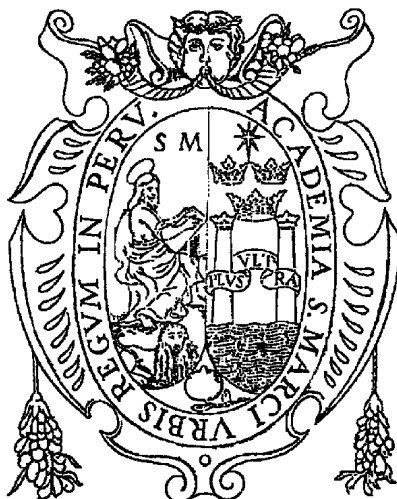


**Universidad Nacional Mayor de San Marcos**  
**Facultad de Ingeniería de Sistemas e Informática**  
**E.P. de Ingeniería de Software**



**PEP-DAS: Diseño de Arquitectura del Sistema**

**Integrantes**

Calle Huamantínco, Luis Eduardo	22200255
Calongos Jara, Leonid	22200102
Flores Cóngora, Paolo Luis	22200232
Matthew Alexandre, Pariona Molina	22200235
Calderón Matias, Diego Alonso	22200074
Luján Vila, Frank José	12200058

**Curso:** Gestión de la Configuración del Software.

**Docente:** Wong Portillo, Lenis Rossi.

**ÍNDICE**

<b>I. Diseño de Capas:</b>	.....
1. Capa de Presentación (Frontend):	.....
2. Capa de Lógica de Negocio (Backend):	.....
3. Capa de Persistencia de Datos (Base de Datos):	.....
<b>II. Protocolos de Comunicación:</b>	.....
1. REST API:	.....
2. Autenticación y Autorización:	.....
3. Contenedores y Despliegue:	.....
<b>III. Diagrama de Secuencia:</b>	.....

## PEP-DAS: Diseño de Arquitectura del Sistema

### I. Diseño de Capas:

La arquitectura está dividida en capas que separan la lógica del negocio, la interfaz y la persistencia de datos, facilitando el mantenimiento y la escalabilidad del sistema.

### **1. Capa de Presentación (Frontend):**

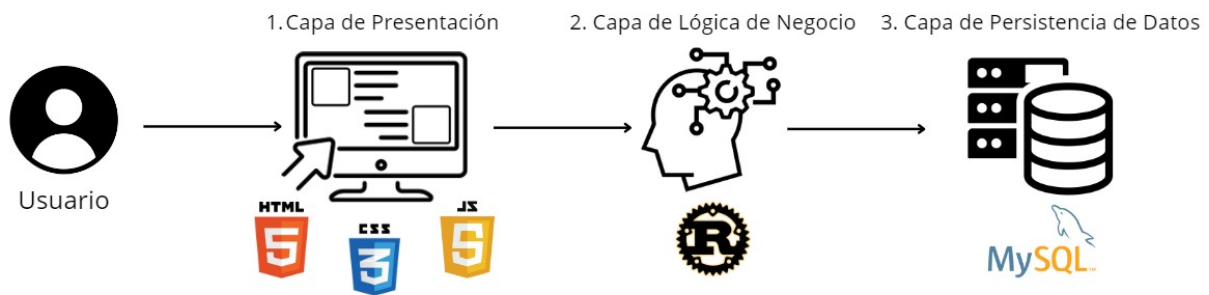
- Tecnologías Utilizadas: HTML, CSS, y JavaScript para la estructura de la interfaz y el diseño visual.
- Función: Ofrecer una interfaz para que los usuarios (estudiantes, administradores) realicen operaciones como la búsqueda de profesores, el registro de calificaciones, y la recuperación de contraseñas.
- Responsabilidad: Interactuar con el servidor mediante solicitudes HTTP (REST API) y renderizar las respuestas para mejorar la experiencia del usuario.

### **2. Capa de Lógica de Negocio (Backend):**

- Componentes: Controladores, servicios de negocio y módulos de seguridad.
- Lenguaje de Programación: Desarrollo de Rust para desarrollar APIs REST y lógica del negocio.
- Función: Procesa las solicitudes que llegan desde la capa de presentación. Los controladores reciben las solicitudes y delegan las operaciones al servicio correspondiente, que realiza las validaciones y ejecuta la lógica de negocio. Rust provee una base sólida para la eficiencia y la concurrencia, beneficios útiles en sistemas de calificaciones y evaluaciones.
- Responsabilidad: Asegurar el flujo correcto de datos entre el frontend y la capa de persistencia. También incluye la verificación de la autenticación y autorización del usuario.

### **3. Capa de Persistencia de Datos (Base de Datos):**

- Estructura del Proyecto: Uso de SQL para manejar el esquema y la inicialización de la base de datos.
- Función: Almacenar datos relacionados con usuarios, profesores, evaluaciones, y calificaciones. Gestionar los datos en una base de datos relacional como MySQL.
- Responsabilidad: Facilitar el acceso y almacenamiento de datos de manera segura y eficiente, garantizando la integridad de los datos de calificaciones, comentarios y perfiles de profesores.



## II. Protocolos de Comunicación:

### 1. REST API:

- La API REST es el principal medio de comunicación entre el frontend y el backend. En Rust, se suelen utilizar frameworks como Actix o Rocket para construir estas APIs.
- La API expone endpoints para realizar operaciones CRUD relacionadas con usuarios, calificaciones, comentarios, y autenticación.

### 2. Autenticación y Autorización:

- La autenticación puede manejarse mediante tokens JWT o similares, para asegurar que los usuarios están autorizados a realizar acciones específicas en el sistema.

### 3. Contenedores y Despliegue:

- El sistema utiliza Docker para empaquetar y desplegar los servicios. Esto permite que el backend, la base de datos y otros servicios relacionados se desplieguen y configuren de forma consistente.
- “docker-compose” facilita la administración de contenedores múltiples, orquestando la conexión entre el backend y la base de datos.

## III. Diagrama de Secuencia:

sd Diagrama de Secuencia

