Tarjetas de Estudio: Databricks Certified Generative AI Engineer Associate

🧩 1. Arquitectura RAG (Retrieval-Augmented Generation)

- **¿Qué es RAG?**
    - Es una arquitectura que combina recuperación de información y generación de texto para proporcionar respuestas más precisas y contextuales.
- **Componentes principales:**
    - **Recuperación:** Busca documentos relevantes utilizando búsquedas semánticas.
    - **Generación:** Utiliza modelos de lenguaje para generar respuestas basadas en los documentos recuperados.
- **Ventajas:**
    - Permite respuestas más precisas sin necesidad de reentrenar modelos.

🧱 2. Preparación de Datos para GenAI

- **Segmentación (Chunking):**
    - Dividir documentos en fragmentos manejables para facilitar la recuperación y el procesamiento.
- **Embeddings:**
    - Representaciones vectoriales de texto que capturan el significado semántico.
- **Almacenamiento:**
    - Utilización de bases de datos vectoriales, como Mosaic AI Vector Search, para almacenar y recuperar embeddings.

🧠 3. Ingeniería de Prompts

- **Técnicas comunes:**
    - **Zero-shot:** El modelo responde sin ejemplos previos.
    - **Few-shot:** Se proporcionan algunos ejemplos para guiar la respuesta.
    - **Chain-of-thought:** Se anima al modelo a razonar paso a paso.LinkedIn+2LinkedIn+2Databricks+2
- **Consejos:**
    - Sé claro y específico en tus instrucciones.
    - Evita ambigüedades para obtener respuestas más precisas.

⚖️ 4. Fine-tuning vs. RAG

- **Fine-tuning:**
    - Reentrenamiento del modelo con datos específicos.
    - Requiere más recursos y tiempo.
- **RAG:**
    - Utiliza recuperación de información para proporcionar contexto.
    - Más eficiente y flexible para incorporar nueva información.

🧪 5. Evaluación y Monitoreo de Modelos

- **Evaluación:**
    - Medición de la precisión, relevancia y coherencia de las respuestas generadas.
- **Monitoreo:**

- Seguimiento del rendimiento del modelo en producción.
- Detección de desviaciones o disminuciones en la calidad de las respuestas.Udemy+2Databricks+2Databricks+2

🛠️ 6. Herramientas Clave en Databricks

- **Mosaic AI Playground:**
  - Entorno para experimentar con modelos de lenguaje y prompts.
- **Mosaic AI Vector Search:**
  - Motor de búsqueda semántica para recuperar información relevante.
- **MLflow:**
  - Plataforma para gestionar el ciclo de vida de modelos de machine learning.
- **LangChain:**
  - Framework para construir aplicaciones que integran modelos de lenguaje con otras herramientas y fuentes de datos.
- **Delta Lake:**
  - Almacenamiento de datos confiable y escalable que permite manejar grandes volúmenes de información.

🧠 Sistemas de IA Compuestos (Compound AI Systems)

Objetivo principal: Aprender a pasar de usar un modelo aislado (como un LLM solo) a construir sistemas más complejos formados por varios componentes que trabajan juntos.

Conceptos clave:

- **Sistemas compuestos de IA**: Son soluciones que combinan varios modelos y herramientas para cumplir tareas complejas.
- **Componentes comunes**: Intent detection (detectar qué quiere el usuario), tareas específicas, y *pipelines* (secuencia de pasos para resolver algo).
- **Intención, tarea y cadena**:
    - **Intención**: Qué quiere lograr el usuario.
    - **Tarea**: Qué acción concreta se debe ejecutar.
    - **Cadena (chain)**: Conjunto de pasos o prompts conectados para lograr un objetivo.

Lo que se aprende:

- Cómo identificar la intención de un usuario.
- Cómo crear cadenas de prompts (prompt chains) para resolver tareas complejas paso a paso.
- Diferencias entre:
    - Una **tarea individual** para un LLM (por ejemplo, generar un resumen).
    - Una **cadena completa** de tareas (por ejemplo, extraer info, clasificarla y luego resumirla).

