

**INSTITUTO TECNOLOGICO DE CHILPANCINGO**

**Cómputo En La Nube Y Grandes Datos**

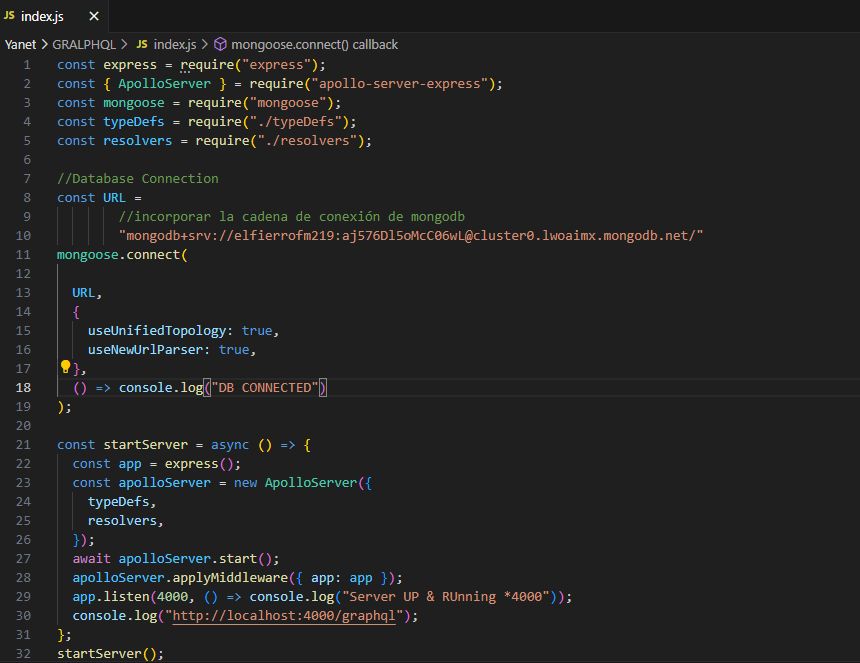
Practica GRALPHQL

**Alumno:**

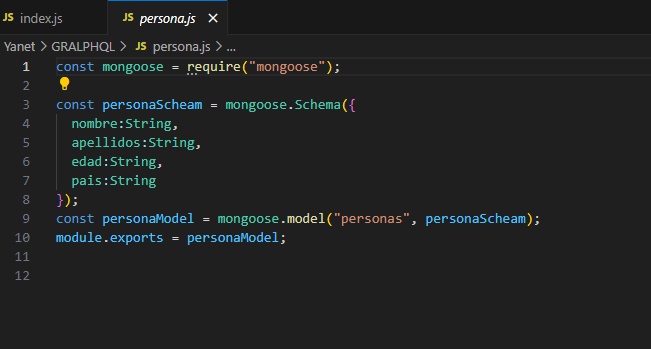
Fierro Monroy Isaac Antonio

Lunes 13 de mayo de 2024

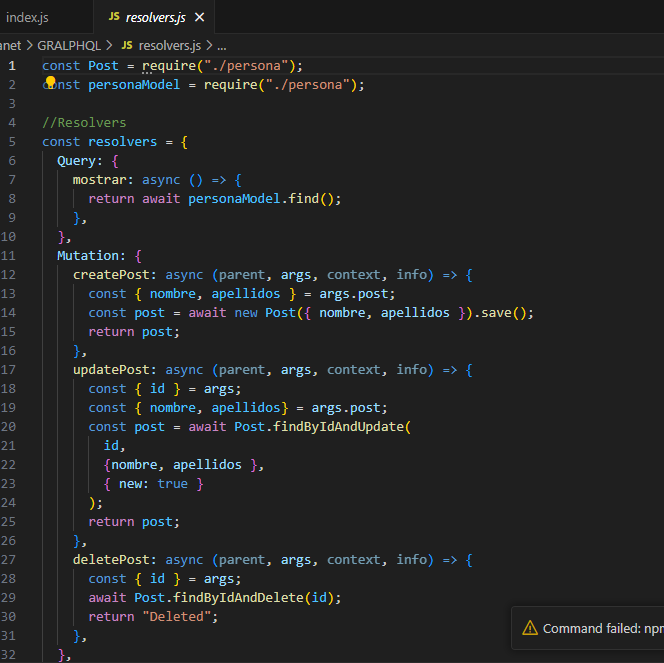
# GRALPHQL CON CAPA

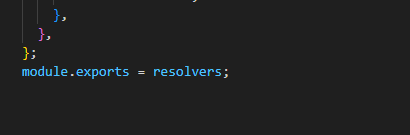


Este código intenta configurar un servidor GraphQL utilizando Express y Apollo Server, conectándose a una base de datos MongoDB utilizando Mongoose. Sin embargo, hay algunos errores tipográficos que deben corregirse para que el código funcione correctamente.

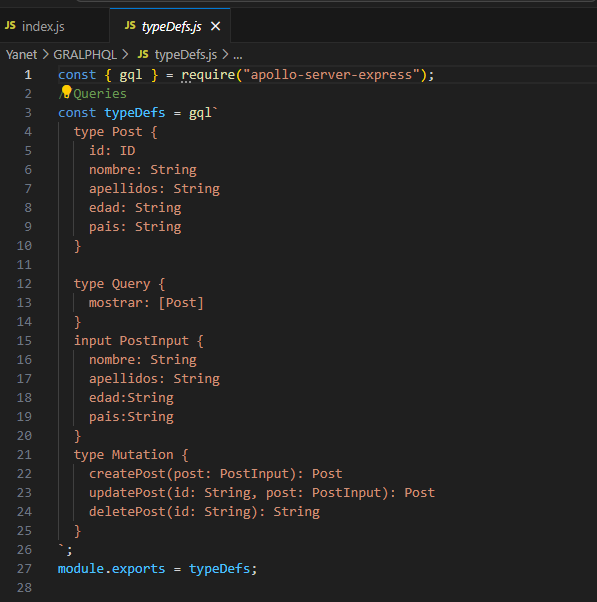


Este código establece un esquema y un modelo de Mongoose para la entidad "persona", que luego puede ser utilizado para interactuar con la base de datos MongoDB.

