

# Языки интернет- программирования

Лекция 2. Общие принципы построения web-приложений

Курс читают:

Шульман В.Д.	@ShtuzerVD
Пелевина Т.В.	@anivelat
Шабанов В.В.	@ZeroHug

# На «ты»

По-прежнему пытаемся общаться по-человечески.

# Рубрика «ответы на вопросы»

Вопрос: Когда откроется 4-я лабораторная работа?

Ответ: Скоро

Вопрос: Можно не приходить на лабу, если нечего сдавать?

Ответ: Можно. Если не приходит никто (или 1-2 человека), то сообщить заранее (мы тоже не придём)

Вопрос: Что будет, если не сдать лабы/РК «в срок»?

Ответ: Появляется риск встретить новый год в любимой бомонке (нам тоже, мы будем не очень довольны)

Вопрос: Список задач/вопросов для РК будет выложен заранее?

Ответ: Нет, но будет шаблон + пример билета

# Примеры проектов для ДЗ

Amazon

Instagram

Wikipedia

Reddit

LinkedIn

Twitter

Pinterest

eBay

Хабр

Пикабу

Авито

КиберЛенинка

ВБ

Дзен

Хэдхантер

Винлайн

Блог

Тематический форум

Интернет-магазин

Социальная сеть

Корпоративная сеть

Беттинг

Новостной портал

# Технические требования к ДЗ

В вашем проекте должен быть полный CRUD, т.е. должны присутствовать операции создания, чтения, обновления и удаления данных (create, read, update, delete).

Также в нем должна присутствовать авторизация

Проект должен состоять из 2 частей – BackEnd и FrontEnd.

FrontEnd часть приложения должна быть реализована в виде SPA (single page application) на базе любого js-фреймворка, например, React

BackEnd часть должна быть реализована на языке Golang

Взаимодействие между FrontEnd и BackEnd должно осуществляться через AJAX запросы

# Работа на семинарах

За работу на семинарах можно получить дополнительные баллы, которые начисляются сверх тех 70, что можно получить за семестр



# Про рабочее окружение

При работе из под Linux / Mac OS проблем возникать не должно.

Если используется Windows, то есть 3 пути:

1. **(рекомендуемый)** Установить Linux (Ubuntu) второй системой по инструкции <https://www.youtube.com/watch?v=BTHPkDIAzdQ>
2. Установить Linux (Ubuntu) на Virtualbox по инструкции <https://www.youtube.com/watch?v=S-GNJNlpLGE>
3. Установить WSL на Windows <https://learn.microsoft.com/ru-ru/windows/wsl/install>

# Лабораторные

Уже доступно 3 штуки по темам:

- Git
- HTML+CSS
- Golang

The screenshot shows a GitHub repository named 'web-core' (Public) by user 'ValeryBMSTU'. The repository has 1 branch (master) and 0 tags. The commit history shows 'Update README.md' by 'ValeryBMSTU' (c65f19c) yesterday, with 11 commits in total. The file list includes 'Lecion\_1.pdf', 'Lecion\_1.pptx', and 'README.md'. The 'README' file is open, showing the title 'Языки интернет-программирования' and the description 'Репозиторий для публикации лекций и ссылок на задания по лабораторным/ПК'. Below this, the section 'Лабораторные' lists three links to other repositories: 'https://github.com/ValeryBMSTU/web-1', 'https://github.com/ValeryBMSTU/web-2', and 'https://github.com/ValeryBMSTU/web-3'. The remaining four items in the list are marked as '<закрыто>'.

web-core Public

master 1 Branch 0 Tags

Go to file t Add file <> Code

ValeryBMSTU Update README.md c65f19c · yesterday 11 Commits

Lecion_1.pdf	Add files via upload	5 days ago
Lecion_1.pptx	Add files via upload	5 days ago
README.md	Update README.md	yesterday

README

## Языки интернет-программирования

Репозиторий для публикации лекций и ссылок на задания по лабораторным/ПК

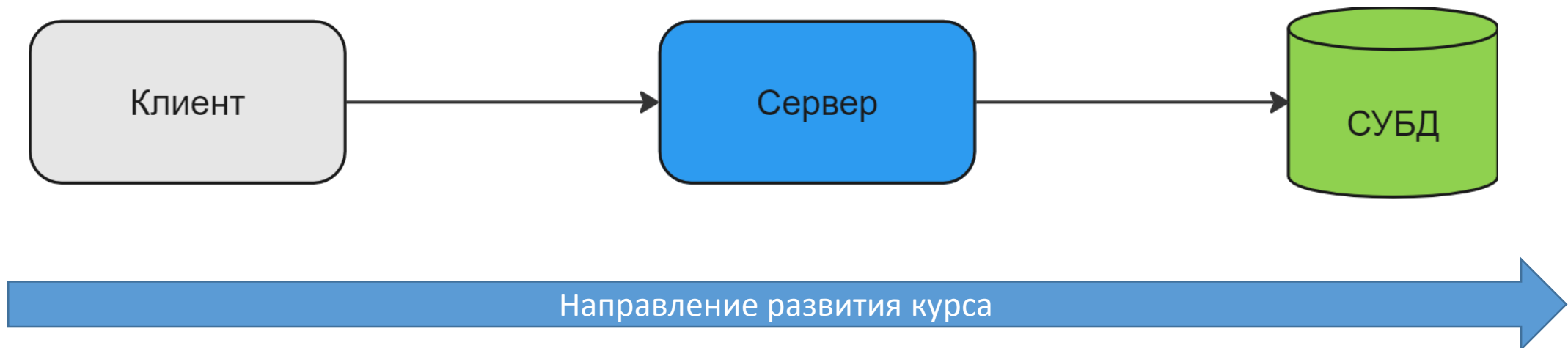
## Лабораторные

- <https://github.com/ValeryBMSTU/web-1>
- <https://github.com/ValeryBMSTU/web-2>
- <https://github.com/ValeryBMSTU/web-3>
- <закрыто>
- <закрыто>
- <закрыто>
- <закрыто>



# Направление развития курса

По мере продвижения по программе лабораторных и лекций мы постепенно будем смещать акцент с клиентской части в сторону серверной + СУБД



# План лекции

- обсудить что такое клиент
- обсудить что такое сервер
- обсудить что такое СУБД
- выяснить как они взаимодействуют друг с другом

# Клиент

А кто это вообще такой?

# Примеры клиентов

Клиент – это, прежде всего, программа



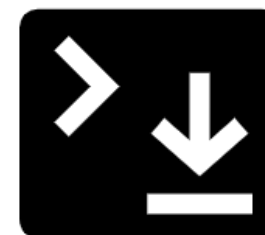
Postman



curl



Insomnia



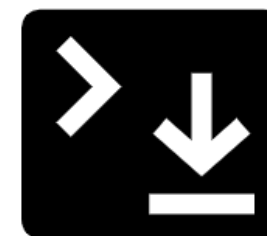
wget

# Примеры клиентов

Эти ребята в рамках нашего курса рассматриваться не будут (почти)

