



Использование Server Side Rendering в React приложении

Мубаракшин Александр
Frontend Tech Lead

C/TRON/UM

Немного истории



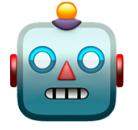
Server Side Rendering уже был

Server Side Rendering уже был

- PHP
- Java
- Ruby
- Python

Server Side Rendering уже был

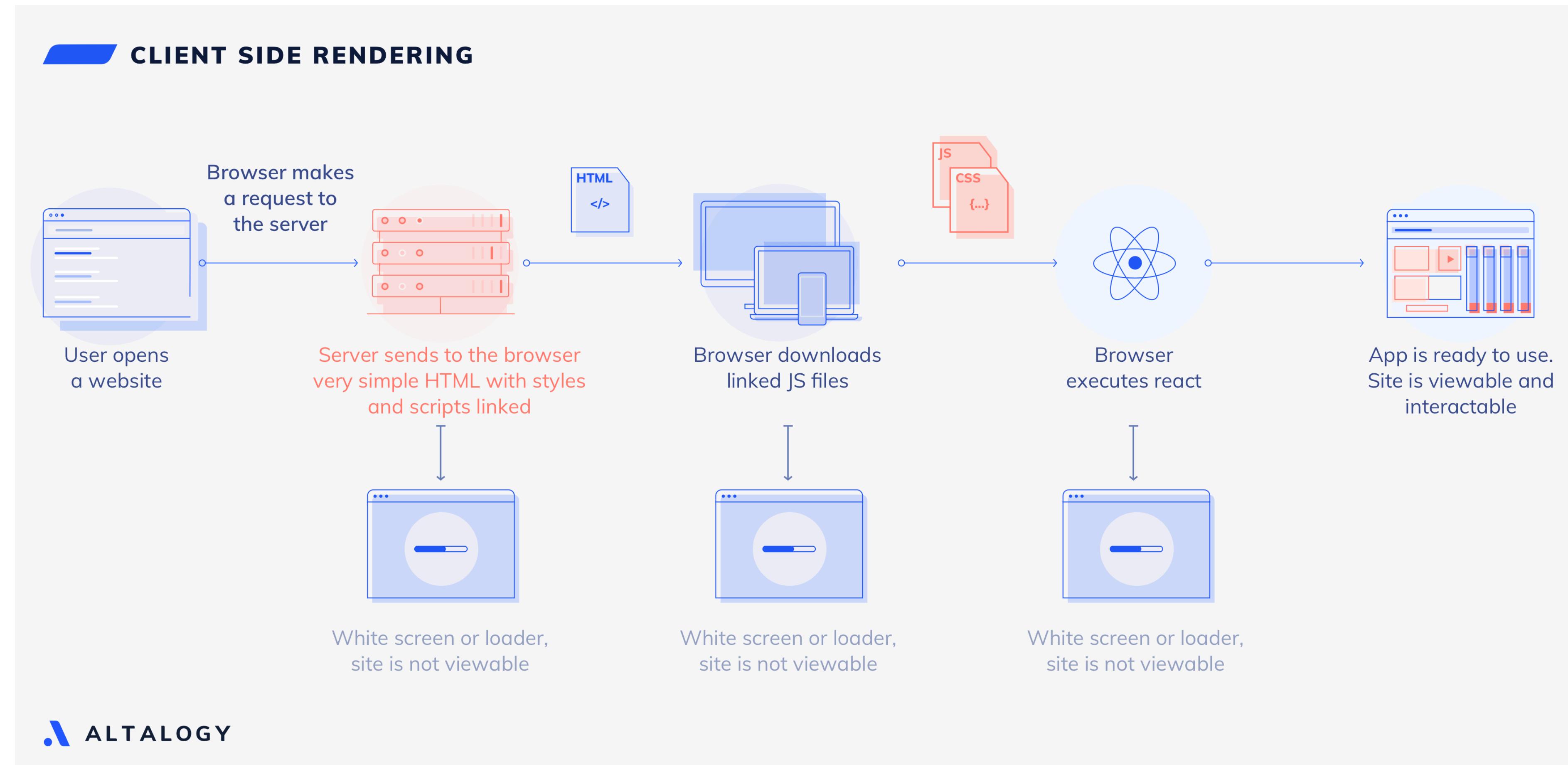
 Низкий TTI (Time To Interactive) и FCP (First Contentful Paint)

