

Implementación de Infraestructura Cloud AWS – Proyecto Promarketing (Casino Online)

Terraform IaC – Despliegue modular con AWS Services

Repo: <https://github.com/JavierPulidoT/reto-cloud-promarketing>

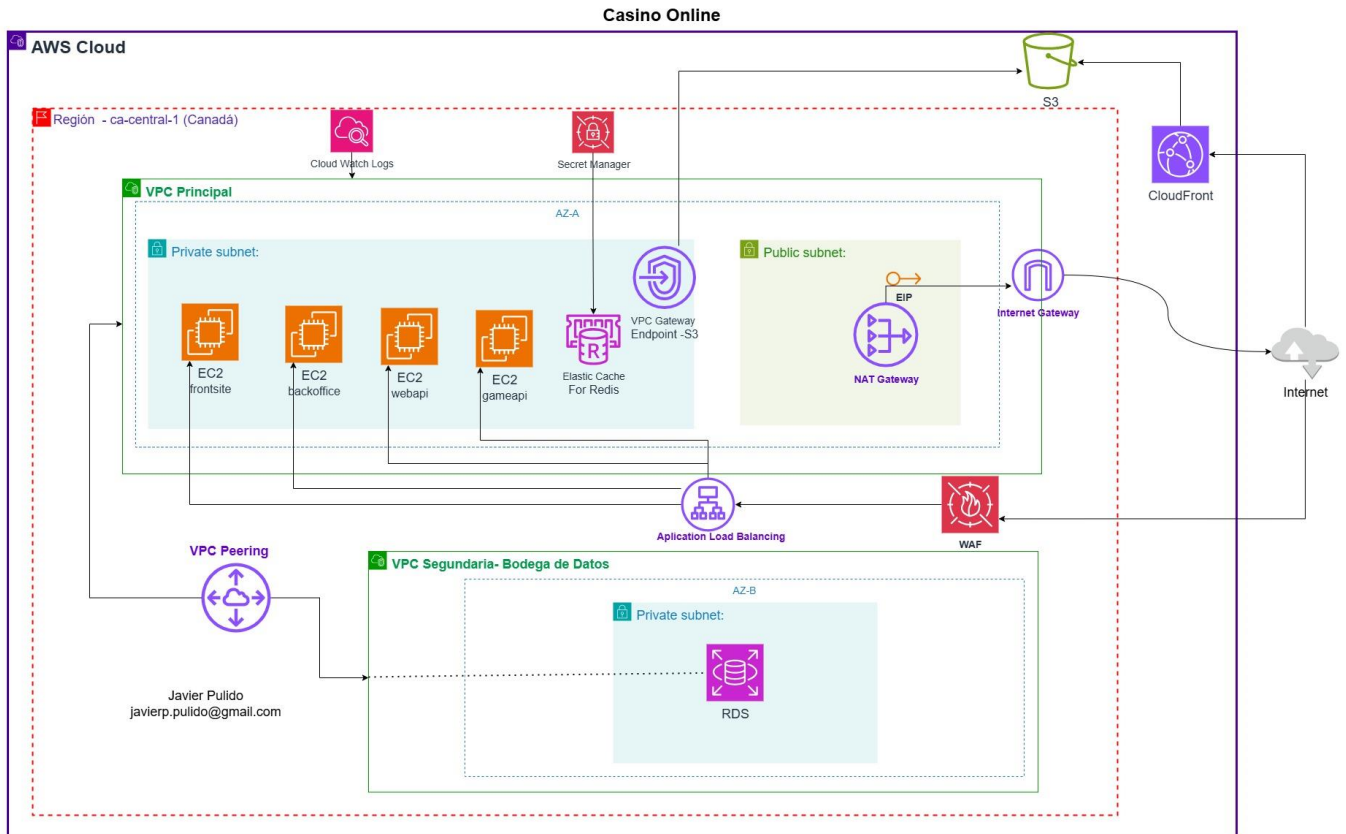
Calculadora:

<https://calculator.aws/#/estimate?id=5293298fce6103a6699da5cbc7280b7f48155b50>

Autor: Javier Pulido (javierp.pulido@gmail.com)

Región: ca-central-1 (Canadá)

Diagrama:



1. Introducción

El presente documento describe el diseño, desarrollo y despliegue de una infraestructura Cloud en AWS, implementada mediante Terraform bajo el enfoque Infrastructure as Code (IaC). El proyecto “Promarketing – Casino Online” contempla la creación de servicios modulares en AWS (VPCs, EC2, ALB, RDS, Redis, S3, CloudFront, Secrets Manager y CloudWatch), garantizando alta disponibilidad, seguridad y automatización del entorno productivo. Se trató de desplegar la mayoría de los recursos por terraform.

2. Arquitectura general

La arquitectura consta de dos VPCs (principal y secundaria), distribuidas en múltiples zonas de disponibilidad. El flujo de tráfico sigue la ruta: CloudFront → WAF → ALB → EC2 Privadas → Redis / RDS.