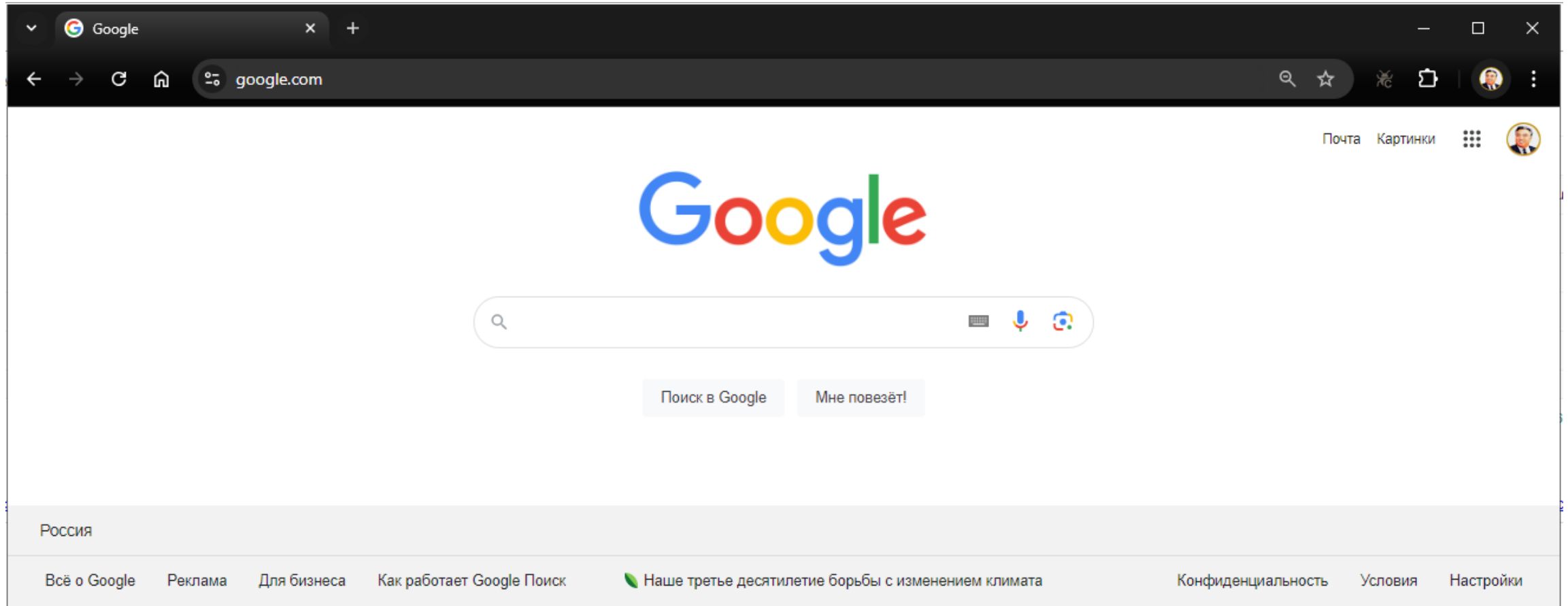


curl

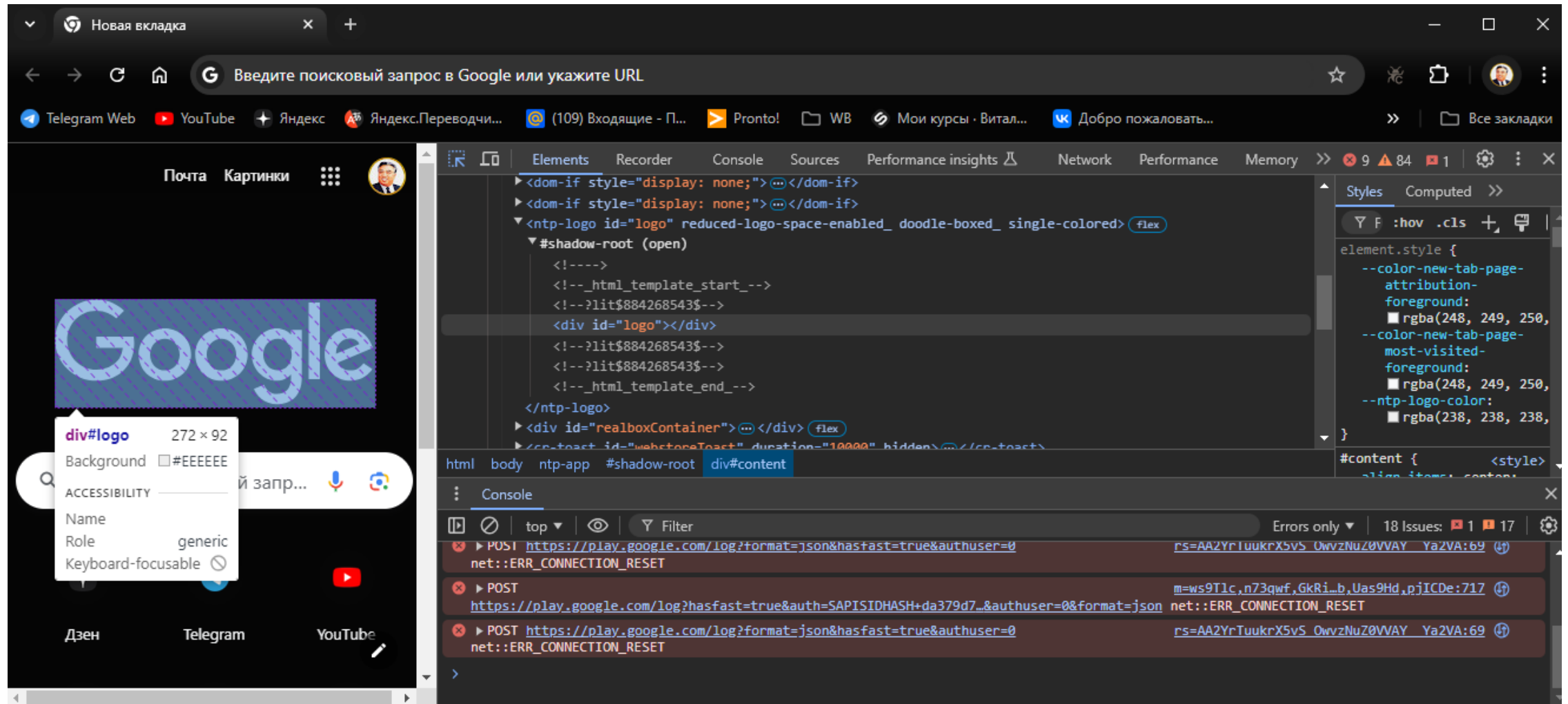


wget

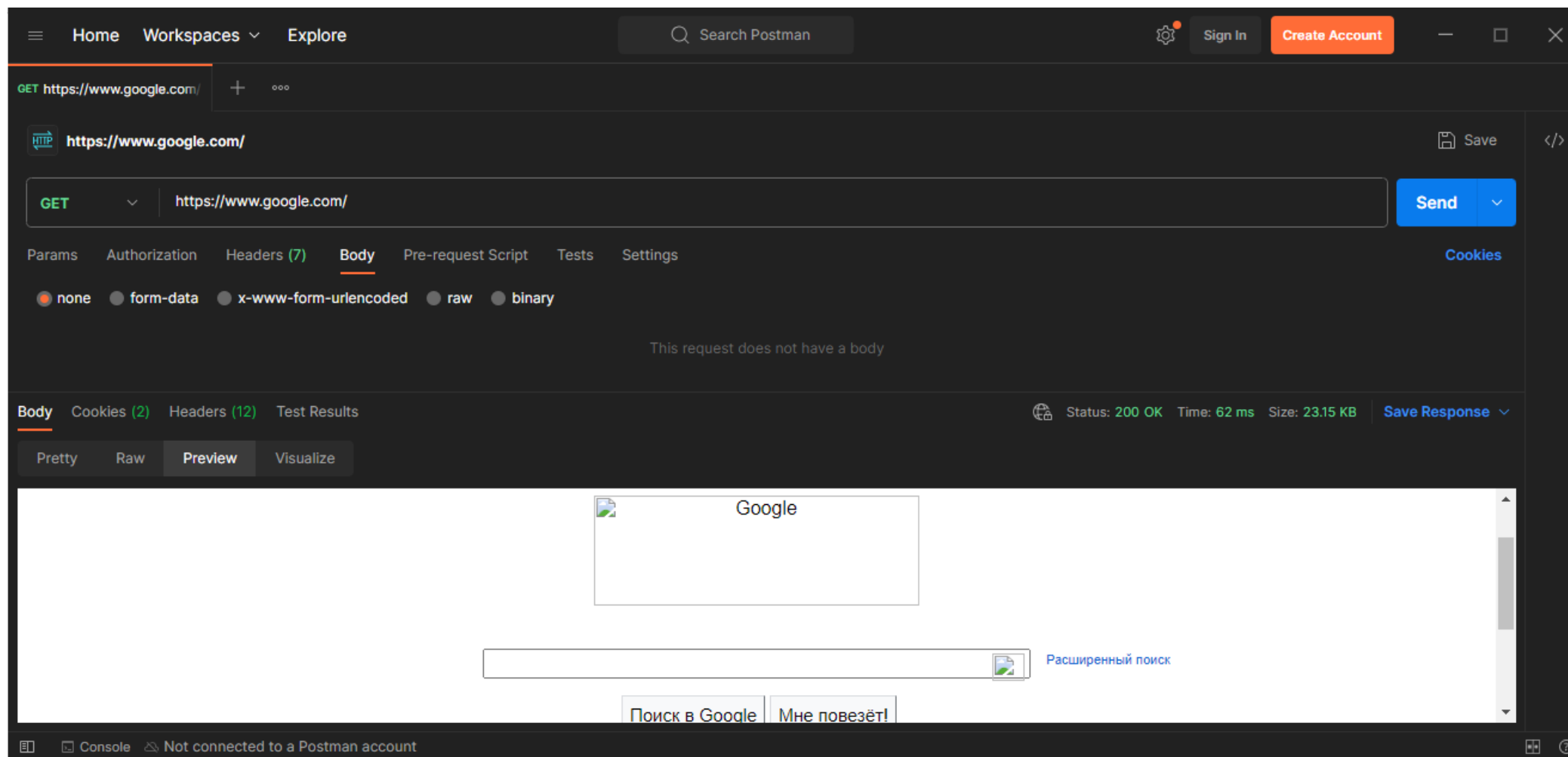
# Google Chrome



# Google Chrome



# Postman





# Curl

```
projects — bash — 155x35
bash-3.2$
bash-3.2$ curl -vvv -X GET "http://www.google.com"
Note: Unnecessary use of -X or --request, GET is already inferred.
* Host www.google.com:80 was resolved.
* IPv6: (none)
* IPv4: 108.177.14.105, 108.177.14.147, 108.177.14.104, 108.177.14.103, 108.177.14.99, 108.177.14.106
* Trying 108.177.14.105:80...
* Connected to www.google.com (108.177.14.105) port 80
> GET / HTTP/1.1
> Host: www.google.com
> User-Agent: curl/8.7.1
> Accept: */*
>
* Request completely sent off
< HTTP/1.1 200 OK
< Date: Mon, 09 Sep 2024 11:01:54 GMT
< Expires: -1
< Cache-Control: private, max-age=0
< Content-Type: text/html; charset=ISO-8859-1
< Content-Security-Policy-Report-Only: object-src 'none';base-uri 'self';script-src 'nonce-exe_Zo5TYwTiZ0lMPMHMzg' 'strict-dynamic' 'report-sample' 'unsafe-eval' 'unsafe-inline' https: http:;report-uri https://csp.withgoogle.com/csp/gws/other-hp
< P3P: CP="This is not a P3P policy! See g.co/p3phelp for more info."
< Server: gws
< X-XSS-Protection: 0
< X-Frame-Options: SAMEORIGIN
< Set-Cookie: AEC=AVYB7crR5o4splAvFNeqU353ccYnv5mL3bD-dYVKKYVivMiQftOqFPeR_g; expires=Sat, 08-Mar-2025 11:01:54 GMT; path=/; domain=.google.com; Secure; HttpOnly; SameSite=lax
< Set-Cookie: NID=517=BbZMKEdw6JP_Yu7_3LGBRhf_WUSM3meIXoFIGJ6IsKTQCnVhg0ghtjklNJBInZnbMDe1aIGraGQsxYDY0_CEY3XbKGpQTF0vDc_DB10F3x9Bh1QH6mk0PaaE6GZrw5MmVtq50abZ2tzoggp8NXs231B200LrucRF080DfQF27pXTblXFoLyFKXHj; expires=Tue, 11-Mar-2025 11:01:54 GMT; path=/; domain=.google.com; HttpOnly
< Accept-Ranges: none
< Vary: Accept-Encoding
< Connection: close
<
<!doctype html><html itemscope="" itemtype="http://schema.org/WebPage" lang="ru"><head><meta content="&#1055;&#1086;&#1080;&#1089;&#1082; &#1080;&#1085;&#1092;&#1086;&#1088;&#1084;&#1072;&#1094;&#1080;&#1080; &#1074; &#1080;&#1085;&#1090;&#1077;&#1088;&#1085;&#1077;&#1090;&#1077;: &#1074;&#1077;&#1073; &#1089
```

# Браузеры



Глобально отвечают за 2 аспекта:

- 2. обмен данными/статикой по сети с сервером
- 3. рендеринг страниц

Под статикой мы понимаем .html, .css, .js + медиа-файлы

# HTML

```
index.html > ? ? ? html
1 <!doctype html>
2 <html>
3
4 <head>
5   <meta charset="utf-8">
6   <link rel="stylesheet" href="css/CARD.css">
7   <link rel="preconnect" href="https://fonts.googleapis.com">
8   <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
9   <link href="https://fonts.googleapis.com/css2?family=Exo+2&family=Montserrat:wght@
10  <title>Card 1.0</title>
11 </head>
12
13 <body>
14   <div class="container">
15     <div class="card">
16       <div class="overflow">
17         <div class="card__image">
21       <div class="card__title">Pragser Wildsee, Italy</div>
22       <div class="card__text">Lorem ipsum dolor sit amet, consectetur adipis
23       <div class="card__btn"><a href="#">Booking</a></div>
24     </div>
25   </div>
26 </div>
27 </body>
28
29 </html>
```



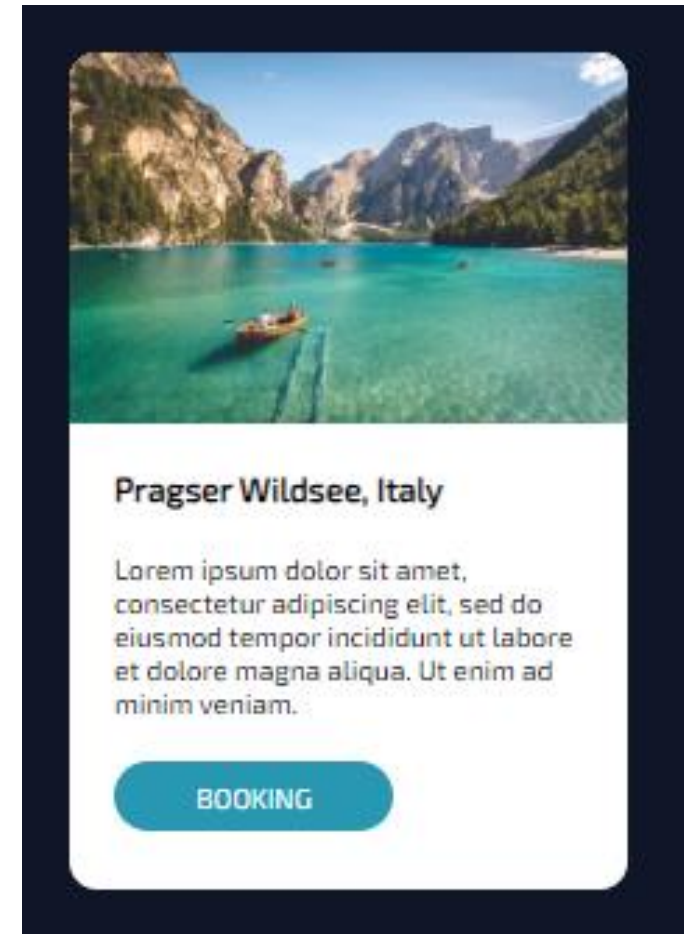
Pragser Wildsee, Italy

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam.

