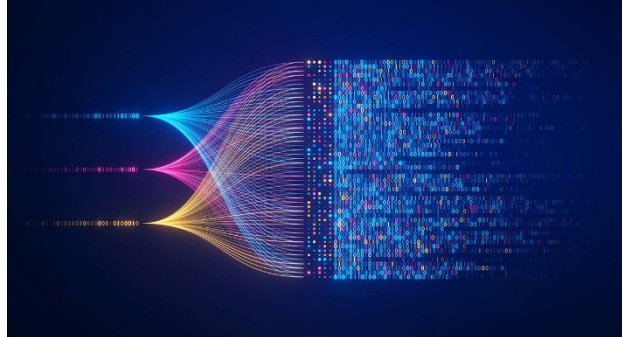




PRESENTACIÓN



NOMBRE

Esteban Josué Mata Acosta

MATRÍCULA

2024-1860

ELECTIVA 1

FECHA DE ENTREGA

13/12/25

TEMA

Proyecto final

INTRODUCCIÓN

Este documento describe el desarrollo de una aplicación web simple con frontend en HTML/CSS/JavaScript, backend en Node.js/Express, base de datos en SQL Server, y un pipeline de CI/CD con GitHub Actions.

El objetivo es demostrar buenas prácticas de desarrollo, integración continua, despliegue automatizado y monitoreo básico.

ARQUITECTURA DEL PROYECTO

La aplicación se compone de los siguientes módulos:

- Frontend: interfaz básica en HTML, CSS y JavaScript.
- Backend: API REST con Node.js y Express.
- Base de datos: SQL Server con tablas iniciales
- Contenedores: Docker para empaquetar y ejecutar servicios.
- Pipeline CI/CD: GitHub Actions para pruebas, construcción y despliegue.
- Monitoreo: logs centralizados y métricas básicas.

PIPELINE CI/CD

El pipeline automatiza el flujo de integración y despliegue:

1. Checkout del código desde GitHub.
2. Instalación de dependencias Node.js.
3. Ejecución de pruebas unitarias e integración.
4. Construcción de imagen Docker.
5. Push de la imagen a DockerHub.
6. (Opcional) Despliegue automático en servidor remoto.

Beneficios:

- Validación automática de cambios.
- Entrega continua sin intervención manual.
- Versionado y despliegue consistente.

LECCIONES APRENDIDAS

- La integración de CI/CD reduce errores humanos y acelera entregas.
- Docker garantiza portabilidad entre entornos.
- Documentar cada etapa facilita colaboración y mantenimiento.
- Las pruebas automatizadas detectan errores antes del despliegue.

DESAFÍOS Y SOLUCIONES

Configurar SQL Server en Docker

- Usar imagen oficial y variables de entorno seguras

Conexión backend ↔ base de datos

- Validar puertos, credenciales y tiempos de espera

Fallos en GitHub Actions

- Separar jobs y usar needs: para dependencias

MANUAL DE OPERACIONES

- Logs: almacenados en tabla Logs de SQL Server.
- Métricas: número de requests, tiempo de respuesta, errores.
- Alertas: notificaciones cuando el backend se detiene o la base de datos no responde.
- Recuperación de fallos:
- Verificar contenedores: `docker ps`
- Reiniciar servicio: `docker-compose restart backend`
- Revisar logs: `docker logs simple-backend`