

Кеширование

HTTP предоставляет встроенные механизмы кэширования. Все, что вам нужно сделать, это включить некоторые дополнительные заголовки исходящего ответа от сервера и в некоторых случаях выполнить небольшую проверку при получении некоторых заголовков запроса от клиента.

Мы будем говорить про кеширование на стороне браузера (клиента), а не на стороне промежуточных серверов или целевого сервера, куда идут запросы от клиента. Потому что на стороне браузера кеширование обычно управляется механизмами HTTP, а все остальные требуют специфичных программ и индивидуальной настройки в зависимости от архитектуры веб-приложения.

Браузер умеет: получать ответы от сервера на свои запросы, сохранять ответы в свой кеш, выбирать кешировать или нет (на основе заголовков ответов от сервера).

Варианты кеширования

1. Не кешировать — конфиденциальные данные/капча для проверки и похожие данные/всё, что часто меняется.
Ставим заголовок в ответе от сервера(то есть даём инструкцию браузеру) — Cache-Control: private, no-cache, no-store
2. Кеширование на время — данные, для которых известно время изменения.
Ставим заголовок к ответу от сервера — Cache-Control: max-age=3600, must-revalidate.
Это значит - можно брать ответ из кеша в течение 3600 секунд с этого времени.
3. Кеширование с валидацией — когда мы хотим закешировать данные, но мы не знаем время их изменения.
Как реализуется? С помощью условных запросов.
 1. сервер отвечает с заголовком Cache-Control: no-cache и Etag: уникальныйID (может быть хеш какого-то ресурса).
 2. потом клиент хочет снова запросить эти данные, но браузер вместо полноценного запроса посылает условный, то есть GET с заголовком (или даже HEAD, еще условным может быть PUT при изменении)
If-None-Match: УНИКАЛЬНЫЙID, например "33a64df551425fcc55e4d42a148795d9f25f89d4"
 3. если значение ресурса на клиенте If-None-Match от клиента = текущему значению ресурса Etag на сервере (ведь ресурс мог измениться за это время, а мог и остаться прежним)
 4. тогда сервер отвечает без тела, но со статусом 304 Not Modified

5. клиент по статусу 304 понимает, что ресурс не менялся (не стал новым) и его можно брать из кеша

Если же ресурс изменился (If-None-Match от клиента !=(не равно) текущему Etag на сервере), тогда сервер отдает ресурс в теле ответа с заголовками Cache-Control: no-cache и Etag: новый уникальныйID

4. Кешировать навсегда и везде (в том числе на прокси-серверах) — можно что-то совсем неуникальное и вечное...

Ставим заголовок в ответ от сервера - Cache-Control: max-age=31536000, public, immutable