1. Arquitectura Técnica (Tecnología y Despliegue)

El proyecto utiliza un enfoque híbrido (SPA-like). La autenticación, Vistas de Perfil y Navegación Principal se gestionan mediante el *routing* de Laravel, mientras que la Lógica de Juego (Tablero y Teclado) se gestiona con JavaScript para ofrecer una experiencia fluida sin recargas constantes.

Capa	Componente	Tecnología	Responsabilidad
Frontend (Cliente)	Vistas, Interfaz, Lógica de Juego	**HTML5, CSS3 Nativo** (Flexbox/Grid), **JavaScript (ES6+)**	Renderizar la UI, manejar el estado de la partida (tecleo, intentos, temporizador), comunicación asíncrona (`fetch`).
Backend (Servidor)	API REST, Autenticación, ORM	**Laravel 10/11** (PHP)	Gestionar rutas (`routing`), autenticación (sesiones), persistencia de datos (partidas, usuarios, diccionario).
Base de Datos	Persistencia de Datos	**MySQL 8.0**	Almacenar usuarios, la tabla de palabras (diccionario) y los resultados de las partidas (ranking).
Servidor Web	Servicio HTTP	**Apache 2.4**	Servir la aplicación, mapear el dominio `lingo.local`, y usar el archivo `.htaccess` de Laravel.

Servicio (Contenedor)	Imagen Base	Puerto Expuesto (Host:Contenedor)	Función
lingo-apache	app-web (Basado en PHP/Apache)	80:80	Servidor web que expone la aplicación. Contiene la configuración lingo.conf para el dominio local.
lingo-mysql	mysql:8.0	N/A	Base de datos persistente. Usa un volumen para guardar los datos.
lingo-phpmyadmin	phpmyadmin/phpmyadmin	8080:80	Herramienta de gestión visual de la BD.
app-node	node:22	5173:5173	Contenedor para tareas de build del frontend (Vite/npm), aunque el código de juego principal es vanilla JS.

2. Flujo de la Aplicación (Usuario y SPA)

El flujo de la aplicación se divide en tres áreas principales: **Autenticación**, **Juego** y **Datos**.

2.1. Flujo de Autenticación (Login/Registro)

1.

Característica	Servidor (Laravel - Rutas/Middleware)	Cliente (Navegador)
Login/Registro	Laravel Breeze define las rutas /register, /login, y usa middleware para proteger rutas.	El navegador hace una petición completa al servidor. La vista se renderiza por Blade/Inertia.
Inicio de Sesión	Tras validar, Laravel crea la sesión y redirige al usuario a la ruta principal (/).	El navegador recibe la respuesta y recarga la página con la vista principal.
Acceso a Perfil	La ruta /profile está protegida por middleware. Laravel sirve la vista Blade de perfil.	El usuario navega haciendo clic en el enlace, lo que provoca una navegación de página completa (recarga).
Cerrar Sesión	El botón de "Cerrar Sesión" envía un POST a /logout. Laravel destruye la sesión y redirige a /login.	La sesión termina, y la página se recarga en la vista de login.

2.2. Flujo de la Partida (Lógica JavaScript Dinámica)

Una vez que el usuario está en la vista principal del juego (servida por Laravel), toda la interacción de la partida se gestiona mediante JavaScript consumiendo *endpoints* de API. **Esta parte funciona como una SPA, sin recargas.**

.

Paso	Cliente (JavaScript - fetch)	Servidor (Laravel - API Controller)
1. Iniciar Partida	JS captura el clic en "Empezar". fetch (POST) a /api/partida/iniciar.	El Controlador selecciona una palabra del Diccionario (Modelo ORM), crea un registro de Partida en BD.
2. Jugada y Teclado	JS maneja el estado de la fila activa, temporizador y la entrada de las letras en el tablero.	N/A (Lógica en el cliente).
3. Validación	El usuario pulsa Enter. JS detiene el temporizador y ejecuta fetch (POST) a /api/partida/validar. Envía la palabra tecleada, ID de partida y tiempo.	Controlador: 1. Valida la palabra contra el Diccionario. 2. Genera el array de pistas (🟩🟧🟥) comparando la palabra con la secreta. 3. Actualiza el registro de la partida en BD.

4. Feedback Visual	JS recibe el array de pistas. Aplica la animación flip y colorea celdas y el teclado virtual. Si el juego termina (ganar/perder), muestra el mensaje de resultado.	N/A (Lógica en el cliente).

2.3. Flujo de Datos (Ranking)

1. **Activación**: El usuario pulsa el botón de Ranking (Header).
2. **Solicitud**: JavaScript ejecuta fetch a la ruta API protegida /api/ranking.
3. **Procesamiento** (Laravel): El Controlador utiliza el Modelo ORM para consultar y ordenar la tabla Partidas, devolviendo el Top 10 en JSON.
4. **Renderizado**: JavaScript recibe el JSON y genera dinámicamente el contenido del Modal/Overlay de Ranking.