 SEO friendly

Θpa SPA

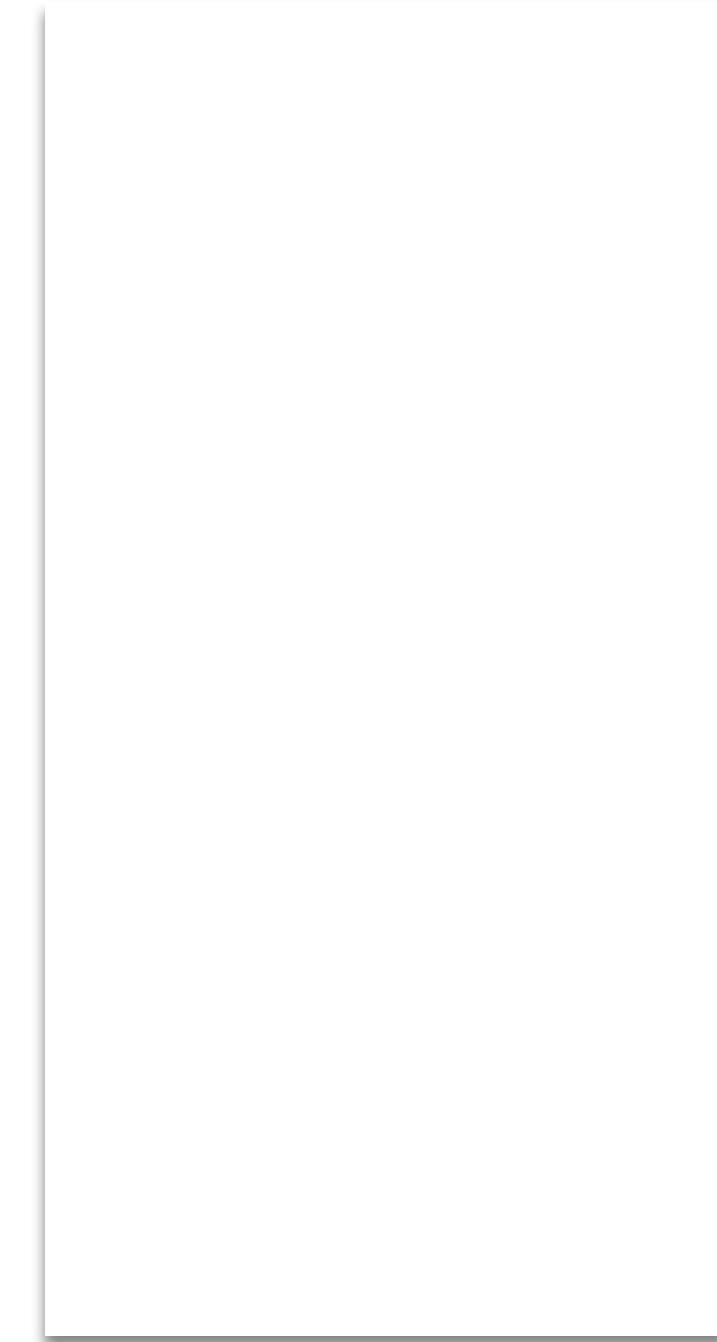
- Backbone 2009
- Angular.js 2010
- ReactJS 2011 (Release 2013)

CSR



Источник: <https://altalogy.com/blog/client-side-rendering-vs-server-side-rendering/>

CSR



Brewdog's beer explorer



Brewdog's beer explorer

Buzz

A light, crisp and bitter IPA brewed with English and American hops. A small batch brewed only once.



Trashy Blonde

A titillating, neurotic, peroxide punk of a Pale Ale. Combining attitude, style, substance, and a little bit of low self esteem for good measure; what would your mother say? The seductive lure of the sassy passion fruit hop proves too much to resist. All that is even before we get onto the fact that there are no additives, preservatives, pasteurization or strings attached. All wrapped up with the customary BrewDog bite and imaginative twist.



