UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

CS2032 - Cloud Computing (Ciclo 2024-2)
Proyecto Final
Semana 11 a Semana 15 (Exposición en Semana 16)

ELABORADO POR: GERALDO COLCHADO

Agenda

Proyecto Final

1. Competencias a lograr
2. Enunciado y Rúbrica
3. Entregables y Plazo

Competencias a lograr
Por el alumno al finalizar el proyecto final

Competencia:
4.1: Crea, selecciona, adapta y aplica técnicas, recursos y herramientas modernas para la práctica de la computación y comprende sus limitaciones. (nivel 3).

Agenda

Proyecto Final

1. Competencias a lograr
2. Enunciado y Rúbrica
3. Entregables y Plazo

Enunciado y Rúbrica
Proyecto Final

- **Integrantes:** Máximo 5 personas por grupo

- **Funcionalidad:** Usted elegirá la funcionalidad a implementar. Ejemplo: Sistema para un Banco, para una compañía de Seguros, para una cadena de cines, para una aerolínea, para una casa de apuestas en línea, para una tienda de Comercio Electrónico, para un videojuego en línea, para streaming de música como Spotify, etc.

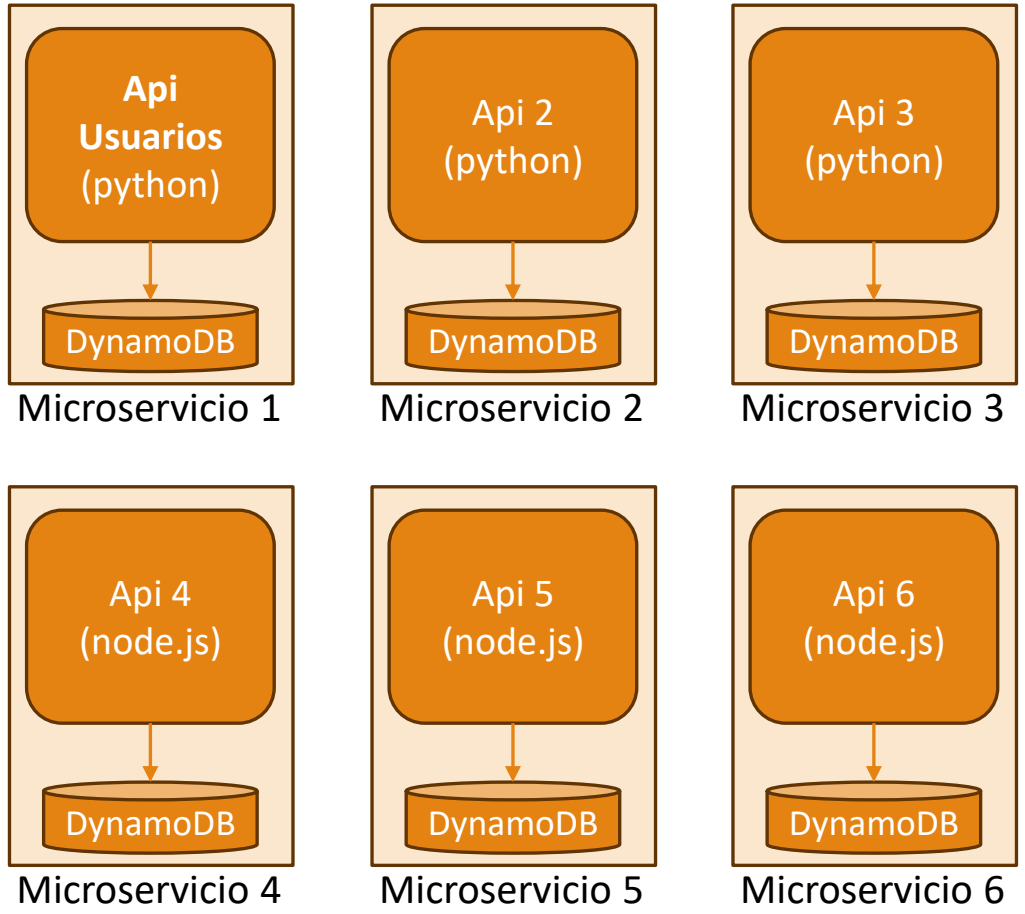
- **Arquitectura:**
  - Multi-tenancy

- **Rúbrica:**
  - BackEnd (7 puntos)
  - FrontEnd (4 puntos)
  - Data Science (6 puntos)
  - Diagrama de Arquitectura Solución (1 punto)
  - Exposición presencial (2 puntos)

  Si no se presenta a la exposición presencial su evaluación será desaprobatoria (sobre nota 10 como máximo).

