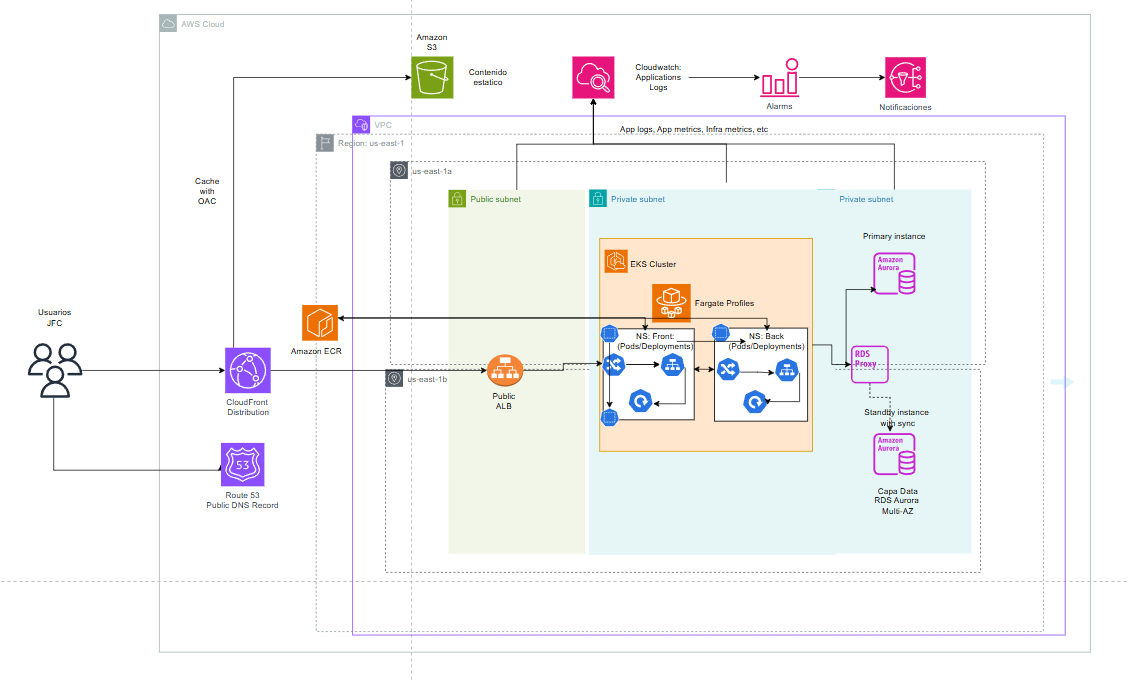
**Diagrama de la Infraestructura del e-commerce JFC en AWS**



He diseñado una arquitectura en AWS para la nueva aplicación de e-commerce de tres capas de JFC, con el objetivo de garantizar escalabilidad, alta disponibilidad, rendimiento óptimo, seguridad y control de costos, minimizando la gestión manual de infraestructura.

La arquitectura propuesta adopta un enfoque moderno basado en microservicios, servicios administrados y, donde es posible, un modelo **serverless** para la capa de cómputo.

A continuación, justifico la selección de cada componente y cómo contribuye a los requisitos del reto:

**a. Acceso de Usuarios y Distribución de Contenido**

* **Amazon Route 53**: Es el punto de entrada para los usuarios. Proporciona un servicio DNS altamente disponible y escalable, esencial para la resolución de nombres de dominio de la aplicación.
* **Amazon CloudFront (CDN)**: Se elige CloudFront para mejorar el rendimiento **y la** disponibilidad.
* **Amazon S3**: Se utiliza como un almacenamiento de objetos altamente duradero y disponible para el contenido estático del frontend de la aplicación.