Mecanismos de seguridad aplicados:

- Subnets privadas para instancias.
- Endpoints VPC para acceso privado a servicios (S3, Secrets Manager).
- CloudWatch Logs y Secrets Manager para gestión segura y observabilidad.

3. Estructura del proyecto Terraform

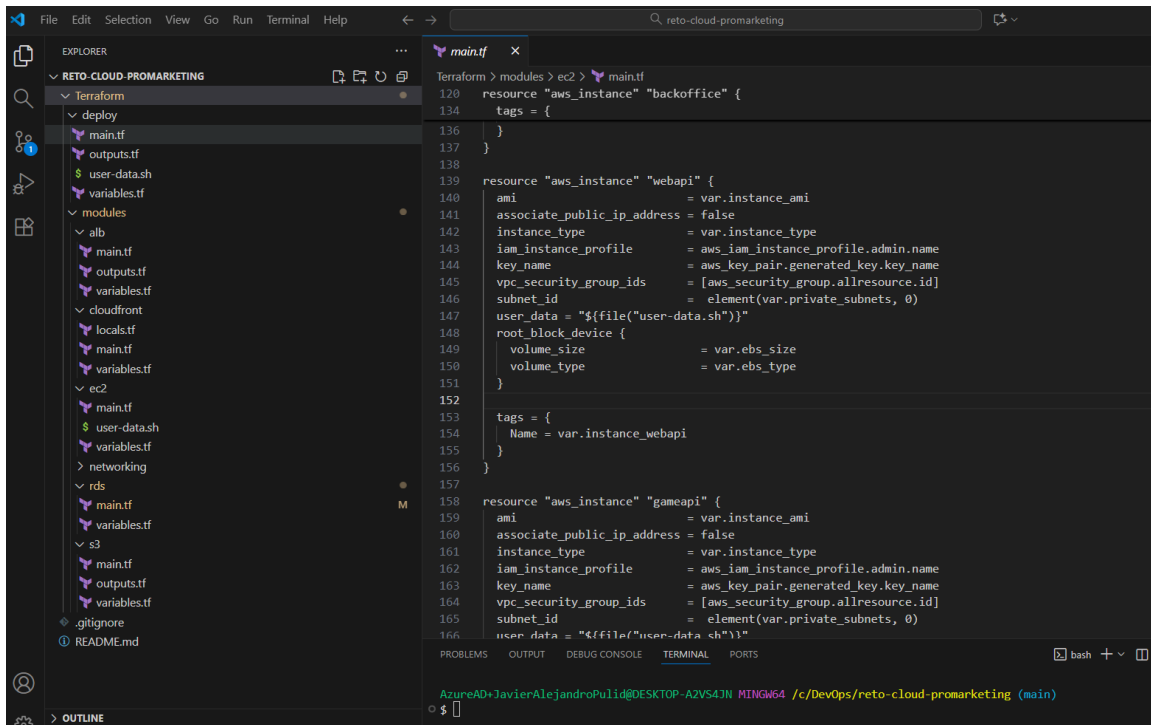
El código se organizó en módulos reutilizables que permiten desplegar la infraestructura por componentes de manera ordenada y mantenible. (Terraform init, plan, apply, destroy)

Terraform:

```
/deploy
├── main.tf
├── variables.tf
├── outputs.tf
└── user-data.sh
```

```
/modules
├── alb/
├── cloudfront/
├── ec2/
├── networking/
├── rds/
└── s3/
```

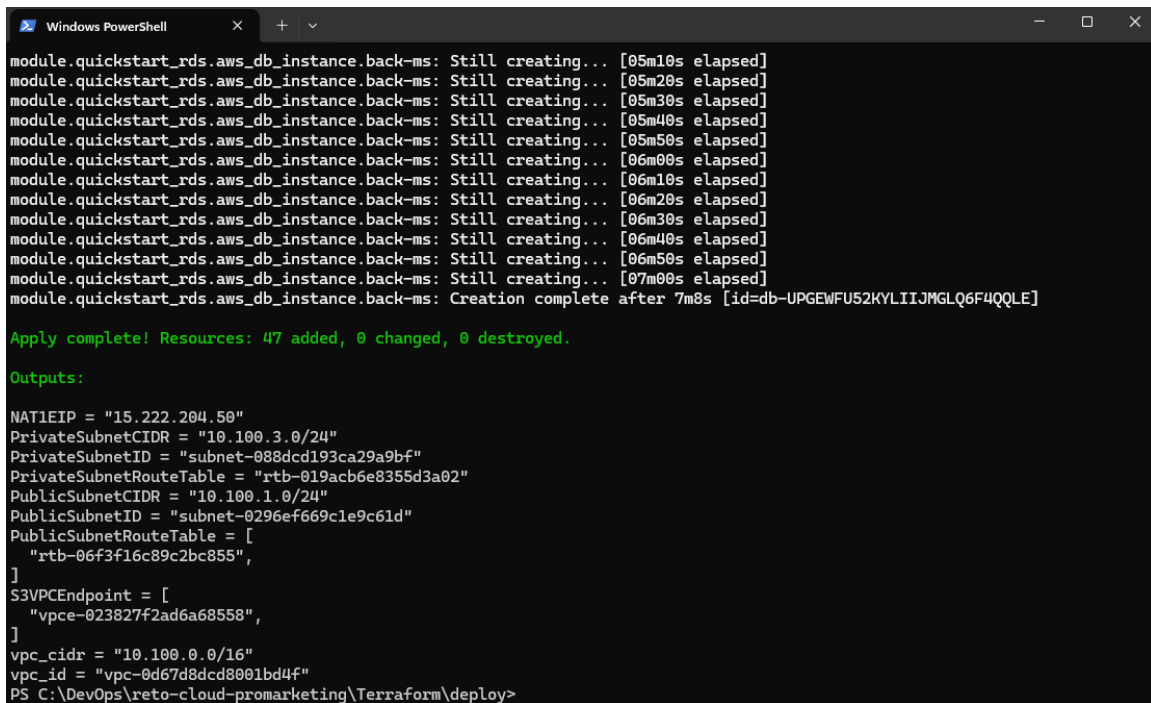
Visual Studio Code.