🔗 Diseño de Sistemas Compuestos y Construcción de Cadenas

- Clasificar la intención del usuario es clave para saber qué pasos (chain) usar.

- Las cadenas deben ser diseñadas cuidadosamente para que cada paso tenga sentido y esté alineado con el objetivo final.

- Importa el orden, la lógica y cómo se conectan los prompts entre sí.

🧩 Construcción de Cadenas de Razonamiento Multietapa (Multi-stage Reasoning Chains)

Objetivo principal: Aprender a resolver problemas complejos usando pasos múltiples, como lo haría un humano.

Conceptos clave:

- **Razonamiento multietapa**: Resolver algo descomponiéndolo en partes más pequeñas y razonando en varias etapas.

- **Frameworks de composición**: Herramientas como LangChain permiten crear estos sistemas fácilmente.

- **LangChain**: Librería popular para construir cadenas LLM, usando pasos como herramientas, memoria, y agentes.

- **Databricks y LLMs**: Databricks ofrece herramientas para construir y escalar estas cadenas, integrando datos, código y modelos.

Ventajas:

- Mejor precisión.

- Más control sobre cada paso del razonamiento.

- Refleja mejor cómo los humanos resuelven problemas (paso a paso).

FLASHCARDS – Generative AI Application Development (Databricks)

🟦 **Flashcard 1**
Pregunta: ¿Qué es un sistema de IA compuesto (Compound AI System)?
Respuesta: Es un sistema que combina múltiples modelos, herramientas y pasos para resolver tareas complejas de manera integrada.

🟦 **Flashcard 2**
Pregunta: ¿Cuál es la diferencia entre un modelo LLM y un sistema de IA compuesto?
Respuesta: Un modelo LLM responde tareas individuales, mientras que un sistema compuesto encadena varios pasos y herramientas para lograr una solución completa.

🟦 **Flashcard 3**
Pregunta: ¿Qué es una cadena (chain) en un sistema de IA?
Respuesta: Es una secuencia de prompts o pasos conectados que resuelven una tarea compleja.

🟦 **Flashcard 4**
Pregunta: ¿Qué es la intención (intent) en un sistema de IA?
Respuesta: Es el objetivo o necesidad que tiene el usuario al interactuar con el sistema.

🟦 **Flashcard 5**
Pregunta: ¿Qué es la clasificación de intención?
Respuesta: Es el proceso de identificar qué quiere lograr el usuario para elegir la cadena adecuada de pasos.

🟦 **Flashcard 6**
Pregunta: ¿Qué es una tarea LLM?
Respuesta: Es una acción específica que realiza el modelo, como traducir, resumir o responder.

🟦 **Flashcard 7**
Pregunta: ¿Qué es una cadena basada en LLM (LLM-based chain)?

Respuesta: Es una serie de tareas conectadas, cada una ejecutada por un modelo LLM, para cumplir una intención más compleja.

🟦 **Flashcard 8**
Pregunta: ¿Qué es el razonamiento multietapa (multi-stage reasoning)?
Respuesta: Es un enfoque donde se divide un problema en varios pasos lógicos para resolverlo de manera más precisa.

🟦 **Flashcard 9**
Pregunta: ¿Qué es LangChain?
Respuesta: Es un framework de Python que permite construir cadenas de razonamiento y sistemas LLM personalizados fácilmente.

🟦 **Flashcard 10**
Pregunta: ¿Qué ofrece Databricks para trabajar con sistemas de IA compuestos?
Respuesta: Herramientas para integrar datos, modelos y cadenas LLM de forma escalable en entornos colaborativos.

🟦 **Flashcard 11**
Pregunta: ¿Por qué es útil usar frameworks de composición como LangChain?
Respuesta: Porque facilitan el desarrollo, el mantenimiento y la reutilización de componentes de IA en cadenas complejas.