Berliner Weisse With



Несколько минусов CSR

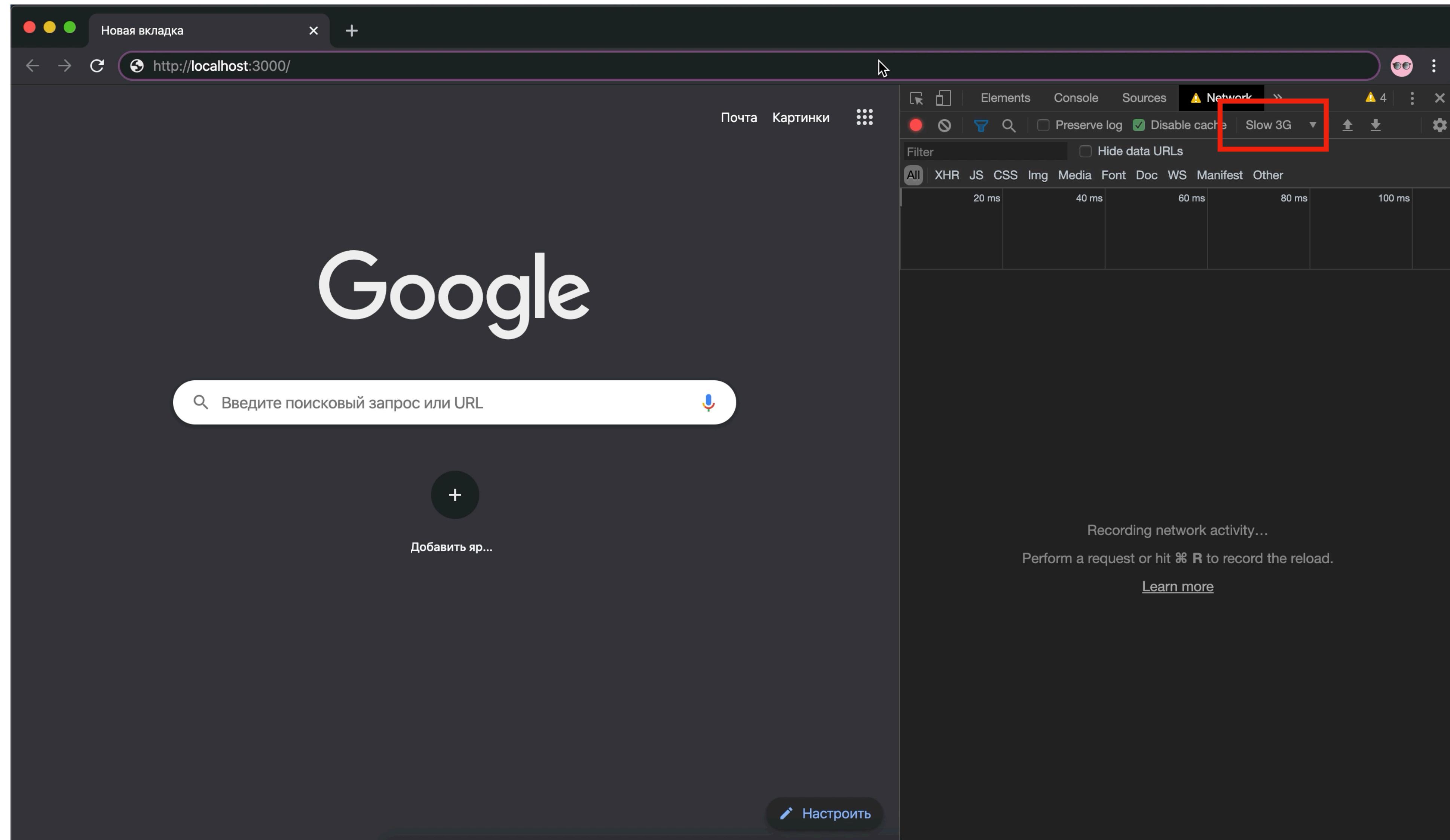


CSR



Где контент?

```
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <title>Brewdog's beer explorer</title>
5     <meta charset="utf-8">
6     <meta name="viewport" content="width=device-width, initial-scale=1">
7     <meta name="theme-color" content="#556cd6">
8     <link rel="shortcut icon" href="/assets/logos/favicon.ico" type="image/x-icon">
9     <link rel="icon" href="/assets/logos/favicon.ico" type="image/x-icon">
10    <link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Roboto:300,400,500,700&display=swap"/>
11  </head>
12  <body>
13    <div class="content">
14      <div id="app" class="wrap-inner"></div>
15    </div>
16    <script src="/client.js"></script>
17  </body>
18 </html>
19
```



Beer explorer

localhost:3000

Обновить эту страницу

Brewdog's beer explorer



Buzz

A light, crisp and bitter IPA brewed with English and American hops. A small batch brewed only once.

KEG ONLY



Trashy Blonde

A titillating, neurotic, peroxide punk of a Pale Ale. Combining attitude, style, substance, and a little bit of low self esteem for good measure; what would your mother say? The seductive lure of the sassy passion fruit hop proves too much to resist. All that is even before we get onto the fact that there are no additives, preservatives, pasteurization or strings attached. All wrapped up with the customary BrewDog bite and imaginative twist.



Berliner Weisse With Yuzu - B-Sides

Japanese citrus fruit intensifies the sour nature of this German classic.



Pilsen Lager

Our Unleash the Yeast series was an epic experiment into the differences in aroma and flavour provided by switching up your yeast. We brewed up a wort with a light caramel note



Avery Brown Dredge

An Imperial Pilsner in collaboration with beer writers. Tradition. Homage. Revolution. We wanted to showcase the awesome backbone of the Czech



Electric India

Re-brewed as a spring seasonal, this beer – which appeared originally as an Equity Punk shareholder creation – retains its trademark spicy, fruity

Минусы CSR

 Большой TTR (Time To Render)