[Booking](#)

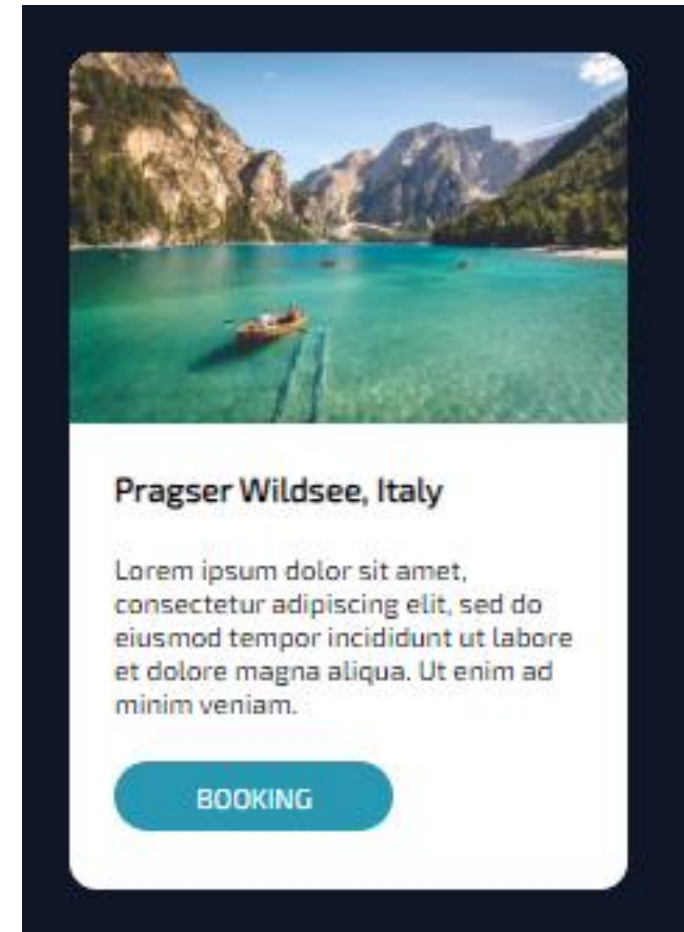
# CSS

```
# style.css > 🚧
1  * {
2    margin: 0;
3    padding: 0;
4  }
5
6  body {
7    font-family: "Exo 2";
8    color: #000;
9    font-size: 10px;
10   background-color: #0e1627;
11 }
12
13 .container {
14   padding-top: 50px;
15   display: flex;
16   justify-content: center;
17   align-items: center;
18 }
19
20 .card {
21   width: 200;
22   height: 300;
23   display: flex;
24   flex-direction: column;
25   justify-content: flex-start;
26   background-color: #fff;
27   border: 1 solid #fff;
28   border-radius: 10px;
29 }
30
31 a {
32   text-decoration: none;
```



# JS

```
index.html > ? > html
1 <!doctype html>
2 <html>
3
4 <head>
5   <meta charset="utf-8">
6   <link rel="stylesheet" href="style.css">
7   <link rel="preconnect" href="https://fonts.googleapis.com">
8   <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
9   <link href="https://fonts.googleapis.com/css2?family=Exo+2&family=Montserrat:wght@500;600;700&family=Pacifico&">
10  <title>Card 1.0</title>
11  <script>
12    function myFunction() {
13      document.getElementById("click_button").style.fontSize = "25px";
14      document.getElementById("click_button").style.color = "red";
15      document.getElementById("click_button").style.backgroundColor = "yellow";
16    }
17  </script>
18 </head>
19
20 <body>
21   <div class="container">
22     <div class="card">
23       <div class="overflow">
24         <div class="card_image">
25       </div>
26     </div>
27     <div class="card_content">
28       <div class="card_title">Pragser Wildsee, Italy</div>
29       <div class="card_text">Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam,
30       <div class="card_btn" id="click_button" onclick="myFunction()"><a href="#">Booking</a></div>
31     </div>
32   </div>
33 </div>
34 </body>
35
36 </html>
```





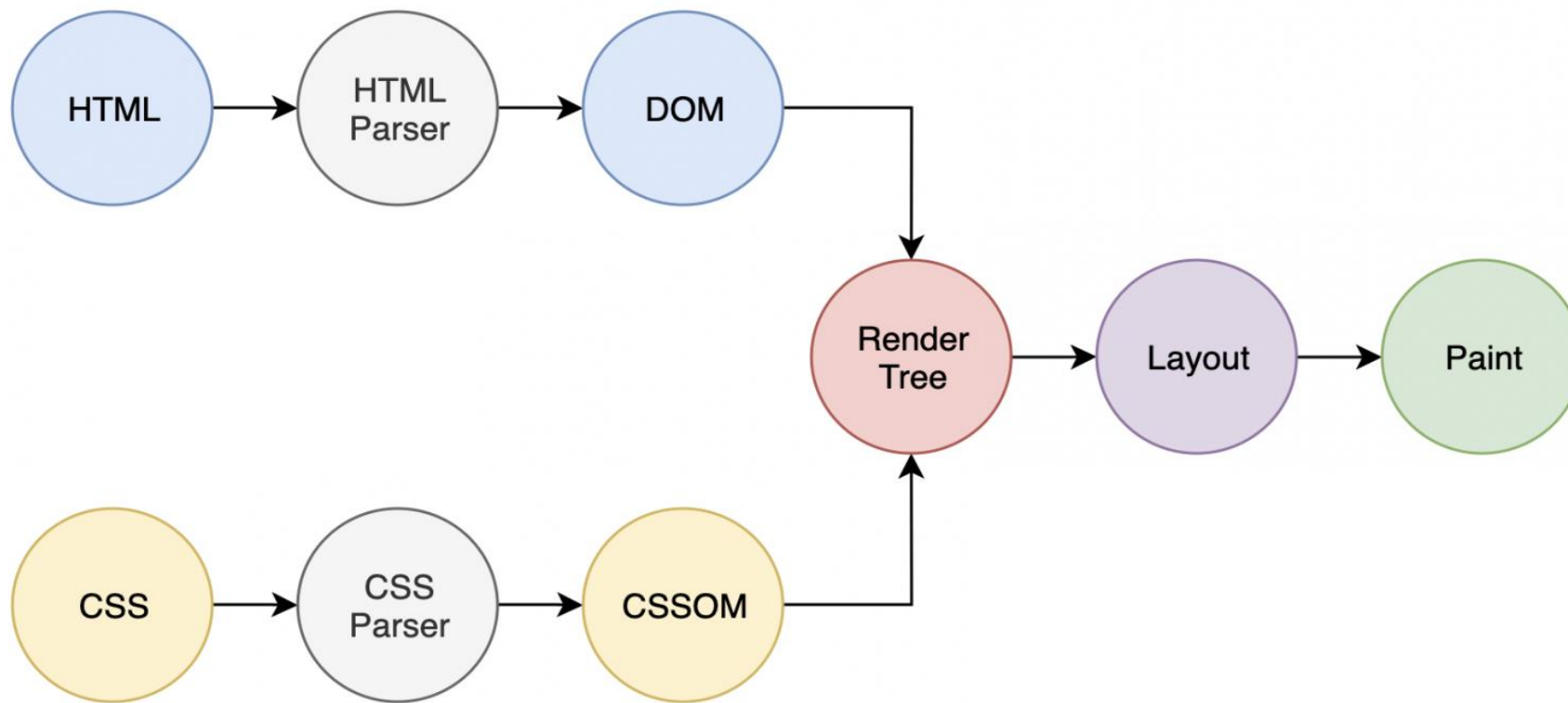
# JS

```
index.html > ? > html
1 <!doctype html>
2 <html>
3
4 <head>
5   <meta charset="utf-8">
6   <link rel="stylesheet" href="style.css">
7   <link rel="preconnect" href="https://fonts.googleapis.com">
8   <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
9   <link href="https://fonts.googleapis.com/css2?family=Exo+2&family=Montserrat:wght@500;600;700&family=Pacifico&">
10  <title>Card 1.0</title>
11  <script>
12    function myFunction() {
13      document.getElementById("click_button").style.fontSize = "25px";
14      document.getElementById("click_button").style.color = "red";
15      document.getElementById("click_button").style.backgroundColor = "yellow";
16    }
17  </script>
18 </head>
19
20 <body>
21   <div class="container">
22     <div class="card">
23       <div class="overflow">
24         <div class="card_image">
25         </div>
26       </div>
27       <div class="card_content">
28         <div class="card_title">Pragser Wildsee, Italy</div>
29         <div class="card_text">Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam,
30         <div class="card_btn" id="click_button" onclick="myFunction()"><a href="#">Booking</a></div>
31       </div>
32     </div>
33   </div>
34 </body>
35
36 </html>
```

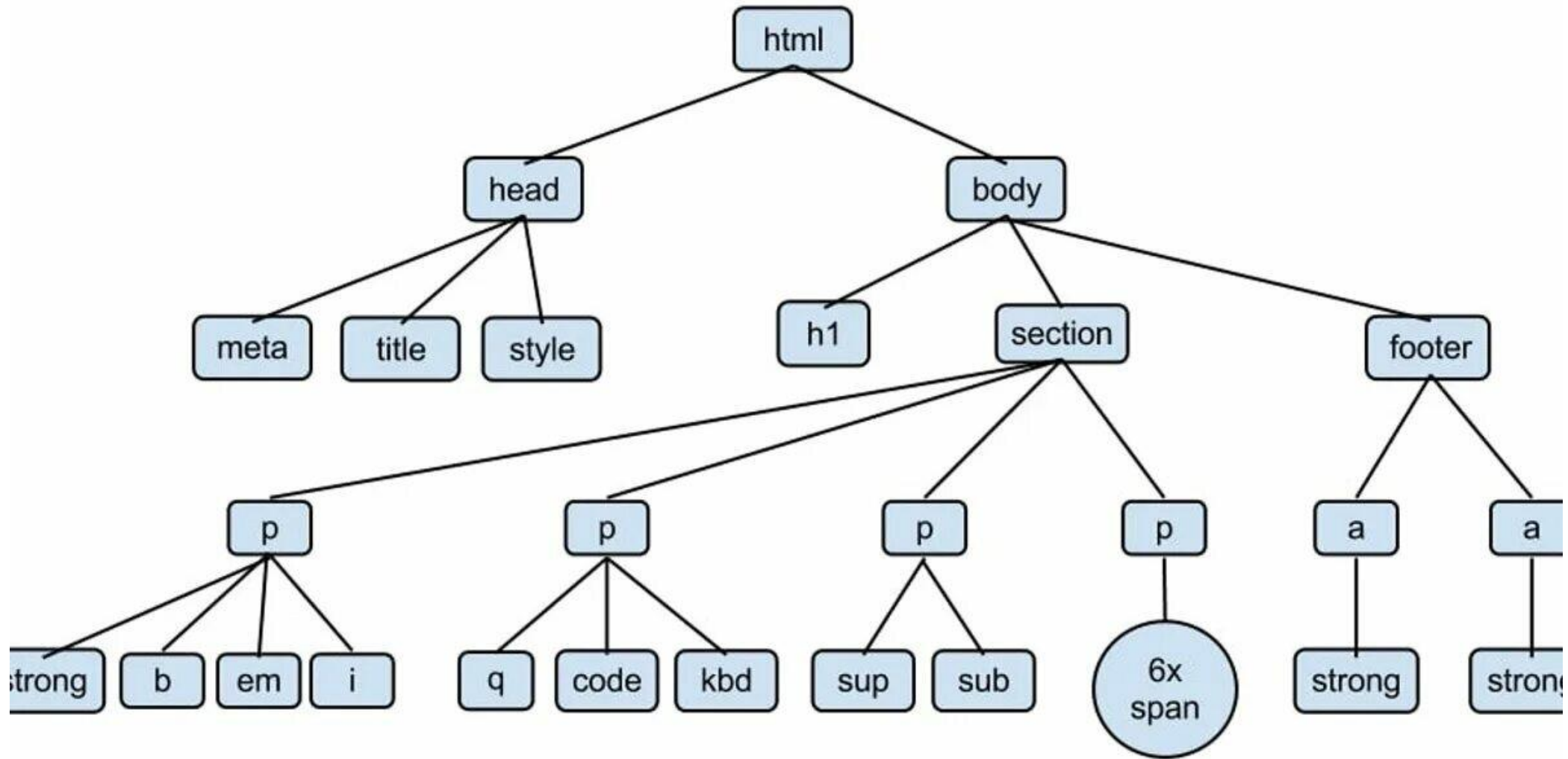


# Рендеринг страницы

Рендеринг страниц в браузере – это процесс преобразования кода .html и .css в пиксели на экране



# DOM





# DOM

DOM (Document Object Model) — это специальная древовидная структура, которая позволяет управлять HTML-разметкой из JavaScript-кода.

