ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFÓRMATICA Y ELECTRONICA

INGENIERIA DE SOFTWARE

APLICACIONES INFORMÁTICAS II

INTEGRANTES:

ALAN NAVIA (6780)

JOHAN GRACIA (7138)

PERIODO ACADEMICO:

OCTUBRE 2024 – FEBRERO 2025

Definición de tecnologías

TABLA COMPARATIVA FRONTEND

ASPECTO	REACT	ANGULAR	VUE.JS	HTML + VANILLA JS
Gestión del estado	Herramientas como Redux o Context API simplifican el manejo de datos dinámicos y sincronización en tiempo real.	Complejo de configurar para aplicaciones interactivas; requiere RxJS para manejo de eventos complejos.	Ofrece herramientas básicas para manejo del estado (Vuex), pero menos robustas que en React.	Requiere programar la lógica manualmente, aumentando la complejidad en proyectos dinámicos.
Accesibilidad	Amplio soporte con librerías como react-aria y configuración simple para cumplir estándares WCAG, ideal para personas con discapacidad auditiva.	Accesibilidad posible, pero con configuraciones más manuales.	Compatible con accesibilidad, pero menos recursos nativos o específicos en comparación con React.	Necesita implementar todo desde cero, lo que requiere más tiempo y es propenso a errores.
Animaciones avanzadas	Librerías como Framer Motion y React Spring permiten animaciones fluidas para lenguaje de señas y gráficos educativos interactivos.	Soporte para animaciones más complejo de implementar; utiliza Angular Animations, pero con más esfuerzo.	Compatible con librerías de animación, pero con menos opciones maduras y soporte en comparación con React.	Animaciones requieren mayor esfuerzo manual o librerías externas complejas.
Componentes reutilizables	Permite crear y reutilizar componentes fácilmente, lo que optimiza el tiempo de desarrollo para múltiples lecciones y actividades.	Componentes reutilizables, pero su estructura es más rígida y compleja.	También permite reutilización, aunque el ecosistema es menos robusto que React.	No hay estructura clara para reutilización; componentes deben codificarse nuevamente.
Integración con APIs	Fácil integración con APIs de lenguaje de señas, reconocimiento de gestos y	Compatible con APIs, pero la configuración es más tediosa y	Menos intuitivo para manejar integraciones complejas con APIs externas.	Integración requiere programar lógica y manejo de

	servicios de texto a voz.	menos flexible que en React.		errores manualmente.
Ecosistema de herramientas	Comunidad y recursos amplios; soporte para herramientas específicas como React Helmet para accesibilidad y React Three Fiber para gráficos 3D interactivos.	Ecosistema robusto, pero menos especializado en herramientas específicas para objetos de aprendizaje y accesibilidad.	Comunidad activa, pero menos enfoque en herramientas avanzadas para aprendizaje inclusivo y visual.	Comunidad básica sin soporte avanzado para herramientas específicas de accesibilidad o aprendizaje interactivo.
Tiempo de desarrollo	Desarrollo rápido gracias a su sintaxis sencilla, modularidad y librerías específicas para objetos de aprendizaje interactivos.	Mayor tiempo inicial debido a su curva de aprendizaje compleja y configuraciones obligatorias.	Tiempo de desarrollo moderado, aunque limitado en funcionalidades avanzadas.	Tiempo significativamente mayor al implementar desde cero funcionalidades como animaciones y accesibilidad.
Escalabilidad	Ideal para crecer con el proyecto; permite agregar nuevas funcionalidades o actividades sin comprometer el rendimiento.	Escalable, pero con mayor complejidad en proyectos pequeños o medianos.	Escalable en proyectos pequeños o medianos, pero no tan robusto como React en proyectos complejos.	Dificultades para escalar sin una estructura sólida definida desde el inicio.

TABLA COMPARATIVA BACKEND

Aspecto	**Express.js**	**Django**	**Ruby on Rails**	**Laravel (PHP)**
Simplicidad y flexibilidad	Minimalista y altamente flexible; permite personalizar completamente la arquitectura del backend para casos específicos como aprendizaje accesible.	Estructura rígida basada en convenciones; menos flexible para configuraciones personalizadas.	Ofrece flexibilidad moderada, pero depende de gems específicas para casos particulares.	Basado en convenciones, menos flexible para aplicaciones personalizadas.