The screenshot shows the Visual Studio Code interface with a Terraform project named 'reto-cloud-promarketing'. The Explorer pane on the left shows the project structure, including a 'modules' directory with 'ec2' and 's3' submodules. The main.tf file is open in the editor, showing Terraform configuration for three AWS instances: 'backoffice', 'webapi', and 'gameapi'. The 'backoffice' instance is an 'ec2' resource. The 'webapi' and 'gameapi' instances are also 'ec2' resources. The 'gameapi' instance is configured with a specific AMI and instance type. The terminal at the bottom shows the command prompt for the user 'AzureAD+JavierAlejandroPulid'.

```
Terraform > modules > ec2 > main.tf
120 resource "aws_instance" "backoffice" {
134   tags = {
136   }
137 }
138
139 resource "aws_instance" "webapi" {
140   ami           = var.instance_ami
141   associate_public_ip_address = false
142   instance_type = var.instance_type
143   iam_instance_profile = aws_iam_instance_profile.admin.name
144   key_name        = aws_key_pair.generated_key.key_name
145   vpc_security_group_ids = [aws_security_group.allresource.id]
146   subnet_id       = element(var.private_subnets, 0)
147   user_data       = "${file("user-data.sh")}"
148   root_block_device {
149     volume_size = var.ebs_size
150     volume_type = var.ebs_type
151   }
152   tags = {
153     Name = var.instance_webapi
154   }
155 }
156
157 resource "aws_instance" "gameapi" {
158   ami           = var.instance_ami
159   associate_public_ip_address = false
160   instance_type = var.instance_type
161   iam_instance_profile = aws_iam_instance_profile.admin.name
162   key_name        = aws_key_pair.generated_key.key_name
163   vpc_security_group_ids = [aws_security_group.allresource.id]
164   subnet_id       = element(var.private_subnets, 0)
165   user_data       = "${file("user-data.sh")}"
166 }
```

