Selección de lenguajes

Con base a la necesidad de trabajar de manera integrada con Web3 en lo procedente al desarrollo del proyecto se han seleccionado los siguientes lenguajes para ser utilizados en el desarrollo tanto del backend y frontend.

Frontend: JavaScript impulsado por React en asociación con SCSS.

Backend: Motoko.

Base de datos: Canisters nativos de internet computer.

Las razones de estas selecciones se presentan a continuación:

1. Motoko: Optimizado para IC

- **Integración Nativa**: Motoko es un lenguaje de programación diseñado específicamente para IC. Esto significa que tiene optimizaciones y características integradas para interactuar directamente con el entorno de IC, como la gestión de estados estables y la persistencia de datos en canisters.
- **Simplicidad y Seguridad**: Motoko proporciona abstracciones que facilitan la programación segura y la gestión de la memoria, lo que es crucial en un entorno descentralizado donde los errores pueden tener grandes repercusiones.
- **Ecosistema y Soporte**: Motoko está respaldado por la Fundación DFINITY, lo que asegura soporte continuo y mejoras específicas para IC así como una basta documentación de uso y una comunidad activa de desarrolladores.

2. React: Potente Framework para Frontend

- **Componentes**: React permite construir interfaces de usuario modulares y reutilizables, facilitando el desarrollo y mantenimiento de aplicaciones complejas.
- **Ecosistema Amplio**: Con una basta comunidad y una amplia gama de bibliotecas y herramientas, React ofrece soluciones para casi cualquier necesidad de frontend.
- **Rendimiento**: React es altamente eficiente en la actualización del DOM, lo que resulta en una experiencia de usuario rápida y fluida.

3. Canisters: Contratos Inteligentes en IC

- **Persistencia de Datos**: Los canisters pueden almacenar datos de forma persistente y distribuirlos automáticamente en la red, lo que asegura alta disponibilidad y redundancia.
- **Escalabilidad**: IC está diseñado para escalar automáticamente, y los canisters pueden aprovechar esta escalabilidad sin necesidad de configuración adicional por parte del desarrollador.
- **Descentralización**: Al utilizar canisters, las aplicaciones pueden beneficiarse de las propiedades de seguridad y resistencia a la censura propias de una red descentralizada.

4. Beneficios Combinados

- **Sin Dependencias en Servidores Centrales**: La combinación de React para el frontend y Motoko para la lógica del backend dentro de canisters elimina la necesidad de servidores centrales tradicionales, reduciendo los puntos de fallo.
- **Seguridad Mejorada**: Con los datos almacenados de manera segura en canisters y gestionados por código seguro en Motoko, se reducen las superficies de ataque y se mejora la integridad de los datos.
- **Experiencia de Usuario Mejorada**: React facilita la creación de interfaces dinámicas y responsivas que pueden comunicarse eficientemente con canisters en tiempo real, ofreciendo una experiencia de usuario superior.

5. Ecosistema de Internet Computer

- **Interoperabilidad**: IC facilita la interoperabilidad entre diferentes canisters y servicios dentro de su red, permitiendo la creación de aplicaciones compuestas y servicios interconectados.
- **Desarrollo Continuo**: La comunidad y el equipo de desarrollo detrás de IC están constantemente mejorando la plataforma, ofreciendo nuevas funcionalidades, optimizaciones y herramientas para desarrolladores.

Ejemplo de Integración

Para ilustrar cómo estos componentes pueden trabajar juntos, es posible definir un flujo simplificado de cómo una aplicación puede operar:

1. **Frontend en React**: Se construye la interfaz de usuario de la aplicación en React. Los componentes de React interactúan con el backend a través de llamadas a canisters.
2. **Backend en Motoko**: Se escribe la lógica de programa y gestión de datos en Motoko, desplegando esta lógica en canisters en la red IC. donde, los datos se almacenan y se procesan de manera descentralizada.
3. **Comunicación Eficiente**: Se utilizan bibliotecas como agent en la aplicación React para realizar llamadas a los canisters, obteniendo y enviando datos de forma segura y eficiente.