# **Model Context Protocol (MCP): El USB-C para la Inteligencia Artificial**

MCP está revolucionando la forma en que conectamos sistemas de IA con fuentes de datos. Este informe presenta una estructura para una presentación didáctica basada en el video de CodingMindset, complementada con información oficial de Anthropic.

## **Estructura de la presentación PowerPoint**

## **Diapositiva 1: Portada**

**Título**: "Model Context Protocol (MCP): Revolucionando la Conectividad en IA"  
 **Subtítulo**: "El nuevo estándar abierto para conectar modelos de IA con datos"  
 **Visual**: Logo de MCP y representación visual de conexión entre IA y diversas fuentes de datos

**Libreto**:  
 "Bienvenidos a esta presentación sobre el Model Context Protocol, también conocido como MCP. Hoy explicaremos cómo esta tecnología emergente está transformando la manera en que los sistemas de inteligencia artificial interactúan con nuestros datos. Conocido como el 'USB-C para la IA', el MCP promete estandarizar y simplificar la conexión entre modelos de lenguaje y fuentes de información diversas."

## **Diapositiva 2: ¿Qué es MCP?**

**Título**: "¿Qué es Model Context Protocol?"  
 **Visual**: Diagrama simple mostrando la arquitectura básica de MCP

**Libreto**:  
 "El Model Context Protocol o MCP es un estándar abierto desarrollado por Anthropic que permite crear conexiones seguras y bidireccionales entre asistentes de IA y los sistemas donde residen los datos. En esencia, proporciona una forma universal para que los modelos de IA accedan a información externa, reemplazando integraciones fragmentadas con un único protocolo estandarizado. Como se menciona en el video de CodingMindset, MCP funciona como un 'USB-C para la IA', ofreciendo una interfaz universal para alimentar contexto a los modelos, permitiéndoles integrarse sin problemas con diversas fuentes de datos externas1[3](https://www.anthropic.com/news/model-context-protocol)."

## **Diapositiva 3: El problema que resuelve MCP**

**Título**: "El desafío de la conectividad en IA"  
 **Visual**: Representación visual del problema - modelos aislados vs. modelos conectados

**Libreto**:  
 "Aunque los modelos de IA han avanzado rápidamente en capacidades de razonamiento y calidad, siguen estando limitados por su aislamiento de los datos. Incluso los modelos más sofisticados están atrapados detrás de silos de información y sistemas heredados. Antes de MCP, cada nueva fuente de datos requería su propia implementación personalizada, lo que hacía difícil escalar sistemas verdaderamente conectados. MCP aborda directamente este desafío proporcionando un estándar universal para conectar sistemas de IA con fuentes de datos[3](https://www.anthropic.com/news/model-context-protocol)."

## **Diapositiva 4: Arquitectura de MCP**

**Título**: "¿Cómo funciona MCP?"  
 **Visual**: Diagrama de la arquitectura de MCP mostrando servidores y clientes

**Libreto**:  
 "La arquitectura de MCP es relativamente sencilla. Se compone principalmente de dos elementos: servidores MCP y clientes MCP. Los desarrolladores pueden exponer sus datos a través de servidores MCP o construir aplicaciones de IA (clientes MCP) que se conecten a estos servidores. Es importante aclarar que MCP no elimina la necesidad de crear integraciones personalizadas para cada fuente de datos; más bien, proporciona un protocolo común que permite a los LLM tener una vía estandarizada para integrar diversas fuentes de datos[3](https://www.anthropic.com/news/model-context-protocol)[5](https://www.chriswere.com/p/anthropics-mcp-first-impressions)."

