Лекция-1

Введение в Frontend. Интернет. WEB.

Введение в Frontend

Что такое Frontend?

Frontend — это часть веб-приложения, с которой взаимодействует пользователь напрямую. Он включает в себя все, что пользователь видит и с чем он взаимодействует в веб-браузере: графический интерфейс, кнопки, текстовые поля, изображения, анимации и прочее.

Основные технологии, используемые для разработки фронтенда, включают HTML, CSS и JavaScript.

Цели и задачи Frontend-разработки:

1. **Создание пользовательского интерфейса (UI);**
2. **Обеспечение пользовательского опыта (UX);**
3. **Кроссбраузерная совместимость;**
4. **Адаптивный дизайн;**
5. **Производительность и оптимизация;**
6. **Взаимодействие с бекендом;**
7. **Обеспечение безопасности;**

Что такое Backend?

Backend — это серверная часть веб-приложения, которая отвечает за логику, обработку данных и взаимодействие с базами данных. Пользователь не видит и не взаимодействует напрямую с backend, однако именно он обеспечивает функциональность и работу всего приложения.

Основные задачи и цели Backend-разработки:

1. **Обработка данных;**
2. **Взаимодействие с базами данных;**
3. **Логика приложения (бизнес-логика);**
4. **Аутентификация и авторизация;**
5. **Работа с API;**
6. **Масштабируемость и производительность;**
7. **Обеспечение безопасности;**

Что такое UI?

UI (User Interface) — это интерфейс пользователя, который представляет собой часть программного обеспечения или устройства, через который пользователь взаимодействует с системой. UI включает в себя различные элементы, которые пользователь видит на экране и с которыми он взаимодействует для выполнения различных задач и операций.

Основные аспекты UI:

1. **Элементы интерфейса**: Это все, что видит и использует пользователь, включая кнопки, поля ввода, меню, иконки, окна, диалоговые окна и т.д.
2. **Цель**: Основная задача UI — обеспечить простоту и эффективность взаимодействия пользователя с приложением или устройством. Хороший UI должен быть интуитивно понятным, легким в использовании и удовлетворять потребности пользователей.
3. **Дизайн интерфейса**: UI дизайн включает в себя выбор цветовой схемы, шрифтов, расположение элементов на экране, стилизацию иконок и другие аспекты, которые делают интерфейс привлекательным и функциональным.
4. **Типы UI**: Существуют различные типы пользовательских интерфейсов, включая графический пользовательский интерфейс (GUI), командный интерфейс (CLI), веб-интерфейсы (Web UI), мобильные интерфейсы (Mobile UI) и т.д.

Интернет

Интернет.

Интернет - глобальная сеть компьютеров, обменивающихся данными по стандартным протоколам. Разработчику важно понимать его работу для создания эффективных, безопасных и масштабируемых приложений.

Клиент-Серверная архитектура:

- **Сервер** - устройство или программа, предоставляющая услуги или ресурсы другим устройствам (клиентам) в сети.
- **Клиент** - устройство или программа, обращающаяся к серверу для получения доступа к услугам, данным или ресурсам.

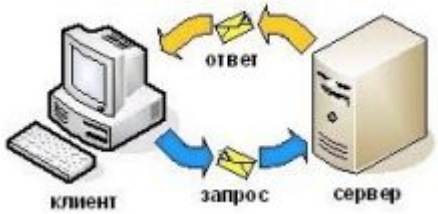
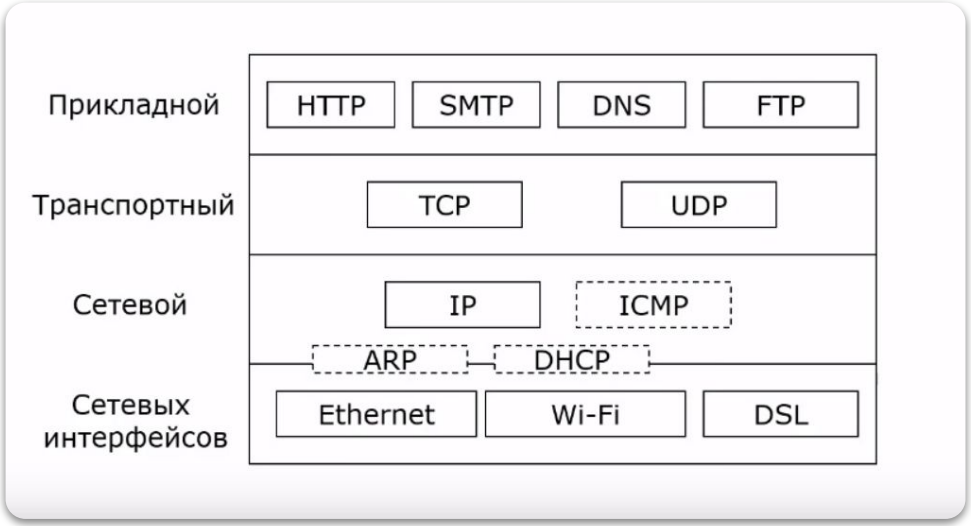


