,Лаба 1

9. Интернет

всемирная сеть, построенная на стеке протоколов TCP/IP.

10. Служба Интернет

это программа, система, предоставляющая услуги клиентам. Сервер + протокол. Стандартные серверы, которые прослушивают стандартные порты (*от 0 до 1024*).

11. Узел сети Интернет

Узел сети Интернет — это любое устройство, подключенное к Интернету, которое может отправлять, получать или обрабатывать данные. Узлами могут быть компьютеры, серверы, маршрутизаторы и другие сетевые устройства.

12. Клиент-серверное приложение

Клиент-серверное приложение — это архитектурная модель, в которой один или несколько клиентов (например, браузеры) запрашивают услуги или ресурсы у сервера, который обрабатывает запросы и возвращает результаты.

13. Сетевой протокол

Сетевой протокол — это набор правил и стандартов, определяющих, как данные передаются и обрабатываются в сети. Протоколы описывают структуру сообщений, порядок их передачи и обработку ошибок.

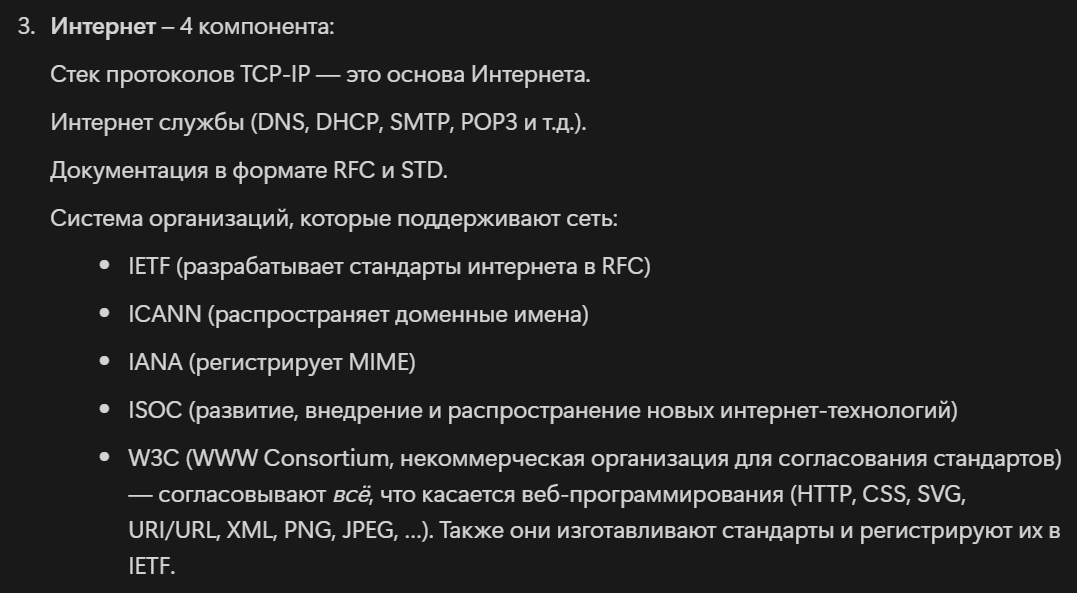
14. Основные свойства протокола HTTP

Безопасность: Поддержка шифрования (HTTPS).

Статeless: Каждый запрос независим от предыдущих.

Расширяемость: Возможность добавления новых методов и заголовков.

Кросс-платформенность: Поддержка различных платформ и устройств.



15. Состав информации, пересылаемой в HTTP-запросе

Метод (GET, POST и др.)

URI (Uniform Resource Identifier)

Версия протокола

Заголовки (информация о клиенте, типе данных и др.)

Тело запроса (в случае методов, передающих данные, например, POST)

16. Состав информации, пересылаемой в HTTP-ответе

Версия протокола

Код состояния (например, 200 OK, 404 Not Found)

Заголовки (информация о сервере, типе контента и др.)

Тело ответа (например, HTML-страница или JSON-данные)

17. Группы заголовков HTTP

Заголовки запроса (например, Accept, User-Agent)

Заголовки ответа (например, Content-Type, Server)

Заголовки сущности (например, Cache-Control)

18. Web-приложение

Web-приложение — это приложение, доступное через веб-браузер, которое использует веб-технологии для взаимодействия с пользователем и обработки данных на сервере.

19. Frontend и Backend

Frontend: Часть веб-приложения, с которой взаимодействует пользователь; включает в себя интерфейс, разработанный с использованием HTML, CSS и JavaScript.

Backend: Серверная часть веб-приложения, отвечающая за обработку данных, бизнес-логику и взаимодействие с базами данных.

20. Кроссплатформенное приложение

Кроссплатформенное приложение — это приложение, которое может работать на разных операционных системах и устройствах без необходимости модификации кода для каждой платформы.

21. Общая схема web-приложения

[ Пользователь ] <--> [ Frontend (браузер) ] <--> [ Backend (сервер) ] <--> [ База данных ]

Пользователь: Взаимодействует с веб-приложением через браузер.

Frontend: Отображает интерфейс и обрабатывает пользовательские запросы.

Backend: Обрабатывает запросы, выполняет бизнес-логику и взаимодействует с базой данных.

База данных: Хранит данные приложения.

22. Основные технологии разработки серверных кроссплатформенных приложений

Node.js

Java (Spring)

Python (Django, Flask)

Ruby (Ruby on Rails)

.NET (ASP.NET Core)

23. Асинхронная операция

Асинхронная операция — это операция, которая выполняется независимо от основного потока выполнения программы. Результат такой операции может быть получен позже, что позволяет программе продолжать выполнение без ожидания завершения операции.

24. Основное назначение Node.js

Node.js предназначен для создания высокопроизводительных, масштабируемых сетевых приложений, использующих асинхронную обработку событий, что позволяет обрабатывать множество соединений одновременно.

25. Основные свойства Node.js

Асинхронность: Поддержка неблокирующего ввода-вывода.

Однопоточная модель: Обработка событий в одном потоке.

Масштабируемость: Легкость в создании масштабируемых приложений.

Использование JavaScript: Позволяет использовать один язык как на клиенте, так и на сервере.