Управление обычно состоит из добавления и удаления элементов, изменения их стилей и содержимого.

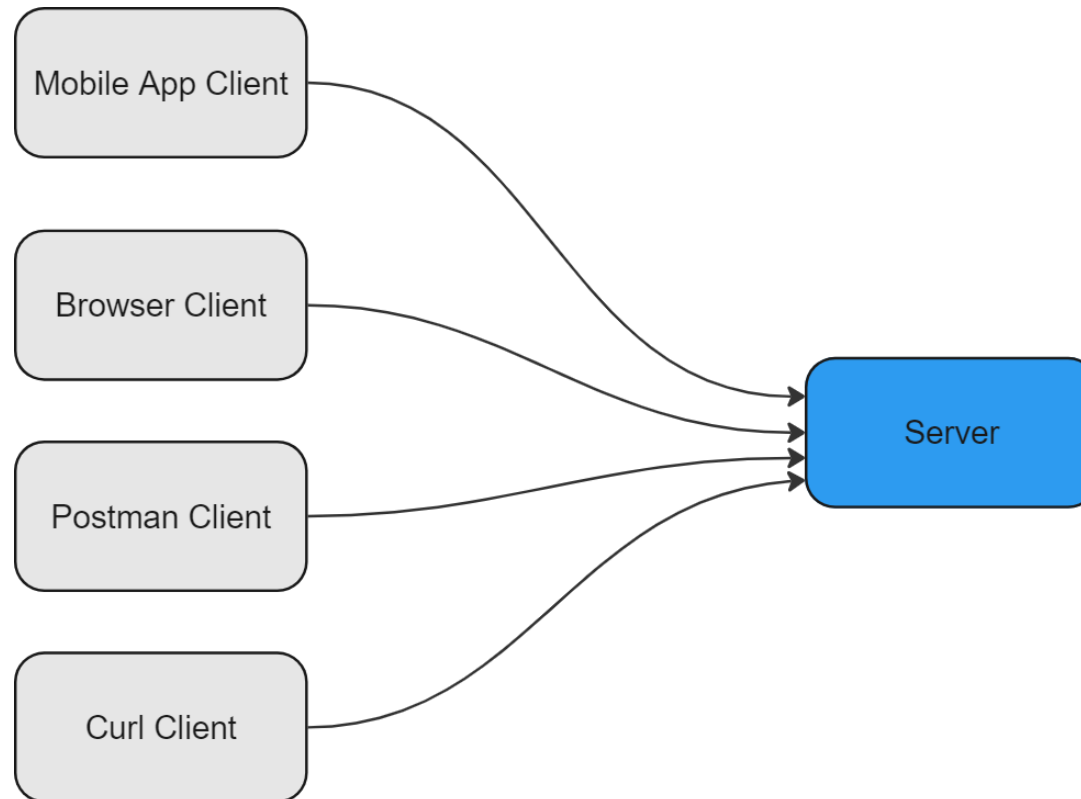
Браузер создаёт DOM при загрузке страницы, складывает его в переменную `document`

# Сервер

А это кто?

# Сервер

В рамках нашего курса сервер – это тоже программный компонент.



# Коробочные решения

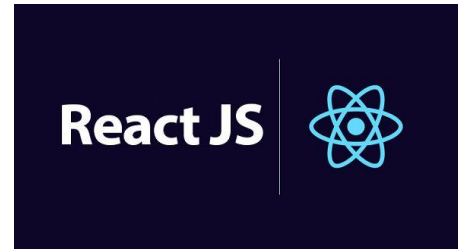


# Примеры языков/фреймворков реализации

FullStack



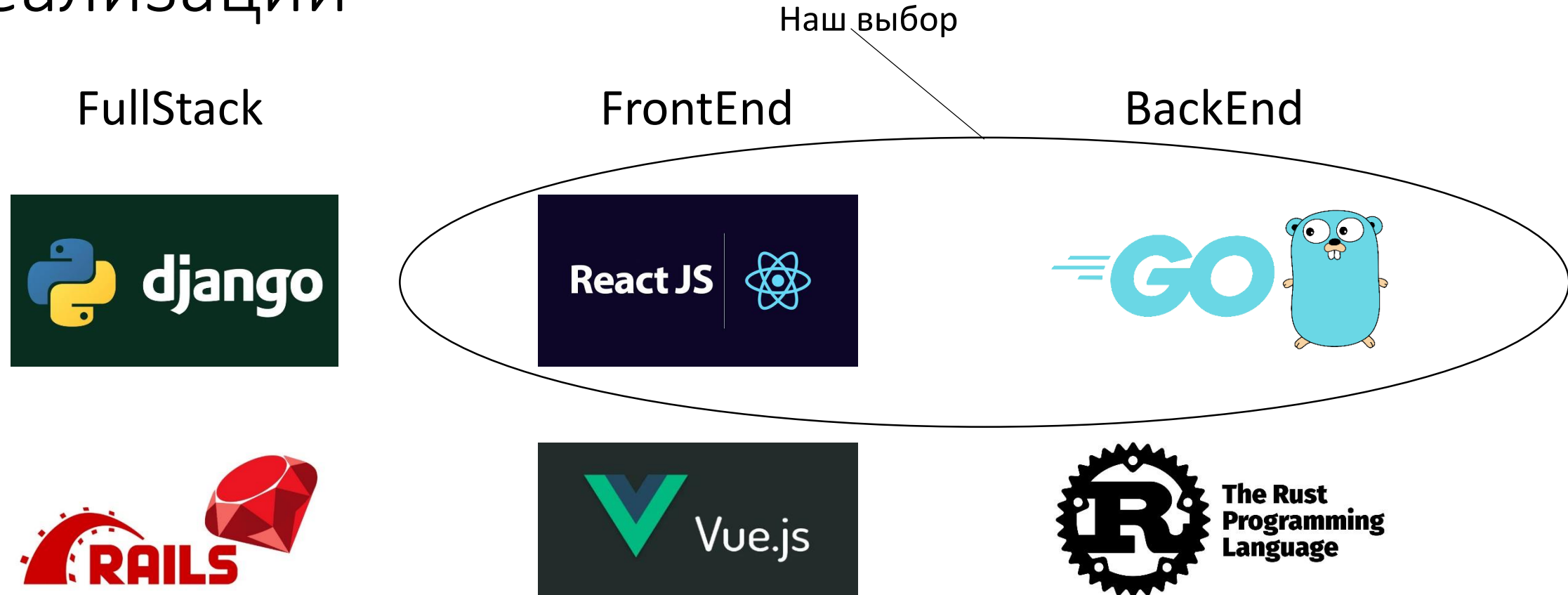
FrontEnd



BackEnd



# Примеры языков/фреймворков реализации



# За что отвечает FrontEnd-сервер?

Эта коробочка отвечает за:

1. Хранение пользовательского интерфейса в виде набора статических файлов (html, css, js и т.д.)
2. Сборку и предоставление SPA-приложения клиентам



# За что отвечает BackEnd-сервер?

Эта коробочка отвечает за:

1. Обработку запросов от клиентов (будь то мобильное приложение или SPA)
2. Основную бизнес-логику веб-приложения
3. Аутентификацию и авторизацию пользователей





# \*Вопрос

В чём разница между идентификацией, аутентификацией и авторизацией?

\*Будет на экзамене

## \* Ответ

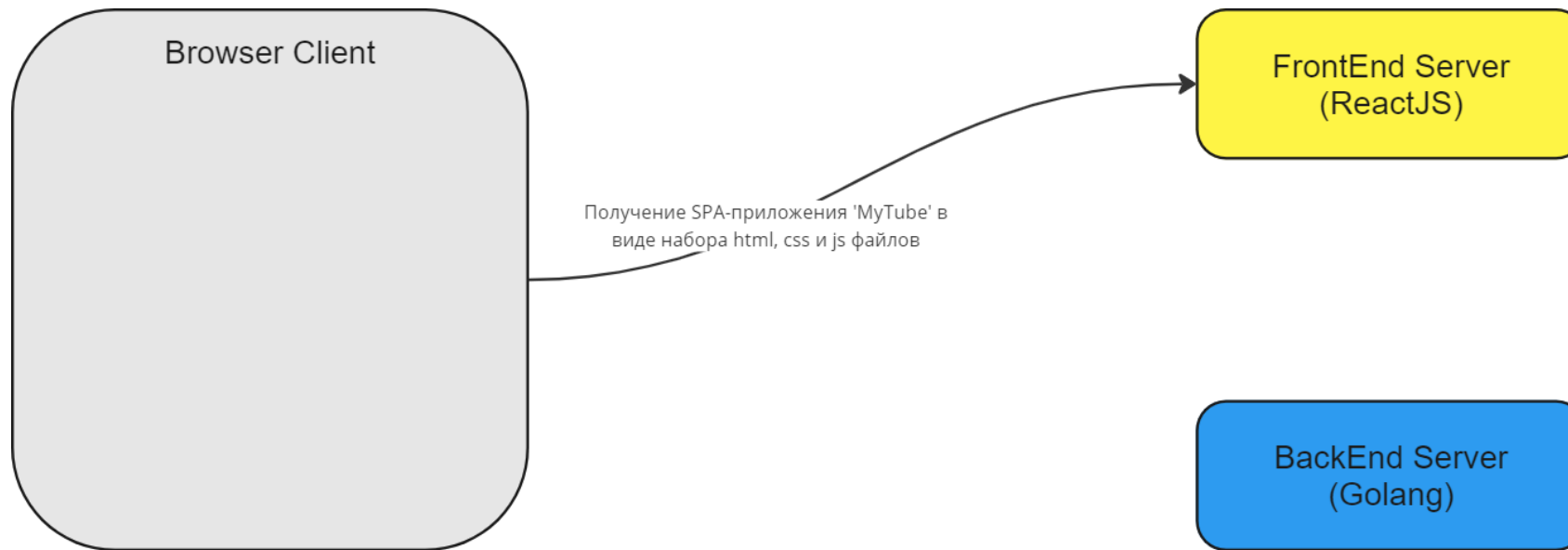
Идентификация – это процесс присвоения пользователю идентификатора

Аутентификация – это процесс проверки, что пользователь действительно тот, за кого себя выдает