Схема TCP/IP



tcp/ip представляет архитектуру - то как устройства в сети взаимодействуют друг с другом.

Прикладной уровень: предоставляет приложениям доступ к сетевым службам и ресурсам. Он содержит различные протоколы и стандарты, используемые для обмена данными между приложениями на разных устройствах.

Транспортный: управляет передачей данных между устройствами-отправителями и устройствами-получателями.

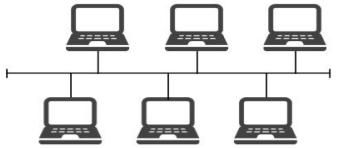
Сетевой: обеспечивает маршрутизацию пакетов данных по сети и их доставку от отправителя к получателю через различные сети.

Сетевых интерфейсов: это уровень доступа к сети. отвечает за физическое подключение устройств к сети и передачу данных через конкретные сетевые технологии, такие как Ethernet или Wi-Fi.

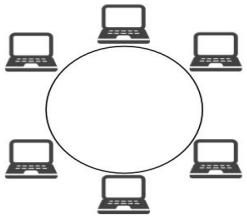
Сети и их классификации.

Сеть - это просто группа компьютеров или устройств, которые связаны друг с другом.

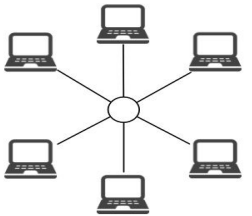
Например, у вас дома может быть своя сеть компьютеров, а у вашего соседа тоже своя. Когда все эти сети объединяются, они образуют Интернет - своего рода сеть сетей.



Шинная топология



Кольцевая топология



Звездная топология

Классификация сетей по размеру:

- **LAN (Local Area Network)** - (локальная сеть) маленькая сеть для дома, офиса или здания. *Например*, сеть WiFi в офисе компании.
- **MAN (Metropolitan Area Network)** - сеть для города. *Например*, соединение нескольких кампусов.
- **WAN (Wide Area Network)** - расширенная сеть для больших расстояний, между городами или странами. (стандартный интернет)
- **PAN (Personal Area Network)** - личная сеть (bluetooth соединение между устройствами)

Классификация сетей по топологии:

- **Звезда** - устройства подключены к центральному хабу или коммутатору.
- **Кольцо** - устройства соединены в кольцо, каждое с двумя соседями.
- **Шинная топология** - устройства подключены к единому кабелю.
- и другие (уже менее важные) Древовидная топология..

Скорость передачи данных.

Скорость передачи данных - количество данных, которые могут быть переданы через сеть или канал связи за определенный промежуток времени. Обычно измеряется в битах в секунду (bps) или его кратных единицах, таких как килобит в секунду (kbps), мегабит в секунду (Mbps) или гигабит в секунду (Gbps).

Скорость передачи данных важна для определения, насколько быстро данные могут быть отправлены или получены на конечных устройствах.

Пропускная способность обозначает максимальную скорость передачи данных через сеть или канал связи. Она измеряется также в битах в секунду (bps), kbps, Mbps, Gbps и т.д.

Проводные сети.