Rendimiento	Ligero y rápido; ideal para manejar solicitudes en tiempo real necesarias para sincronizar objetos de aprendizaje y actividades interactivas.	Más pesado debido a su enfoque completo y características integradas.	Rendimiento moderado, pero menos eficiente en comparación con Express.js para aplicaciones en tiempo real.	Rendimiento más lento debido al consumo de recursos y su naturaleza monolítica.
Escalabilidad	Escalable para manejar múltiples usuarios y tareas simultáneamente; bien soportado por herramientas como Clústeres de Node.js.	Escalabilidad sólida, pero más adecuada para aplicaciones monolíticas que para microservicios.	Escalable, pero no tan eficiente como Express.js en arquitecturas distribuidas o microservicios.	Escalable, aunque con mayor complejidad al dividir funcionalidades en microservicios.
Soporte para APIs en tiempo real	Compatible con Socket.IO para WebSockets, ideal para implementar comunicación en tiempo real como videollamadas, chat o animaciones interactivas.	Soporte limitado para WebSockets; requiere librerías externas y mayor configuración.	Posible con gems como ActionCable, pero menos optimizado para alto tráfico.	Soporte para WebSockets mediante librerías como Ratchet, pero más difícil de configurar.
Integración con tecnologías modernas	Fácil integración con bases de datos NoSQL (MongoDB) y SQL (PostgreSQL, MySQL), ideal para almacenar objetos de aprendizaje multimedia y metadatos de accesibilidad.	Compatible con múltiples bases de datos, pero más optimizado para SQL (PostgreSQL).	Compatible con bases de datos SQL, pero menos eficiente para NoSQL.	Principalmente optimizado para bases de datos SQL; integración con NoSQL requiere más trabajo.
Curva de aprendizaje	Curva de aprendizaje baja para desarrolladores con experiencia en JavaScript; ideal para	Curva de aprendizaje más alta debido a su sintaxis y herramientas específicas de Python.	Curva de aprendizaje moderada, pero menos amigable para principiantes.	Curva de aprendizaje moderada; requiere conocimientos avanzados de PHP y su ecosistema.

	proyectos que requieren rápido desarrollo y adaptación.			
Comunidad y recursos	Gran comunidad y ecosistema activo; muchas librerías y middleware específicos para manejar accesibilidad, seguridad, y gestión de usuarios.	Comunidad robusta, con gran cantidad de recursos educativos, aunque centrados en aplicaciones tradicionales.	Comunidad más pequeña y menos recursos dedicados a aplicaciones accesibles o aprendizaje interactivo.	Amplia comunidad, pero con menos librerías especializadas para accesibilidad o aprendizaje interactivo.
Costo de infraestructura	Ligero, consume menos recursos en servidores, lo que reduce costos; ideal para entornos interactivos con alta concurrencia.	Más consumo de recursos debido a su enfoque completo y características integradas.	Requiere más recursos para manejar cargas altas en comparación con Express.js.	Consumo moderado de recursos, pero menos eficiente que Express.js para manejar muchas solicitudes concurrentes.
Soporte para accesibilidad	Integración directa con herramientas como APIs de texto a voz, conversión de lenguaje de señas y almacenamiento multimedia para personas con discapacidad auditiva.	Compatible con herramientas de accesibilidad, pero requiere más trabajo manual para configuraciones específicas.	Menos librerías dedicadas a accesibilidad; se necesita esfuerzo adicional para integrar características inclusivas.	Compatible con librerías de accesibilidad, pero no está optimizado para casos complejos como aprendizaje interactivo accesible.
Tiempo de desarrollo	Desarrollo rápido debido a su simplicidad y amplia gama de middleware preconstruido.	Desarrollo más lento debido a la configuración inicial y la integración de herramientas avanzadas.	Desarrollo moderado, pero con mayor esfuerzo para configuraciones personalizadas.	Más lento al manejar configuraciones específicas y acceso a tecnologías modernas como NoSQL o WebSockets.

TABLA COMPARATIVA BASES DE DATOS

Aspecto	PostgreSQL	MySQL	MongoDB	SQLite
Compatibilidad con datos estructurados y no estructurados	Soporta datos relacionales (SQL) y almacenamiento JSON nativo, ideal para mezclar información estructurada y contenidos multimedia no estructurados.	Optimizado principalmente para datos relacionales, con soporte JSON limitado.	Diseñado para datos no estructurados, pero menos eficiente para relaciones complejas entre datos.	Solo soporta datos relacionales; no es adecuado para manejar contenido multimedia o JSON de manera eficiente.
Escalabilidad	Escalable horizontal y verticalmente, con soporte para grandes volúmenes de datos y consultas complejas, ideal para gestionar recursos educativos multimedia.	Escalable, pero menos eficiente en consultas complejas debido a su diseño optimizado para velocidad sobre consistencia.	Escalabilidad horizontal robusta, pero con limitaciones en consultas transaccionales complejas.	No es adecuado para aplicaciones que requieren manejar grandes volúmenes de datos; está limitado a entornos pequeños.
Consultas complejas	Soporta consultas avanzadas como CTE (Common Table Expressions), ventanas y funciones analíticas, ideales para análisis detallados del progreso de los usuarios.	Consultas menos avanzadas; carece de funcionalidades como índices GIN/GIN para búsquedas rápidas en JSON.	Consultas avanzadas más limitadas debido a su modelo orientado a documentos.	Consultas básicas, no apto para análisis o búsquedas avanzadas.
Soporte para multimedia	Compatible con extensiones como PostGIS para almacenar y consultar datos multimedia o geoespaciales, y almacenamiento	Soporte básico para datos binarios, pero no optimizado para multimedia o extensiones avanzadas como PostGIS.	Ideal para almacenar multimedia en formato de documentos JSON, pero no es eficiente para consultas	Soporte muy limitado para datos multimedia; no es práctico para proyectos de esta escala.

