# Prueba Técnica de Desarrollador Fullstack Backend

**Bienvenido/a.**

Esta prueba técnica está diseñada para evaluar tus habilidades en las tecnologías y lenguajes clave de nuestro stack. Nos interesa más la calidad de tu código y la claridad de tu solución que la velocidad con la que la entregues.

---

## Objetivo del Desafío

Tu tarea es construir una aplicación web completa y monolítica para gestionar una lista de tareas (un "to-do list"). Todo, tanto el backend (API) como el frontend, debe residir en la misma base de código.

---

## Requisitos del Proyecto

### 1. Backend (API y Lógica de Negocio)

Deberás desarrollar la lógica de la aplicación en **PHP utilizando el framework CodeIgniter**. Tu aplicación debe ser capaz de gestionar tareas (CRUD). La persistencia de los datos debe manejarse en una base de datos **SQL**.

A continuación, se detalla la estructura de la tabla que debes usar:

```
CREATE TABLE tasks (
  id INT AUTO_INCREMENT PRIMARY KEY,
  title VARCHAR(255) NOT NULL,
  completed BOOLEAN DEFAULT FALSE,
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Los siguientes endpoints o métodos de tu controlador deben estar disponibles para ser consumidos por el frontend:

- **GET /tasks**: Obtiene y retorna una lista de todas las tareas en formato JSON.
- **GET /tasks/{id}**: Obtiene y retorna una tarea específica por su ID en formato JSON.
- **POST /tasks**: Crea una nueva tarea.
- **PUT /tasks/{id}**: Actualiza una tarea existente.
- **DELETE /tasks/{id}**: Elimina una tarea por su ID.

### 2. Frontend (Interfaz de Usuario)

Dentro de la misma estructura de CodeIgniter, crea una interfaz de usuario simple en **HTML, CSS y JavaScript** puro. Esta interfaz debe:

- Mostrar la lista de tareas obtenida del backend.
- Tener la funcionalidad para crear nuevas tareas.
- Permitir editar el título de las tareas existentes.
- Permitir eliminar tareas.
- **Utilizar AJAX para todas las interacciones con el backend**.

### 3. Tecnologías Adicionales

- **Docker**: El proyecto debe ser ejecutable a través de Docker. Se requiere un archivo docker-compose.yml que levante el servicio de la aplicación (CodeIgniter) y un servicio de base de datos SQL (por ejemplo, MySQL o PostgreSQL).
- **Pruebas unitarias**: Deberás escribir al menos **3 pruebas unitarias** para un controlador o un modelo de tu aplicación utilizando **phpUnit**.
- **Control de versiones**: Todo el código debe ser gestionado con **Git**.

---

## Entrega del Proyecto

1. Sube tu proyecto completo a un repositorio público de **Git** (GitHub, GitLab, etc.).
2. Crea un archivo README.md en la raíz del repositorio con las siguientes instrucciones:
    - Pasos claros para clonar y levantar el proyecto usando Docker.
    - Instrucciones para acceder al frontend.
    - Instrucciones para ejecutar las pruebas unitarias.
3. Comparte el **enlace del repositorio Git** con nosotros.

---

## Criterios de Evaluación

Valoraremos lo siguiente:

- **Calidad del Código**: Claridad, organización y adhesión a las buenas prácticas.
- **Uso de Docker**: Correcta configuración del archivo docker-compose.yml.
- **Manejo de AJAX**: Correcta implementación de las peticiones asíncronas desde el frontend.
- **Pruebas Unitarias**: Cobertura y calidad de las pruebas.
- **Git**: Uso correcto y claro del control de versiones,incluyendo la gestión de múltiples ramas (al menos 2 ramas).
- **Documentación**: Claridad y suficiencia del archivo README.md.

¡Mucho éxito en la prueba! Esperamos ver tu trabajo pronto.