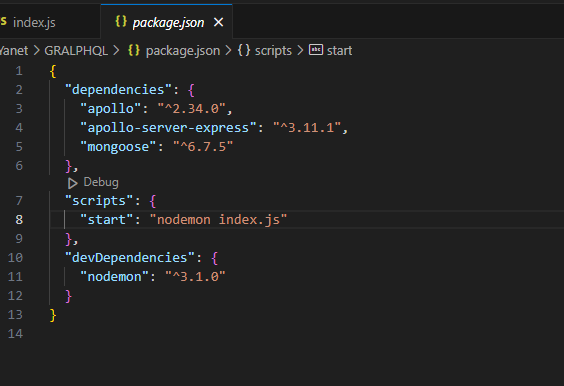


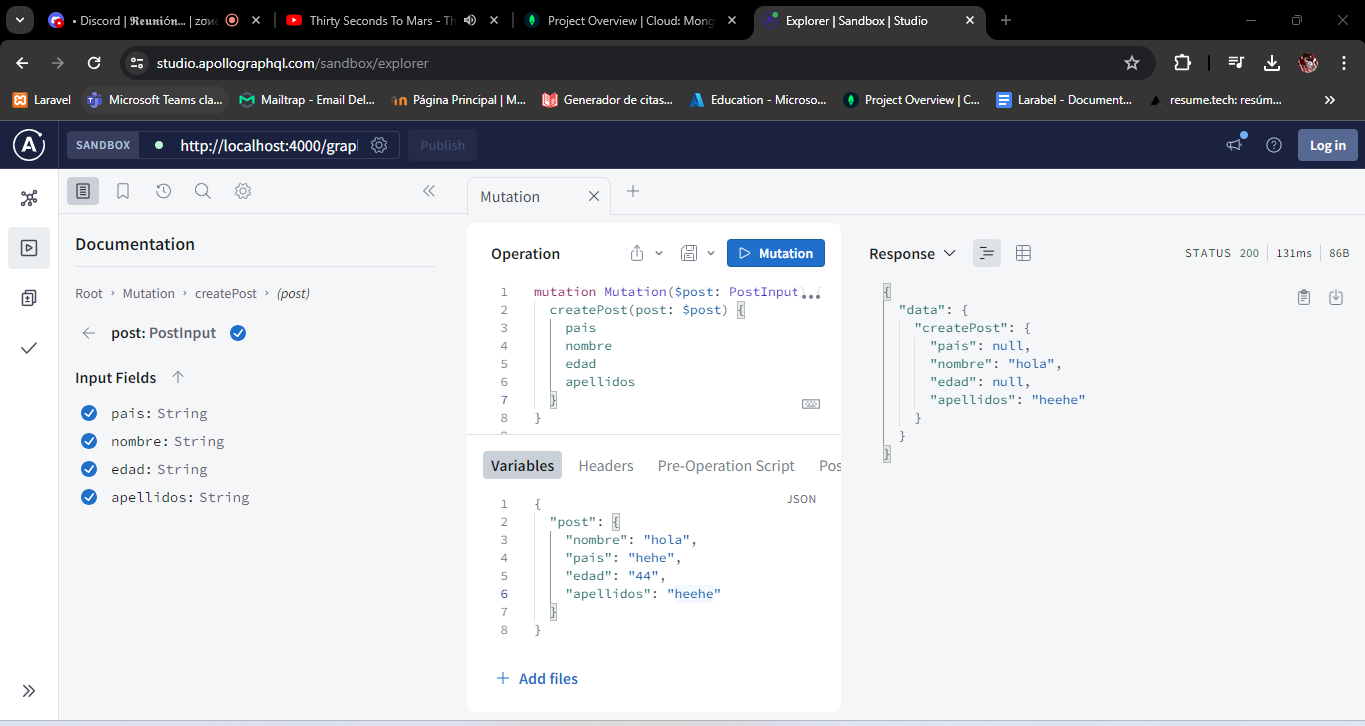


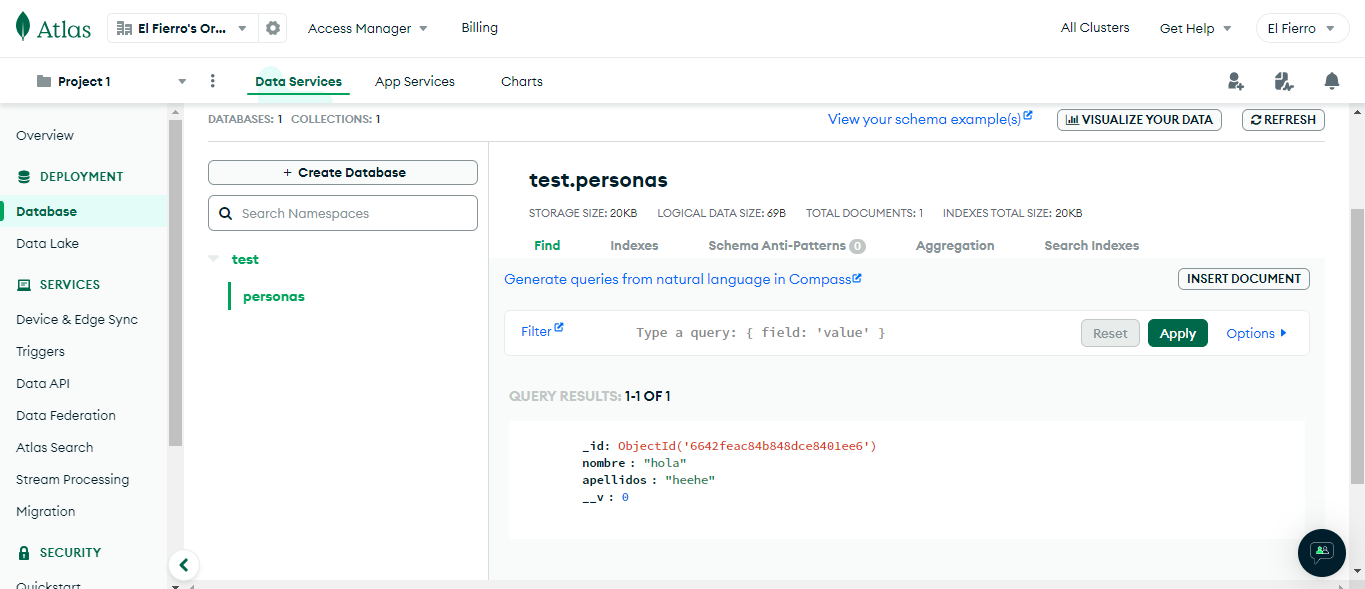
Estos resolutores proporcionan la lógica necesaria para realizar consultas y modificaciones en la base de datos MongoDB a través de tu API GraphQL.