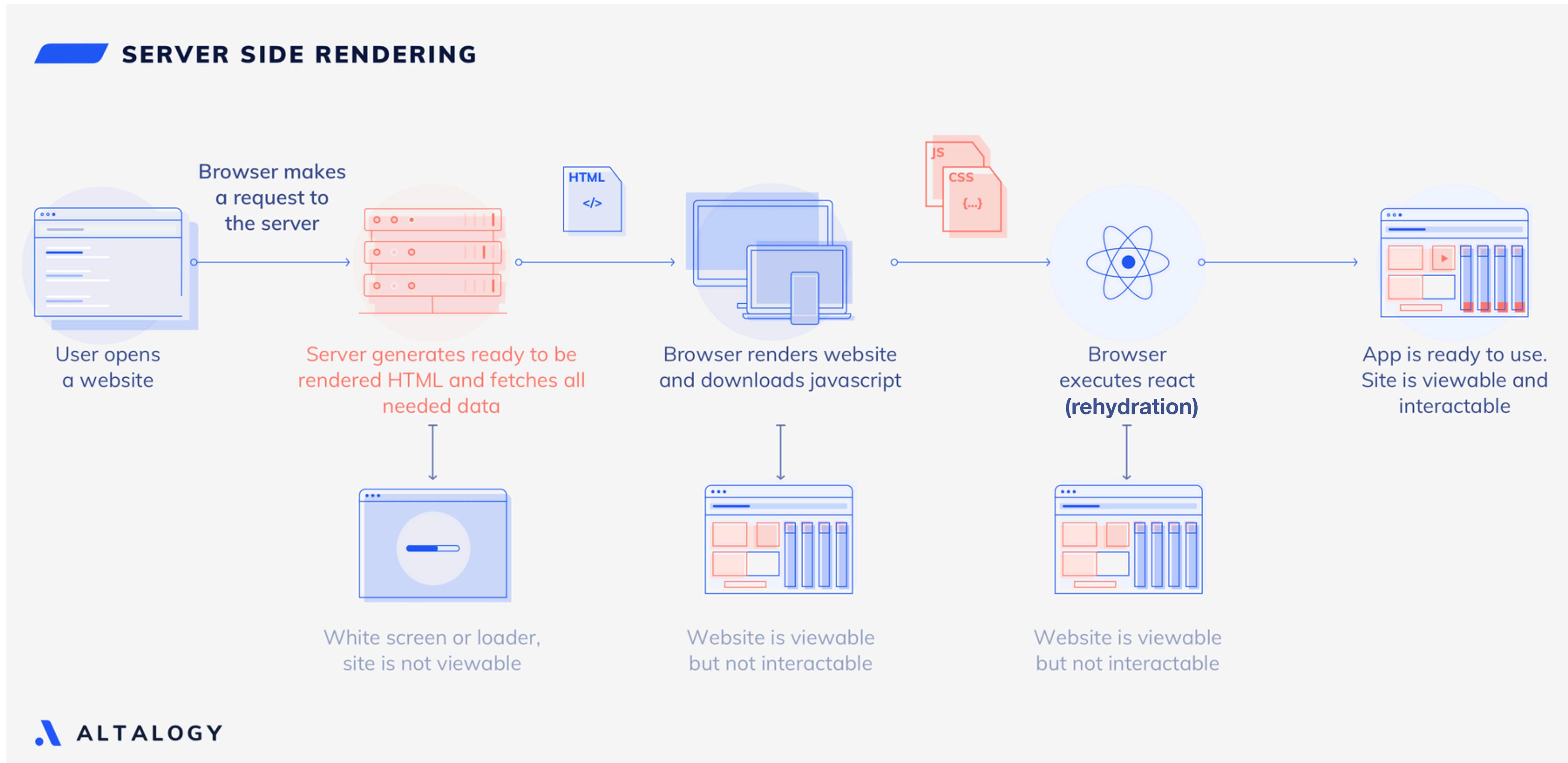
 Проблемы при медленном интернете

 Отсутствует индексация*

**Как получить преимущества
CSR и SSR?**

SSR with (re)hydration

SSR



Источник: <https://altalogy.com/blog/client-side-rendering-vs-server-side-rendering/>