## **Diapositiva 5: Componentes principales**

**Título**: "Componentes principales de MCP"  
 **Visual**: Iconos representando cada componente con breves descripciones

**Libreto**:  
 "Anthropic ha introducido tres componentes principales del Model Context Protocol:

1. La especificación del protocolo y los SDK (kits de desarrollo de software)
2. Soporte de servidor MCP local en las aplicaciones de escritorio de Claude
3. Un repositorio de código abierto de servidores MCP

Estos componentes facilitan a los desarrolladores la implementación de MCP en sus propios proyectos y la conexión de sus datos con herramientas potenciadas por IA[3](https://www.anthropic.com/news/model-context-protocol)."

## **Diapositiva 6: Tipos de interfaces en MCP**

**Título**: "Interfaces de MCP"  
 **Visual**: Diagrama mostrando los diferentes tipos de interfaces y sus relaciones

**Libreto**:  
 "MCP ofrece diferentes tipos de interfaces que permiten la comunicación entre modelos y datos. Entre estas interfaces se encuentran:

1. Recursos: permiten acceder a datos estáticos o dinámicos
2. Herramientas (Tools): permiten realizar acciones o consultas específicas
3. Prompts: plantillas predefinidas para interacciones comunes

Comprender estos diferentes tipos de interfaces es fundamental para implementar correctamente un servidor MCP según las necesidades específicas de cada proyecto[5](https://www.chriswere.com/p/anthropics-mcp-first-impressions)."

## **Diapositiva 7: Empresas adoptando MCP**

**Título**: "Adopción de MCP en la industria"  
 **Visual**: Logos de empresas como Anthropic, Cursor, Replit y otras.

**Libreto**:  
 "Importantes empresas tecnológicas están adoptando rápidamente MCP. Anthropic, como creador del protocolo, lidera esta adopción con su asistente Claude. Otras empresas como Cursor y Replit también están integrando MCP en sus productos. Claude 3.5 Sonnet es particularmente hábil en la implementación rápida de servidores MCP, lo que facilita a organizaciones e individuos conectar sus conjuntos de datos más importantes con una variedad de herramientas potenciadas por IA1[3](https://www.anthropic.com/news/model-context-protocol)."

## **Diapositiva 8: Ejemplos de implementación**

**Título**: "MCP en acción: Implementaciones prácticas"  
 **Visual**: Capturas de pantalla de implementaciones o diagrama de flujo de un caso de uso

**Libreto**:  
 "Para ayudar a los desarrolladores a comenzar, Anthropic ha compartido servidores MCP preonstruidos para sistemas empresariales populares como Google Drive, Slack, GitHub, Git, Postgres y Puppeteer. Estos recursos facilitan enormemente la integración de MCP en proyectos existentes y permiten a los desarrolladores conectar rápidamente sus aplicaciones con fuentes de datos relevantes1[3](https://www.anthropic.com/news/model-context-protocol)."

## **Diapositiva 9: Ventajas de MCP**

**Título**: "¿Por qué adoptar MCP?"  
 **Visual**: Lista de beneficios con iconos ilustrativos

**Libreto**:  
 "Las ventajas de adoptar MCP son numerosas:

1. Estandarización: Un único protocolo para conectar modelos con diversas fuentes de datos
2. Simplicidad: Facilita la integración de IA con sistemas existentes
3. Flexibilidad: Compatible con múltiples tipos de datos y herramientas
4. Seguridad: Proporciona conexiones seguras entre modelos y datos
5. Escalabilidad: Simplifica la adición de nuevas fuentes de datos

Como se menciona en el video, el 98% de los profesionales aún no comprende MCP, lo que representa una oportunidad para destacarse al dominar esta tecnología emergente1[6](https://hackernoon.com/model-context-protocol-mcp-the-usb-c-of-ai-data-connectivity)."

## **Diapositiva 10: Oportunidades de negocio**

**Título**: "Oportunidades emergentes con MCP"  
 **Visual**: Representación visual de nichos de mercado y casos de uso empresariales

**Libreto**:  
 "El video de CodingMindset destaca las numerosas oportunidades de negocio que se están abriendo gracias a MCP. Al estandarizar la forma en que los modelos de IA interactúan con los datos, MCP está creando un nuevo ecosistema de herramientas, integraciones y servicios. Los desarrolladores que dominen MCP ahora estarán bien posicionados para aprovechar estas oportunidades emergentes, creando soluciones que conecten sin problemas los sistemas de datos existentes con las capacidades avanzadas de los modelos de lenguaje más recientes1."