Este código define los tipos de datos y operaciones que se pueden realizar en tu esquema GraphQL utilizando GraphQL Schema Definition Language (SDL).

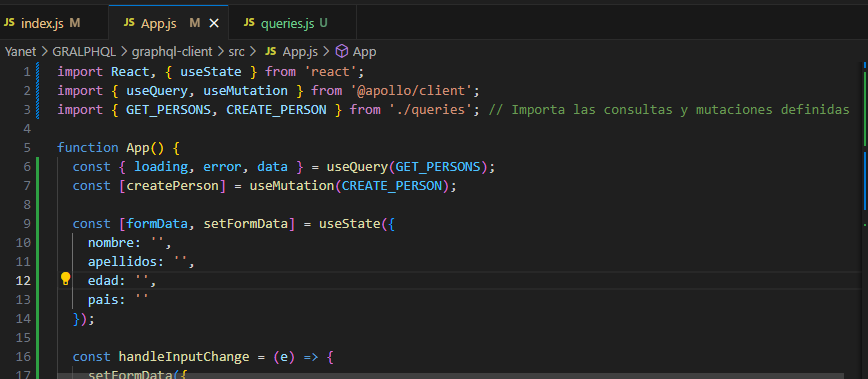


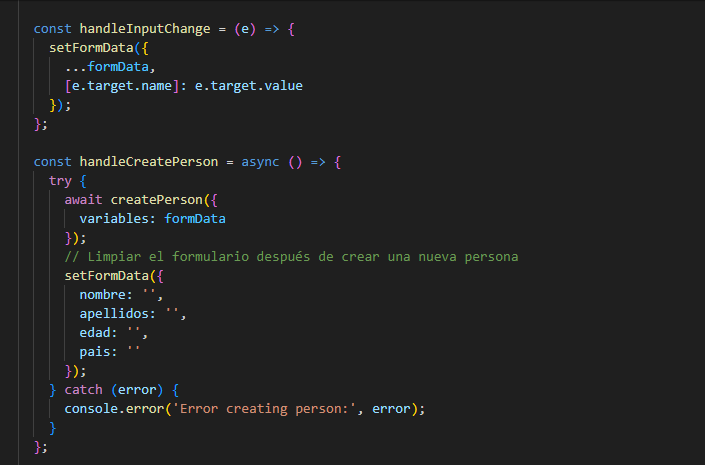


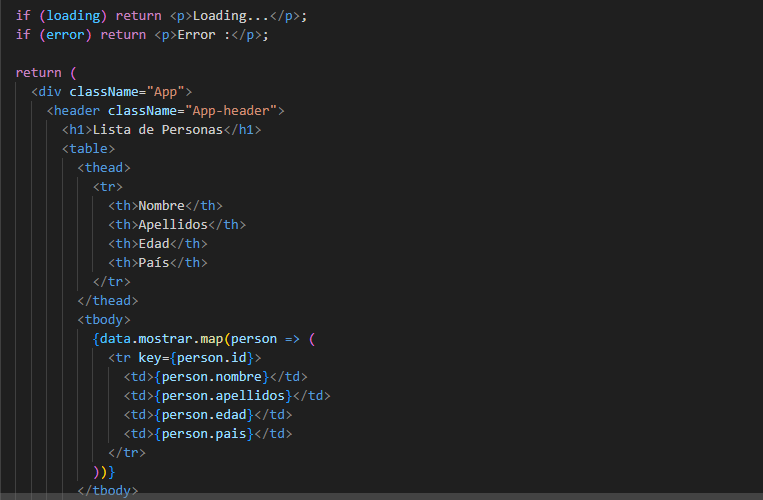


Esto solo usando el backend.

El frontend seria de esta manera.