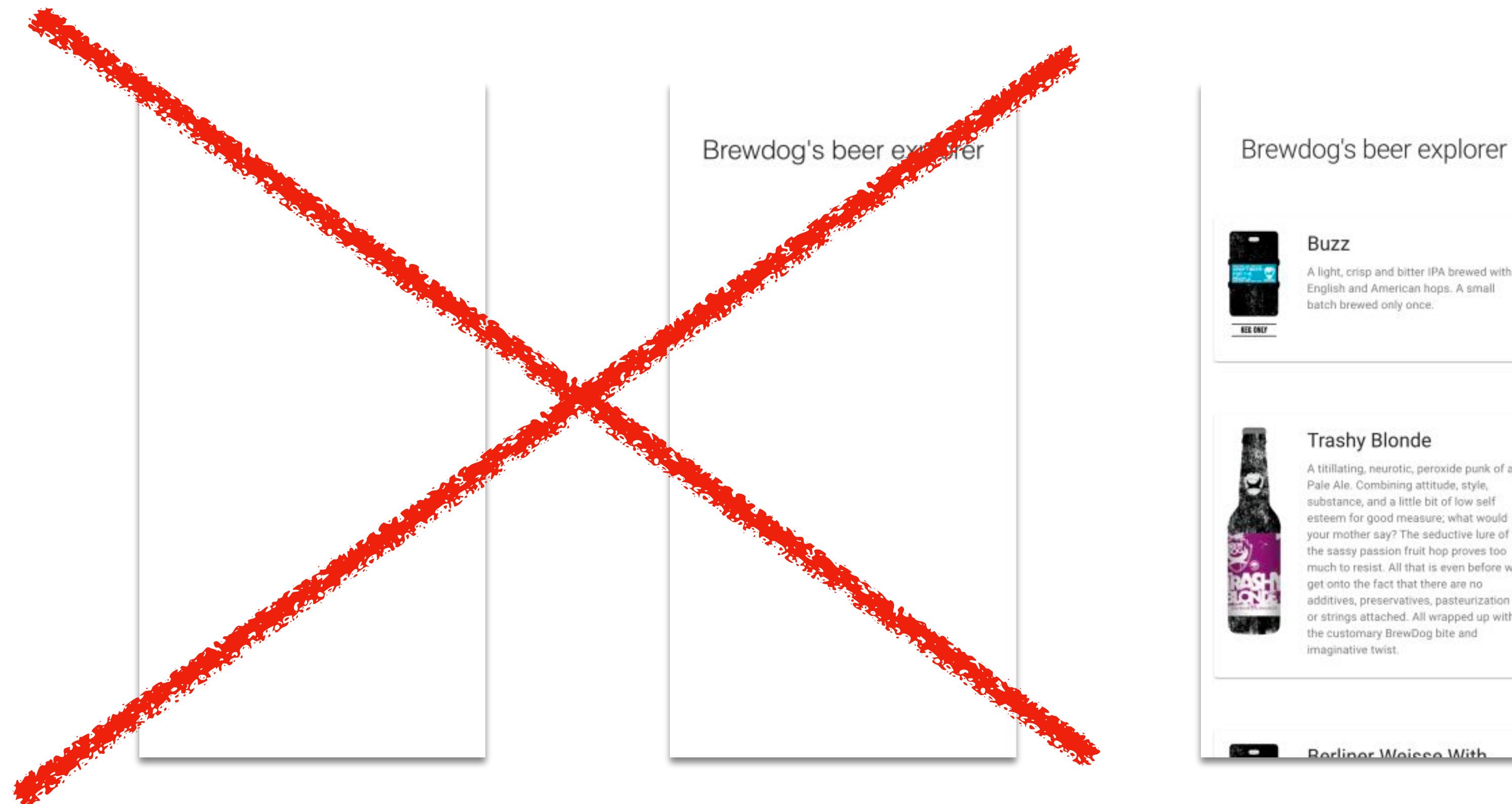
Регистрация ?!?!?!

- 🎧 Повесить Event Listener'ы на существующую разметку
- 🔍 Проверить разницу между разметкой полученной с помощью сервера и клиента
- 🌳 Пересоздать дерево на Virtual DOM
- 🎉 Выполнить life cycle методы

Решаем проблемы CSR



Большой TTR (Time To Render)