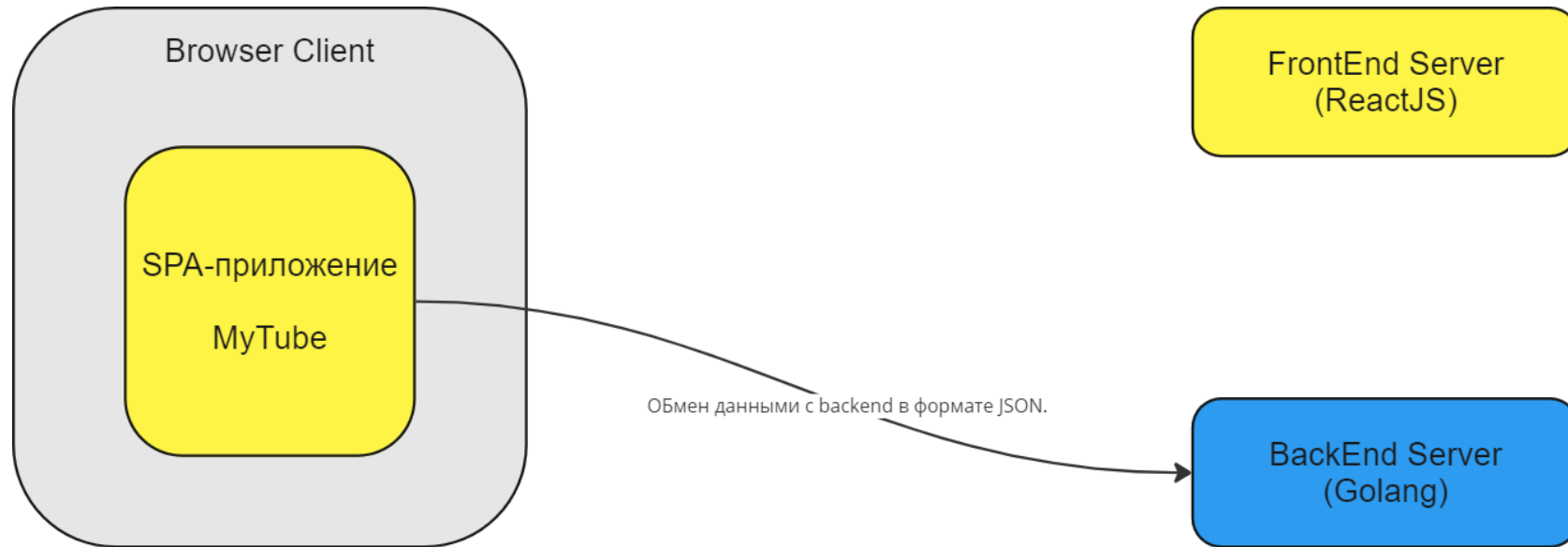
Авторизация – это процесс проверки, что пользователь имеет права на те действия, которые пытается совершить

\* На экзамене требуется более развернутый ответ

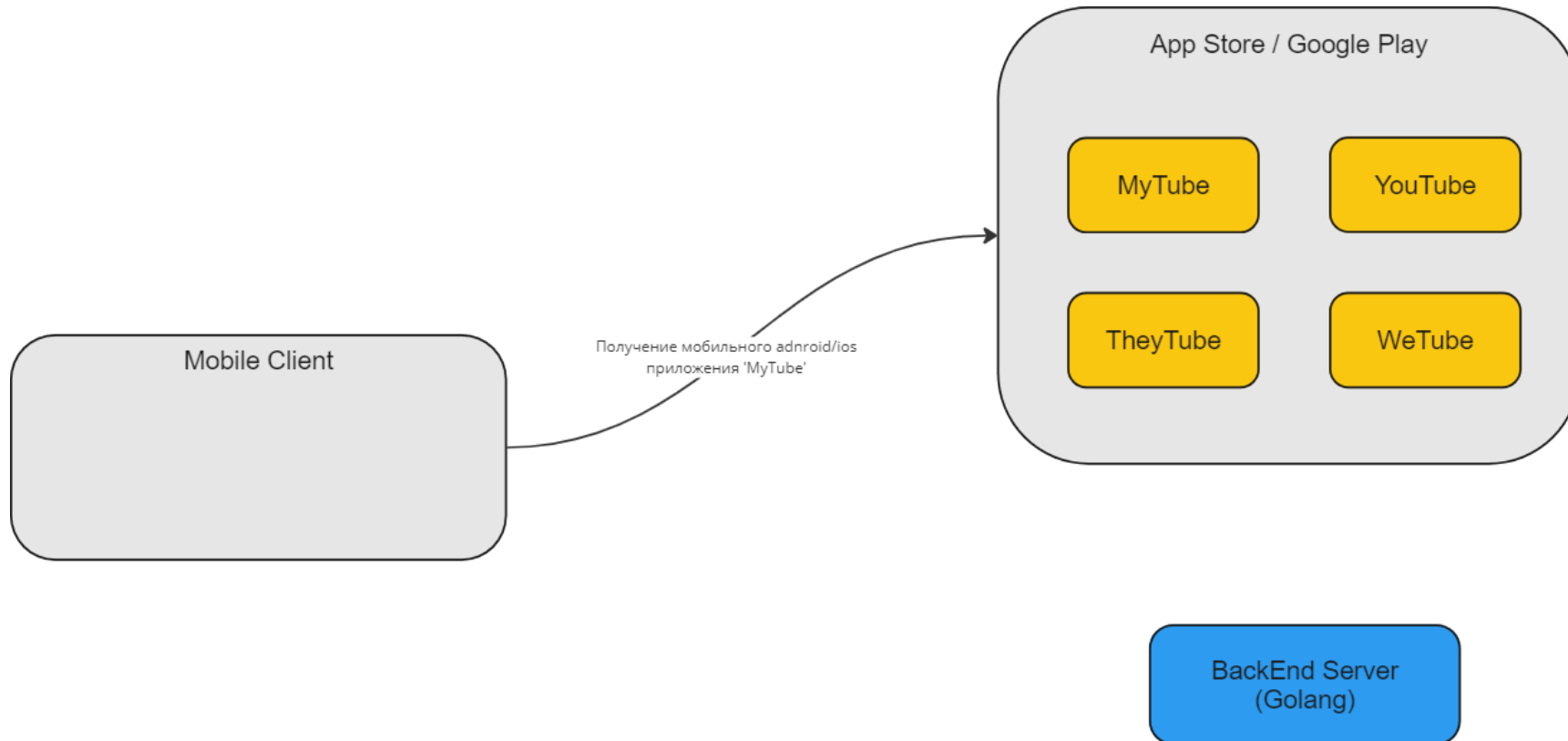
# Как связаны между собой Front и Back?



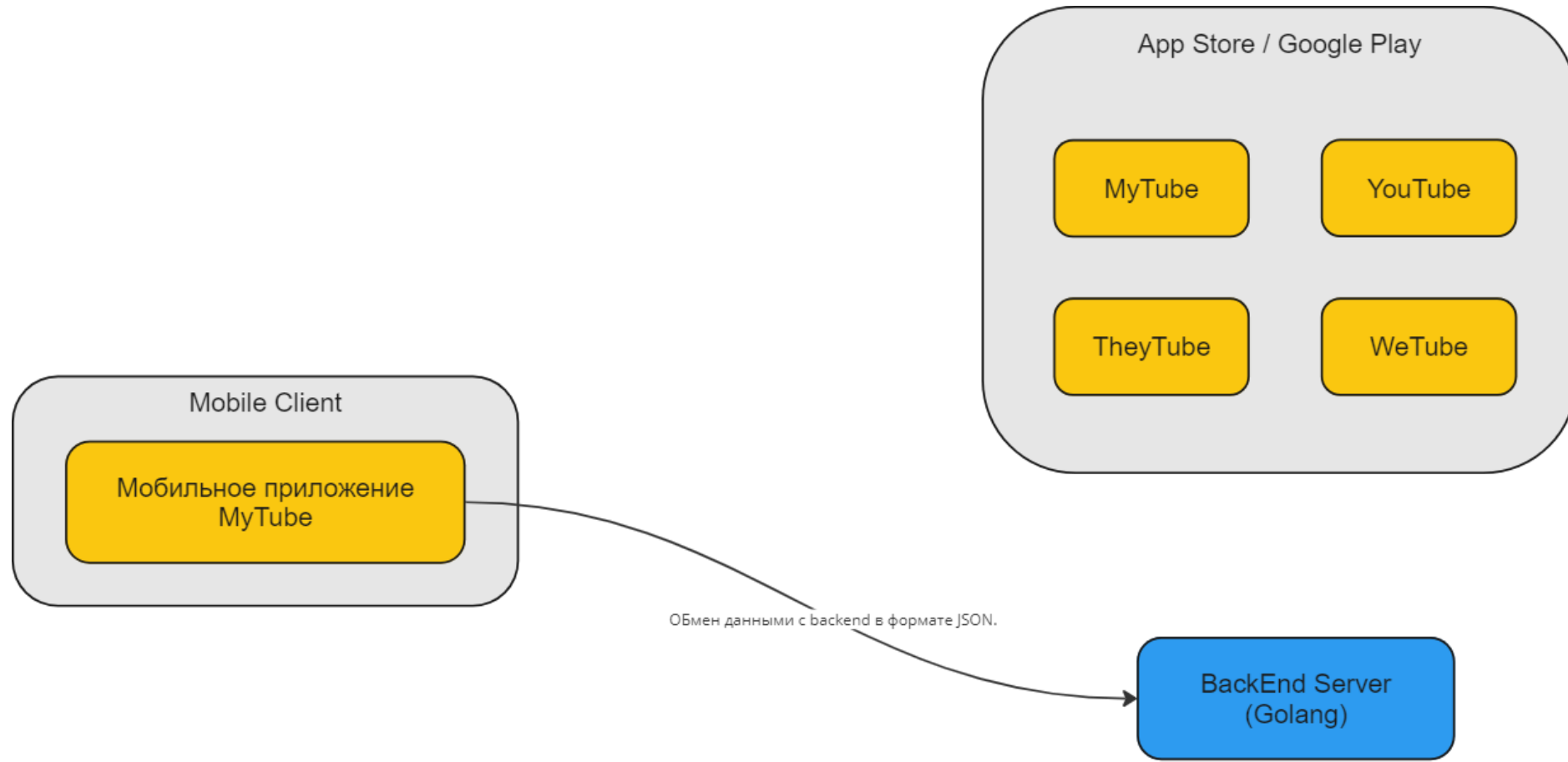
# Как связаны между собой Front и Back?