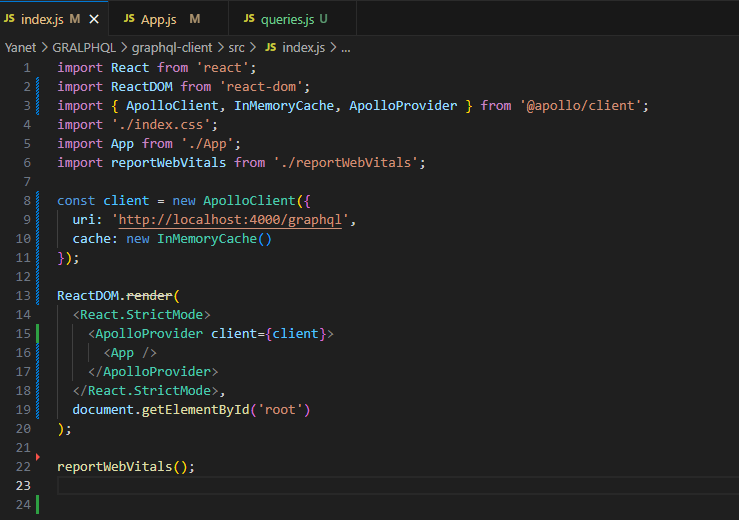




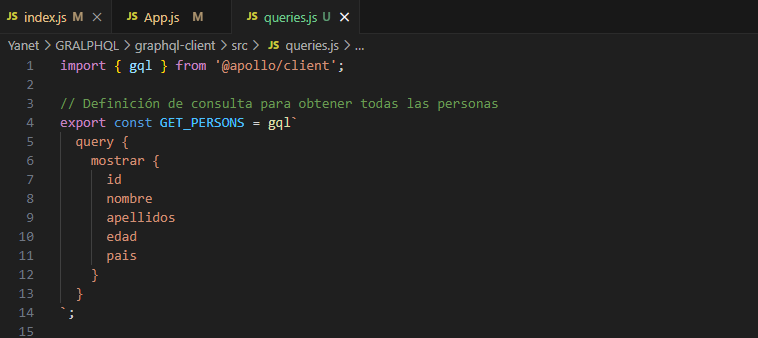


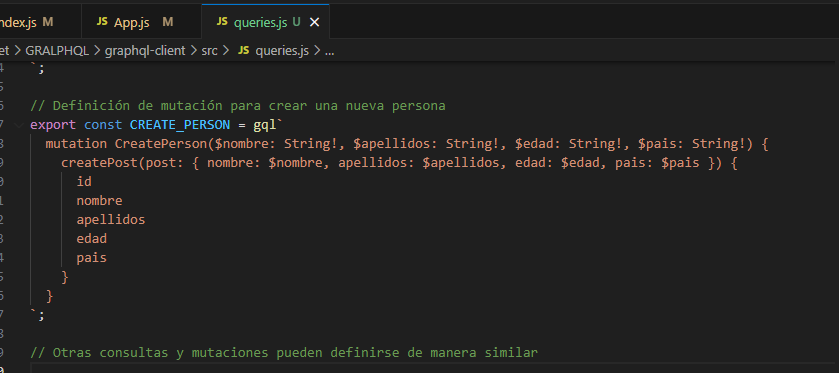


El componente App muestra una lista de personas y permite al usuario crear nuevas personas a través de un formulario. Utiliza Apollo Client para realizar consultas y mutaciones GraphQL en el backend.