Решаем проблемы CSR



Проблемы при медленном интернете

Решаем проблемы CSR

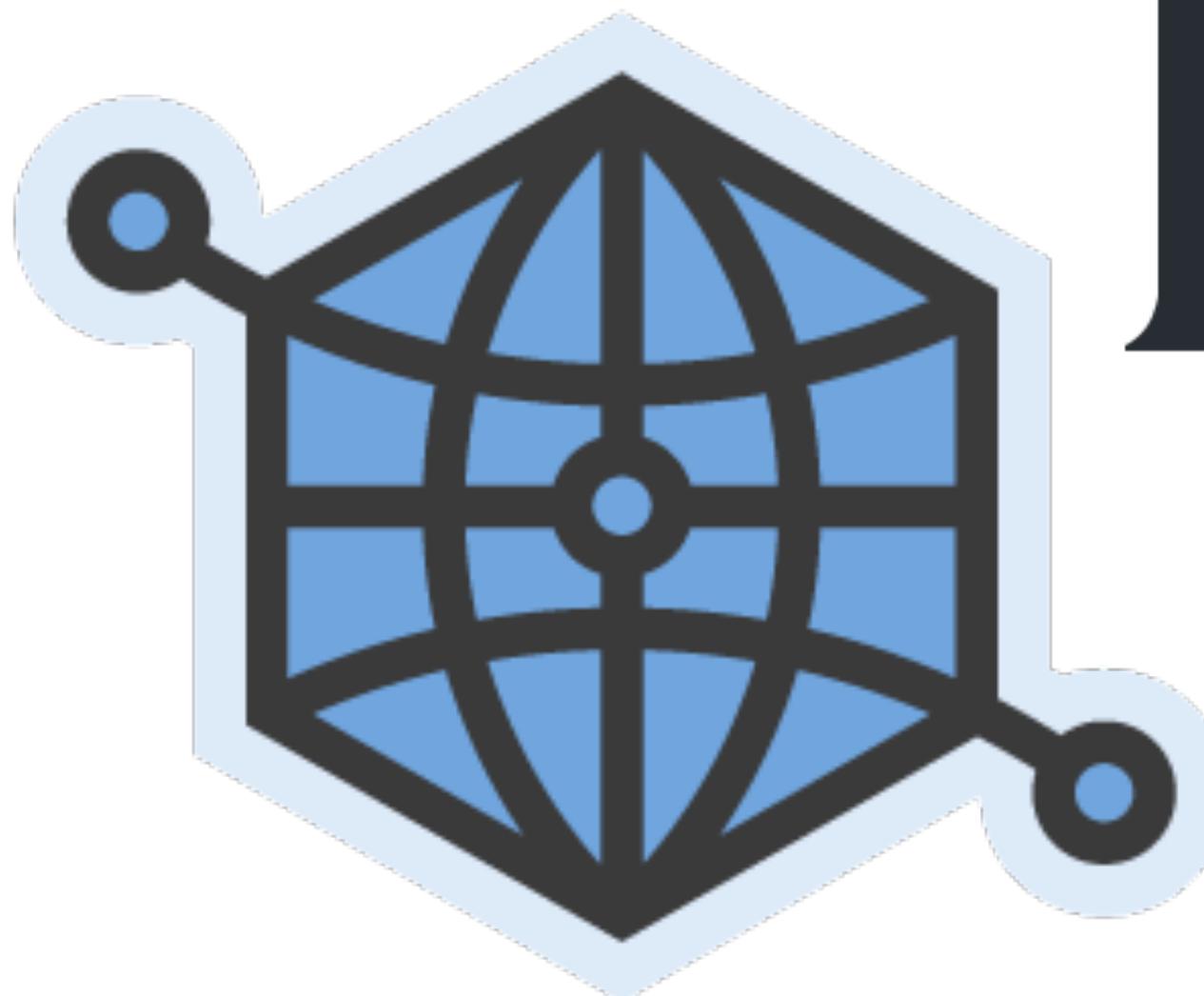


Отсутствует индексация

Плюсы SSR

- 🔍 Поисковая оптимизация
- 😊 Дружелюбный UX
- 📣 Оптимизация под социальные сети

📢 Оптимизация под социальные сети



Open Graph

<https://www.ozon.ru/context/detail/id/28336354/>

OZON.ru

Цифровая книга "Чистый код: создание, анализ и рефакторинг" — купить в интернет-магазине OZON.ru. OZON предлагает выгодные цены и отличный сервис. Цифровая книга "Чистый код: создание, ...