Enunciado
Backend (7 puntos) - Multi-tenancy y Serverless

- **Bases de Datos:** Deberá usar DynamoDB en todas las Apis. Debe diseñar la clave de partición y clave de ordenamiento como Multi-tenancy y priorizando consultas por query y no por scan. Debe tener como mínimo 1 “Global secondary index (GSI)” y 1 “Local secondary index (LSI)”. Debe configurar en cada tabla DynamoDB la funcionalidad de “Copias de seguridad - Recuperación a un momento dado (PITR)”. Debe presentar un diagrama donde relacione lógicamente todas las tablas DynamoDB y se muestre la estructura json de cada tabla.
- **Seguridad:** Deberá tener un Api Usuarios Multi-tenancy que permita crear usuario, login usuario y validar token de acceso. Todas las demás Apis deben estar protegidas con token de acceso.
- **Microservicios:** Debe implementar como mínimo 6 Apis Multi-tenancy con Lambdas y Api Gateway incluyendo al Api Usuarios. Debe automatizar su despliegue con framework serverless en 3 stages (dev, test y prod) incluyendo las tablas DynamoDB. Debe implementar la mitad de lambdas en **python** y la otra mitad en **node.js**.
- **Datos de prueba:** Debe insertar masivamente, por única vez, datos ficticios (fake data) en todas las tablas DynamoDB (Mínimo 10,000 registros x tabla repartidos entre 3 tenant_id diferentes).
- **Documentación**: Debe documentar las 6 apis para visualizarlas en swagger-ui.
- **Código Fuente:** Debe incluir enlaces a repositorios públicos de github con las fuentes.



Enunciado
FrontEnd (4 puntos) - Multi-tenancy y Serverless

- Debe implementar una **página web** Multi-tenancy con **login de usuario** y desplegarla en un **bucket S3** de AWS (Simple Storage Service) y que **invoque a los 6 microservicios** (como mínimo a 1 método de cada api rest). Deben existir 3 buckets S3, uno por cada stage (dev, test, prod) y que invoquen a las apis respectivas.

- Puede usar el **framework web** del lado del cliente **de su preferencia**. Ejemplos:
    - Javascript puro
    - Javascript con framework: react.js, angular.js, vue.js

- Debe incluir enlace a repositorio público de github con las fuentes.

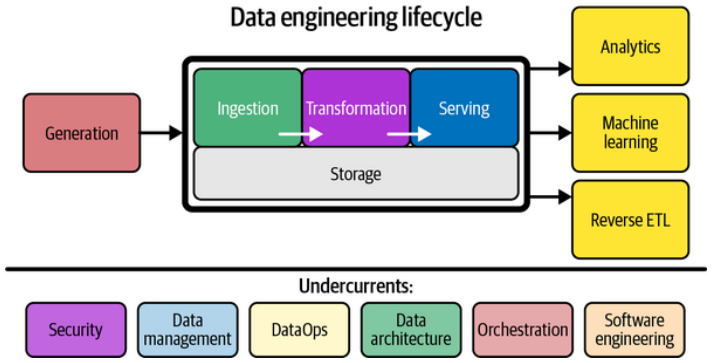
Enunciado
Data Science (6 puntos) - Multi-tenancy y Máquina Virtual

