Evaluación Módulo 6: Diseño de Infraestructura Cloud y DevOps para una Plataforma de Video Bajo Demanda

🌟Contexto

Una empresa de tecnología desea crear una plataforma de video bajo demanda (VOD), similar a Netflix. Los usuarios podrán registrarse, seleccionar contenido y hacer streaming desde diversos dispositivos. Se requiere una infraestructura cloud escalable, segura y eficiente, capaz de atender a miles de usuarios simultáneos.

🦜Objetivo del Informe

Diseñar una infraestructura cloud moderna para la plataforma VOD, justificando cada elección respecto a:

- Servicios de almacenamiento.
- Cómputo y escalabilidad.
- Redes y seguridad.
- CI/CD e infraestructura como código.

---

1. Diseño de la Infraestructura en la Nube (1.5 pt)

🐐Modelo de implementación

Tipo: Nube pública (AWS)

Justificación:

- Alta disponibilidad global.
- Red de entrega de contenido (CDN).
- Reducción de costos en mantenimiento y hardware.

🌳Modelo de servicio

Modelo híbrido: IaaS (para EC2 y redes), PaaS (para RDS), y FaaS (para tareas programadas o lógica liviana con Lambda).

🕳️Almacenamiento

- **S3:** Contenido multimedia estático (videos).
- **Glacier:** Backup y almacenamiento a largo plazo.
- **RDS PostgreSQL:** Base de datos relacional para usuarios, reproducciones, catálogo, etc.

Diagrama de Infraestructura

(*Imagen adjunta - Diagrama en Draw.io*)

---

2. Arquitectura de Cómputo y Escalabilidad (1.5 pt)

🐒Elección de servicios

- **ECS con Fargate:** Backend en contenedores autogestionados.
- **S3 + CloudFront:** Frontend estático distribuido globalmente.

⬛Escalabilidad

- **Auto Scaling Groups** para EC2 o ECS.
- **Load Balancer (ALB)** para distribuir el tráfico.

Orquestador

- **Amazon ECS** como gestor de contenedores (alternativa: Kubernetes/EKS).

---

3. Redes y Seguridad en la Nube (1.5 pt)

🐁Red (VPC)

- Subredes separadas:
- Pública (frontend, balanceador).
- Privada (RDS, ECS containers).
- Tablas de ruteo y Gateway NAT.

🖌️Seguridad

- **IAM Roles y Policies.**
- **Security Groups y NACLs.**
- **Secrets Manager** para credenciales.
- **VPN y IPSec** para administración segura.

🌎CDN

- **CloudFront** para distribución eficiente de videos.

---

4. Automatización e Integración Continua (1.5 pt)

CI/CD Pipeline (GitHub Actions)

```
name: CI/CD - Plataforma VOD

on:
```

```
  push:
    branches: [ "main", "release/**" ]

jobs:
  build:
    runs-on: ubuntu-latest
    steps:
    - uses: actions/checkout@v3
    - uses: actions/setup-node@v3
      with:
        node-version: '18'
    - run: npm install
    - run: npm test

  deploy:
    needs: build
    runs-on: ubuntu-latest
    steps:
    - uses: appleboy/ssh-action@master
      with:
        host: ${{ secrets.EC2_HOST }}
        username: ${{ secrets.EC2_USER }}
        key: ${{ secrets.EC2_SSH_KEY }}
        script: |
          cd /home/ec2-user/vod-backend
          git pull origin main
          npm install
          pm2 restart app.js
```

🕊️Infraestructura como Código (Terraform)

Ejemplo de Terraform:

```
resource "aws_instance" "vod_backend" {
  ami           = "ami-0c55b159cbfafe1f0"
  instance_type = "t3.micro"
  subnet_id     = aws_subnet.public_subnet.id
  vpc_security_group_ids = [aws_security_group.web_sg.id]
  user_data = <<-EOF
              #!/bin/bash
              yum update -y
              docker run -d -p 80:80 nginx
              EOF
}
```

🦬Monitoreo

- **CloudWatch** (logs, métricas).
- **SNS** para alertas.

5. Antes y Después del Ejercicio

🦡Antes:

- Servidores VPS inseguros.
- Deploy manual por FTP.
- Sin pipelines ni automatización.
- Sin monitoreo ni control de versiones.

🦜Después:

- AWS bien estructurado (VPC, S3, RDS, ECS).
- CI/CD con GitHub Actions.
- Terraform para IaC.
- GitFlow aplicado.

🐔GitFlow (Mermaid)

```
gitGraph
   commit id: "Inicio"
   branch develop
   checkout develop
   commit id: "Desarrollo inicial"
   branch feature/perfil
   checkout feature/perfil
   commit id: "Agrega perfiles"
   checkout develop
   merge feature/perfil
   branch release/v1.0
   commit id: "QA"
   checkout main
   merge release/v1.0 tag: "v1.0"
   branch hotfix/login
   commit id: "Fix login"
   merge hotfix/login
```

🦌 Entrega

- Formato: PDF.
- Nombre del archivo: `ArquitecturaCloud_M6_DevOps_LuisMontero.pdf`
- Adjuntar:
- Diagrama (.png / Draw.io).
- Código (GitHub o .zip).
- Enlace compartido: Google Drive, OneDrive o GitHub.

🐂Herramientas Utilizadas

- Terraform
- GitHub Actions
- AWS (EC2, S3, ECS, RDS, CloudWatch, IAM)
- Draw.io (diagrama)
- Mermaid (gitflow)
- Node.js + PM2 (backend)

📅**Entrega dentro del plazo suma 1 punto extra**.

🔗**Puntaje máximo: 7 puntos**