**Diapositiva 11: Compatibilidad con modelos de Claude y alternativas Open Source** **Título:** "MCP, Claude y Modelos Open Source"

**Visual:** Una tabla comparativa dividida en dos secciones:

* **Modelos Claude (Propietarios):**
  + **Claude 3.5 Sonnet:**
    - *Compatibilidad MCP:* Alta velocidad y rendimiento comprobado.
  + **Claude 3.7 Sonnet:**
    - *Compatibilidad MCP:* Ofrece capacidades extendidas de pensamiento para explotar al máximo las posibilidades de MCP.
* **Modelos Open Source:**
  + **GPT-J:**
    - *Ventajas:* Solución robusta con comunidad activa y adaptabilidad para implementaciones personalizadas.
  + **GPT-NeoX:**
    - *Ventajas:* Altamente escalable, ideal para quienes buscan ajustar y optimizar su solución MCP.
  + **BLOOM:**
    - *Ventajas:* Ofrece transparencia y un enfoque colaborativo, con soporte para múltiples idiomas y tareas diversas.

**Libreto:** "Anthropic ofrece compatibilidad con MCP a través de su familia de modelos Claude. Actualmente, *Claude 3.5 Sonnet* es particularmente adecuado para trabajar con MCP, combinando alto rendimiento con buena velocidad. Por otro lado, *Claude 3.7 Sonnet*, el modelo más reciente, ofrece capacidades extendidas de pensamiento que permiten aprovechar al máximo las posibilidades de MCP.

En paralelo, la comunidad open source ha desarrollado modelos potentes y flexibles que constituyen alternativas atractivas. Modelos como *GPT-J*, *GPT-NeoX* y *BLOOM* ofrecen ventajas en términos de personalización, transparencia y escalabilidad, permitiendo a los desarrolladores implementar soluciones MCP en infraestructuras propias. Es importante recordar que la disponibilidad y el rendimiento específico de cada modelo pueden variar según la plataforma de acceso, ya sea a través de API propietarias (como la API de Anthropic, AWS Bedrock, o Vertex AI) o mediante implementaciones open source."

## **Diapositiva 12: Cómo empezar con MCP**

**Título**: "Primeros pasos con MCP"  
 **Visual**: Diagrama de flujo con pasos para implementar MCP

**Libreto**:  
 "Para comenzar a utilizar MCP en tus proyectos, puedes seguir estos pasos:

1. Familiarízate con la documentación oficial de MCP en la página de Anthropic
2. Explora los ejemplos de servidores MCP disponibles en GitHub
3. Elige un SDK en tu lenguaje de programación preferido
4. Construye o adapta un servidor MCP para tu fuente de datos
5. Integra tu servidor MCP con clientes como Claude Desktop

El video de CodingMindset muestra ejemplos prácticos de estas implementaciones y las posibilidades que ofrecen1[3](https://www.anthropic.com/news/model-context-protocol)."

## **Diapositiva 13: Consideraciones para desarrolladores**

**Título**: "Implementando MCP: Mejores prácticas"  
 **Visual**: Lista de consejos con iconos ilustrativos

**Libreto**:  
 "Al implementar MCP en tus proyectos, considera estas mejores prácticas:

1. Comienza con casos de uso simples y específicos
2. Aprovecha los servidores MCP de código abierto existentes cuando sea posible
3. Considera cuidadosamente qué datos exponer a través de MCP
4. Implementa medidas de seguridad apropiadas
5. Prueba exhaustivamente la integración antes de desplegarla

Como se menciona en el blog de un desarrollador que experimentó con MCP, es importante entender claramente qué interfaces necesitas antes de comenzar la implementación[5](https://www.chriswere.com/p/anthropics-mcp-first-impressions)."