Проводные сети используют физические кабели для передачи данных между устройствами. Это обеспечивает высокую скорость, надежность и безопасность соединения. Проводные сети могут использовать медные или оптоволоконные кабели в зависимости от требований к пропускной способности и расстоянию передачи данных.

Типы медных кабелей:

- **Twisted Pair (Витая пара):** Наиболее распространенный тип кабеля для локальных сетей. Состоит из пары медных проводов, скрученных вместе для уменьшения электромагнитных помех.
  - **Категории витой пары:**
    - **Cat5e:** Поддерживает скорости до 1 Гбит/с на расстоянии до 100 метров.
    - **Cat6:** Поддерживает скорости до 10 Гбит/с на расстоянии до 55 метров.
    - **Cat6a:** Поддерживает скорости до 10 Гбит/с на расстоянии до 100 метров.
    - **Cat7:** Поддерживает скорости до 10 Гбит/с и выше с улучшенной защитой от помех.
- **Coaxial Cable (Коаксиальный кабель):** Используется в телевизионных антеннах, кабельном телевидении и некоторых типах локальных сетей. Обеспечивает хорошую защиту от электромагнитных помех.
- **Ethernet кабели:** Обычно используют витую пару и могут иметь различные категории (Cat5e, Cat6, Cat7) в зависимости от требуемой скорости и дальности передачи данных.

и другие…

Плюсы и минусы медных кабелей.

Преимущества медных кабелей:

- Легкость установки и прокладки.
- Доступная стоимость.
- Широкая совместимость с сетевым оборудованием.

Недостатки медных кабелей:

- Ограниченная пропускная способность по сравнению с оптоволоконными кабелями.
- Подверженность электромагнитным помехам и наводкам.
- Ограниченное расстояние передачи данных (до 100 метров без повторителей).

Беспроводные сети.

Беспроводные сети используют радиоволны для передачи данных между устройствами, что обеспечивает гибкость и удобство подключения без использования физической кабельной инфраструктуры.

Wi-Fi

Типы и стандарты Wi-Fi:

1. **802.11a:** Работает в диапазоне 5 ГГц, скорость до 54 Мбит/с.
2. **802.11b:** Работает в диапазоне 2.4 ГГц, скорость до 11 Мбит/с.
3. **802.11g:** Работает в диапазоне 2.4 ГГц, скорость до 54 Мбит/с.
4. **802.11n:** Работает в диапазонах 2.4 и 5 ГГц, скорость до 600 Мбит/с.
5. **802.11ac:** Работает в диапазоне 5 ГГц, скорость до 3.2 Гбит/с.
6. **802.11ax (Wi-Fi 6):** Работает в диапазонах 2.4 и 5 ГГц, скорость до 9.6 Гбит/с.

Bluetooth

Типы и версии Bluetooth:

1. **Bluetooth 1.0 - 3.0:** Обеспечивают скорости передачи данных от 1 до 24 Мбит/с.
2. **Bluetooth 4.0:** Включает режим Low Energy (LE), что позволяет экономить энергию при подключении устройств.
3. **Bluetooth 5.0:** Увеличивает дальность передачи данных и скорость до 2 Мбит/с в режиме LE.

Плюсы и минусы WiFi.

Преимущества Wi-Fi:

- Удобство и мобильность: устройства могут подключаться без кабелей.
- Простота установки и настройки.
- Возможность подключения множества устройств к одной точке доступа.

Недостатки Wi-Fi:

- Ограниченная пропускная способность и скорость передачи по сравнению с проводными сетями.
- Зависимость от качества сигнала и покрытия сети.
- Подверженность помехам от других беспроводных устройств и препятствий (стены, мебель).

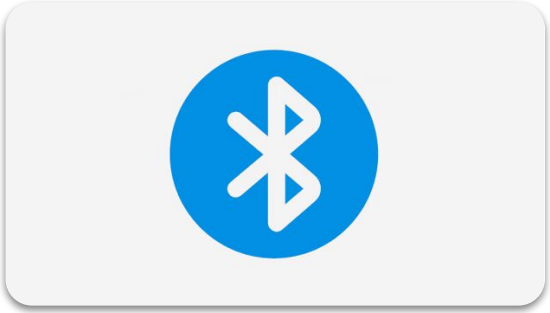
Плюсы и минусы Bluetooth.