Deploy



The screenshot shows a Windows PowerShell terminal window displaying the output of a Terraform deployment. The output shows the progress of creating various resources, including 'module.quickstart_rds.aws_db_instance.back-ms'. The deployment is complete after 7m8s. The terminal also shows the 'Apply complete' message and the 'Outputs' section, which lists various network and endpoint information.

```
module.quickstart_rds.aws_db_instance.back-ms: Still creating... [05m10s elapsed]
module.quickstart_rds.aws_db_instance.back-ms: Still creating... [05m20s elapsed]
module.quickstart_rds.aws_db_instance.back-ms: Still creating... [05m30s elapsed]
module.quickstart_rds.aws_db_instance.back-ms: Still creating... [05m40s elapsed]
module.quickstart_rds.aws_db_instance.back-ms: Still creating... [05m50s elapsed]
module.quickstart_rds.aws_db_instance.back-ms: Still creating... [06m00s elapsed]
module.quickstart_rds.aws_db_instance.back-ms: Still creating... [06m10s elapsed]
module.quickstart_rds.aws_db_instance.back-ms: Still creating... [06m20s elapsed]
module.quickstart_rds.aws_db_instance.back-ms: Still creating... [06m30s elapsed]
module.quickstart_rds.aws_db_instance.back-ms: Still creating... [06m40s elapsed]
module.quickstart_rds.aws_db_instance.back-ms: Still creating... [06m50s elapsed]
module.quickstart_rds.aws_db_instance.back-ms: Still creating... [07m00s elapsed]
module.quickstart_rds.aws_db_instance.back-ms: Creation complete after 7m8s [id=db-UPGEWFU52KYLIIJMLQ6F4QQLE]

Apply complete! Resources: 47 added, 0 changed, 0 destroyed.

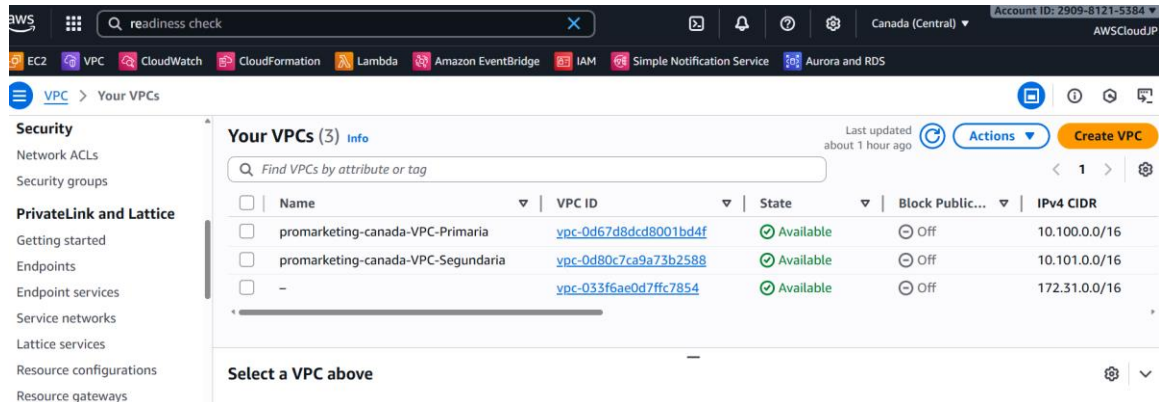
Outputs:

NAT1EIP = "15.222.204.50"
PrivateSubnetCIDR = "10.100.3.0/24"
PrivateSubnetID = "subnet-088dcd193ca29a9bfb"
PrivateSubnetRouteTable = "rtb-019acb6e8355d3a02"
PublicSubnetCIDR = "10.100.1.0/24"
PublicSubnetID = "subnet-0296ef669c1e9c61d"
PublicSubnetRouteTable = [
  "rtb-06f3f16c89c2bc855",
]
S3VPCEndpoint = [
  "vpce-023827f2ad6a68558",
]
vpc_cidr = "10.100.0.0/16"
vpc_id = "vpc-0d67d8dcd8001bd4f"
PS C:\DevOps\reto-cloud-promarketing\Terraform\deploy>
```

4. Descripción de módulos

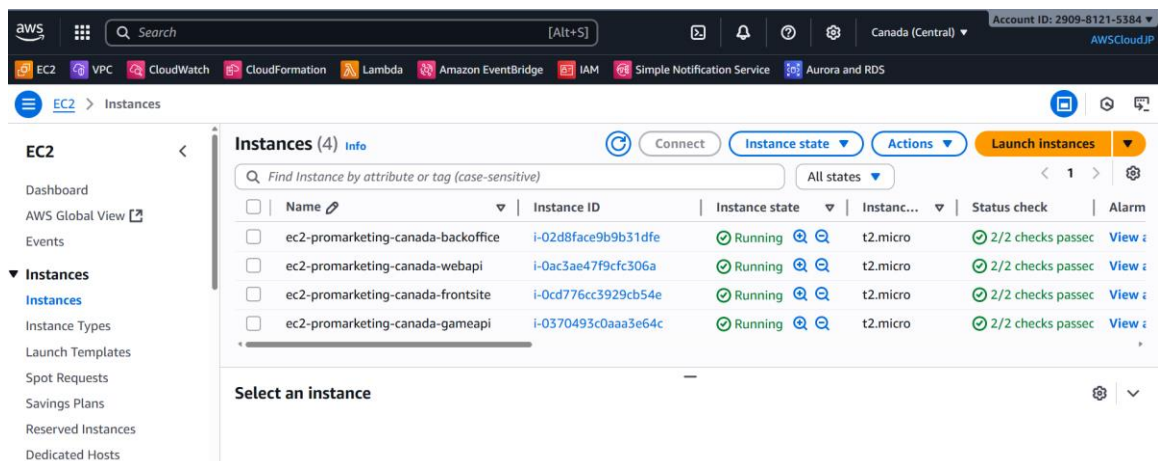
Networking

- Crea la VPC principal y secundaria, subnets públicas/privadas, IGW, NAT y tablas de ruteo.
- Incluye VPC Peering y Endpoints (S3 Gateway, Secrets Manager Interface).
- Configura etiquetas estándar: vpc-casino-prod-01-ca-central-1.



EC2

- Crea instancias frontsite, backoffice, webapi y gameapi en subredes privadas.
- Ejecuta user-data.sh para instalar el SSM Agent y habilitar acceso por AWS Systems Manager. Y Configura Security Groups y etiquetas.



No se considera tener BastionHost por el momento, las EC2 tienen SSM pre-instalado, razón que pueden ser administradas de manera mas confiable nivel SO.