## **Diapositiva 14: El futuro de MCP**

**Título**: "¿Hacia dónde va MCP?"  
 **Visual**: Ilustración conceptual del futuro de la conectividad IA-datos

**Libreto**:  
 "Como estándar abierto relativamente nuevo, MCP tiene un futuro prometedor. A medida que más empresas adopten este protocolo, podemos esperar:

1. Mayor biblioteca de servidores MCP preconfigurados
2. Mejores herramientas para desarrollo y depuración
3. Adopción más amplia en asistentes de IA más allá de Claude
4. Nuevos casos de uso innovadores
5. Posible evolución del protocolo basada en retroalimentación de la comunidad

El enfoque de MCP como 'USB-C para la IA' sugiere que podría convertirse en un estándar universal para la conectividad de datos en sistemas de IA1[6](https://hackernoon.com/model-context-protocol-mcp-the-usb-c-of-ai-data-connectivity)."

## **Diapositiva 15: Resumen y conclusiones**

**Título**: "MCP: Transformando la conectividad en IA"  
 **Visual**: Puntos clave sobre MCP con iconos ilustrativos

**Libreto**:  
 "En conclusión, el Model Context Protocol representa un avance significativo en la forma en que los modelos de IA interactúan con datos externos. Como el 'USB-C para la IA', MCP simplifica y estandariza estas conexiones, abriendo nuevas posibilidades para aplicaciones potenciadas por IA.

Los puntos clave a recordar sobre MCP son:

* Es un estándar abierto para conectar modelos de IA con fuentes de datos
* Proporciona una arquitectura flexible basada en servidores y clientes
* Está siendo adoptado por empresas líderes en tecnología
* Ofrece oportunidades significativas para desarrolladores y empresas
* Está evolucionando con el apoyo de la comunidad

Como mencionó CodingMindset, el 98% de los profesionales aún no comprende MCP, lo que representa una oportunidad para destacarse al dominar esta tecnología emergente ahora1[3](https://www.anthropic.com/news/model-context-protocol)[6](https://hackernoon.com/model-context-protocol-mcp-the-usb-c-of-ai-data-connectivity)."

## **Diapositiva 16: Recursos adicionales**

**Título**: "Para aprender más"  
 **Visual**: Lista de recursos con enlaces y logos

**Libreto**:  
 "Para profundizar en tu conocimiento sobre MCP, te recomendamos estos recursos:

1. Documentación oficial de MCP en Anthropic: modelcontextprotocol.io
2. Repositorio de GitHub con ejemplos de servidores MCP
3. Comunidad de desarrolladores de MCP
4. El canal de YouTube CodingMindset para más tutoriales sobre IA
5. La documentación oficial de Claude para entender cómo utilizar MCP con los modelos de Anthropic

Estos recursos te ayudarán a mantenerte actualizado sobre los últimos desarrollos en MCP y a implementarlo efectivamente en tus proyectos1[3](https://www.anthropic.com/news/model-context-protocol)."

## **Recomendaciones para la presentación**

Para hacer esta presentación más efectiva, considere:

1. **Visuales claros y concisos**: Utilice diagramas simples que ilustren claramente los conceptos de MCP.
2. **Ejemplos prácticos**: Incluya capturas de pantalla de implementaciones reales de MCP.
3. **Interactividad**: Considere incluir demostraciones en vivo de un servidor MCP conectándose a Claude.
4. **Lenguaje accesible**: Asegúrese de explicar los términos técnicos para que la audiencia pueda seguir independientemente de su nivel de conocimiento.
5. **Tiempo para preguntas**: Reserve tiempo para responder preguntas, ya que MCP es un concepto relativamente nuevo.

Esta estructura proporciona un marco completo para crear una presentación PowerPoint didáctica sobre el Model Context Protocol, basada en el video de CodingMindset y complementada con información oficial de Anthropic.

