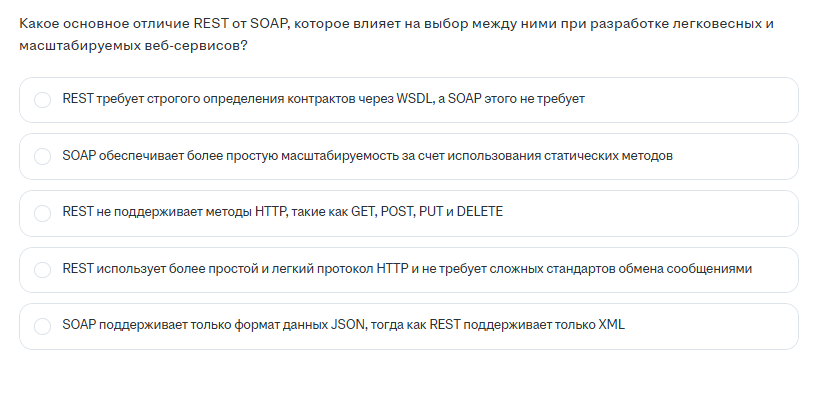
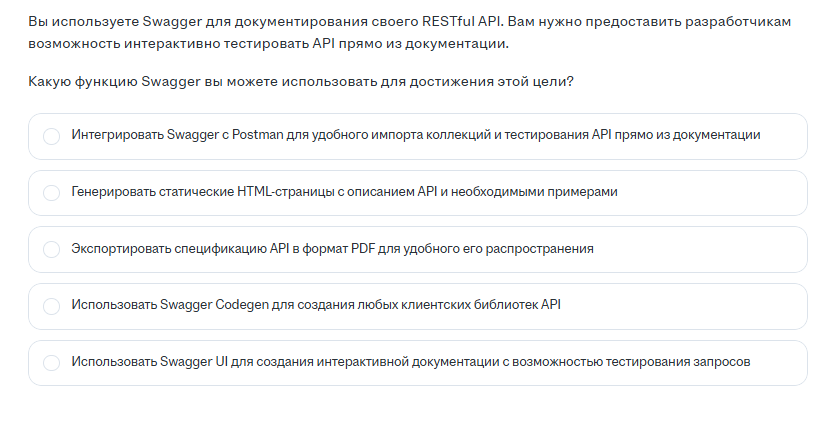
1)



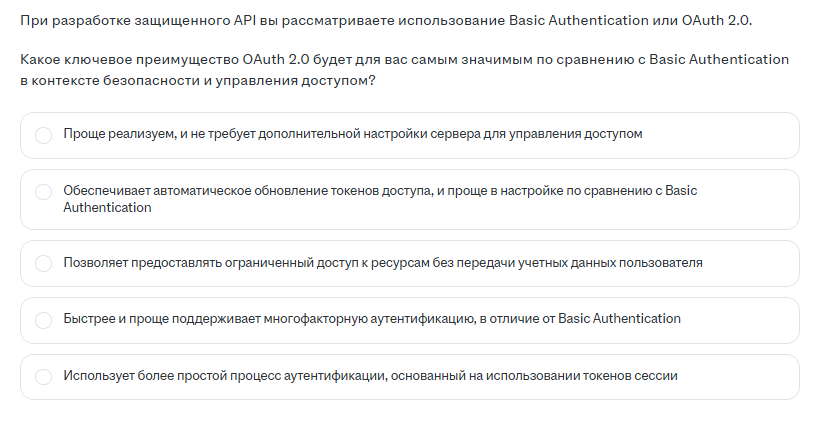
**REST использует более простой и легкий протокол HTTP и не требует сложных стандартов обмена сообщениями**

2)



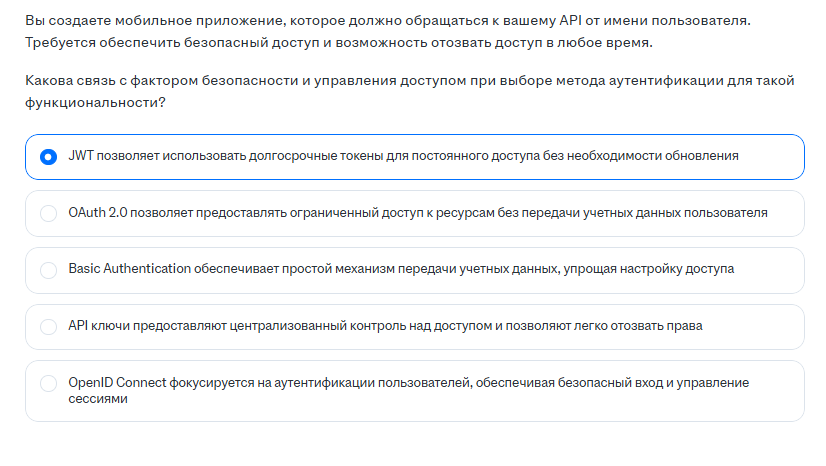
**Использовать Swagger UI для создания интерактивной документации с возможностью тестирования запросов**