Ejemplo , una de las EC2s, Se valida conexión y comunicación a (Salida a Internet)

```
Session ID: root-htqh9uershrhdjr2hna8lqs4a8 Shortcuts Instance ID: i-0cd776cc3929cb54e Terminate

[root@ip-10-100-3-111 bin]# ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=110 time=2.15 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=110 time=1.79 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=110 time=1.80 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=110 time=1.76 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=110 time=1.82 ms
64 bytes from 8.8.8.8: icmp_seq=6 ttl=110 time=1.85 ms
64 bytes from 8.8.8.8: icmp_seq=7 ttl=110 time=1.77 ms
64 bytes from 8.8.8.8: icmp_seq=8 ttl=110 time=1.78 ms
^C
--- 8.8.8.8 ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 7013ms
rtt min/avg/max/mdev = 1.764/1.839/2.146/0.118 ms
[root@ip-10-100-3-111 bin]#
```

Application Load Balancer (ALB)

- Expone tráfico HTTP/HTTPS mediante listeners en puertos 80 y 443.
- Integra certificado SSL/TLS de ACM.
- Enruta a target groups asociados a las EC2 privadas.

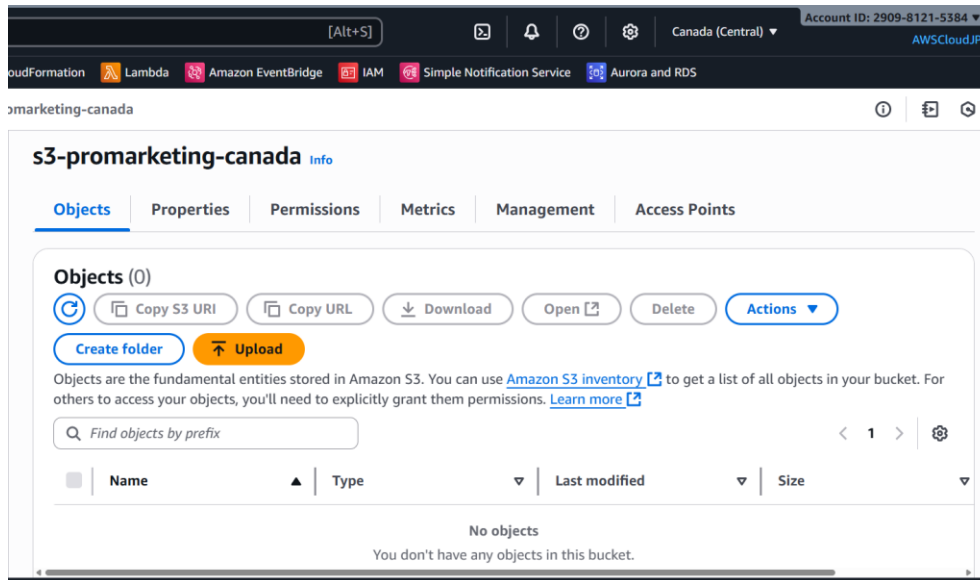
The screenshot displays the AWS Management Console interface for the 'Load balancers' section. At the top, there's a navigation bar with various AWS services and the account ID '2909-8121-5384'. The main content area shows a table of load balancers with one entry: 'alb-promarketing-canada', which is in an 'Active' state, of type 'application', and is 'Internet-facing' using 'IPv4' address type. It is associated with VPC ID 'vpc-0d67d8dcd8001bd4f' and spans '2 Availability Zones'. Below the table, the 'Details' tab is selected for the chosen load balancer, showing tabs for 'Listeners and rules', 'Network mapping', 'Resource map', 'Security', 'Monitoring', 'Integrations', 'Attributes', and 'Capacity'.

Name	State	Type	Scheme	IP address type	VPC ID	Availability Zones
alb-promarketing-canada	Active	application	Internet-facing	IPv4	vpc-0d67d8dcd8001bd4f	2 Availability Zones

S3

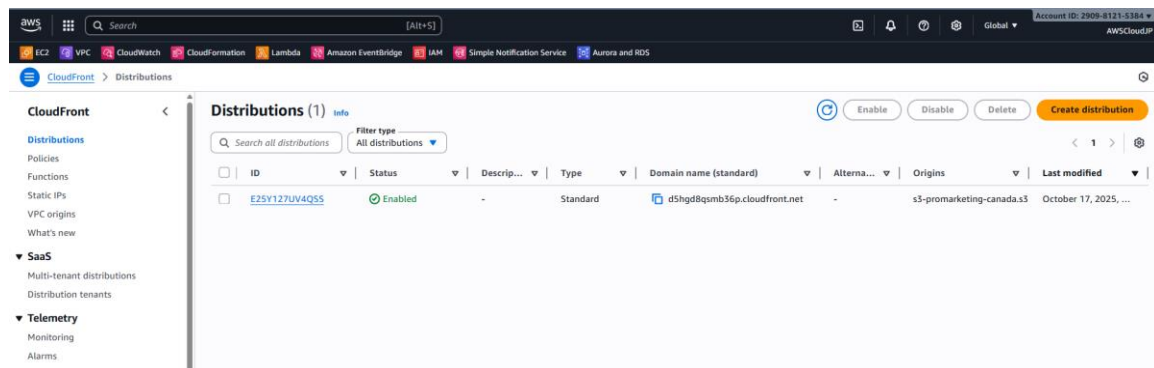
- Bucket privado con cifrado SSE (AES256).
- Bloqueo público y política para permitir acceso solo a CloudFront (OAC).