	eficiente de datos binarios (BLOB).		relacionales complejas.	
Consistencia y fiabilidad	Alta consistencia y fiabilidad gracias a transacciones ACID completas, esencial para garantizar que los datos educativos y de usuario se gestionen correctamente.	Soporta ACID, pero con limitaciones en algunas configuraciones que afectan la consistencia bajo alta carga.	Consistencia eventual, lo que puede ser un problema para aplicaciones que requieren datos educativos sincronizados y precisos.	Soporta ACID, pero no es apto para manejar múltiples usuarios concurrentes en proyectos grandes.
Compatibilidad con accesibilidad	Permite almacenar datos complejos como metadatos de accesibilidad (subtítulos, descripciones de lenguaje de señas, etc.) y recuperarlos eficientemente.	Menor flexibilidad para manejar metadatos complejos debido a su diseño tradicional relacional.	Bueno para metadatos no estructurados, pero ineficiente para relaciones o búsquedas que combinen múltiples fuentes de datos.	Limitado para manejar y optimizar metadatos o datos multimedia requeridos para accesibilidad.
Extensiones	Compatible con extensiones avanzadas como PostGIS (datos geoespaciales), pg_trgm (búsqueda de texto), ideal para herramientas educativas personalizadas.	Extensiones limitadas comparadas con PostgreSQL; carece de soporte avanzado para búsquedas complejas y datos especializados.	No soporta extensiones avanzadas como PostGIS; está más orientado a documentos y análisis básicos.	No soporta extensiones; extremadamente limitado para personalizar funcionalidades avanzadas.
Integración con tecnologías modernas	Compatible con múltiples lenguajes y frameworks modernos; ideal para trabajar con APIs que manejan accesibilidad y multimedia.	Amplia integración con lenguajes modernos, pero menos soporte para funcionalidades avanzadas como JSON y búsquedas complejas.	Muy integrado con tecnologías modernas, especialmente para aplicaciones que manejan documentos JSON.	Menor integración con tecnologías modernas; diseñado principalmente para aplicaciones locales o de pequeño alcance.
Costo de implementación	Open source con funcionalidades empresariales incluidas; ideal	Open source en su versión básica, pero algunas funciones	Open source, pero con costos potenciales en servicios alojados	Open source y gratuito, pero no adecuado para escalar o manejar

	para proyectos educativos accesibles con presupuestos limitados.	avanzadas están restringidas a versiones comerciales (MySQL Enterprise).	como MongoDB Atlas para manejar grandes volúmenes de datos.	requisitos avanzados.
Rendimiento general	Alto rendimiento en consultas complejas y operaciones de escritura/lectura para grandes cantidades de datos educativos.	Mejor rendimiento en operaciones de lectura simples, pero menos eficiente en consultas complejas.	Bueno para grandes volúmenes de datos no relacionales, pero rendimiento bajo en sistemas híbridos o relacionales complejos.	Rendimiento limitado a aplicaciones pequeñas o con pocos usuarios concurrentes.

CONCLUSIONES

PostgreSQL es la mejor opción para proyectos que integren objetos de aprendizaje en entornos virtuales accesibles debido a:

- **Capacidad para manejar datos híbridos (relacionales y no relacionales)**, ideal para objetos de aprendizaje interactivos y metadatos de accesibilidad.
- **Consultas avanzadas y extensiones especializadas**, como PostGIS y pg_trgm, que amplían sus capacidades más allá de otras bases de datos.
- **Alta consistencia y fiabilidad**, esencial para datos educativos que requieren precisión y sincronización.
- **Ecosistema open source completo**, sin necesidad de licencias adicionales para funcionalidades avanzadas.

En comparación con otras bases de datos, PostgreSQL combina flexibilidad, rendimiento y escalabilidad, lo que lo hace ideal para proyectos educativos inclusivos y accesibles.

Express.js sobresale en el desarrollo de objetos de aprendizaje para entornos virtuales accesibles gracias a:

- **Alto rendimiento** para manejar solicitudes en tiempo real y actividades interactivas.
- **Flexibilidad** para personalizar el backend según las necesidades específicas del proyecto.
- **Compatibilidad con tecnologías modernas**, como WebSockets y bases de datos NoSQL, esenciales para contenido multimedia dinámico.

- **Eficiencia en costos e infraestructura**, ideal para aplicaciones escalables e interactivas.

Esto lo convierte en una opción superior para crear espacios de comunicación y aprendizaje inclusivos, en comparación con sistemas más rígidos o monolíticos como Django o Laravel.

React ofrece ventajas claras frente a otros sistemas:

- **Mejor gestión del estado** para entornos dinámicos e interactivos.
- **Soporte avanzado de accesibilidad** con menos esfuerzo de configuración.
- **Librerías especializadas** para animaciones y gráficos educativos.
- Mayor **flexibilidad y escalabilidad** para ampliar el entorno con nuevos objetos de aprendizaje.

Estas características hacen que React sea la opción más adecuada para desarrollar espacios inclusivos y accesibles, diseñados específicamente para personas con discapacidad auditiva.