3)



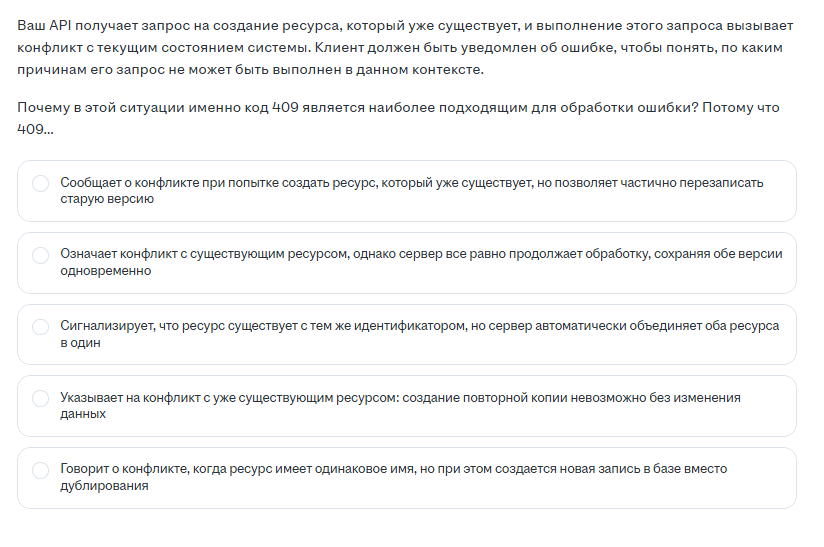
**Позволяет предоставлять ограниченный доступ к ресурсам без передачи учетных данных пользователя**

4)



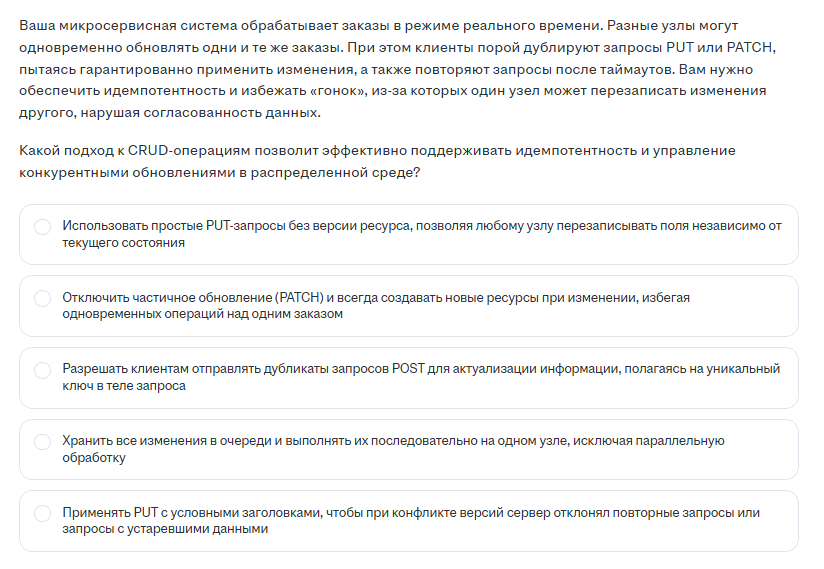
**OAuth 2.0 позволяет предоставлять ограниченный доступ к ресурсам без передачи учетных данных пользователя**

5)