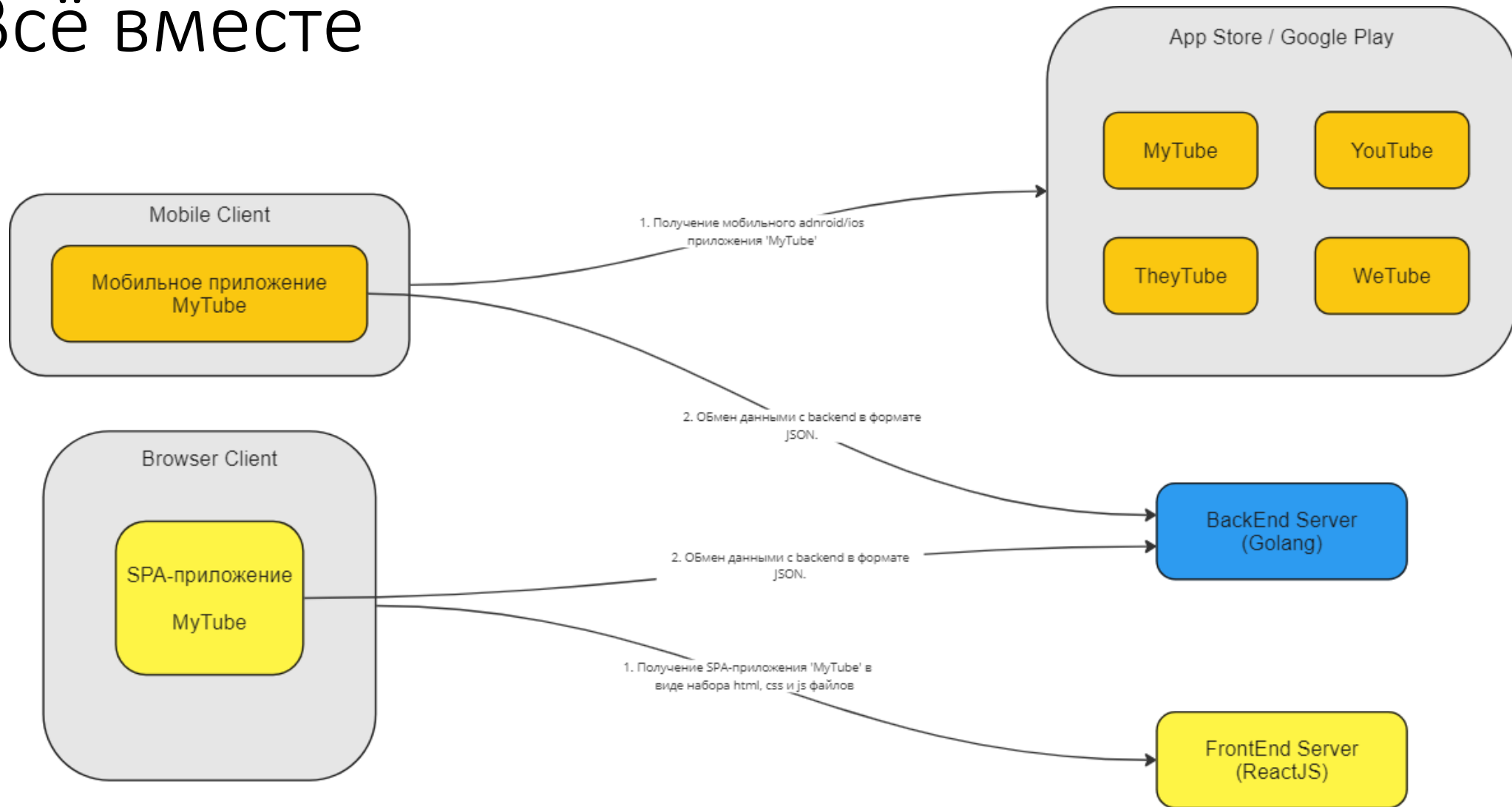
# Аналогия с Mobile App



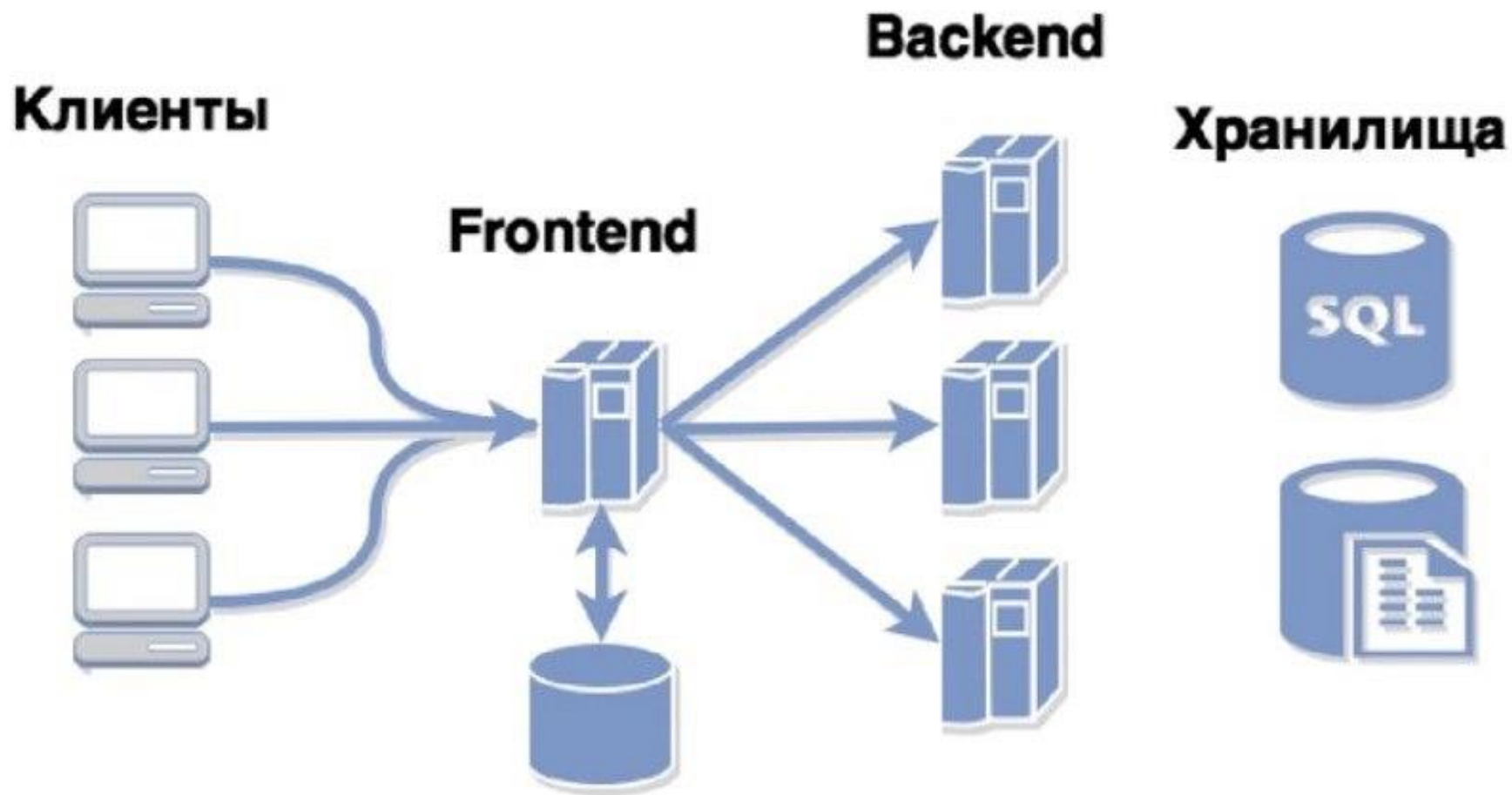
# Аналогия с Mobile App



# Всё вместе



# Альтернативная схема





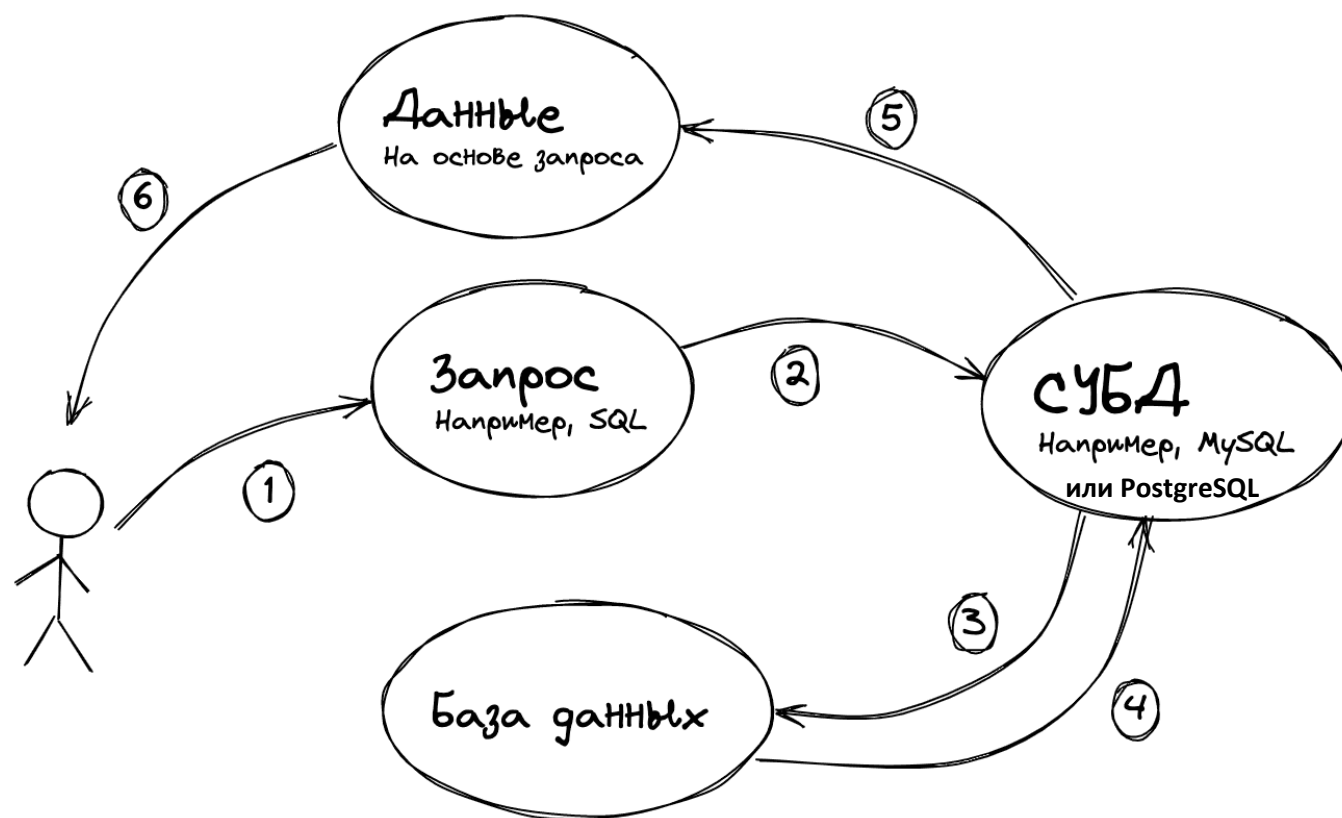
СУБД

?

# СУБД

Система управления базами данных (СУБД) — это система, которая обеспечивает долговременное хранение данных, а также безопасные операции записи, чтения, обновления и удаления

База данных (БД) - это набор данных, хранящихся в структурированном виде



# Основные виды

## Реляционные (SQL)



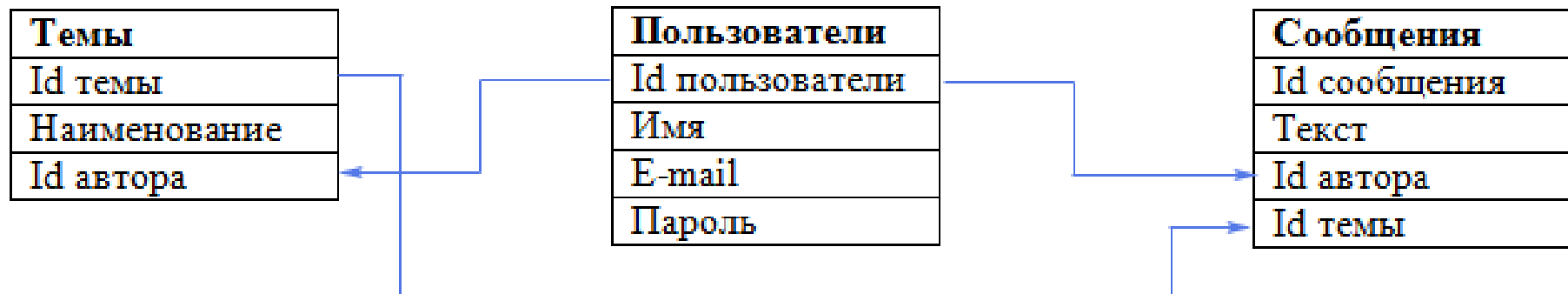
## Не реляционные (NoSQL)



# Реляционные

Наиболее известными реляционными базами данных являются Open Source проекты PostgreSQL и MySQL, а также проприетарные решения Oracle и Microsoft SQL Server

Суть реляционных баз в хранении данных в связанных таблицах



# Реляционные

Есть 4 типа основных операций над данными в SQL

- SELECT ... FROM ... WHERE ...
- INSERT INTO ... VALUES ...
- UPDATE ... SET ... WHERE ...
- DELETE FROM ... WHERE ...