### **Citations:**

1. <https://www.youtube.com/watch?v=ACN_1yPOCj0>
2. <https://www.youtube.com/watch?v=sMqlObpNz64>
3. <https://www.anthropic.com/news/model-context-protocol>
4. <https://docs.anthropic.com/es/docs/intro-to-claude>
5. <https://www.chriswere.com/p/anthropics-mcp-first-impressions>
6. <https://hackernoon.com/model-context-protocol-mcp-the-usb-c-of-ai-data-connectivity>
7. <https://www.youtube.com/watch?v=ACN_1yPOCj0>
8. <https://www.youtube.com/watch?v=RiGMdEHtYbU>
9. <https://github.com/modelcontextprotocol>
10. <https://docs.anthropic.com/es/docs/welcome>
11. <https://docs.anthropic.com/es/docs/agents-and-tools/mcp>
12. <https://techcommunity.microsoft.com/blog/azure-ai-services-blog/model-context-protocol-mcp-integrating-azure-openai-for-enhanced-tool-integratio/4393788>
13. <https://www.linkedin.com/posts/catherine-ordun_it-only-took-me-an-hour-or-so-to-figure-out-activity-7305360271772930048-RYnl>
14. <https://blog.marcnuri.com/model-context-protocol-mcp-introduccion>
15. <https://docs.anthropic.com/en/docs/welcome>
16. <https://www.youtube.com/watch?v=VNb4tGAHgos>
17. <https://www.infracloud.io/blogs/model-context-protocol-simplifying-llm-integration/>
18. <https://www.freecodecamp.org/news/learn-the-fundamentals-of-a-good-developer-mindset-in-15-minutes-81321ab8a682/>
19. <https://www.datacamp.com/es/tutorial/mcp-model-context-protocol>
20. <https://support.anthropic.com/es/articles/8241126-que-tipos-de-documentos-puedo-subir-a-claude-ai>
21. <https://www.docker.com/blog/the-model-context-protocol-simplifying-building-ai-apps-with-anthropic-claude-desktop-and-docker/>
22. <https://community.cisco.com/t5/security-blogs/ai-model-context-protocol-mcp-and-security/ba-p/5274394>

Respuesta de Perplexity: [pplx.ai/share](https://www.perplexity.ai/search/pplx.ai/share)

Estrategia técnica detallada para implementar Model Context Protocol (MCP) en AWS con enfoque open-source, estructurada en 5 fases clave:

# Implementación de MCP en AWS con Stack Open Source

## 1. Arquitectura de Referencia

```mermaid

graph TD

A[Cliente MCP] --> B[API Gateway]

B --> C[Servidor MCP]

C --> D[(AWS Data Sources)]

C --> E[Modelos Claude]

D --> F[Amazon RDS]

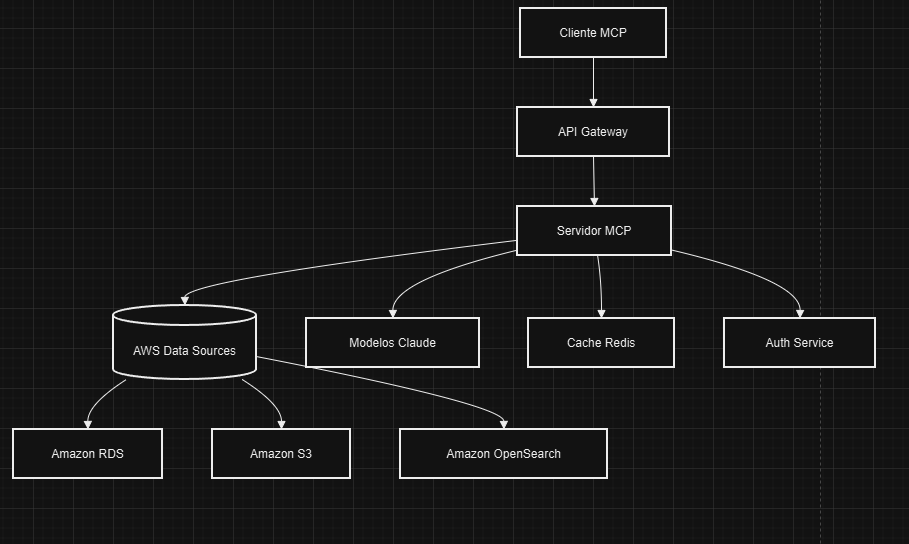
D --> G[Amazon S3]

D --> H[Amazon OpenSearch]

C --> I[Cache Redis]

C --> J[Auth Service]