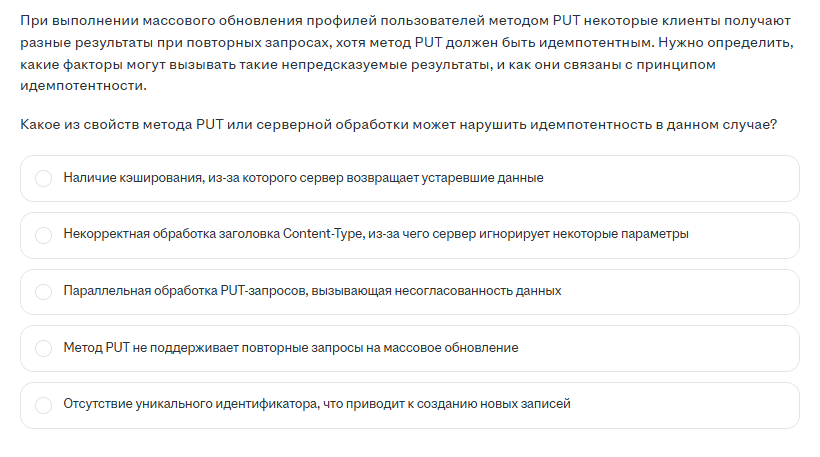
**Указывает на конфликт с уже существующим ресурсом: создание повторной копии невозможно без изменения данных**

6)



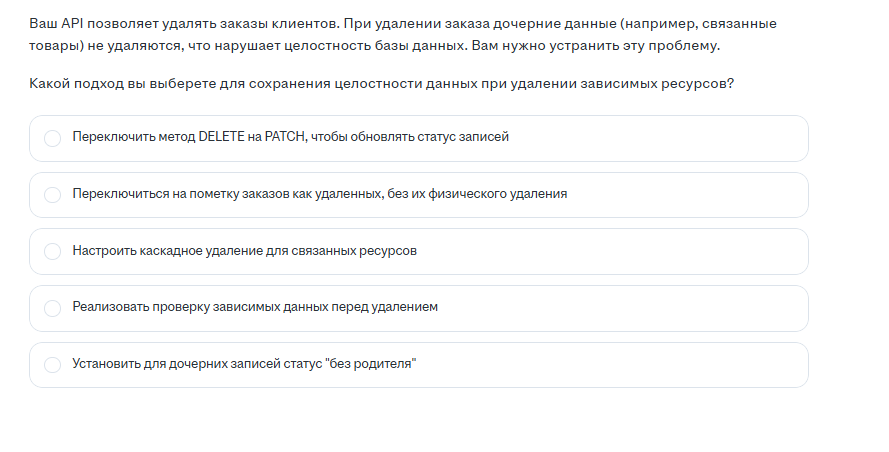
**Применять PUT с условными заголовками, чтобы при конфликте версий сервер отклонял повторные запросы или запросы с устаревшими данными**

7)



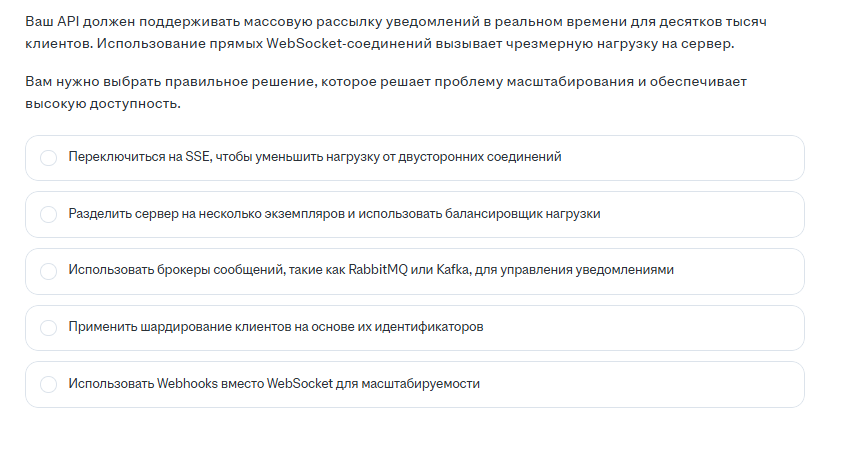
**Параллельная обработка PUT-запросов, вызывающая несогласованность данных**

8)