# \*Вопрос

Одна из ключевых фич реляционных СУБД – это транзакционность.

Любое изменение данных в СУБД – это транзакция, которая должна обладать свойствами, но какими?

\*Будет на экзамене

## \* Ответ

Одна из ключевых фич реляционных СУБД – это транзакционность.

Любое изменение данных в СУБД – это транзакция, которая должна обладать свойствами ACID:

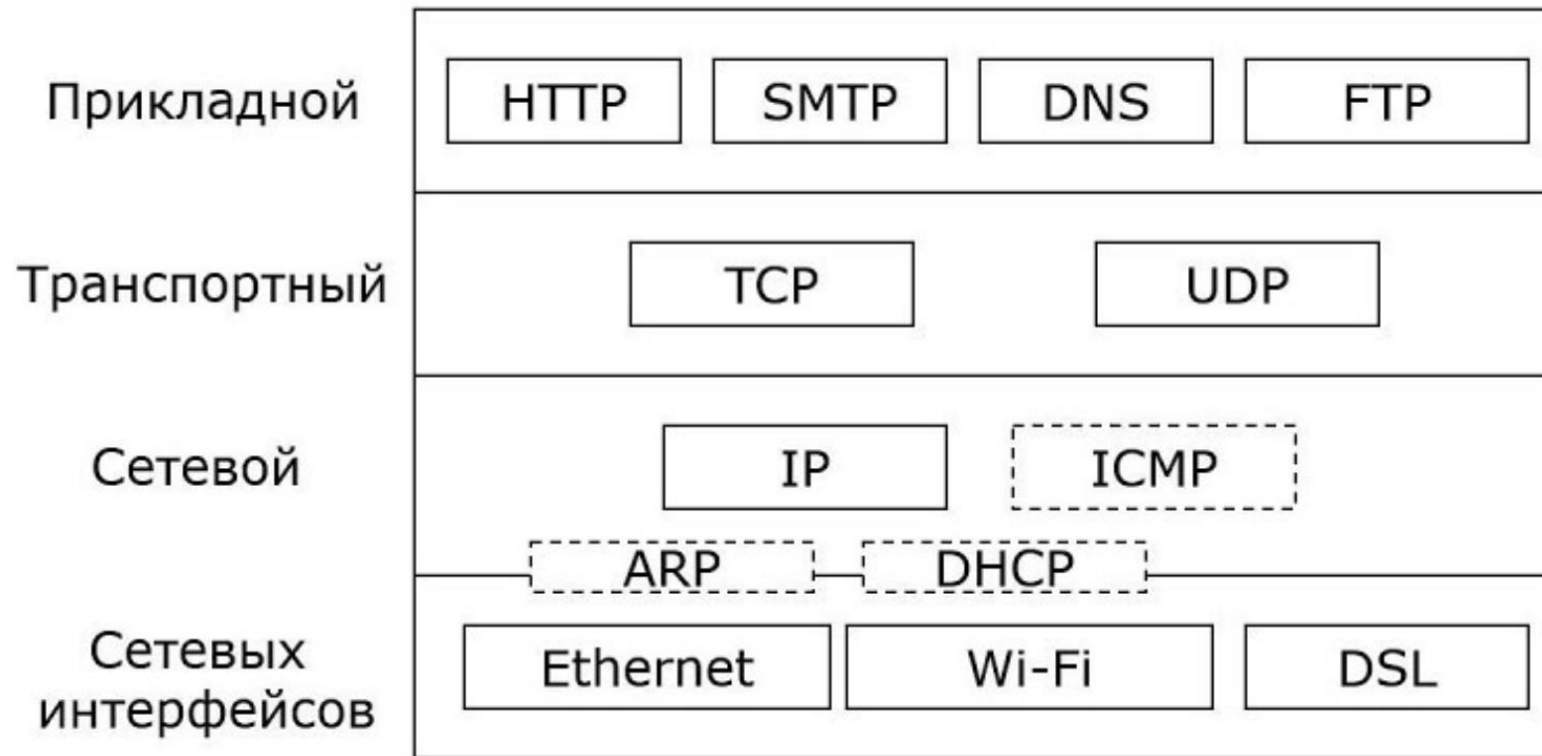
- Atomicity — Атомарность
- Consistency — Согласованность
- Isolation — Изолированность
- Durability — Надёжность

Подробнее можно почитать здесь: <https://habr.com/ru/articles/555920/>

\*На экзамене требуется более развернутый ответ

# Клиент-серверное взаимодействие

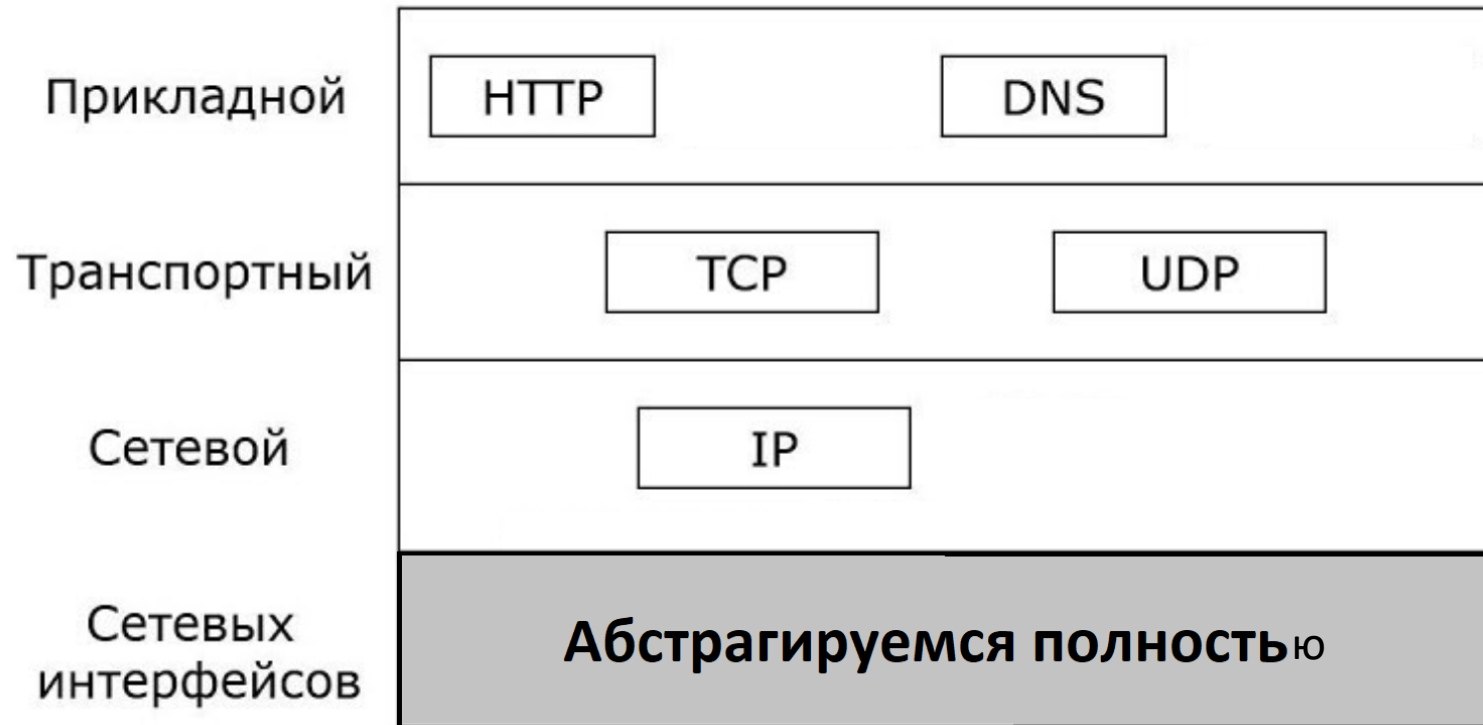
Осуществляется на базе стека протоколов TCP/IP