🟦 **Flashcard 12**
Pregunta: ¿Cómo se parece el razonamiento multietapa al pensamiento humano?
Respuesta: Porque, como las personas, descompone problemas grandes en pasos lógicos que se resuelven uno a uno.

FLASHCARDS – Evaluación de Aplicaciones de GenAI

🟦 **Flashcard 1**
Pregunta: ¿Por qué es importante evaluar una aplicación GenAI?
Respuesta: Para asegurar que sea útil, legal, ética y no cause daños.

🟦 **Flashcard 2**
Pregunta: ¿Qué partes de un sistema GenAI deben evaluarse?
Respuesta: El input, la salida, y el comportamiento del sistema ante distintos usuarios.

🟦 **Flashcard 3**
Pregunta: ¿Qué tipo de riesgos legales existen en GenAI?
Respuesta: Uso de datos sin licencia, violación de privacidad o derechos de autor.

🟦 **Flashcard 4**
Pregunta: ¿Qué es un "guardrail" en IA generativa?
Respuesta: Un filtro o límite que protege al sistema de dar respuestas dañinas o inadecuadas.

🟦 **Flashcard 5**
Pregunta: ¿Qué riesgos puede causar el mal uso por parte de usuarios?
Respuesta: Pueden intentar generar spam, contenido tóxico o manipular la IA.

🟦 **Flashcard 6**
Pregunta: ¿Por qué es difícil evaluar salidas generativas?
Respuesta: Porque pueden ser muy variadas, no siempre hay una "respuesta correcta".

🟦 **Flashcard 7**
Pregunta: ¿Qué es el "data licensing" en IA generativa?
Respuesta: Asegurarse de que los datos usados están autorizados legalmente.

FLASHCARDS – Seguridad y Gobernanza de GenAI

🟦 **Flashcard 1**
Pregunta: ¿Por qué es clave asegurar y gobernar una aplicación de GenAI?
Respuesta: Para proteger datos, evitar usos indebidos y cumplir normas legales y éticas.

🟦 **Flashcard 2**
Pregunta: ¿Qué hace difícil la gobernanza de GenAI?
Respuesta: Las salidas son impredecibles, los riesgos son múltiples y se necesita control constante.

🟦 **Flashcard 3**
Pregunta: ¿Cuál es el rol de data scientists y desarrolladores en la seguridad de GenAI?
Respuesta: Diseñar sistemas seguros, aplicar filtros, controlar permisos y colaborar en gobernanza.

🟦 **Flashcard 4**
Pregunta: ¿Qué es Unity Catalog en Databricks?
Respuesta: Es la herramienta de gobernanza de datos que permite controlar accesos, rastrear el uso y auditar acciones.

🟦 **Flashcard 5**
Pregunta: ¿Para qué sirve el **lineage** en Unity Catalog?
Respuesta: Para ver el origen y destino de los datos, ayudando a entender cómo se procesan.

🟦 **Flashcard 6**
Pregunta: ¿Qué es el **Safety Filter**?
Respuesta: Un filtro que bloquea respuestas peligrosas, ofensivas o inadecuadas en aplicaciones GenAI.

🟦 **Flashcard 7**
Pregunta: ¿Qué es **Llama Guard**?
Respuesta: Un framework de seguridad para proteger y controlar el uso de modelos generativos (LLMs).

FLASHCARDS – Técnicas de Evaluación en GenAI

🟦 **Flashcard 1**
Pregunta: ¿Cuál es una gran diferencia entre evaluar ML tradicional y LLMs?
Respuesta: LLMs generan texto libre, así que es más difícil medir objetivamente la calidad.

🟦 **Flashcard 2**
Pregunta: ¿Qué es la *perplexity*?
Respuesta: Una medida de qué tan bien el modelo predice la próxima palabra. Menor perplexity = mejor.

🟦 **Flashcard 3**
Pregunta: ¿Qué mide la métrica *toxicity*?
Respuesta: Si el contenido generado por el modelo es ofensivo, dañino o inadecuado.