17:04 ✓

Реальность

Реальность

✉️ Реализация получения начальных асинхронных данных

❗ Code Splitting

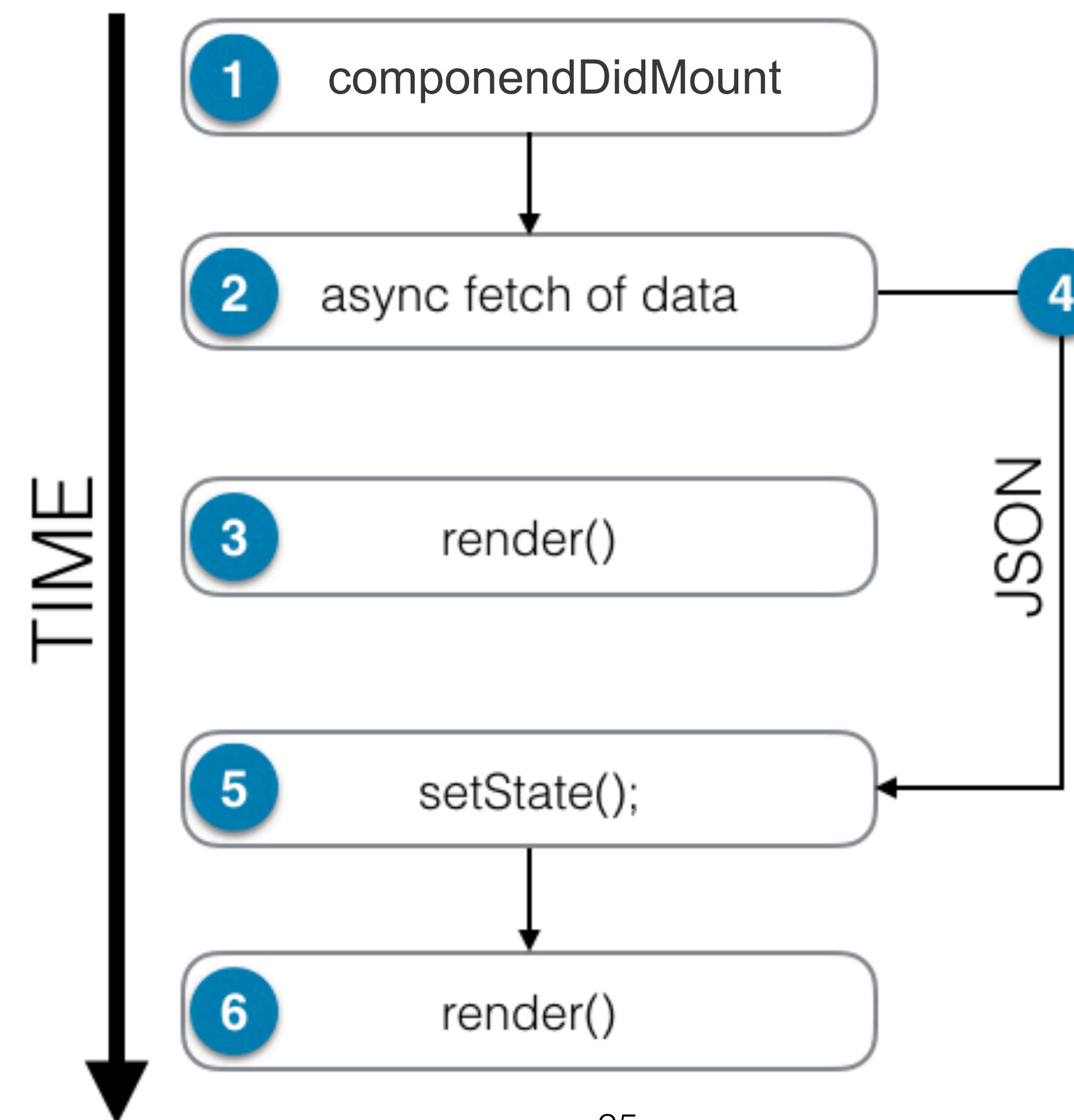
🔒 Безопасность

💀 Проблемы со стабильностью и утечками памяти

❗ Дублирование бизнес логики

Реальность

✉️ Получение начальных асинхронных данных



Реальность



Code Splitting



Синхронная загрузка модулей на сервер



Собрать набор модулей, участвовавших в серверном и клиентском рендеринге



Асинхронная загрузка модулей на клиенте

Реальность



Безопасность

```
1 <script>
2   window.__STATE__ = JSON.stringify({ someVal: '</script><script>alert(document.cookie)' });
3 </script>
```



The screenshot shows a web browser window displaying the Apollo GraphQL Client documentation. The title bar reads "Server-side rendering - Client (R X)". The URL in the address bar is "apollographql.com/docs/react/performance/server-side-rendering/#store-rehydration".

The left sidebar contains a navigation menu with the following sections and links:

- Client (React)**
- FETCHING DATA**
- CACHING**
- DEVELOPMENT & TESTING**
- PERFORMANCE**
 - Improving performance
 - Optimistic UI
 - Server-side rendering
 - Compiling queries with Babel
- INTEGRATIONS**
- NETWORKING**
- API REFERENCE**

The main content area has a search bar labeled "Search Apollo Docs" and a button "Launch Graph Manager →". The current section is "Store rehydration".

Store rehydration

For applications that can perform some queries on the server prior to rendering the UI on the client, Apollo allows for setting the initial state of data. This is sometimes called rehydration, since the data is "dehydrated" when it is serialized and included in the initial HTML payload.

For example, a typical approach is to include a script tag that looks something like:

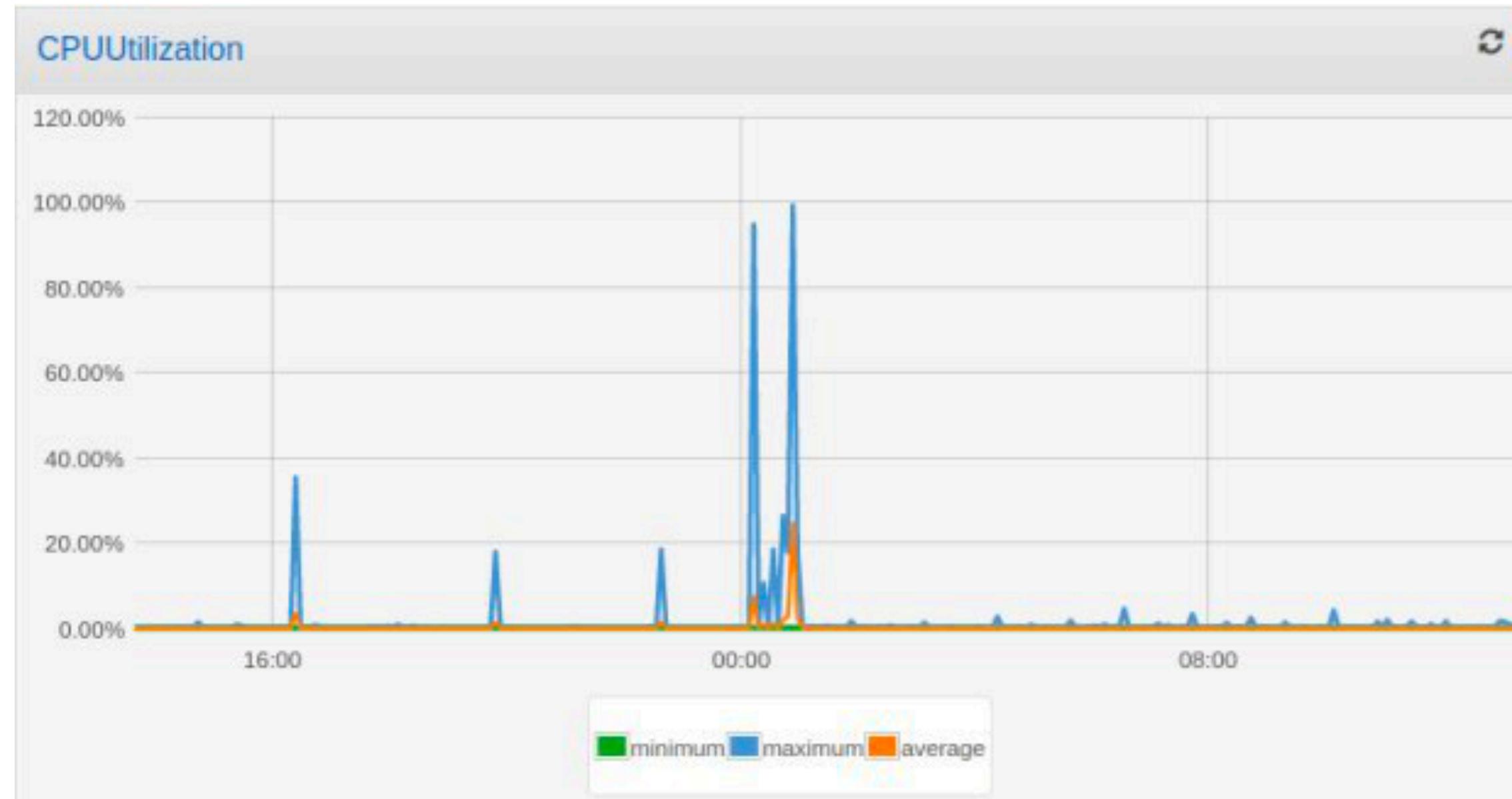
```
1 <script>
2   window.__APOLLO_STATE__ = JSON.stringify(client.extract());
3 </script>
```

