Eduardo David Tejada Moreta

davidtejadamoreta26@gmail.com

+18296204529

PERFIL PROFESIONAL

Analista de Automatización y Desarrollador de Software con experiencia en optimización de procesos y creación de soluciones personalizadas. Especializado en:

- **Automatización, captura y validación de datos mediante scripting.**
- **Mejora de la eficiencia operativa y reducción de tareas manuales.**
- **Prácticas de DevOps: contenedores, despliegue continuo y monitorización de aplicaciones.**

EDUCACIÓN

- **Bachillerato (2018-2023)**
  Institución: Profesora Celeste Aida del Villar
- **Desarrollo de Software (2024 - Presente)**
  Institución: ITLA (Instituto Tecnológico de las Américas)

CERTIFICACIONES

- **Técnico de Ciberseguridad** – DarFe Learning Consulting S.L. (Enero 2023)
- **Cyber Security** – Xford Home Study Centre (Dic 2022**)**
- **Python Fundamentals for Beginners –** Great Learning (Sep 2023)
- **Excel –** Universidad Psicología Industrial Dominicana (UPID) (2024)

HABILIDADES Y CONOCIMIENTOS

Desarrollo de Software y APIs

- **Lenguajes:** Python
- **Frameworks** y APIs: FastAPI, Flask, Tornado, Django, Dash, RESTful APIs, API Gateway
- **ORMs:** SQLAlchemy, DRF

Contenedores y Orquestación

- **Contenedores:** Docker

- **Orquestación:** Kubernetes, Docker-compose

Servidores HTTP

- **Proxy Inverso y Balanceo:** Nginx
- **Servidores ASGI/WSGI:** Uvicorn, Gunicorn
- **Administración remota mediante SSH.**
- **Despliegue.**

Bases de Datos y Almacenamiento

- **DBMS:** PostgreSQL, MySQL
- **DB en memoria:** Redis

Monitoreo y Dashboard

- Prometheus, Grafana
- Power Bi

Pruebas de carga, unitaria y de integración:

- Locust
- Pytest

DevOps e Infraestructura

- **Version Control:** Git
- **Load Testing:** Locust
- **Túneles:** Ngrok
- **Cloud:** Azure, Render
- **CI/CD:** Jeankis, Github Action
- **Sistemas Operativos:** Windows, Ubuntu Server, Debian

Arquitectura de Software

- Microservicios, Clean Architecture, Monolítica

M365

- Excel, Word

IDIOMAS

- **Español: Nativo (★★★★★)**

- **Inglés: Intermedio (★★★☆☆)**

EXPERIENCIA LABORAL

Soporte Técnico
Junta Central Electoral (JCE), Santo Domingo Oeste
Junio 2023 – Octubre 2023

- Instalación y mantenimiento de equipos.
- Registro y resolución de incidencias en sistemas y dispositivos.
- Configuración, monitoreo y soporte técnico general**.**

Analista de Datos
INAFOCAM
Mayo 2024 – Actualidad

- Automatización de captura, validación y transformación de datos mediante scripting.
- Desarrollo de software a medida para automatizar procesos específicos.
- Creación y despliegue de aplicaciones web internas o scripts para los pipeline
- Logros: Reducción significativa del tiempo de procesamiento manual al implementar herramientas personalizadas.

Logros Destacados

Optimización del Proceso de Validación de Participantes

El proceso de validación de participantes solía ser tedioso y propenso a errores debido a registros incompletos o mal redactados. Para solucionarlo, se desarrolló una plataforma en Python que permite:

- Cargar listados en Excel.
- Normalizar cédulas.
- Capturar información.
- Separar registros no válidos.

La herramienta utiliza técnicas de programación asíncrona (asyncio/uvloop), caching, multithreading, event loops, semáforos y un pool de conexiones, lo que le permite procesar hasta 10,000 registros en aproximadamente 30 segundos (en condiciones sin cache), con posibilidad de escalar aumentando workers o el límite del pool. Este logro demuestra experiencia en procesos asíncronos, programación concurrente/paralela, manejo eficiente de operaciones I/O-bound, segmentación en batches y gestión de tareas mediante semáforos.

Orquestación de Servicios mediante Docker Compose y Kubernetes

La instalación de dependencias y la configuración de entornos pueden generar conflictos y aumentar la complejidad. Mediante Docker Compose se orquestaron los servicios necesarios para la plataforma,

permitiendo implementaciones multientorno con pocos comandos (si se cumple el prerrequisito de contar con Docker). Este enfoque simplifica la administración de servicios y garantiza una implementación consistente y escalable en diversos ambientes.