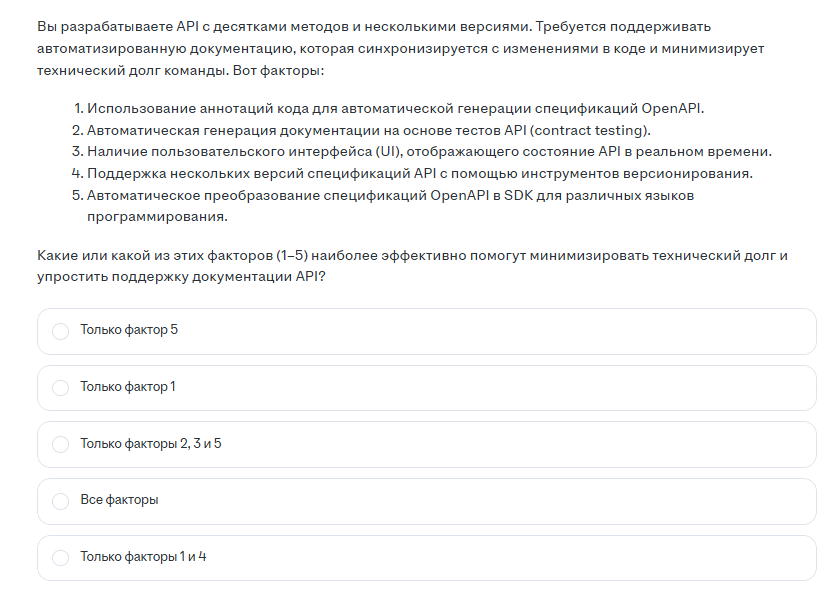
**Настроить каскадное удаление для связанных ресурсов**

9)



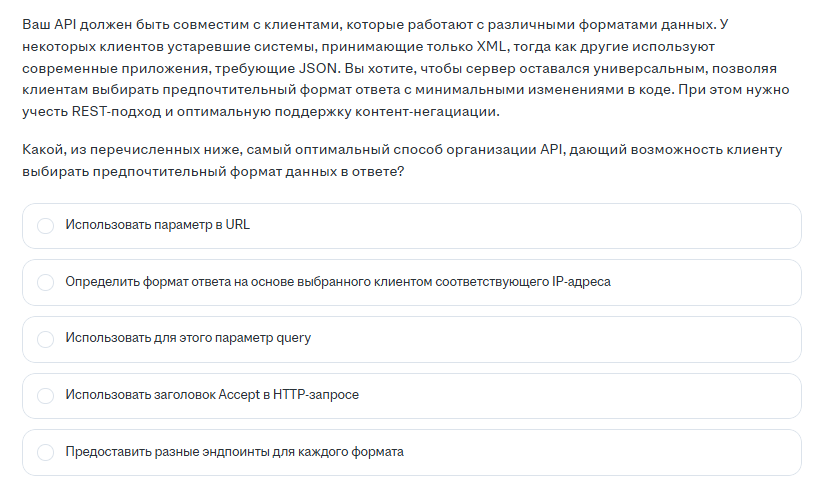
**Применить брокер сообщений (например, RabbitMQ, Kafka) для асинхронного управления уведомлениями**

10)



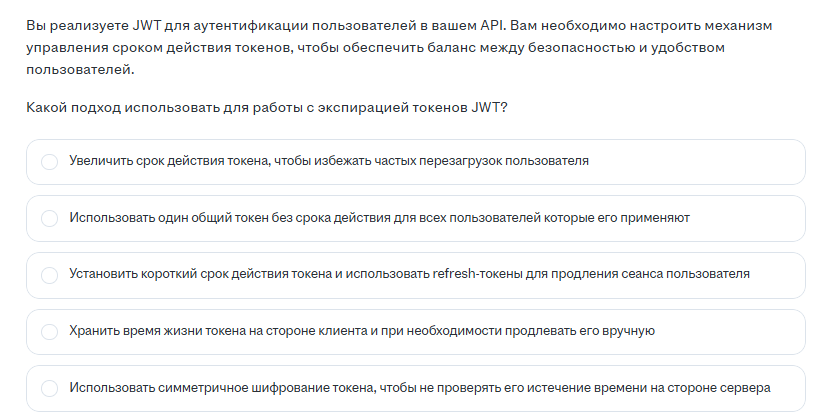
**только 1 фактор и 2 , но тут нет такого варианта так что только 1**

11)



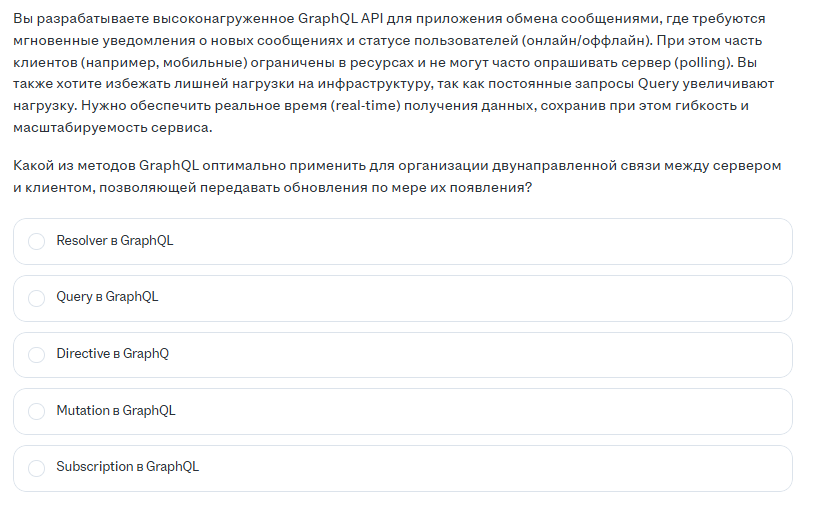
**Использовать заголовок Accept в HTTP-запросе**

