

И вот какие могут быть возможные решения.

- Низкие баллы (8-12): простой REST, SOAP или JSON-RPC API с возможной согласованностью и базовой обработкой ошибок.

Более близки к свойствам BASE, поскольку рассчитаны на согласованность в конечном счете и позволяют более гибко хранить и извлекать данные.

Может не подходить для приложений, требующих строгих транзакционных гарантий и немедленной согласованности.

- Средние баллы (13-20): REST, GraphQL, gRPC API с сильной согласованностью, расширенной обработкой ошибок и некоторыми функциями реального времени.

Смесь свойств ACID и BASE, поскольку они обеспечивают сильную согласованность, а также включают некоторые функции реального времени.

Эти варианты API подходят для приложений, которым необходим баланс между надежной согласованностью данных и гибкостью в обработке различных типов данных, запросов и обновлений.

- Высокие баллы (21-24): сочетание API REST, GraphQL, gRPC с WebSocket или Server-Sent Events для обновления в реальном времени, строгой согласованности и устойчивости системы.

Более близки к свойствам ACID, поскольку подчеркивают строгую согласованность, гарантии транзакций и отказоустойчивость.

Эти варианты API подходят для приложений, требующих обновлений в реальном времени, немедленной согласованности и высокой доступности.

Однако эти варианты могут быть менее гибкими в плане хранения и извлечения данных по сравнению с API со свойствами BASE, так как в них приоритет отдается согласованности и надежности.

Помните, что выбор типа API должен основываться на уникальных требованиях вашего приложения, и важно выбрать тот тип API, который наилучшим образом отвечает этим требованиям. Будьте готовы вносить коррективы по мере развития и масштабирования вашего приложения и всегда учитывайте компромиссы при выборе наиболее подходящего дизайна API.

Мы просто предоставили вам пример того, как может выглядеть архитектурное решение.