You can then rehydrate the client using the initial state passed from the server:

```
1 const client = new ApolloClient({
2   cache: new InMemoryCache().restore(window.__APOLLO_STATE__),
3   link,
4 });
```

Реальность

💀 Проблемы со стабильностью и утечками памяти



Реальность

!! Дублирование бизнес логики

SSR with (re)hydration

👍 Низкий TTR

👍 Индексируемый

👍 Дружелюбный UX

👎 Высокий TTFB

👎 Гидратация ест ресурсы

CSR

👍 Гибкий

👍 Низкий TTFB

👎 Медленно работает на слабых устройствах

👎 Отсутствует индексация

Подходы для рендера приложения

- Client Side Rendering
- Rendering with (re)hydration
- Dynamic Rendering by route
- Dynamic Rendering at component level
- Static Rendering
- Static Rendering at component level
- Progressive Hydration

Dynamic Rendering by route

Пользователю CSR, роботу - SSR



Низкий TTFB



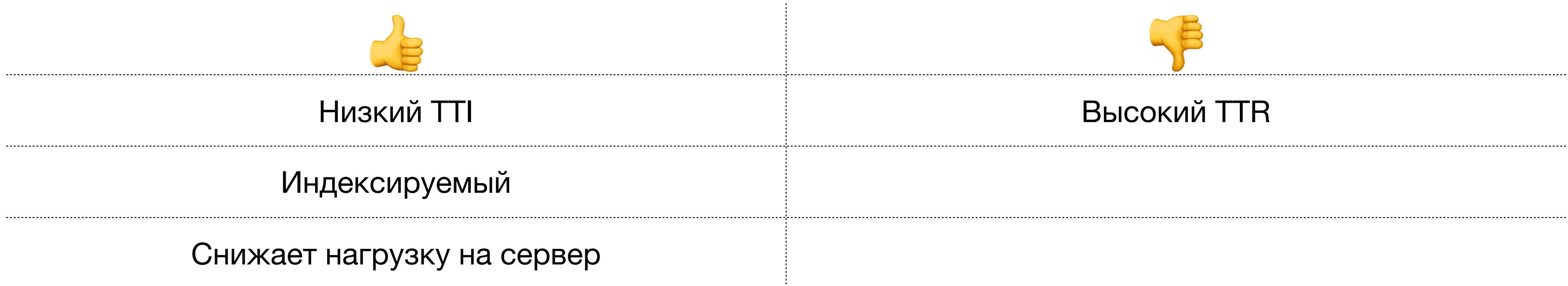
Высокий TTR

Индексируемый

Низкая нагрузка на сервер

Dynamic Rendering at component level

Пользователю рендерит контент постепенно, работе - полностью



Static Rendering

Генерит статику на каждый роут



Низкий FCP и TTFB

Индексируемый



Отсутствие гибкости

Static Rendering at component level

**Отключаем гидратацию у компонентов
которым не нужно состояние и life cycle**



Низкий ТТИ



Снижается интерактивность

Индексируемый

Снижает нагрузку на клиент

Progressive Hydration

Выполняем гидратацию компонента, когда он появляется во viewport



Низкий ТТИ



Гидратация все еще ест ресурсы

Индексируемый

Снижает нагрузку на клиент

Готовые решения



Gatsby

NE~~X~~T.JS



NUXTJS

Мои контакты:



AlexMubarakshin



@A1ex_dev

Полезные ссылки

- Miguel Angel Duran Garcia – React rendering strategies
- Guillermo Rauch – Client rendering, server rendering, pre rendering
- Rendering on the Web