Proyecto Transversal: Caso LogiData S.A.S.

INGENIERÍA DE DATOS

Propósito del reto

El proyecto transversal busca integrar todos los conocimientos del programa en un caso de negocio realista, demostrando la capacidad del estudiante para **construir una plataforma moderna de datos en AWS**, con flujos **batch y streaming**, aplicando principios de **gobernanza, automatización, calidad y visualización**.

El reto representa una situación realista en la que **cada estudiante** deberá **aplicar progresivamente los conceptos vistos en clase**, construir entregas bajo el lenguaje de **historias de usuario**, y presentar una **solución integral validable técnica y conceptualmente** ante un panel experto.

Contexto de la empresa

LogiData S.A.S. es una empresa colombiana de logística inteligente que presta servicios de transporte y entrega de última milla para comercios electrónicos, supermercados y farmacéuticas.

Actualmente procesa más de **50.000 pedidos diarios** y **3.000 vehículos conectados con sensores IoT**, que reportan temperatura, ubicación y estado en tiempo real.

Su crecimiento acelerado generó una gran dispersión de información entre sistemas legados, archivos planos y fuentes externas.

El reto del estudiante de Ingeniería de Datos es crear una **plataforma moderna en AWS** que consolide toda esta información y permita decisiones operativas basadas en datos.

Reto de negocio

LogiData necesita una plataforma de datos centralizada, escalable y gobernada que:

- Reduzca retrasos logísticos.
- Monitoree condiciones de transporte en tiempo real.
- Unifique indicadores de desempeño y cumplimiento.
- Permita decisiones basadas en datos confiables.

Desafío del estudiante

Cada estudiante deberá **diseñar, construir y desplegar de manera individual** la nueva plataforma de datos de LogiData, integrando flujos batch y streaming, aplicando buenas prácticas de ingeniería de datos, calidad y automatización.

El resultado final debe incluir: modelo de datos, procesamiento, gobierno, orquestación, despliegue y dashboard funcional.

Base de datos unificada

Todos los estudiantes trabajarán sobre la **misma base de datos LogiData**, entregada por AceleraTI en formato CSV y con un endpoint simulado.

Esto garantiza que las comparaciones técnicas y los resultados sean consistentes.

Tablas incluidas

- **Pedidos:** id_pedido, fecha, cliente, monto, estado.
- **Entregas:** id_pedido, hora_programada, hora_real, zona, conductor.
- **Clientes:** id_cliente, nombre, zona, tipo_cliente.
- **Sensores IoT:** id_vehículo, temperatura, latitud, longitud, evento, timestamp.
- **Catálogo de productos:** id_producto, categoría, precio, tipo_entrega.

Esquema

clientes.csv

- `id_cliente` (string) — PK
- `nombre` (string)
- `zona` (string) ∈ {Norte, Sur, Oriente, Occidente, Centro}
- `tipo_cliente` (string) ∈ {Retail, Farmacéutico, Supermercado, Ecommerce, Restaurante}

catalogo.csv

- `id_producto` (string) — PK
- `categoria` (string)
- `precio` (float)
- `tipo_entrega` (string) ∈ {Same Day, Next Day, Programada, Express}

pedidos.csv

- `id_pedido` (string) — PK
- `id_cliente` (string) — FK → clientes
- `id_producto` (string) — FK → catálogo
- `fecha` (datetime, UTC-agnóstico)
- `monto` (float)
- `estado` (string) ∈ {CREADO, EN_DESPACHO, ENTREGADO, CANCELADO}

entregas.csv (solo pedidos no cancelados)

- `id_pedido` (string) — FK → pedidos
- `hora_programada` (datetime)
- `hora_real` (datetime)
- `zona` (string)
- `conductor` (string, ej. C0001…C0300)
- `vehiculo` (string, ej. V0001…V0500)

sensores.csv (eventos IoT simulados)

- `vehiculo` (string) — FK lógico → entregas.vehiculo
- `timestamp` (datetime)
- `latitud` (float)
- `longitud` (float)
- `temperatura` (float, °C)
- `evento` (string) ∈ {OK, TEMP_CRITICA}

Notas para la clase y la sustentación

- Los datos cubren **~300 clientes**, **200 productos**, **2000 pedidos**, **~entregas** para pedidos no cancelados y **10.000 eventos de sensores**.
- Las claves **referenciales** están alineadas para que puedan construir fácilmente el **modelo relacional**, el **NoSQL** y el **modelo dimensional**.
- Los timestamps abarcan **~60 días recientes**, ideal para flujos batch y simulación de streaming.

Estructura del proyecto: módulos, historias de usuario y entregas

Cada fase del proyecto corresponde a los contenidos vistos en clase.

Las entregas se formulan como **historias de usuario (HU)** para reforzar el enfoque práctico y medible del reto.

Módulo	Nombre del módulo	Historias de Usuario (HU)	Entregables esperados
1	Fundamentos de Ingeniería de Datos y Cloud	**HU1:** Como ingeniero de datos, quiero comprender la arquitectura de una plataforma de datos moderna para estructurar la solución de LogiData. **HU2:** Como ingeniero, quiero definir los servicios AWS que usaré en la plataforma.	- Diagrama de arquitectura general. - Documento técnico base con descripción de componentes AWS y flujo general.