Nombre: s3-casino-prod-01-ca-central-1.



CloudFront

- Distribución conectada a S3 mediante Origin Access Control (OAC).
- Certificado ACM configurado, viewer protocol HTTPS.
- TTL y políticas de caché



RDS (Bodega de datos)

- Base de datos PostgreSQL en la VPC secundaria.
- y acceso vía VPC Peering.

The screenshot shows the AWS Management Console interface for an Amazon RDS instance. The top navigation bar includes the AWS logo, a search bar, and various service icons. The main content area displays the details for the instance 'rds-promarketing-canada'.

Summary

DB identifier	Status	Role	Engine	Recommendations
rds-promarketing-canada	Available	Instance	PostgreSQL	
CPU	Class	Current activity	Region & AZ	
4.28%	db.t3.micro	0 Connections	ca-central-1a	

Connectivity & security

Endpoint & port	Networking	Security
Endpoint rds-promarketing-canada.chk82cgc.ihx.ca-central-1.rds.amazonaws.com	Availability Zone ca-central-1a VPC	VPC security groups db-security-group-promarketing-canada (sg-02d6352b626dbbf8) Active

ElastiCache (Redis), Valkey caches compatible.

- Cluster Redis.

The screenshot shows the AWS Management Console interface for an Amazon ElastiCache cluster. The top navigation bar includes the AWS logo, a search bar, and various service icons. The main content area displays the details for the cluster 'elasticache-casino-prod-01'.

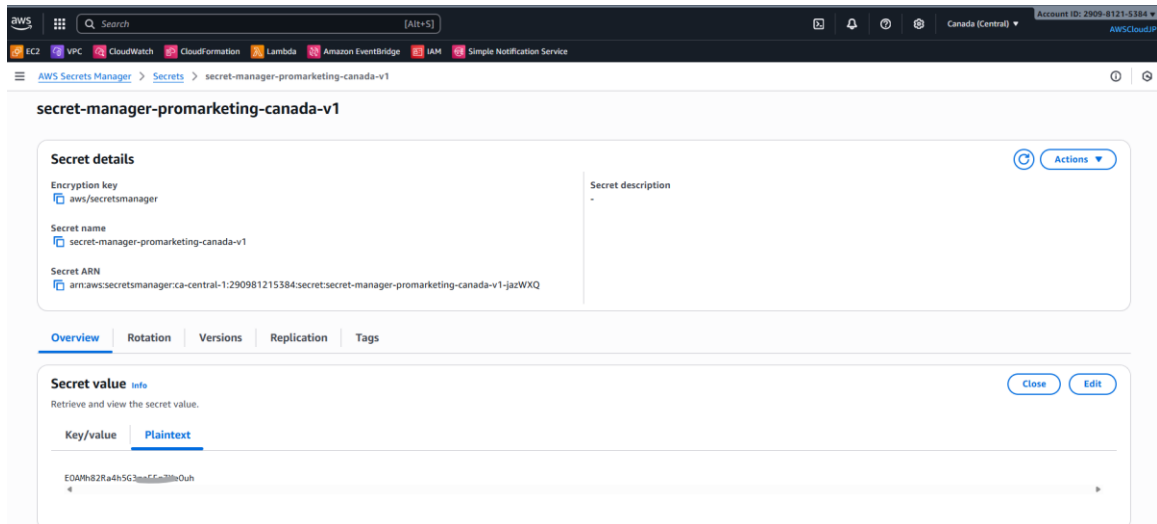
Cluster details

Cluster name	Description	Node type	Status
elasticache-casino-prod-01	Redis cache for WebAPI and GameAPI services	cache.t3.micro	Available
Engine	Engine version	Global datastore	Global datastore role
Valkey	8.2.0	-	-
Update status	Cluster mode	Shards	Number of nodes
Up to date	Disabled	1	3
Data tiering	Multi-AZ	Auto-failover	Encryption in transit
Disabled	Enabled	Enabled	Enabled
Encryption at rest	Parameter group	Outpost ARN	Transit encryption mode
Enabled	default.valkey8	-	Required
Configuration endpoint	Primary endpoint	Reader endpoint	ARN
-	master.elasticache-casino-prod-01.lev4aa.cac.1.cache.amazonaws.com:6379	replica.elasticache-casino-prod-01.lev4aa.cac.1.cache.amazonaws.com:6379	arn:aws:elasticache:ca-central-1:29098121538:4:replicationgroup:elasticache-casino-prod-01
Data migration			
No active migrations			

Connectivity and security - new | Nodes | Metrics | Logs | Maintenance and backups | Service updates | Tags