🟦 **Flashcard 4**
Pregunta: ¿Qué es BLEU y para qué sirve?
Respuesta: Es una métrica para evaluar traducción automática comparando con textos de referencia.

🟦 **Flashcard 5**
Pregunta: ¿Qué es ROUGE y en qué se usa?
Respuesta: Es una métrica que mide calidad de resúmenes, viendo si contienen info clave.

🟦 **Flashcard 6**
Pregunta: ¿Qué significa *LLM-as-a-judge*?
Respuesta: Usar un LLM para evaluar la calidad de las respuestas de otro modelo.

🟦 **Flashcard 7**
Pregunta: ¿Por qué se necesitan métricas específicas según la tarea?
Respuesta: Porque cada tipo de tarea tiene criterios distintos para saber si está bien hecha.

FLASHCARDS – Evaluación de Sistemas Completos de GenAI

🟦 **Flashcard 1**
Pregunta: ¿Por qué es importante evaluar el sistema GenAI completo?
Respuesta: Porque el rendimiento general y el costo dependen de todos los componentes, no solo del modelo.

🟦 **Flashcard 2**
Pregunta: ¿Qué incluye una arquitectura de sistema GenAI?
Respuesta: Modelo LLM, prompts, herramientas externas, lógica de cadena, filtros de seguridad.

🟦 **Flashcard 3**
Pregunta: ¿Cómo se mejora el rendimiento total de un sistema GenAI?
Respuesta: Evaluando y ajustando cada componente por separado para optimizar su aporte al sistema.

🟦 **Flashcard 4**
Pregunta: ¿Qué es una métrica personalizada?
Respuesta: Es una métrica adaptada a un componente específico del sistema, como rapidez o relevancia.

🟦 **Flashcard 5**
Pregunta: ¿Qué es la evaluación online?
Respuesta: Evaluación continua con usuarios reales, útil para medir el desempeño a largo plazo y en escala.

FLASHCARDS – Fundamentos de Despliegue de Modelos

🟦 **Flashcard 1**
Pregunta: ¿Qué es un despliegue por lotes (batch)?
Respuesta: Cuando el modelo procesa grandes cantidades de datos en intervalos programados.

🟦 **Flashcard 2**
Pregunta: ¿Cuándo se recomienda el despliegue en tiempo real?
Respuesta: Cuando se necesita una respuesta inmediata, como en chatbots o recomendaciones.

🟦 **Flashcard 3**
Pregunta: ¿Cuál es la diferencia entre stream y real-time?
Respuesta: Stream procesa datos constantemente, real-time responde al instante a eventos individuales.

🟦 **Flashcard 4**
Pregunta: ¿Qué son los Model Flavors en MLflow?
Respuesta: Formatos compatibles que permiten desplegar modelos de diferentes frameworks.

🟦 **Flashcard 5**
Pregunta: ¿Qué hace el MLflow Deploy Client?
Respuesta: Permite desplegar modelos en diferentes entornos desde MLflow de forma sencilla.

🟦 **Flashcard 6**
Pregunta: ¿Por qué usar Unity Catalog para registrar modelos?
Respuesta: Mejora la seguridad, la organización y el seguimiento de modelos registrados.

FLASHCARDS – Batch Deployment

🟦 **Flashcard 1**
Pregunta: ¿Qué es el batch deployment?
Respuesta: Es ejecutar un modelo sobre un conjunto de datos grande en momentos programados.

🟦 **Flashcard 2**
Pregunta: ¿Cuándo es ideal usar batch deployment?
Respuesta: Cuando no se necesita una respuesta inmediata, como reportes o análisis periódicos.

🟦 **Flashcard 3**
Pregunta: Menciona una ventaja del batch deployment.
Respuesta: Es más eficiente y económico para procesar grandes volúmenes de datos.