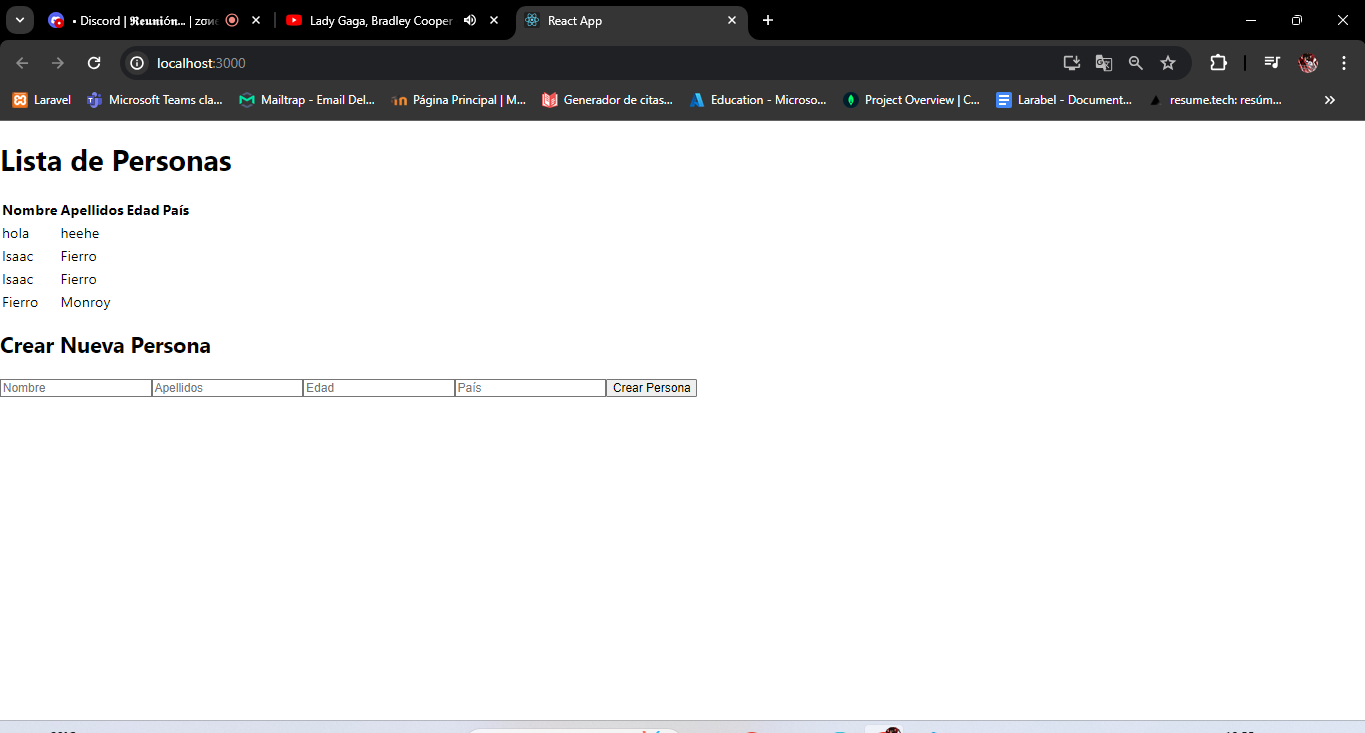


Este archivo configura Apollo Client y luego renderiza tu aplicación dentro del div con el id "root", asegurando que tu componente App tenga acceso a los datos GraphQL proporcionados por Apollo Client. Si tienes alguna pregunta o necesitas más detalles sobre algún aspecto específico, ¡no dudes en preguntar