Преимущества Bluetooth:

- Низкое энергопотребление, особенно в режиме LE.
- Простота использования и автоматическое сопряжение устройств.
- Широкая совместимость с различными устройствами (наушники, клавиатуры, мыши, смартфоны и т.д.).

Недостатки Bluetooth:

- Ограниченная дальность передачи (обычно до 10 метров).
- Низкая скорость передачи данных по сравнению с Wi-Fi и проводными сетями.
- Подверженность помехам от других беспроводных устройств.



Мобильные сети.

Мобильные сети — это системы беспроводной связи, которые позволяют мобильным устройствам, таким как смартфоны, планшеты и ноутбуки, обмениваться данными и голосовыми вызовами через радиоволны. Мобильные сети предоставляют пользователям возможность оставаться на связи практически в любом месте, обеспечивая доступ к интернету, голосовым услугам и мультимедийным сервисам.

Поколения мобильной сети:

- **1G** — первое поколение мобильных сетей, появившееся в 1980-х годах; (аналоговая передача данных) AMPS
- **2G** — второе поколение, введенное в 1990-х годах; (цифровая передача данных) GSM, CDMA
- **3G** — третье поколение, появившееся в начале 2000-х годов; (высокоскоростная передача данных до 2 Мбит/с) UMTS
- **4G** — четвертое поколение, запущенное в конце 2000-х и начале 2010-х годов. LTE, и до 1 Гбит/секунду
- **5G** — пятое поколение, которое начало разворачиваться в 2020-х годах; до 10 Гбит/секунду

Сети и связь с Frontend.

Мобильные сети и другие типы сетей играют важную роль в работе фронтенд-разработчиков, поскольку они влияют на доступность, производительность и пользовательский опыт веб-приложений и сайтов. Фронтенд-разработчики создают веб-приложения, которые должны быть доступны пользователям через различные типы сетей, включая мобильные сети (3G, 4G, 5G), Wi-Fi и проводные сети. Для обеспечения хорошего пользовательского опыта важно учитывать скорость и надежность соединения.

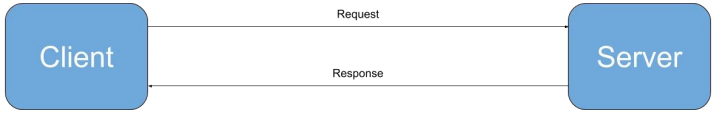
Скорость сети и её задержки влияют на время загрузки страниц и взаимодействие с веб-приложением. Фронтенд-разработчики должны оптимизировать производительность своих приложений, чтобы минимизировать влияние медленных или нестабильных сетей. Основные аспекты оптимизации включают:

- **Минимизация размера ресурсов:** Уменьшение размера CSS, JavaScript и изображений через сжатие и минимизацию.
- **Использование CDN (Content Delivery Network):** Распространение контента через сервера, расположенные ближе к пользователям, для ускорения загрузки.
- **Асинхронная загрузка:** Загрузка скриптов и других ресурсов асинхронно, чтобы не блокировать отображение страницы.
- **Кэширование:** Настройка кэширования для уменьшения количества запросов к серверу и ускорения повторных загрузок.

Пользователи могут получать доступ к веб-приложениям через различные устройства (смартфоны, планшеты, настольные компьютеры) и сети (медленные 3G или быстрые Wi-Fi). Адаптивный дизайн позволяет веб-приложениям корректно отображаться и работать на всех устройствах и при любых скоростях соединения.

Веб-приложения часто взаимодействуют с серверами через API для получения данных и выполнения операций. Фронтенд-разработчики должны учитывать задержки и потенциальные потери данных при работе с сетями. Сетевые соединения могут быть уязвимы для атак, таких как перехват данных или вмешательство в передаваемые данные. Фронтенд-разработчики должны учитывать безопасность при работе с сетями. (применение HTTPS, CORS и тп)

HTTP и HTTPS



HTTP (HyperText Transfer Protocol) и **HTTPS (HyperText Transfer Protocol Secure)** являются основными протоколами для передачи данных в интернете. HTTP - протокол передачи данных между клиентом и сервером. Браузер отправляет запрос, сервер отвечает. HTTPS - защищенная версия HTTP, обеспечивает безопасное взаимодействие.

