1. **Интернет**: Глобальная сеть компьютеров, соединенных между собой по всему миру, позволяющая обмен информацией и ресурсами.
2. **Служба Интернет**: В контексте Интернета, это программное обеспечение или сервис, предоставляемый через Интернет, который выполняет определенные функции или предоставляет доступ к определенным данным.
3. **Узел сети Интернет**: Один компьютер или другое устройство, подключенное к Интернету, способное отправлять и получать данные через сеть.
4. **Клиент-серверное приложение**: Тип приложения, в котором клиентское устройство (клиент) и сервер взаимодействуют через сеть. Клиент отправляет запросы к серверу, а сервер предоставляет ответы и обрабатывает запросы.
5. **Сетевой протокол**: Соглашение или набор правил, определяющий формат и последовательность данных, передаваемых между устройствами в сети.
6. **Основные свойства протокола HTTP**:

Простота взаимодействия.

Без состояния (stateless).

Основанный на тексте (text-based).

Использует методы запросов (GET, POST, etc.).

Поддерживает клиент-серверную архитектуру.

1. **Состав информации, пересылаемой в HTTP-запросе**: Метод (GET, POST, и т. д.), URI (Uniform Resource Identifier), версия протокола, заголовки и тело запроса.
2. **Состав информации, пересылаемой в HTTP-ответе**: Версия протокола, статусный код, статусное сообщение, заголовки и тело ответа.
3. **Группы заголовков HTTP**: Заголовки запроса и заголовки ответа. Заголовки могут быть общими, специфичными для запроса или ответа, и сущностными.
4. **Web-приложение**: Программное приложение, доступное через веб-браузер, которое позволяет пользователям взаимодействовать с данными и выполнять задачи через интернет.
5. **Frontend и Backend**: Frontend - часть web-приложения, отвечающая за визуальное представление и пользовательский интерфейс. Backend - часть, занимающаяся бизнес-логикой и обработкой данных на сервере.
6. **Кроссплатформенное приложение**: Приложение, которое может работать на разных операционных системах или платформах без необходимости переписывать код.
7. **Общая схема web-приложения**: Клиент (браузер) <-> Интернет <-> Сервер (Backend) <-> База данных. Клиент отправляет запросы на сервер, который обрабатывает их и возвращает ответы.
8. **Технологии разработки серверных кроссплатформенных приложений**: Node.js, Python (с использованием фреймворков), Ruby (с использованием Ruby on Rails), Java, PHP и другие.
9. **Асинхронная операция**: Операция, которая выполняется в фоновом режиме или без блокировки выполнения других операций. В Node.js асинхронность осуществляется с использованием колбэков или промисов
10. **Назначение NODE.JS**: Node.js - это среда выполнения JavaScript на стороне сервера, предназначенная для создания масштабируемых и высокопроизводительных приложений.
11. **Основные свойства NODE.JS**:

Асинхронное и событийное программирование.

Однопоточная, неблокирующая архитектура.

Поддержка платформы сервера.

Модульная система с поддержкой пакетов NPM.

Высокая производительность и эффективность ввода-вывода.