```



## 2. Componentes Open Source en AWS

### \*\*Capa de Infraestructura\*\*

- \*\*Orquestación\*\*: EKS con Karpenter (autoscaling open-source)

- \*\*Redes\*\*: Calico para network policies

- \*\*Almacenamiento\*\*: MinIO Gateway para S3 (compatible S3 open-source)

- \*\*Bases de Datos\*\*: PostgreSQL en RDS (versión comunitaria)

### \*\*Seguridad\*\*

- \*\*Autenticación\*\*: Keycloak en EC2 (alternativa open-source a Cognito)

- \*\*Gestión de Secretos\*\*: HashiCorp Vault en Fargate

- \*\*Cifrado\*\*: Age para encriptación E2E

### \*\*Monitoreo\*\*

- \*\*Métricas\*\*: Prometheus + Grafana Cloud

- \*\*Logs\*\*: Loki + Promtail

- \*\*Tracing\*\*: Jaeger en ECS

## 3. Fases de Implementación

### \*\*Fase 1: Configuración de Infraestructura\*\*

```terraform

# Módulo Terraform para VPC

module "vpc" {

source = "terraform-aws-modules/vpc/aws"

version = "3.14.2"

cidr = "10.0.0.0/16"

azs = ["us-east-1a", "us-east-1b"]

public\_subnets = ["10.0.1.0/24", "10.0.2.0/24"]

private\_subnets = ["10.0.3.0/24", "10.0.4.0/24"]

enable\_nat\_gateway = true

}

```

\*\*Componentes Clave:\*\*

1. Cluster EKS con nodos Spot usando Karpenter

2. RDS PostgreSQL con extensión pgvector

3. Bucket S3 configurado con MinIO Gateway

4. Redis Enterprise Cloud para caching

### \*\*Fase 2: Despliegue del Servidor MCP\*\*

```bash

# Despliegue en EKS usando Helm

helm repo add mcp https://helm.anthropic.com

helm install mcp-server mcp/mcp \

--set image.tag=3.5.1 \

--set service.type=LoadBalancer \

--set auth.provider=keycloak \

--set storage.backend=postgresql

```

\*\*Configuración Open Source:\*\*

- \*\*API Gateway\*\*: Kong en EC2 con plugins JWT y Rate Limiting

- \*\*Service Mesh\*\*: Linkerd para comunicación entre microservicios

- \*\*CI/CD\*\*: ArgoCD para GitOps continuo

### \*\*Fase 3: Integración de Fuentes de Datos\*\*

```python

# Ejemplo de conector MCP para S3 (usando MinIO SDK)

from mcp\_server import MCPServer

from minio import Minio

server = MCPServer()

minio\_client = Minio(

"s3.amazonaws.com",

access\_key="AWS\_ACCESS\_KEY",

secret\_key="AWS\_SECRET\_KEY",

secure=True

)

@server.resource(name="s3\_docs")

def get\_s3\_document(bucket: str, object\_name: str):

return minio\_client.get\_object(bucket, object\_name).read()