Особенности HTTP:

- HTTP обычно использует порт 80.
- HTTP-запросы состоят из строки запроса (метод, URL, версия протокола), заголовков (headers) и, иногда, тела запроса.
- использует методы GET, POST, PUT, DELETE и другие.
- использует статусы ответов (HTTP-коды).

Особенности HTTPS:

- HTTPS обычно использует порт 443.
- HTTPS использует SSL (Secure Sockets Layer) или его более современную версию TLS (Transport Layer Security) для шифрования данных.
- Для использования HTTPS требуется цифровой сертификат, который подтверждает подлинность веб-сайта и обеспечивает безопасное соединение.
- Повышенная безопасность за счет шифрования данных, защита от атак типа "человек посередине" (MITM), аутентификация сервера.

HTTP-коды.

HTTP-код состояния — это числовой код, включенный в ответ HTTP-сервера, который сообщает клиенту (например, веб-браузеру) о результате выполнения запроса. Каждый код состояния состоит из трех цифр и имеет определенное значение, которое указывает на успешность выполнения запроса или на наличие ошибки.

Информационные (1xx)

- **100 Continue:** Сервер получил начальные данные запроса и клиент может продолжать отправку тела запроса.
- **101 Switching Protocols:** Сервер понимает запрос клиента на смену протокола и согласен на это.

Успешные (2xx)

- **200 OK:** Запрос выполнен успешно. Ответ содержит запрашиваемый ресурс.
- **201 Created:** Запрос выполнен, и в результате был создан новый ресурс.
- **202 Accepted:** Запрос принят для обработки, но обработка еще не завершена.
- **204 No Content:** Запрос выполнен успешно, но тело ответа пустое.

Перенаправления (3xx)

- **301 Moved Permanently:** Запрашиваемый ресурс был перемещен на постоянный новый URL.
- **302 Found (Moved Temporarily):** Запрашиваемый ресурс временно доступен по другому URL.
- **304 Not Modified:** Запрашиваемый ресурс не был изменен с момента последнего запроса.
- **307 Temporary Redirect:** Запрашиваемый ресурс временно доступен по другому URL, но следует использовать исходный URL для будущих запросов.

Клиентские ошибки (4xx)

- **400 Bad Request:** Сервер не может обработать запрос из-за ошибки клиента (например, неверный синтаксис).
- **401 Unauthorized:** Для доступа к запрашиваемому ресурсу требуется аутентификация.
- **403 Forbidden:** Сервер понимает запрос, но отказывается его выполнять из-за недостатка прав.
- **404 Not Found:** Запрашиваемый ресурс не найден на сервере.
- **405 Method Not Allowed:** Метод, указанный в запросе, не поддерживается для данного ресурса.
- **408 Request Timeout:** Сервер ожидает запрос от клиента слишком долго и завершает соединение.
- **409 Conflict:** Запрос не может быть выполнен из-за конфликта с текущим состоянием ресурса.

Серверные ошибки (5xx)

- **500 Internal Server Error:** Внутренняя ошибка сервера. Сервер не может выполнить запрос из-за непредвиденной ситуации
- **501 Not Implemented:** Сервер не поддерживает функциональность, необходимую для выполнения запроса.
- **502 Bad Gateway:** Сервер получил недействительный ответ от вышестоящего сервера или прокси.
- **503 Service Unavailable:** Сервер временно недоступен из-за перегрузки или обслуживания.
- **504 Gateway Timeout:** Сервер не получил вовремя ответ от вышестоящего сервера или прокси.

Методы взаимодействия с сервером.

Методы взаимодействия с сервером представляют собой совокупность технических способов, используемых для эффективного обмена информацией между клиентскими устройствами и удаленными серверами. Эти методы включают различные протоколы и типы запросов, направленные на получение, изменение или удаление данных на сервере в соответствии с потребностями приложений и пользователей.