2	Bases de Datos Relacionales y NoSQL	**HU3:** Como ingeniero, quiero diseñar el modelo relacional que represente los datos transaccionales de pedidos, entregas y clientes. **HU4:** Como ingeniero, quiero definir un modelo NoSQL para los datos de sensores IoT.	- Modelo relacional (PostgreSQL). - Modelo NoSQL (MongoDB o DynamoDB). - Carga inicial con datos de prueba. - Documentación de diseño.
3	Arquitectura Escalable y Data Lakes	**HU5:** Como ingeniero, quiero construir un Data Lake en S3 que reciba los datos crudos desde distintas fuentes. **HU6:** Como analista, quiero catalogar la información en AWS Glue para habilitar consultas gobernadas.	- Data Lake estructurado (zonas raw y curated). - Scripts de ingesta en Python. - Glue Catalog configurado. - Lake Formation documentado.
4	Procesamiento Distribuido con Spark	**HU7:** Como ingeniero, quiero transformar los datos de pedidos y entregas en PySpark para generar KPIs de cumplimiento y eficiencia.	- ETL batch funcional en PySpark. - Transformaciones y limpieza. - Escritura en S3 o Redshift. - Documento técnico del flujo.
5	Procesamiento de Datos en Tiempo Real	**HU8:** Como sistema de monitoreo, quiero recibir datos de sensores en tiempo real para detectar anomalías de temperatura. **HU9:** Como ingeniero, quiero integrar esos eventos en un flujo de streaming continuo.	- Flujo funcional con Kafka o Kinesis. - Productor, consumidor y simulador de eventos IoT. - Logs y evidencia de ejecución.
6	DataOps y Automatización	**HU10:** Como líder técnico, quiero orquestar los flujos de datos usando Airflow. **HU11:** Como ingeniero de calidad, quiero validar los datos procesados con Great Expectations.	- DAGs activos en Airflow (mín. 2). - Validaciones automáticas de calidad. - Diagrama del flujo completo. - Documentación CI/CD (GitHub Actions).
7	Modelado de Datos y Bodegas	**HU12:** Como analista BI, quiero diseñar un modelo dimensional para analizar el cumplimiento de entregas por zona y tiempos. **HU13:** Como ingeniero, quiero preparar los datos para consumo analítico.	- Modelo dimensional (hechos y dimensiones). - Tablas creadas o script DDL. - Documentación de relaciones y KPIs.
8	Cloud & DevOps en AWS	**HU14:** Como ingeniero DevOps, quiero desplegar toda la infraestructura del proyecto con Terraform. **HU15:** Como gerente, quiero visualizar los indicadores operativos en QuickSight.	- Terraform modular (S3, Glue, Redshift, Lambda, IAM, CloudWatch). - Dashboard en QuickSight con mínimo 3 métricas. - Evidencia funcional y monitoreo.

Orden lógico de avance

Para evitar dependencias entre módulos (por ejemplo, que el módulo 2 dependa del 7), el avance se realizará de forma progresiva:

1. **Diseño y arquitectura (Módulo 1)** → visión general del sistema.
2. **Modelado inicial (Módulo 2)** → solo modelo transaccional y NoSQL.
3. **Ingesta (Módulo 3)** → carga inicial al Data Lake.
4. **Procesamiento (Módulos 4 y 5)** → KPIs batch y streaming.
5. **Orquestación (Módulo 6)** → automatización y calidad.
6. **Modelado dimensional (Módulo 7)** → creación de bodega analítica sobre datos ya transformados.
7. **Despliegue y visualización (Módulo 8)** → infraestructura + dashboard final.

Evaluación y validación

Etapa	Criterio de evaluación	Validación en clase / panel
Diseño	Claridad conceptual, coherencia técnica	Presentación del diagrama y explicación individual
Modelado	Aplicación correcta de relaciones y estructuras	Revisión de esquemas SQL y NoSQL
Ingesta	Integración y gobierno de datos	Ejecución de scripts y Glue Catalog
Procesamiento	Lógica y eficiencia del ETL	Ejecución individual en PySpark
Streaming	Funcionamiento en tiempo real	Logs y simulador de eventos IoT
Automatización	Orquestación y control de calidad	DAGs corriendo + validaciones GE
Modelado BI	Coherencia de dimensiones y hechos	Revisión de modelo y consultas
Despliegue y Dashboard	Visualización y modularidad IaC	Demo personal en QuickSight y Terraform

Sustentación final ante panel

Durante la sustentación, **cada estudiante deberá**:

1. Contar la historia del caso LogiData y el problema resuelto.
2. Explicar su arquitectura y decisiones técnicas.

3. Demostrar en vivo la ejecución de sus pipelines y dashboards.
4. Argumentar cómo garantizó la calidad, seguridad y automatización.
5. Mostrar apropiación del conocimiento en todos los módulos.

El panel evaluará tanto la **solidez técnica** como la **comprensión conceptual** y la **autonomía en la ejecución**.

Estructura del repositorio sugerida

```
/docs
  ├── arquitectura_logidata.pdf
  ├── modelo_datos_logidata.pdf
/src
  ├── ingest/
  ├── processing/
  ├── streaming/
  ├── airflow_dags/
  ├── terraform/
  ├── tests/
/dashboards
  └── quicksight_logidata.pdf
README.md
```

Notas finales

- Las entregas se formulan como **historias de usuario individuales**, no en equipo.
- Cada entrega debe mostrar **apropiación de los conceptos vistos en clase**, no solo ejecución técnica.
- La **base de datos LogiData** es común para todos los estudiantes y no debe modificarse estructuralmente.
- Las validaciones se realizarán en clase con acompañamiento docente.