.com	TXT	Simple	-	"heritage=external-dns,external-dns/owner=my-identifier,external-dns/resource=ingress/staging/sportgully-ingress-stag"
.beta.sportgully.com	A	Simple	-	k8s-sportsgullydevsta-a318187ecd-297556309.ap-south-1.elb.amazonaws.com.
.beta.sportgully.com	TXT	Simple	-	"heritage=external-dns,external-dns/owner=my-identifier,external-dns/resource=ingress/staging/sportgully-ingress-stag"

Private hosted zone records

Record name	Type	Routin...	Differ...	Value/Route traffic to
sportgully.com.private	NS	Simple	-	ns-1536.awsdns-00.co.uk. ns-0.awsdns-00.com. ns-1024.awsdns-00.org. ns-512.awsdns-00.net.
sportgully.com.private	SOA	Simple	-	ns-1536.awsdns-00.co.uk. awsdns-hostmaster.amazon.com. 1 7200 900 1209600 86400
rds-readreplica.sportgully.com.private	CNAME	Simple	-	sportgully.clebetqfwpjj.ap-south-1.rds.amazonaws.com
rds.sportgully.com.private	CNAME	Simple	-	sportgully.clebetqfwpjj.ap-south-1.rds.amazonaws.com
redis-cache.sportgully.com.private	CNAME	Simple	-	sportgully.jxi59f.ng.0001.aps1.cache.amazonaws.com
redis-leaderboard.sportgully.com.private	CNAME	Simple	-	sportgully-2.jxi59f.ng.0001.aps1.cache.amazonaws.com

ALB

- **Note: We are provisioning ALB (Application Load Balancer) with ALB controller ingress deployed in Cluster.**

- Here's the link to deploy/set up the ALB controller in the cluster.
- [Installing the AWS Load Balancer Controller add-on - Amazon EKS]
- Currently, we have created only 2 Load balancers for this deployment using Kubernetes ingress.
  Its currently used for Production deployment and staging deployment

- **Also when we move towards the final production deployment we'll create the 2 load balancers as per the deployment environments (Production, Staging).**
- **We have attached the WAF with ALB by using Ingress annotations.**

Setting Up External DNS:

- In Order to add the entries to the Route53 hosted zone we have set up the External DNS component to our Kubernetes Cluster.
- [Setup External DNS - AWS LoadBalancer Controller]

Annotations Used in ALB:

```
  -   alb.ingress.kubernetes.io/wafv2-acl-arn: ARN of WAF

  -   kubernetes.io/ingress.class: alb

  -   alb.ingress.kubernetes.io/group.name: for grouping the ingress for different
      namespace to same alb

  -   alb.ingress.kubernetes.io/scheme: internet-facing

  -   alb.ingress.kubernetes.io/load-balancer-attributes: idle_timeout.timeout_seconds=600

  -   alb.ingress.kubernetes.io/backend-protocol: HTTP or HTTPS ( based on our backend )

  -   alb.ingress.kubernetes.io/certificate-arn:  ARN of ACM certificate we have issued

  -   alb.ingress.kubernetes.io/subnets:  Subnet ID of public subnets

  -   alb.ingress.kubernetes.io/listen-ports: '[{"HTTPS":443}, {"HTTP":80}]'

  -   alb.ingress.kubernetes.io/actions.ssl-redirect: '{"Type": "redirect",
      "RedirectConfig": { "Protocol": "HTTPS", "Port": "443", "StatusCode": "HTTP_301"}}'

  -   external-dns.alpha.kubernetes.io/hostname: for External DNS to enter the Records to
      route53 Hosted Zone

  -   alb.ingress.kubernetes.io/tags:  give Tags to ALB.
```

Name	DNS name	State	VPC ID	Availability Zones	Ty
api-sportgully-com-ALB	api-sportgully-com-ALB-1159292434.ap-south-1.elb.amazonaws.com	Active	vpc-062b37184fc015c44	ap-south-1b, ap-south-1a	ap
k8s-sportsgullydevsta-a318187ecd	k8s-sportsgullydevsta-a318187ecd-297556309.ap-south-1.elb.amazonaws.com	Active	vpc-0379e1a81109e2826	ap-south-1a, ap-south-1c, ap-south-1b	ap
k8s-sportsgullyprod-4763fa4285	k8s-sportsgullyprod-4763fa4285-1127418738.ap-south-1.elb.amazonaws.com	Active	vpc-0379e1a81109e2826	ap-south-1a, ap-south-1c, ap-south-1b	ap