Основные методы:

- **GET**: Этот метод используется для запроса данных от сервера. Клиент отправляет GET-запрос, указывая URL ресурса, который требуется получить. Ответ сервера содержит запрошенные данные. GET-запросы обычно являются безопасными и идемпотентными, то есть они не должны изменять состояние сервера.
- **POST**: POST используется для отправки данных на сервер для обработки. В отличие от GET, который просто запрашивает данные, POST отправляет данные на сервер в теле запроса. Этот метод может использоваться для создания новых ресурсов на сервере или для выполнения операций, требующих отправки большого объема данных.
- **PUT**: PUT используется для обновления существующего ресурса на сервере. В запросе PUT клиент отправляет обновленные данные ресурса на сервер, заменяя текущее состояние ресурса полностью или частично.
- **DELETE**: DELETE используется для удаления ресурса на сервере по указанному URL. Этот метод предполагает, что удаление ресурса на сервере необратимо.
- **PATCH**: PATCH похож на PUT, но он используется для частичного обновления ресурса на сервере. То есть в запросе PATCH клиент отправляет только те данные, которые нужно изменить, без необходимости отправлять все данные ресурса.

TCP и UDP.

TCP (Transmission Control Protocol) и UDP (User Datagram Protocol) являются двумя основными протоколами транспортного уровня в сетях компьютеров. Они определяют, как данные передаются между устройствами в сети, но имеют разные характеристики и применения:

TCP (Transmission Control Protocol):

1. **Характеристики**: TCP является надежным и ориентированным на соединение протоколом. Это означает, что он обеспечивает установление соединения между отправителем и получателем данных и гарантирует, что данные будут доставлены в правильном порядке и без ошибок.
2. **Функциональность**: TCP используется для передачи данных, где важна надежность и точность, например, при передаче веб-страниц, электронной почты, файлов и других приложений, где ошибки или потеря данных могут привести к проблемам.

UDP (User Datagram Protocol):

1. **Характеристики**: UDP является простым и без соединения протоколом. Он не обеспечивает надежной доставки данных и не гарантирует порядок доставки пакетов. UDP работает быстрее и имеет меньшую нагрузку на сеть, чем TCP.
2. **Функциональность**: UDP часто используется в приложениях, где скорость и эффективность передачи данных важнее, чем точность и надежность, например, в видео- и аудио-потоках, онлайн играх, VoIP и других реальном времени приложениях.

Отличия TCP & UDP

	TCP	UDP
Состояние соединения	Требуется установленное соединение для передачи данных (после завершения передачи соединение должно быть закрыто)	Протокол без обязательного соединения и требований к открытию, поддержанию или прерыванию соединения
Гарантия доставки	Может гарантировать доставку данных получателю	Нет гарантии доставки данных получателю
Повторная передача данных	Повторная передача пакетов в случае потери одного из них	Нет повторной передачи потерянных пакетов
Проверка ошибок	Выполняется полная проверка ошибок	Действует базовый механизм проверки ошибок. Использует вышестоящие протоколы для проверки целостности
Метод передачи	Данные считываются как поток байтов, информация передаётся по границам сегментов	У пакетов есть сформированные границы. Пакеты отправляются по отдельности и проверяются на целостность при получении
Сферы применения	Используется для передачи сообщений электронной почты, HTML-страниц браузеров	Подходит для видеоконференций, потокового вещания, DNS, VoIP, IPTV

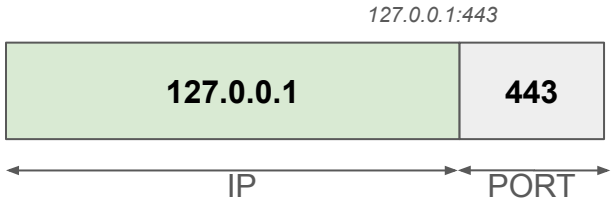
Что такое IP? Что такое порт?

TCP/IP - набор протоколов для надежной передачи данных в сети, где TCP управляет передачей, а IP - маршрутизацией и адресацией.

IP адрес (Internet Protocol address) является уникальным числовым идентификатором каждого устройства (компьютера, маршрутизатора и т.д.), подключенного к сети Интернет или локальной сети. IP адрес позволяет маршрутизаторам и другим сетевым устройствам направлять пакеты данных по сети от отправителя к получателю.