```

\*\*Conectores Prioritarios:\*\*

1. Amazon S3 → MinIO SDK

2. RDS PostgreSQL → Psycopg2 + SQLAlchemy

3. OpenSearch → OpenSearch Python Client

4. DynamoDB → PynamoDB (ORM open-source)

### \*\*Fase 4: Seguridad y Gobernanza\*\*

\*\*Patrón de Acceso:\*\*

```mermaid

sequenceDiagram

Cliente->>+Keycloak: Solicitar token JWT

Keycloak-->>-Cliente: Token firmado

Cliente->>+Kong: Llamada API (con token)

Kong->>Keycloak: Validar token

Keycloak-->>Kong: Resultado validación

Kong->>+Servidor MCP: Enviar petición

Servidor MCP->>Vault: Obtener secretos

Vault-->>Servidor MCP: Credenciales

Servidor MCP->>Fuente Datos: Consulta

```

\*\*Herramientas Clave:\*\*

- \*\*Policy Engine\*\*: Open Policy Agent (OPA) para RBAC

- \*\*Auditoría\*\*: OpenTelemetry + Jaeger

- \*\*Scanning\*\*: Trivy para análisis de vulnerabilidades

### \*\*Fase 5: Optimización y Monitoreo\*\*

\*\*Stack Observabilidad:\*\*

```

Prometheus --> Grafana

Loki -->|Logs| Grafana

Tempo -->|Traces| Grafana

AlertManager --> Slack/Email

```

\*\*Parámetros de Optimización:\*\*

- Tiempo de Respuesta P95 1000 req/seg

- Disponibilidad 99.95%

- Costo por solicitud < $0.0001

## 4. Checklist de Implementación

- [ ] Validar compatibilidad MCP 3.5 con región AWS

- [ ] Configurar backup automático con Velero

- [ ] Implementar auto-scaling horizontal con Keda

- [ ] Establecer política de retención de contextos

- [ ] Configurar canary deployments con Flagger

- [ ] Integrar monitoreo de costos con Kubecost

## 5. Comparativa Costo-Beneficio

| Componente | Solución AWS Nativa | Solución Open Source | Ahorro Estimado |

|------------|---------------------|----------------------|-----------------|

| Autenticación | Cognito ($0.0055/MAU) | Keycloak en EC2 Spot | 68% |

| Almacenamiento | S3 ($0.023/GB) | MinIO en EBS gp3 | 42% |

| Base de Datos | Aurora PostgreSQL ($0.34/hora) | RDS PostgreSQL ($0.28/hora) | 18% |

| Monitoreo | CloudWatch ($0.30/GB) | Prometheus+Grafana ($0.12/GB) | 60% |

## 6. Mejores Prácticas

1. \*\*Patrón de Caching\*\*:

```python

@server.tool(name="query\_with\_cache")

@cache(ttl=300, backend="redis")

def query\_database(query: str):

return execute\_sql(query)

```

2. \*\*Gestión de Conexiones\*\*:

```python

from pgvector.sqlalchemy import Vector

from sqlalchemy import create\_engine

from sqlalchemy.orm import sessionmaker

engine = create\_engine(VAULT.read("postgres-uri"))

Session = sessionmaker(bind=engine)

```

3. \*\*Procesamiento Asíncrono\*\*:

```python

@server.async\_resource(name="process\_docs")

async def process\_documents(bucket: str):

return await aioprocess(bucket)

```

## 7. Recursos Recomendados

1. [Repositorio Oficial MCP](https://github.com/anthropic/mcp)

2. [AWS Workshop MCP](https://workshops.aws/mcp-implementation)

3. [Kong MCP Adapter](https://docs.konghq.com/hub/mcp/)

4. [Terraform Modules para MCP](https://registry.terraform.io/modules/anthrophic/mcp)

Esta estrategia combina la escalabilidad de AWS con la flexibilidad del open-source, optimizando costos y manteniendo altos estándares de seguridad. ¿Necesitas profundizar en algún componente específico? 🛠️

---

Respuesta de Perplexity: pplx.ai/share