Balanceo de Carga mediante Docker Compose, Kubernetes y Nginx

Muchos problemas de rendimiento surgen por el mal escalamiento vertical, ya que éste no siempre soluciona el cuello de botella. Para abordar este desafío, se implementaron estrategias de escalamiento horizontal utilizando Docker Compose y Kubernetes, permitiendo distribuir la carga de manera óptima entre múltiples instancias. Además, se integró Nginx para balancear la carga mediante técnicas como round-robin, least_conn e ip-hash, asegurando una distribución eficiente de las solicitudes. Ejemplos de estas configuraciones están disponibles en el repositorio público: Mircroservicios.

Identificación de Cuellos de Botella en Aplicativos

Se realizó un análisis exhaustivo utilizando herramientas como Locust, Apache Benchmark y Docker Stats. Los resultados indicaron que:

- La subida concurrente de archivos y la ejecución intensiva en un entorno asíncrono provocaban un notable incremento en el consumo de CPU y RAM.
- El uso intensivo del pool de conexiones en la arquitectura asíncrona incrementaba el consumo de CPU a lo largo del tiempo, afectando tanto a los servicios externos como a la estabilidad interna.

Para mitigar estos cuellos de botella, se implementó una segmentación lógica de datos en batches y se reguló la ejecución de tareas mediante semáforos, reduciendo el consumo de recursos sin comprometer los tiempos de respuesta. Este enfoque, validado mediante monitoreo continuo a nivel del sistema operativo y pruebas de carga, permitió optimizar significativamente el flujo de trabajo y la eficiencia del aplicativo.

PROYECTOS PERSONALES

1. **Maqueta para Desarrollo de Microservicios**
    - **Objetivo:** Facilitar la configuración de infraestructura y servicios, permitiendo a los desarrolladores enfocarse en la lógica de negocio.
    - **Características:**
        - Infraestructura adaptable con Docker Compose o Kubernetes.
        - Arquitectura de microservicios con API Gateways preconfiguradas (Nginx, FastAPI).
        - Módulos para enrutamiento, autenticación, balanceo de carga y rate limit**.**
    - **Repositorio: Mircroservicios**
2. **Servicio de Geolocalización IP (Microservicios)**

- **Objetivo:** Obtener información geográfica de direcciones IP (IPv4/IPv6) mediante una arquitectura modular.
- **Características:**
  - Endpoints desarrollados con FastAPI (por ejemplo, /get-ip).
  - Caché implementado con Redis para mejorar el rendimiento.
  - Uso de Nginx como proxy inverso y balanceador de carga.
  - Interfaz gráfica en Angular y orquestación con Docker Compose.
- **Repositorio:** whatismyip

3. **API Provincias, Municipios y Sectores de la República Dominicana**
   - **Duración: Ene. 2025 – Actualidad**
   - **Objetivo:** Desarrollar una API pública que devuelva, en formato JSON, todas las provincias, municipios y sectores de la República Dominicana, presentados en forma jerárquica.
   - **Tecnologías**: FastAPI, Docker, PostgreSQL
   - **URL:** Provincias y Sectores
   - **Estado:** En desarrollo.
4. **API de Instituciones Públicas de la República Dominicana**
   - **Duración: Dic. 2024 – Actualidad**
   - **Objetivo:** Centralizar la información de las instituciones públicas del país para fomentar su uso y establecer buenas prácticas en el desarrollo de APIs REST.
   - **Características:**
     - Desarrollada con FastAPI y optimizada para tiempos de respuesta.
     - **Ejemplo de atributos destacados:**
       - **"nombre": "Centro de Atención Integral para la Discapacidad"**
       - **"sigla": "CAID"**
       - **"ubicacion": "Centro de Atención Integral Para la Discapacidad, Avenida Gregorio Luperón, Santo Domingo de Guzmán, Distrito Nacional, 11108, República Dominicana"**
       - **"lat_min": 18.4409577, "lat_max": 18.4421027**
       - **"lon_min": -69.9796326, "lon_max": -69.9784095**
     - Pruebas realizadas en contenedores Docker; despliegue en Render (versión gratuita).

        - **Rendimiento:** Reducción del tiempo de respuesta de 8 segundos (121 respuestas/seg) a 2 ms (479 solicitudes/seg), con mejoras a 0.69 ms y 692 respuestas/seg mediante workers, esta misma pudiendo soportar mas de 100k solicitudes entre 1k usuarios, sin aumentar los recursos consumidos.
    - **Tecnologías:** Python, FastAPI, SQLite, Angular
    - **URL:** Instituciones Públicas
    - **Repositorio de datos:** Datos Instituciones

PROYECTOS LABORALES

Plataforma de Validación

- **Objetivo:** Desarrollar un aplicativo para la validación de cédulas y extracción automatizada de información docente.
- **Tecnologías: Python, Streamlit**