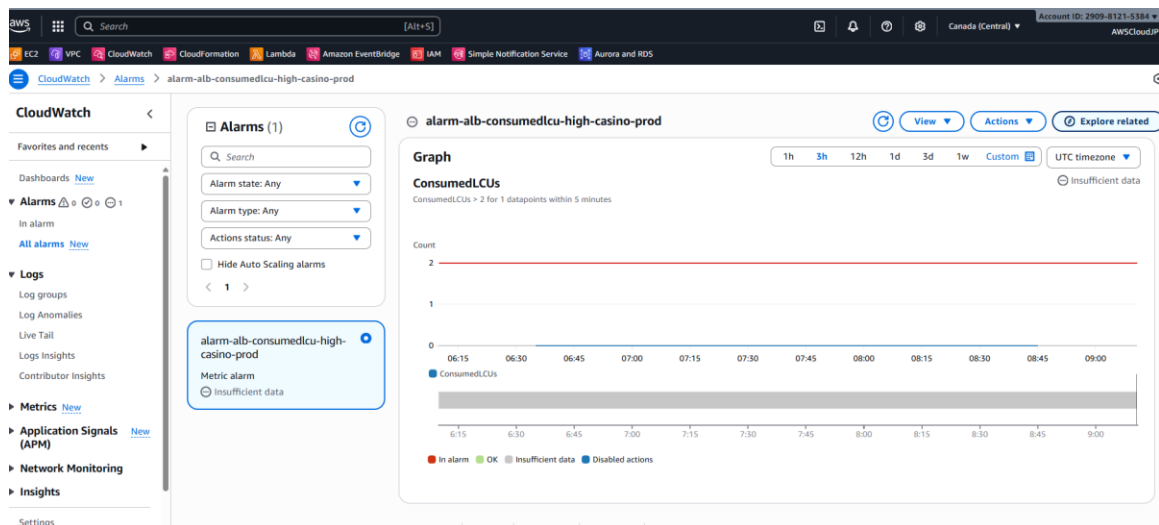
Secrets Manager

- Almacena credenciales de RDS y claves de integración.
- Acceso mediante VPC Endpoint Interface.



CloudWatch

- Recibe logs de EC2, ALB y RDS.
- Servicio regional dentro de ca-central-1.



5. Variables y outputs

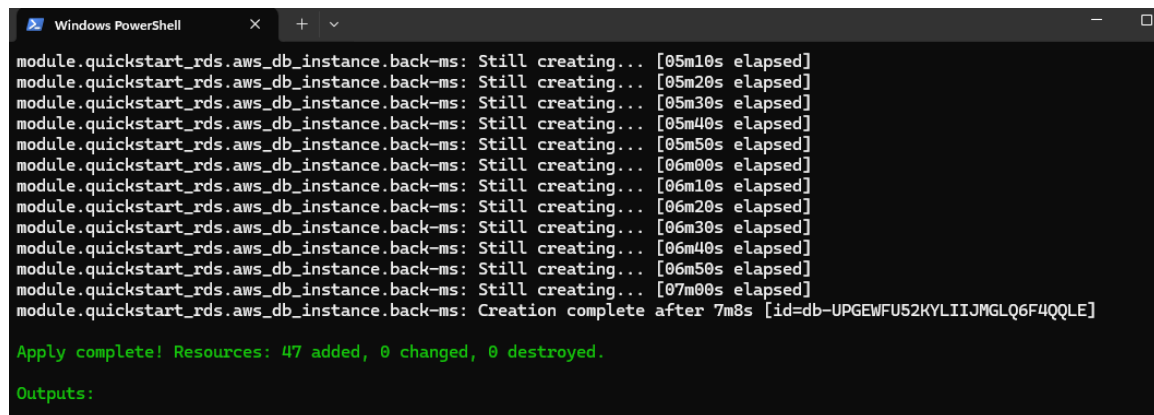
El proyecto utiliza variables reutilizables y outputs para exportar identificadores de recursos, como `vpc_id`, `subnet_ids`, `alb_dns`, `ec2_private_ips` y `rds_endpoint`. Esto permite una integración sencilla entre módulos y facilita la trazabilidad del despliegue.

6. Seguridad

- No se implementó Bastion Host, el acceso se realiza por AWS Systems Manager Session Manager.
- Tráfico restringido por SGs y NACLs.
- S3 y Secrets Manager accesibles solo por endpoints privados.

7. Resultados y despliegue

El código fue ejecutado con éxito mediante Terraform CLI. La mayoría de los recursos fueron desplegados en la consola AWS y validados visualmente. (terraform apply)