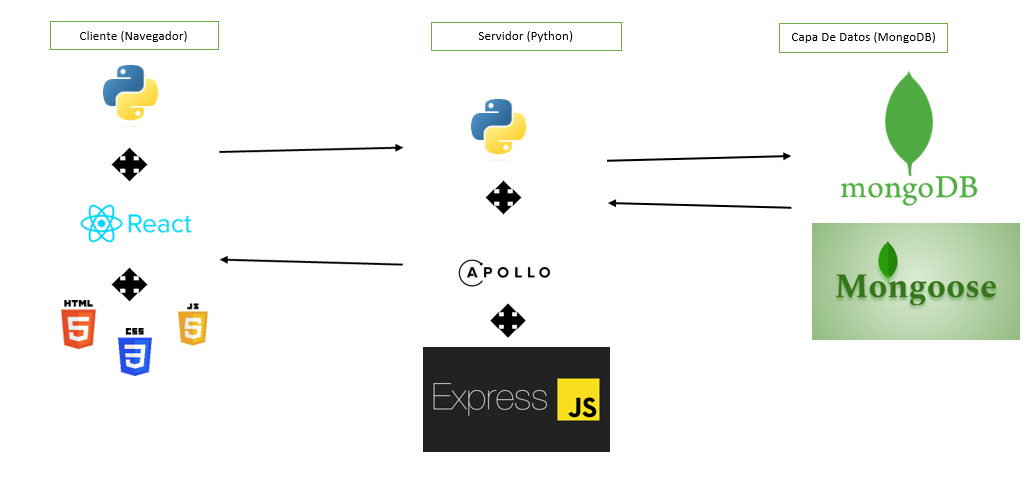




Ambas consultas y mutaciones están definidas utilizando la función gql proporcionada por Apollo Client para interpretar cadenas GraphQL. Estas consultas y mutaciones se pueden importar y utilizar en tus componentes de React para interactuar con tu servidor GraphQL y tu base de datos.



# Estructura



# CONCLUSION

El desarrollo de una aplicación web moderna utilizando GraphQL implica la integración de un cliente en el frontend y un servidor en el backend. En este proyecto, se ha utilizado Apollo Client para el frontend y Apollo Server Express para el backend, junto con otras tecnologías como React, Express, Mongoose y MongoDB. A continuación, se presentan las conclusiones clave de este proyecto:

* Eficiencia en el intercambio de datos: GraphQL ofrece una forma eficiente de intercambiar datos entre el cliente y el servidor al permitir que el cliente solicite solo los campos necesarios en cada consulta. Esto reduce el ancho de banda necesario y mejora el rendimiento de la aplicación.
* Facilidad de uso con Apollo Client: Apollo Client facilita la integración de GraphQL en aplicaciones React al proporcionar hooks como useQuery y useMutation, que simplifican la realización de consultas y mutaciones GraphQL desde los componentes de React.
* Flexibilidad en el diseño de esquemas y resolutores: Con Apollo Server Express, se puede diseñar fácilmente un esquema GraphQL que se ajuste a las necesidades específicas de la aplicación. Los resolutores permiten definir la lógica para resolver los campos del esquema, lo que proporciona flexibilidad en la forma en que se acceden y manipulan los datos.
* Escalabilidad con MongoDB y Mongoose: La combinación de MongoDB y Mongoose ofrece una solución escalable y flexible para almacenar y manipular datos en el backend. MongoDB es una base de datos NoSQL que se integra bien con GraphQL, y Mongoose simplifica la interacción con la base de datos al proporcionar un modelado de datos claro y estructurado.
* Experiencia de usuario mejorada: La implementación de GraphQL en el frontend y el backend permite una experiencia de usuario mejorada al proporcionar datos de manera eficiente y permitir operaciones complejas de lectura y escritura. Esto se traduce en una interfaz de usuario más receptiva y una aplicación más rápida y fluida en general.

En resumen, este proyecto demuestra las ventajas de utilizar GraphQL en el desarrollo de aplicaciones web modernas, proporcionando una experiencia de desarrollo eficiente, una interfaz de usuario mejorada y una arquitectura escalable y flexible en el frontend y el backend.