- **IPv4 (Internet Protocol version 4)**: стандартный формат IP адресов, который используется на сегодняшний день в большинстве сетей. IPv4 адрес состоит из четырех десятичных чисел, разделенных точками (например, 192.168.0.1). Каждое число (октет) может принимать значения от 0 до 255.
- **IPv6 (Internet Protocol version 6)**: IPv6 был разработан как следующее поколение IP адресов для замены IPv4. Адрес IPv6 состоит из восьми групп по четыре шестнадцатеричных символа, разделенных двоеточиями (например, 2001:0db8:85a3:0000:0000:8a2e:0370:7334).

Порт (port) — это числовой идентификатор, который используется для идентификации конкретного процесса или сервиса, который работает на устройстве в сети. В TCP/IP модели порт является частью адреса, который определяет, к какому процессу на конкретном устройстве должны быть направлены данные.



Доменное имя и система преобразований DNS.

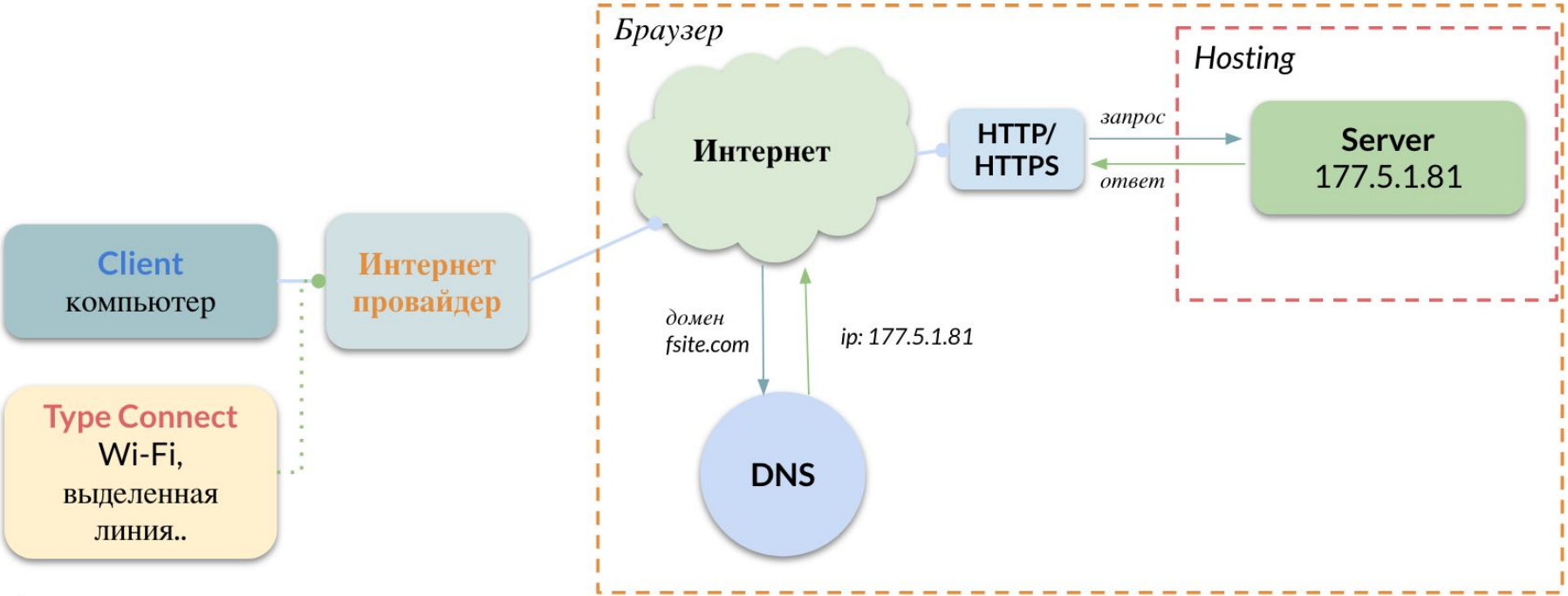
Доменное имя - это читаемое человеком имя, используемое для идентификации веб-сайта, например, google.com, gmail.com, и др. DNS преобразует доменные имена в IP-адреса, что позволяет устройствам найти правильные серверы.

