1. **Проанализируйте диаграмму системы и её описание.** Решите, какую часть системы имеет смысл закешировать.

Поскольку основной пул проблем сосредоточен в MES, ее мы и будем кешировать в первую очередью. В дальнейшем, имеет смысл внедрить кеширование в систему интернет-магазина.

1. **Добавьте в файл раздел «Мотивация».** Опишите здесь, почему вы предлагаете внедрить кеширование, какие проблемы оно должно решить и какие элементы системы вы планируете включить в кеширование.

В нашем случае внедрение кеширования в первую очередь позволит решить проблемы с производительностью:

Скорость доступа:

Кеширование позволяет значительно ускорить доступ к данным, так как браузеры, серверы и приложения могут обращаться к кешу вместо извлечения данных непосредственно из базы данных или удаленного сервиса.

Снижение времени отклика:

При использовании кеша пользователи могут получать данные гораздо быстрее, что улучшает общий пользовательский опыт. Это критично для веб-приложений, где задержки могут негативно сказываться на взаимодействии.

1. **Добавьте раздел «Предлагаемое решение».** Определите, какое кеширование вы будете внедрять — клиентское или серверное. Объясните, почему, на ваш взгляд, нужно использовать именно его. Если вы решите куда-то внедрить серверное кеширование, то поясните, какой паттерн будете применять — Cache-Aside, Write-Through или Refresh-Ahead. А также объясните, почему вы выбрали этот паттерн и почему остальные паттерны не подойдут или покажут себя хуже.

Поскольку для операторов важна актуальная информация по заказам, мы будем использовать серверное кеширование, поскольку сервер может управлять актуальностью кешированных данных, предоставляя свежую информацию и минимизируя риск устаревания данных. Кроме того, так как мы имеем дело в 3D моделями изделий, серверное кеширование предпочтительнее из-за допустимого объема кешируемых данных. На мой взгляд, наиболее подходящим паттерном кеширования является комбинация паттернов Refresh-Ahead/Write-behind, так как он обеспечивает наибольшую согласованность записей кеша, к которым часто обращаются пользователи, в отличие от Cache-Aside, и более высокую скорость передачи данных, поэтому и не Write-Through

1. **Нарисуйте** [**диаграмму последовательности действий (Sequence diagram)**](https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D0%B0%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0_%D0%BF%D0%BE%D1%81%D0%BB%D0%B5%D0%B4%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%BE%D1%81%D1%82%D0%B8)**.** Отобразите там, как проходит операция чтения списка заказов и запись об изменении статуса заказа. Там же опишите процесс кеширования с указанием всех сущностей, которые участвуют в кешировании. Добавьте диаграмму в раздел «Предлагаемое решение».

<https://drive.google.com/file/d/1OsRDDqrVTXzCxA1gqkDODffOyS6m7sld/view?usp=drive_link>

1. **В блоке «Предлагаемое решение» опишите стратегию инвалидации кеша, которую вы планируете использовать.** Объясните, какую стратегию инвалидации вы предлагаете (временную, по ключу, программную или другие), почему она подойдёт и почему не подойдут другие стратегии. Не всегда очевидно, какое решение лучше. Чтобы выбрать оптимальный вариант, можете сделать сравнительный анализ в виде таблицы. Например, так:

| **Первая стратегия** | **Вторая стратегия** | **Остальные стратегии** |
| --- | --- | --- |
| Лучше подходит, потому что... Есть особенности: 1)… 2)… | Позволяет сделать… Не позволяет сделать... | … |

В рамках нашего паттерна инвалидация кеша может происходить следующим образом:

* Настройка TTL:

Устанавливается значение TTL для кеша, после которого данные считаются устаревшими.

Когда данные устаревают, происходит инициирование фонового процесса обновления.

* Обработка уведомлений об изменениях:

Если данные в базе были изменены (например, при операциях создания, обновления или удаления), система получает уведомление и инвалидирует соответствующие кешированные записи.