DOCUMENTO DE ARQUITECTURA DEL SISTEMA DE GESTIÓN DE TUTORÍAS

Laura Ayala

Gabriela Palmezano

Valentina Carreño

Introducción a la Ingeniería

Yuli Andrea Álvarez Pizarro

16 de febrero de 2026

Documento de Arquitectura del Sistema de Gestión de Tutorías

1. Introducción

1.1 Propósito

El presente documento describe la **arquitectura del Sistema de Gestión de Tutorías**, el cual permitirá administrar tutorías ofrecidas en una institución educativa. Este documento define los componentes principales del sistema, su organización, tecnologías utilizadas y flujo de información, proporcionando una guía técnica para el desarrollo, mantenimiento y escalabilidad futura del sistema.

1.2 Alcance

El sistema permitirá:

- Registro y gestión de usuarios (estudiantes, docentes y administrador).
- Creación, visualización y gestión de tutorías.
- Reserva de tutorías por parte de los estudiantes.
- Supervisión de usuarios y tutorías por parte del administrador.

El sistema será **web responsive**, accesible desde escritorio y dispositivos móviles, y asegurará seguridad mediante control de roles.

2. Visión General de la Arquitectura

El sistema seguirá una **arquitectura en capas** basada en **cliente-servidor**, donde:

- **Capa de presentación (Frontend)**: HTML, CSS y Bootstrap para la interfaz visual, y JavaScript para interactividad y validaciones dinámicas.
- **Capa de lógica de negocio (Backend)**: PHP, responsable del manejo de autenticación, gestión de tutorías, control de reservas y permisos por rol.
- **Capa de datos (Base de datos)**: MySQL administrado con PHPMyAdmin, para almacenamiento de usuarios, tutorías, reservas y registros históricos.

Flujo general del sistema:

1. El usuario accede al sistema a través del navegador.
2. El Frontend envía solicitudes HTTP al Backend (PHP).
3. PHP procesa la solicitud y realiza consultas a la Base de Datos MySQL.

4. La respuesta es enviada al Frontend, que actualiza la interfaz de forma dinámica (JavaScript/Bootstrap).

3. Componentes del Sistema

3.1 Frontend

- **HTML**: Estructura de páginas y formularios.
- **CSS / Bootstrap**: Estilos, diseño responsive, tablas, botones, modales.
- **JavaScript**: Interactividad, validaciones, filtrado dinámico de tutorías, alertas y confirmaciones.

Ejemplos de uso:

- Filtrar tutorías por fecha, hora o docente.
- Mostrar lista de reservas de un estudiante.
- Validar formularios de registro o creación de tutorías.

3.2 Backend

- **PHP**:
    - Gestión de sesiones y autenticación.
    - Control de roles (estudiante, docente, administrador).
    - Lógica de negocio: creación, actualización y eliminación de tutorías, reservas y usuarios.
    - Comunicación con MySQL mediante consultas SQL.

3.3 Base de Datos

- **MySQL / PHPMyAdmin** Tablas principales:

tb_rol		
Campo	**Tipo**	**Descripción**
id_rol	INT (PK, AI)	Identificador del rol
nombre	VARCHAR(50)	Nombre del rol (estudiante/profesor/administrador)

tb_usuarios		
Campo	**Tipo**	**Descripción**
id_usuario	INT (PK, AI)	Identificador del usuario
nombre	VARCHAR(100)	Nombre completo
correo	VARCHAR(100) UNIQUE	Correo del usuario
password	VARCHAR(255)	Contraseña encriptada

id_rol	INT (FK)	Relación con tabla roles
activo	TINYINT(1)	Indica si el usuario está activo
fecha_creacion	TIMESTAMP	Fecha de registro

tb_tutorias		
Campo	**Tipo**	**Descripción**
id_tutoria	INT (PK, AI)	Identificador de la tutoría
id_profesor	INT (FK)	Usuario profesor que crea la tutoría
fecha	DATE	Fecha de la tutoría
hora_inicio	TIME	Hora de inicio
hora_fin	TIME	Hora de finalización
tema	VARCHAR(255)	Tema de la tutoría
cupos	INT	Número de estudiantes permitidos
estado	ENUM	disponible, reservada, cancelada
fecha_creacion	TIMESTAMP	Fecha de creación

tb_reservas		
Campo	**Tipo**	**Descripción**
id_reserva	INT (PK, AI)	Identificador de la reserva
id_tutoria	INT (FK)	Tutoría reservada
id_estudiante	INT (FK)	Usuario estudiante
estado	ENUM	activa, cancelada, asistida
fecha_reserva	TIMESTAMP	Fecha en que se hizo la reserva

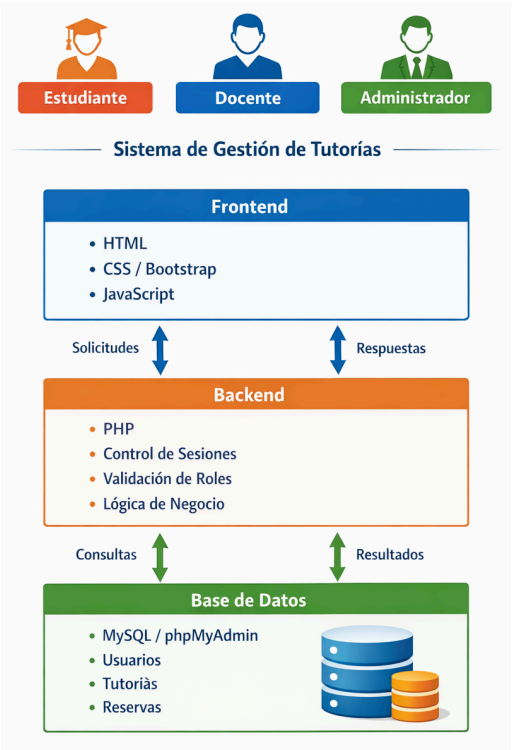
Consideraciones:

- Integridad referencial (FK entre tutorias y usuarios).
- Evitar sobre-reservas mediante validaciones en PHP y consultas transaccionales.
- Posibilidad de exportar scripts SQL para compartir cambios con otros desarrolladores.

3.4 Seguridad

- Control de acceso por **roles** (estudiante, docente, administrador).
- Autenticación de usuarios mediante sesión en PHP.

4. Diagrama de Arquitectura



Flujos importantes:

1. Estudiante: visualiza tutorías → reserva → actualización en BD → confirmación en interfaz.
2. Docente: crea tutoría → almacenamiento en BD → listado actualizado.
3. Administrador: gestiona usuarios y tutorías → cambios reflejados en BD y frontend.

5. Consideraciones Técnicas

- **Responsive:** Compatible con móviles y escritorio.
- **Performance:** Respuesta en menos de 2 segundos para consultas normales.
- **Colaboración:** Uso de GitHub para versionamiento de código; scripts SQL para replicar base de datos.
- **Escalabilidad:** Arquitectura modular permite agregar funcionalidades futuras (notificaciones, historial de tutorías, reportes).

6. Conclusión

El sistema propuesto es **web-based**, con arquitectura **cliente-servidor**, modular y segura. La combinación de PHP, MySQL, HTML, CSS, Bootstrap y JavaScript garantiza que el desarrollo sea ágil, funcional y compatible con los objetivos del proyecto, permitiendo a cada rol interactuar según sus permisos y ofreciendo una experiencia de usuario consistente y eficiente.