Things to Know

Netflix-nextjs

Prismadb.ts

* **const client = global.prismadb || new PrismaClient();** creates a new PrismaClient instance and assigns it to the **client** variable. If the global **prismadb** variable is defined, it will use that instead of creating a new instance. This is useful because you typically want to reuse the same PrismaClient instance across your application to take advantage of connection pooling.
* **if (process.env.NODE\_ENV !== 'production') global.prismadb = client** sets the **global.prismadb** variable to the **client** instance if the **NODE\_ENV** environment variable is not set to **'production'**. This allows the **client** instance to be reused across multiple requests during development, which can help with performance.

Prisma/schema.prisma

1. user User @relation(fields: [userId], references: [id], onDelete: Cascade)

* **User** is the name of the model being defined.
* **@relation** is a decorator that specifies the relation between this model and another model.
* **fields: [userId]** specifies that this relation is based on the **userId** column in this model. This means that there is a foreign key relationship between the **User** model and another model (not shown here) where **userId** is the foreign key.
* **references: [id]** specifies that the foreign key in the related model references the **id** column in that model. This means that the related model has a primary key column named **id** that is used as the target of the foreign key relationship.
* **onDelete: Cascade** specifies that when a **User** is deleted, all related records in the related model should also be deleted. This is known as cascading deletion and is a common strategy for maintaining data integrity in databases.

1. @@unique([provider, providerAccountId])

* **@@unique** is a Prisma schema decorator that specifies that a unique constraint should be created on a combination of columns in the database table.
* **[provider, providerAccountId]** is an array that specifies the combination of columns that should be unique. In this case, the unique constraint will be created on the **provider** and **providerAccountId** columns in the table.
* A unique constraint ensures that no two rows in the table have the same combination of values in the specified columns. This can be useful for enforcing data integrity and preventing duplicate data from being inserted into the table.

SWR – React Hook for Data Fetching

The name “SWR” is derived from stale-while-revalidate, a HTTP cache invalidation strategy popularized by [HTTP RFC 5861(opens in a new tab)](https://tools.ietf.org/html/rfc5861). SWR is a strategy to first return the data from cache (stale), then send the fetch request (revalidate), and finally come with the up-to-date data.