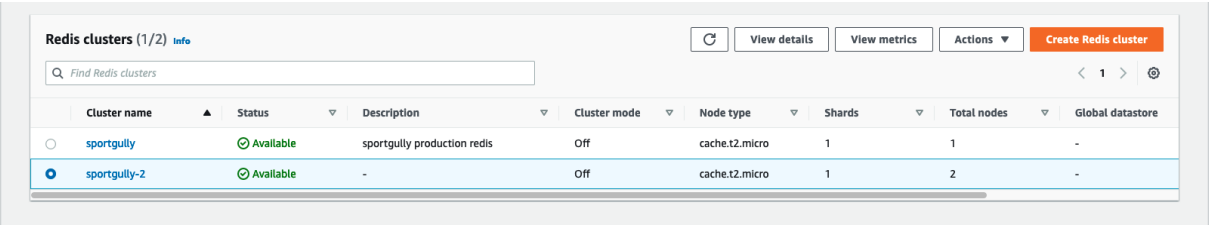
Redis

- We have deployed the Redis in Kubernetes local environment for staging namespace
- For all the separate Namespaces the Redis deployment differs.
- For Staging, Redis deployment is deployed in Kubernetes with a 5GB EBS volume attached.
- **For final production namespace deployment, we are going to deploy the elasticsearch cluster**

Note: we have provisioned 2 Redis instances because **1st is for Leaderboard caching** and **2nd is for General Caching**.

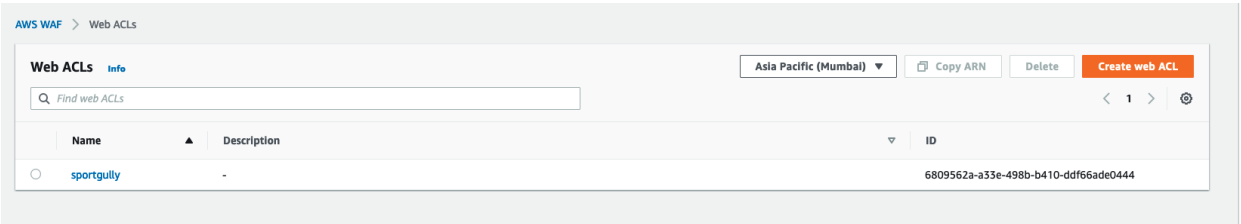
Cluster name	**fantasy**
Engine	**Redis**
Engine version	**6.2.5**
Node type	**cache.t2.micro**
Cluster mode	**Off**
Number of nodes	**1**
Multi-AZ	**Disabled**
Auto-failover	**Disabled**
Parameter group	**default.redis6.x**
Primary endpoint	**fantasy.jxi59f.ng.0001.aps1.cache.amazonaws.com:6379**
Reader endpoint	**fantasy-ro.jxi59f.ng.0001.aps1.cache.amazonaws.com:6379**
Slow logs	**Disabled**
Engine logs	**Disabled**

VPC ID and name	**fantasy (vpc-0379e1a81109e2826)**
Security groups	**sg-0f5c87886a9d77f5a**
Maintenance window	**Saturday 22:30 - Saturday 23:30 UTC**
Auto upgrade minor versions	**Enabled**
Automatic backups	**Disabled**
Tags	**Key Environment: Production** **Contact: Aman Makwana**



WAF

- For security, we have used AWS WAF for various purposes like Rate Limiting & blocking server IPs and it's attached with ALB to protect.
- We have attached a load balancer to WAF with **ingress annotations**.



- If you don't find Web ACLs please check you are in the Mumbai region

MongoDB

- **Currently, we are using the M10 cluster - MongoDB**
- **M10 cluster features include - VPC peering, enhanced monitoring, and automated backups.**
- For security purposes, we have also ensured the DB backup cron in Kubernetes which dumps the DB to the S3 bucket name - fantasy-mongodbdumps-1638949745, fantasy-mysqldumps-1638949745.

Connect to EKS cluster

To connect to an EKS cluster you need to create a user access key/ secret key with permission EKS cluster access policy or EKS full access policy & configure AWS CLI Version 2 to your system. I am sharing one link you can follow that link to install and configure AWS CLI with created user keys.
https://docs.aws.amazon.com/cli/latest/userguide/install-cliv2.html

You can follow the above link for the AWS CLI configuration.
After configuration of the AWS CLI, you can verify the AWS CLI Version with

```
aws --version
```

Now connect to the cluster using the below commands
Install kubectl to interact with the EKS cluster

https://docs.aws.amazon.com/eks/latest/userguide/install-kubectl.html

Follow the above steps to install kubectl

```
aws eks update-kubeconfig --name fantasy --region ap-south-1
```

Now you can contact Aman Makwana , Jay Dobariya to update your details to the AWS config file, and then you can able to access the cluster.