DNS - система преобразования доменных имен в IP-адреса для правильного направления запросов на серверы.

DNS выполняет несколько ключевых функций:

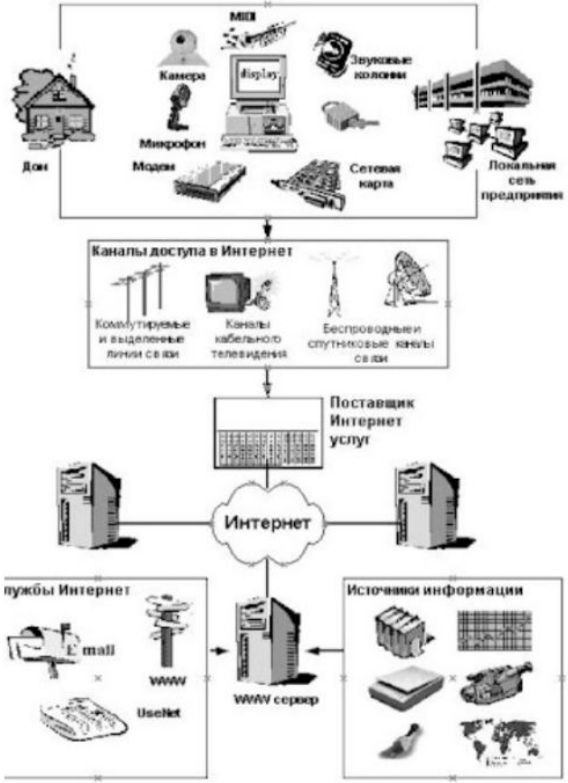
- **Резольвинг**: Преобразование доменных имен в соответствующие IP-адреса.
- **Обратное разрешение**: Преобразование IP-адресов обратно в доменные имена.
- **Кэширование**: Хранение результатов предыдущих запросов для улучшения производительности и снижения нагрузки на сеть.

Упрощенная схема работы интернета.



*__интернет провайдер__ - предоставляет доступ к сети интернет, а __type connect__ - тип соединения с провайдером.
в более сложной схеме еще нужен маршрутизатор, proxy-server, или др сетевые устройства.

Схема-2


MIDI
Камера
Звуковые колонки
Микрофон
Модем
Сетевая карта
Локальная сеть предприятия
Дом
Каналы доступа в Интернет
Коммутируемые и выделенные линии связи
Каналы кабельного телевидения
Беспроводные и спутниковые каналы связи
Поставщик Интернет услуг
Интернет
Источники информации
E-mail
WWW
UseNet
WWW сервер


1 - компьютеры пользователей сети Интернет
2 - сервер Интернет провайдера
3 - серверы сети Интернет
Компьютер
Запрос
Ответ
Запросы
Ответы (запрашиваемая информация)
inetgramotnost.ru
900 × 572

WEB1.0
WEB2.0
WEB3.0

WEB

WEB1.0: Первое поколение веба, преимущественно статические страницы без интерактивности, представленные в основном в виде информационных брошюр.

WEB2.0: Эволюция веба, где пользователи стали активными участниками, создающими и обменивающимися контентом. Появление социальных сетей, блогов, вики, видеохостингов и других платформ, где пользовательский контент стал ключевым.

WEB3.0: Концепция развития веба, фокусирующаяся на персонализированных и интеллектуальных услугах. WEB3.0 стремится к созданию интернета, где данные становятся более структурированными, алгоритмы становятся более интеллектуальными, и пользователи имеют больше контроля над своими данными и опытом в сети. Технологии, такие как искусственный интеллект, блокчейн и интернет вещей, могут играть ключевую роль в реализации этой концепции.

Веб-браузеры. Веб-хостинг.

Браузеры отправляют запросы на серверы и отображают веб-страницы, обрабатывая HTML, CSS и JavaScript.
Хостинг - это услуга предоставления места на сервере для размещения сайтов, обеспечивая их доступность и стабильную работу.