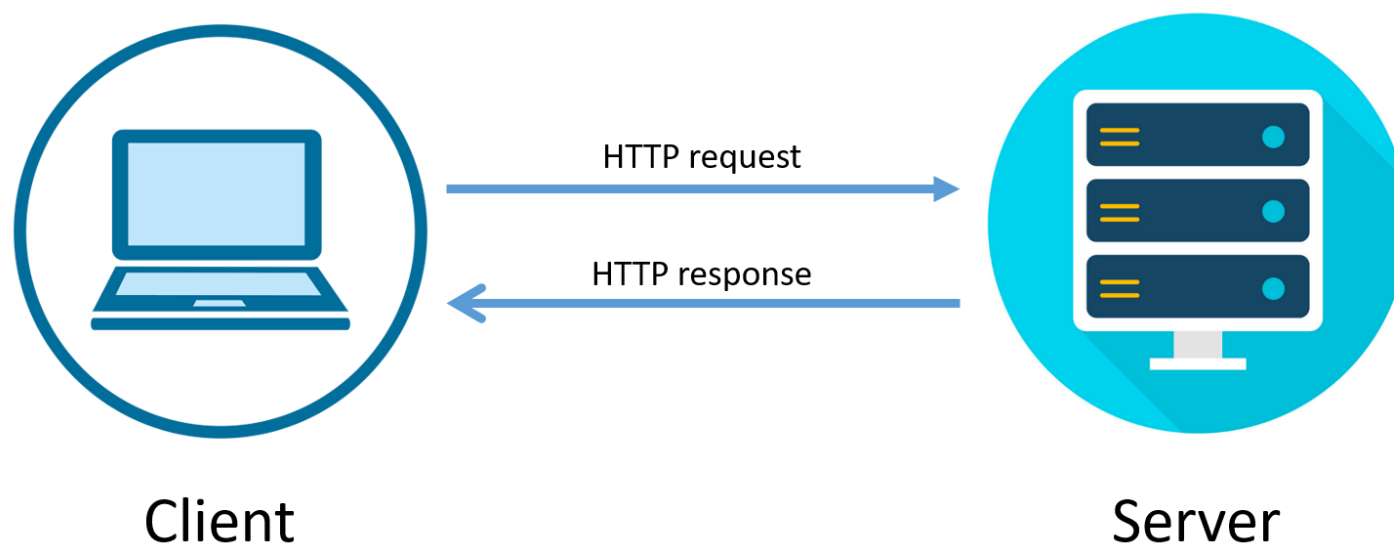
# Клиент-серверное взаимодействие

К счастью, нам нужно не так много

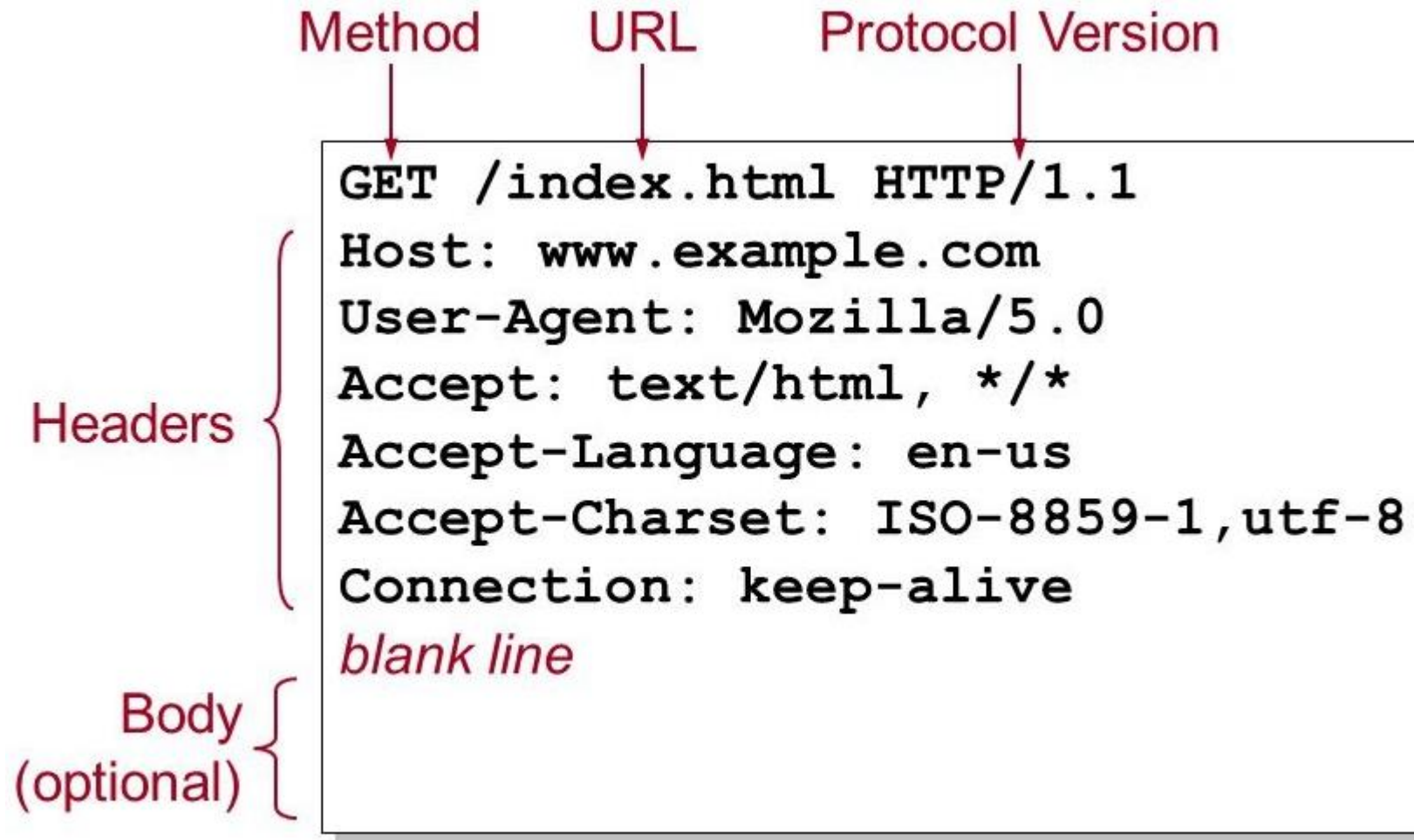


# HTTP

Взаимодействие ведется по HTTP в формате запрос-ответ. При этом, запрос всегда отправляет клиент, а ответ отдает сервер



# Структура HTTP-запроса



# Структура HTTP-ответа



# URI



# Адрес web-сервера

Представляет собой пару из IP и порта.

IP (например, 127.0.0.1) – это сущность протокола IP. Это уникальный сетевой идентификатор машины (виртуальной или физической) в сети

Порт (например, 8080)– это сущность протокола TCP. Это уникальный сетевой идентификатор программы, которая запущена на машине (виртуальное или физическое)

Вместе обычно записывается как:

`http://127.0.0.1:8080/<какой-то путь>`

# \*Вопрос

Почему мы обычно обращаемся к ресурсам в сети по доменным именам, а не ip+port, и почему это вообще работает?

\*Будет на экзамене

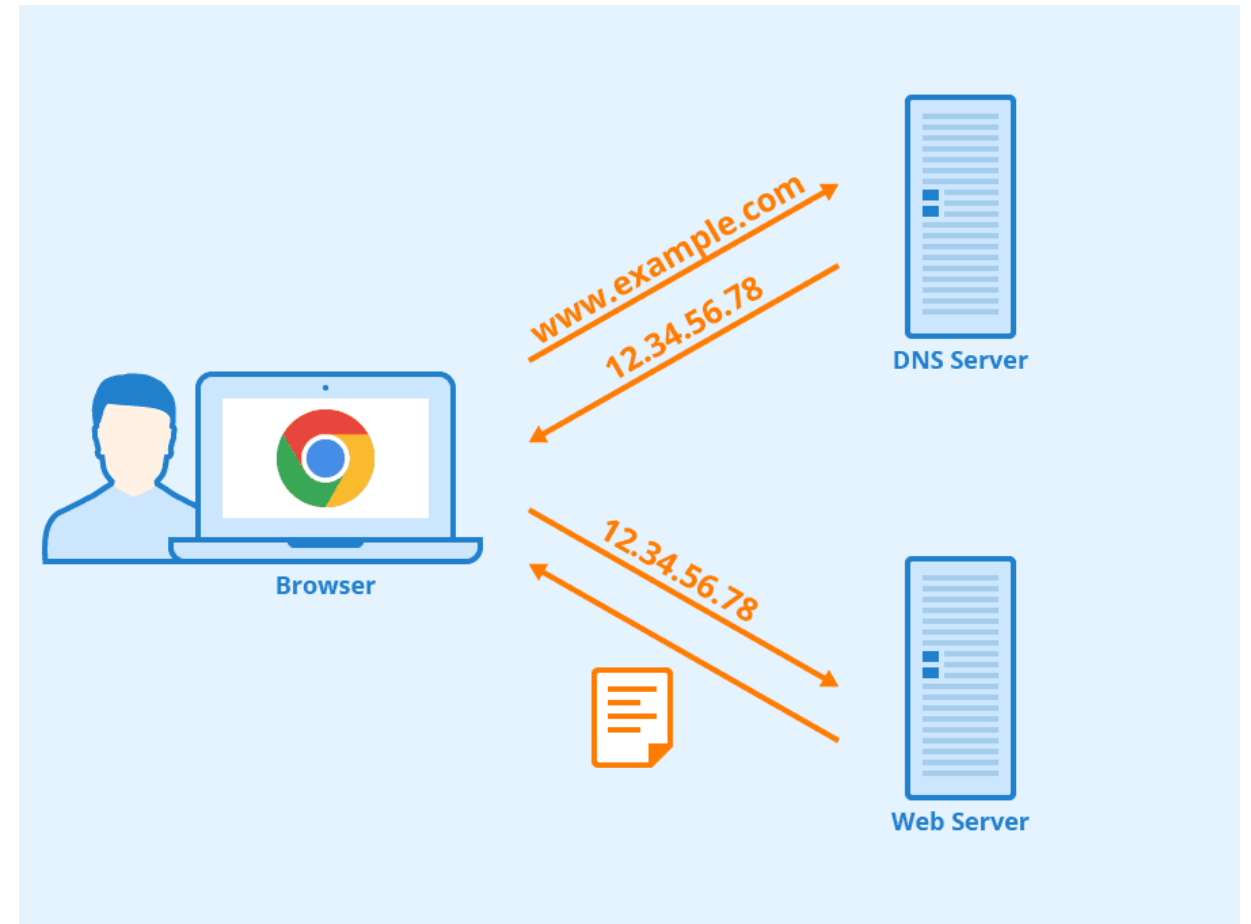
# \* Ответ

Мы не можем просто так обратиться к ресурсу по доменному имени (yandex.ru), нам сначала необходимо получить его ip-адрес через DNS

Подробнее можно посмотреть тут:




<https://howdns.works/ep1/>

\*На экзамене требуется более развернутый ответ





# BackEnd API

 **swagger**     Explore

**test : does fascinating stuff**

Show/Hide | List Operations | Expand Operations | Raw

**admin : Rest api for do operations on admin**

Show/Hide | List Operations | Expand Operations | Raw

GET	/admin/{userName}	Get specific admin
GET	/admin/admin-list/{item_per_list}/{page_number}	Get list of admin users
POST	/admin	Save specific admin
PUT	/admin	Update specific admin

**me : Me first swagger integration**

Show/Hide | List Operations | Expand Operations | Raw

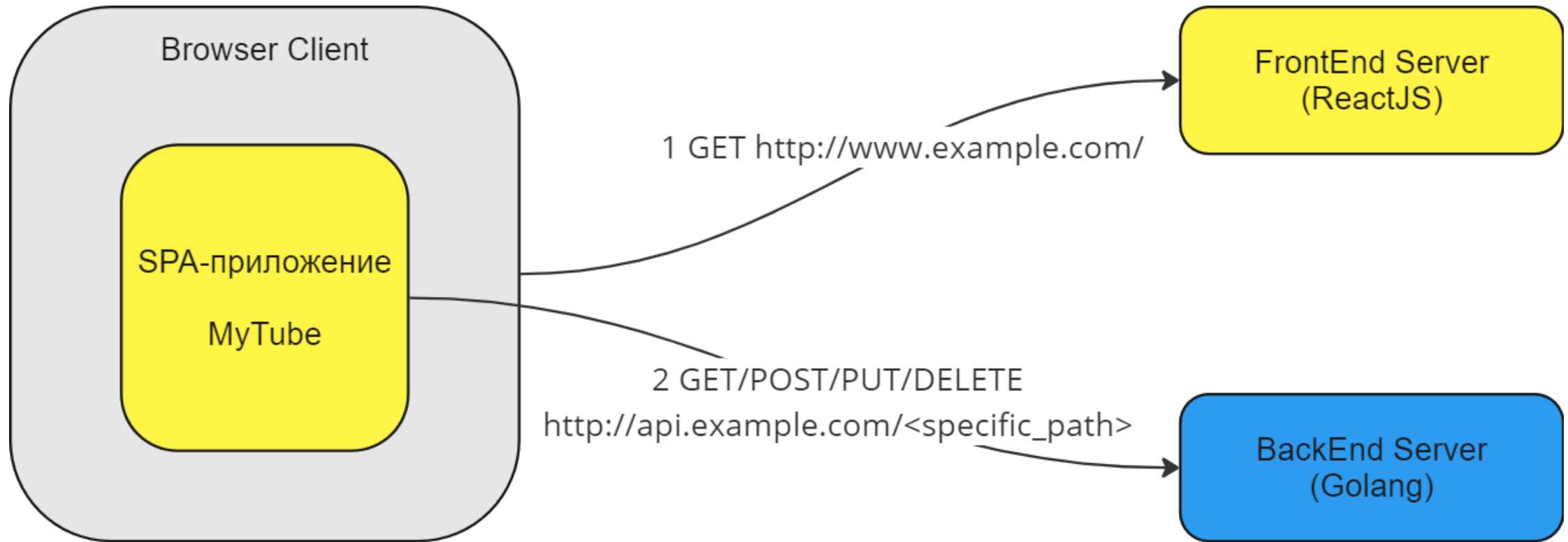
[ BASE URL: http://localhost:8080/SW-Protection-Backend-1.0-SNAPSHOT/rest/api-docs , API VERSION: 3 ]

# FrontEnd API

Обычно поддерживает просто GET запрос на получения SPA-приложения целиком, либо каких-то отдельных фрагментов html+css+js

GET <https://www.google.com/>

# Взаимодействие с API



# Взаимодействие с СУБД



# CRUD-операции для HTTP и SQL

Operation	SQL	HTTP
Create	INSERT	PUT / POST
Read (Retrieve)	SELECT	GET
Update (Modify)	UPDATE	PUT / POST / PATCH
Delete (Destroy)	DELETE	DELETE