A screenshot of a Windows PowerShell terminal window. The title bar shows 'Windows PowerShell' with standard window controls. The terminal displays the output of a Terraform apply command. It shows a series of lines for 'module.quickstart_rds.aws_db_instance.back-ms' in a 'Still creating...' state, with elapsed times ranging from [05m10s] to [07m00s]. The final line indicates 'Creation complete after 7m8s' with an ID. Below this, a green message states 'Apply complete! Resources: 47 added, 0 changed, 0 destroyed.' and the word 'Outputs:' is visible at the bottom.


```
Windows PowerShell
module.quickstart_rds.aws_db_instance.back-ms: Still creating... [05m10s elapsed]
module.quickstart_rds.aws_db_instance.back-ms: Still creating... [05m20s elapsed]
module.quickstart_rds.aws_db_instance.back-ms: Still creating... [05m30s elapsed]
module.quickstart_rds.aws_db_instance.back-ms: Still creating... [05m40s elapsed]
module.quickstart_rds.aws_db_instance.back-ms: Still creating... [05m50s elapsed]
module.quickstart_rds.aws_db_instance.back-ms: Still creating... [06m00s elapsed]
module.quickstart_rds.aws_db_instance.back-ms: Still creating... [06m10s elapsed]
module.quickstart_rds.aws_db_instance.back-ms: Still creating... [06m20s elapsed]
module.quickstart_rds.aws_db_instance.back-ms: Still creating... [06m30s elapsed]
module.quickstart_rds.aws_db_instance.back-ms: Still creating... [06m40s elapsed]
module.quickstart_rds.aws_db_instance.back-ms: Still creating... [06m50s elapsed]
module.quickstart_rds.aws_db_instance.back-ms: Still creating... [07m00s elapsed]
module.quickstart_rds.aws_db_instance.back-ms: Creation complete after 7m8s [id=db-UPGEWUFU52KYLIIJMLQ6F4QQLE]

Apply complete! Resources: 47 added, 0 changed, 0 destroyed.

Outputs:
```

Prueba Peering From EC2 gameapi To RDS

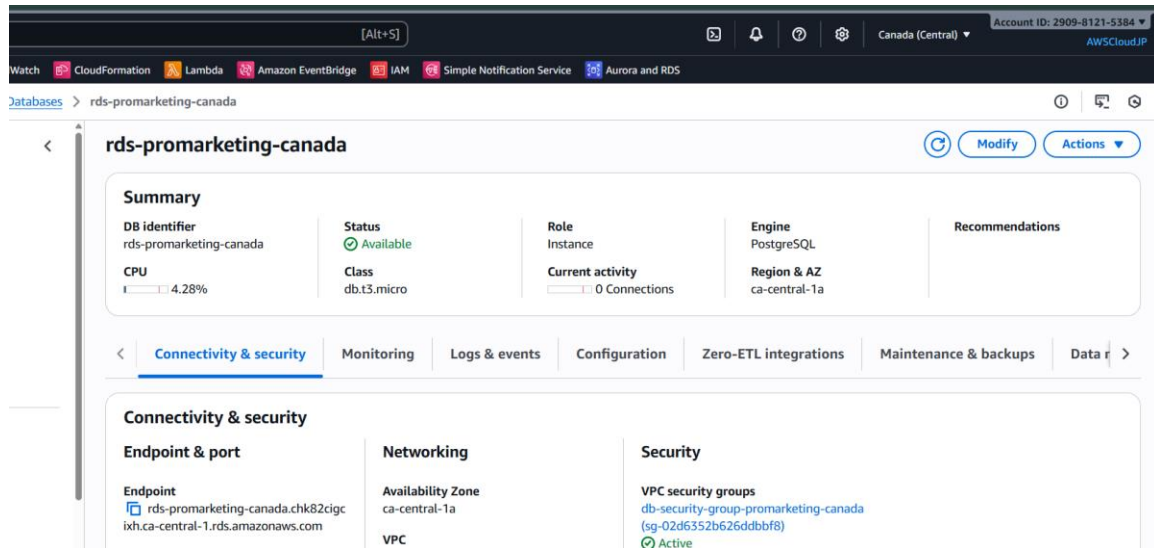
Session ID: root-j33jdshi7nlx7ko4ayc78krt8u

 Shortcuts

Instance ID: i-0370493c0aaa3e64c

Terminate

```
[root@ip-10-100-3-49 bin]# telnet rds-promarketing-canada.chk82cigcixh.ca-central-1.rds.amazonaws.com 5432
Trying 10.101.4.69...
Connected to rds-promarketing-canada.chk82cigcixh.ca-central-1.rds.amazonaws.com.
Escape character is '^]'.
```



The screenshot shows the AWS Management Console interface for the RDS instance 'rds-promarketing-canada'. The top navigation bar includes the AWS Cloud logo, account ID, and region (Canada (Central)). The left sidebar shows various AWS services. The main content area displays the instance details under the 'Databases' section.

Summary

DB identifier	Status	Role	Engine	Recommendations
rds-promarketing-canada	Available	Instance	PostgreSQL	

CPU 4.28%

Class db.t3.micro

Current activity 0 Connections

Region & AZ ca-central-1a

Connectivity & security | Monitoring | Logs & events | Configuration | Zero-ETL integrations | Maintenance & backups | Data r

Endpoint & port

Endpoint
rds-promarketing-canada.chk82cigcixh.ca-central-1.rds.amazonaws.com

Networking

Availability Zone
ca-central-1a

VPC

Security

VPC security groups
db-security-group-promarketing-canada (sg-02d6352b626ddbfb8)

Active

8. Conclusión

La infraestructura desplegada cumple con los lineamientos de seguridad, modularidad y buenas prácticas de AWS. El proyecto me ayudo a fortalecer practicas de IaC, capacidad de implementar un entorno cloud completo, automatizado y seguro mediante Terraform, cumpliendo la mayoría de requerimientos del reto Promarketing