**b. Capa de Cómputo (Microservicios)**

* **Amazon ECR (Elastic Container Registry)**: Un repositorio de Docker privado y gestionado que almacena las imágenes de los microservicios de Frontend y Backend.
* **VPC con Subredes Públicas y Privadas**: Esta segmentación es fundamental para la seguridad.
  + La subred pública aloja el ALB, que necesita ser accesible desde internet.
  + Las subredes privadas albergan el clúster EKS y la base de datos, asegurando que estos recursos críticos no sean directamente accesibles desde la red pública, protegiendo los datos de usuarios y transacciones.
* **Application Load Balancer (ALB)**: Se selecciona por su capacidad para manejar tráfico HTTP/HTTPS y sus características avanzadas, como el enrutamiento basado en rutas.

Distribuye las solicitudes entrantes a los Pods de EKS a través de múltiples Zonas de Disponibilidad (us-east-1a y us-east-1b), eliminando un punto único de fallo lo que garantizaque la aplicación esté disponible y funcionando en todo momento, además que permite balancear grandes cantidades de solicitudes simultáneas.

* **Amazon EKS (Elastic Kubernetes Service) con AWS Fargate**: Esta combinación es la piedra angular para la escalabilidad, rendimiento y minimización de gestión.
  + EKS es el orquestador de los microservicios, proporcionando la flexibilidad para gestionar los despliegues de Frontend y Backend de manera independiente.
  + **Serverless (Fargate)**: La elección de Fargate para los perfiles de EKS es estratégica. Elimina la necesidad de gestionar manualmente la infraestructura de EC2 subyacente. AWS Fargate provisiona y escala automáticamente los recursos de cómputo (vCPU, memoria) a nivel de Pod, lo que permite a la aplicación manejar picos de tráfico y grandes cantidades de solicitudes simultáneas de manera eficiente, y contribuye a la optimización de costos al pagar únicamente por los recursos consumidos.
  + **Escalabilidad Automática**: El uso de Kubernetes permite el escalado horizontal de Podsbasado en métricas como el uso de CPU o las solicitudes por segundo.
  + **Alta Disponibilidad**: Al distribuir los Pods a través de múltiples Zonas de Disponibilidad dentro del clúster EKS, la aplicación mantiene su operatividad incluso si una AZ experimenta una interrupción.

**c. Capa de Datos**

* **Amazon RDS Proxy**: Mantiene un pool de conexiones a la base de datos, lo que reduce la carga sobre la instancia de base de datos durante picos de tráfico y mejora la velocidad de respuesta de la aplicación, facilita la conmutación por error a una instancia en espera de Aurora sin interrupciones significativas para la aplicación.
* **Amazon Aurora Multi-AZ**: Se elige Aurora por ser una base de datos confiable y escalable, diseñada para cargas de trabajo de producción y e-commerce.
  + **Servicio Administrado**: AWS gestiona el parcheo, las copias de seguridad y la replicación, reduciendo la carga operativa.

**d. Observabilidad y Gestión**

* **Amazon CloudWatch**: Es el servicio para la observabilidad y monitoreo donde se centralizan logs de aplicaciones (desde EKS/Fargate), métricas de infraestructura (ALB, EKS, Aurora) y métricas de aplicación, además la configuración de **CloudWatch Alarms** basadas en umbrales de métricas (CPU, memoria, latencia, errores) permite la detección proactiva de fallos y una respuesta rápida, garantizando la salud de la aplicación.

**Conclusión**

Esta arquitectura no solo cumple con todos los requisitos mínimos de la prueba técnica, sino que también incorpora principios de diseño avanzados de AWS para garantizar una solución robusta, eficiente y escalable a largo plazo. La combinación de EKS con Fargate, Aurora Multi-AZ, CloudFront y un sólido sistema de observabilidad proporciona una base sólida para el crecimiento futuro de la aplicación de e-commerce de JFC, con una carga operativa reducida y costos optimizados. Estoy convencido de que esta propuesta no solo satisface las necesidades actuales del negocio, sino que también está preparada para futuras expansiones.