🟦 **Flashcard 4**
Pregunta: ¿Cuál es una desventaja del batch deployment?
Respuesta: No es útil si se requiere respuesta inmediata; puede haber retrasos.

🟦 **Flashcard 5**
Pregunta: ¿Qué pasos implica un batch deployment en Databricks?
Respuesta: Registrar modelo → Crear job → Cargar modelo → Inferencia → Guardar resultados.

🟦 **Flashcard 6**
Pregunta: ¿Qué función se usa para cargar un modelo en MLflow para batch inference?
Respuesta: `pyfunc.load_model()`.

FLASHCARDS – Real-Time Deployment

🟦 **Flashcard 1**
Pregunta: ¿Qué es real-time deployment?
Respuesta: Es desplegar un modelo para que responda inmediatamente a cada solicitud individual.

🟦 **Flashcard 2**
Pregunta: ¿Cuándo se necesita despliegue en tiempo real?
Respuesta: En apps donde la respuesta debe ser instantánea, como chatbots o detección de fraude.

🟦 **Flashcard 3**
Pregunta: Menciona un reto del real-time deployment.
Respuesta: Lograr baja latencia y alta disponibilidad con muchos usuarios.

🟦 **Flashcard 4**
Pregunta: ¿Qué permite hacer Databricks Model Serving?
Respuesta: Crear endpoints REST de forma sencilla para servir modelos en tiempo real.

🟦 **Flashcard 5**
Pregunta: ¿Qué pasos se siguen para servir un modelo en Databricks?
Respuesta: Registrar modelo → Activar Model Serving → Crear endpoint → Hacer solicitudes → Monitorear.

🟦 **Flashcard 6**
Pregunta: ¿Qué ventaja tiene usar la UI de Databricks para servir modelos?
Respuesta: Permite activar el endpoint fácilmente sin necesidad de código.

FLASHCARDS – AI System Monitoring

🟦 **Flashcard 1**
Pregunta: ¿Por qué es importante monitorear un sistema de IA?
Respuesta: Para detectar errores, mantener el rendimiento y asegurar calidad y seguridad a lo largo del tiempo.

🟦 **Flashcard 2**
Pregunta: ¿Qué tipo de métricas se monitorean en el modelo?
Respuesta: Precisión, latencia, errores, costo por uso.

🟦 **Flashcard 3**
Pregunta: ¿Qué se puede monitorear en las herramientas externas del sistema de IA?
Respuesta: Tiempo de respuesta, disponibilidad, fallos.

🟦 **Flashcard 4**
Pregunta: ¿Qué es Lakehouse Monitoring?
Respuesta: Es una herramienta de Databricks para supervisar el rendimiento online de aplicaciones GenAI.

🟦 **Flashcard 5**
Pregunta: ¿Qué beneficios tiene usar Lakehouse Monitoring?
Respuesta: Visualización en tiempo real, detección de anomalías y monitoreo unificado de datos y modelos.

FLASHCARDS – LLMOps Concepts

🟦 **Flashcard 1**
Pregunta: ¿Qué es LLMOps?
Respuesta: Es la práctica de manejar el ciclo de vida completo de modelos de lenguaje grande (LLMs) en producción.

🟦 **Flashcard 2**
Pregunta: ¿En qué se diferencia LLMOps de MLOps?
Respuesta: LLMOps maneja modelos más grandes, con más riesgos, costos y complejidad que los modelos ML tradicionales.

🟦 **Flashcard 3**
Pregunta: ¿Qué componentes incluye una arquitectura típica de LLMOps?
Respuesta: Evaluación, serving, monitoreo, control de versiones y mejora continua.

🟦 **Flashcard 4**
Pregunta: ¿Qué se hace en la etapa de experimentación de LLMOps?
Respuesta: Ingeniería de prompts y pruebas con diferentes LLMs.

🟦 **Flashcard 5**
Pregunta: ¿Qué se monitorea en una solución de LLMOps?
Respuesta: Latencia, calidad de respuestas, toxicidad, costos y seguridad.