- **Cómputo:** Debe crear una máquina virtual "**MV Ciencia Datos**" por cada stage (dev, test, prod) con tipo de instancia "t2.medium" (2 vCPU, 4 GB RAM).
- **Almacenamiento:** Debe crear un **bucket S3** por cada stage (dev, test, prod) para almacenar la ingesta de datos.
- **Ingesta de datos:** Debe implementar **5 contenedores docker** en python para la ingesta de datos con **estrategia pull** del 100% de los registros de las tablas DynamoDB. Cada contenedor ingestará la data de 1 microservicio y generará archivos csv o json normalizados que cargue en el bucket S3. Se recomienda usar la funcionalidad de scan de DynamoDB (*) tomando en cuenta que DynamoDB pagina los resultados de las operaciones de scan. Con la paginación, los resultados del scan se dividen en "páginas" de datos que tienen un tamaño de 1 MB (o menos).
- **Catálogo de Datos:** Debe implementar un **catálogo de datos** en **AWS Glue** por cada archivo con estructura diferente que cargue al bucket S3. Debe crear un diagrama **Entidad / Relación** que relacione **todas las tablas** del catálogo de datos.
- **Analítica de Datos:** Debe mostrar evidencia de como mínimo 6 **consultas SQL** Multi-tenancy que unan varias tablas con **AWS Athena** y crear como mínimo 3 vistas Multi-tenancy.
- **Transformación:** Debe implementar **1 contenedor docker ETL (Extract, Transform, Load)** en python que ejecute queries SQL con Athena e inserte los resultados en tablas resúmenes de BD MySQL (Utilizar MySQL en contenedor en "MV Ciencia Datos"). Debe crear un diagrama **Entidad / Relación** que relacione **todas las tablas** resúmenes.
- **Logs:** Los 6 contenedores deberán dejar archivos de logs con el mismo formato en un mismo directorio de la máquina virtual "**MV Ciencia Datos**" que tengan como mínimo: fecha_hora (hasta milisegundo), tipo_log (INFO, WARNING, ERROR, CRITICAL), nombre_contenedor, mensaje. Deberá mostrar evidencia de ejecución de queries sobre los logs con **lnav** para mostrar por cada contenedor su hora de inicio y fin, cantidad de registros procesados y los pasos más importantes del proceso con la información más relevante.
- **Dashboard o Panel de Control:** Debe implementar un Dashboard (Informe) Multi-tenancy en Google Looker Studio que utilice como fuente de datos la BD MySQL y tenga 10 gráficos de 5 tipos diferentes y 2 filtros como mínimo. Revise estos enlaces para Looker Studio: enlace1, enlace2.
- **Código Fuente:** Debe incluir enlaces a repositorios públicos de github con las fuentes.

(*) Referencia: https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/Scan.html

Enunciado

Data Science (6 puntos)



Fuentes de datos

DynamoDB
Microservicio 2

DynamoDB
Microservicio 3

DynamoDB
Microservicio 4

DynamoDB
Microservicio 5

DynamoDB
Microservicio 6

Ingesta de datos

Contenedor ingesta02

Contenedor ingesta03

Contenedor ingesta04

Contenedor ingesta05

Contenedor ingesta06

MV Ciencia Datos

Almacenamiento

Bucket S3

Transformación

Catálogo de Datos (Glue)

MySQL (Tablas resumen)

Consultas **SQL** (Athena)

Contenedor ETL

MV Ciencia Datos

Analítica de datos

Usuario

Consultas **SQL** (Athena)

Google Looker Studio

Dashboard o Panel de Control

Generation | **Ingestion** | **Storage** | **Transformation** | **Analytics**

Enunciado
Diagrama de Arquitectura de Solución (1 punto)

- Debe elaborar un Diagrama de Arquitectura de Solución que incluya y relacione todos los componentes de:
  - BackEnd
  - FrontEnd
  - Data Science

Agenda

Proyecto Final

1. Competencias a lograr
2. Enunciado
3. Entregables y Plazo

Entregables y Plazo
Grupos de hasta 5 personas

Entregables	Plazo (Fin de Semana 15)
• *Informe en word o pdf con evidencia de todo lo solicitado.* • *Resumen en power point con todo lo solicitado.* • *Exposición presencial y demo durante la semana 16.*	*Máximo hasta Domingo 1-Diciembre-2024 23:59 (En Canvas)*