12)



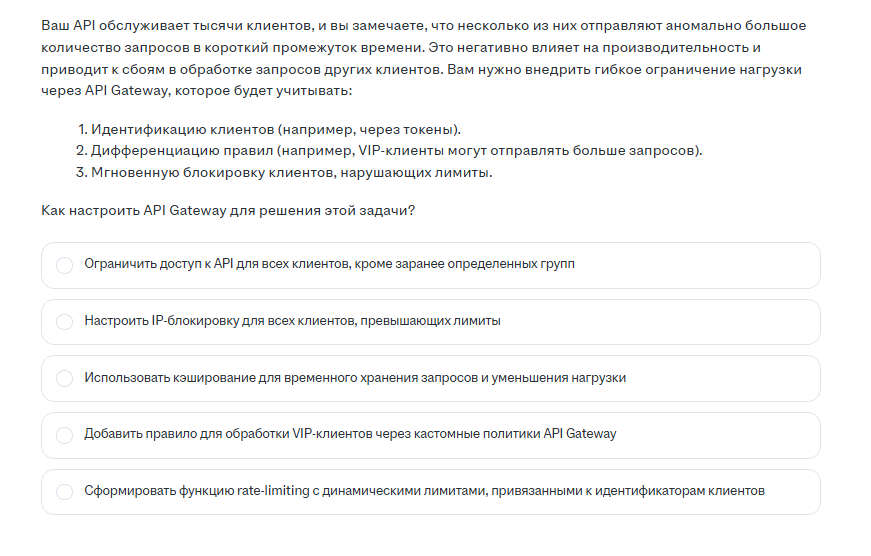
использовать **токены с ограниченным сроком действия и refresh-токены для продления сеанса**.

13)



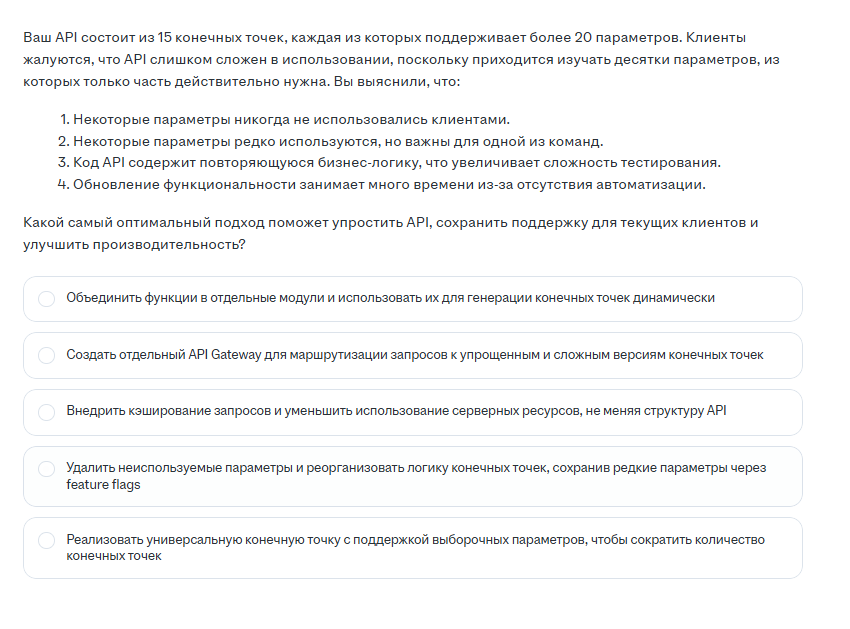
**Subscription в GraphQL**

14)



**Сформировать функцию rate-limiting с динамическими лимитами, привязанными к идентификаторам клиентов**

15)



**Удалить неиспользуемые параметры и реорганизовать логику конечных точек, сохранив редкие параметры через feature flags**