Лабораторная работа 01

ПвИ

**Задание 01**

1. Установите Node.js.
2. Проверьте работоспособность Node.js.

**Задание 02**

1. Разработайте серверное приложение **01-01**, использующее модуль http и возвращающее ответ с разметкой ***<h1>Hello World</h1>*.**
2. Проверьте работоспособность приложения с помощью браузера.
3. Проверьте работоспособность приложения с помощью приложения ***POSTMAN***.

**Задание 03**

1. Разработайте серверное приложение **01-02**, на основе разработанного в задании 2, которое в ответе сервера пересылает html-страницу с содержимым запроса (метод, uri, версия протокола, заголовки, тело).
2. Выполните GET-запрос к серверу с помощью браузера и убедитесь в работоспособности сервера.
3. Выполните POST-запрос (содержащий данные в теле сообщения) к серверу с помощью POSTMAN и убедитесь в работоспособности сервера.

**Задание 04.** Ответьте на следующие вопросы.

1. **Дайте определение понятию «Интернет».**

Интернет - всемирная система объединённых компьютерных сетей для хранения и передачи информации.

1. **Дайте определение понятию «Служба Интернет».**

Служба Интернета — это программное обеспечение или сервис, предоставляемый через Интернет для выполнения определенных функций или обеспечения определенных возможностей. Примерами могут служить электронная почта, поисковые системы, социальные сети и др.

1. **Дайте определение понятию «Узел сети Интернет».**

Узел сети Интернет - это физический или логический элемент, подключенный к Интернету, способный обмениваться данными с другими узлами в сети. Это может быть компьютер, сервер, маршрутизатор, мобильное устройство и так далее.

1. **Дайте определение понятию «клиент-серверное приложение».**

Клиент-серверное приложение - это приложение, которое разделяет работу между клиентской и серверной частями. Клиентская часть обычно отвечает за пользовательский интерфейс и взаимодействие с пользователем, а серверная часть обрабатывает бизнес-логику, хранит данные и предоставляет их клиентской части.

1. **Дайте определение понятию «сетевой протокол».**

Сетевой протокол - это набор правил и соглашений, которые определяют способы и форматы обмена данными между устройствами в компьютерных сетях. Протоколы определяют, как данные упаковываются, отправляются, передаются и принимаются.

1. **Перечислите основные свойства протокола HTTP.**

Основные свойства протокола HTTP (Hypertext Transfer Protocol) включают:

Безсостоятельность: Каждый запрос и ответ независимы друг от друга.

Состояние: Протокол HTTP сам по себе не сохраняет состояние между запросами, но это может быть обеспечено с помощью механизмов, таких как куки или сеансы.

Текстовая основа: HTTP использует текстовые заголовки и данные, что делает его читаемым и отладочно-понятным.

1. **Перечислите состав информации, пересылаемой в HTTP-запросе.**

В HTTP-запросе содержится:

Метод запроса (GET, POST, PUT, DELETE и другие).

URL ресурса.

Заголовки, содержащие метаданные запроса.

Тело запроса (необязательно, например, для POST-запросов).

1. **Перечислите состав информации, пересылаемой в HTTP-ответе.**

В HTTP-ответе содержится:

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

Заголовки, содержащие метаданные ответа.

Тело ответа, содержащее запрашиваемые данные или ресурсы.

1. **Перечислите группы заголовков HTTP.**

Группы заголовков HTTP включают:

Заголовки запроса: Отправляемые клиентом при отправке запроса.

Заголовки ответа: Возвращаемые сервером в ответ на запрос.

Общие заголовки: Применяются как к запросам, так и к ответам и имеют общий характер.

Заголовки сущности: Относятся к данным в теле запроса или ответа.

1. **Дайте определение понятию «web-приложение».**

Web-приложение - это программное приложение, которое доступно через веб-браузер и обычно выполняется на удаленном сервере. Оно позволяет пользователям взаимодействовать с данными и выполнять различные задачи через интерфейс, предоставляемый браузером.

1. **Дайте определение понятиям «frontend» и «backend».**

Frontend - это клиентская часть web-приложения, которая отвечает за пользовательский интерфейс и взаимодействие с пользователем. Backend - это серверная часть приложения, которая обрабатывает запросы от клиента, управляет базой данных и выполняет бизнес-логику.

1. **Дайте определение понятию «кроссплатформенное приложение».**

Кроссплатформенное приложение - это приложение, которое может быть запущено на разных операционных системах или платформах без необходимости переписывать его с нуля. Обычно такие приложения разрабатываются с использованием фреймворков или инструментов, способных обеспечить совместимость на разных платформах.

1. **Изобразите и поясните общую схему web-приложения.**

Общая схема web-приложения включает клиентскую часть (frontend), взаимодействующую с пользователем через браузер, и серверную часть (backend), которая обрабатывает запросы клиента, взаимодействует с базой данных и возвращает данные клиенту. Клиент и сервер обмениваются данными с использованием HTTP или других сетевых протоколов.

1. **Назовите основные технологии разработки серверных кроссплатформенных приложений.**

Основные технологии разработки серверных кроссплатформенных приложений включают Node.js, Ruby on Rails, Django (Python), Express.js (для Node.js), ASP.NET (C#), и другие.

1. **Поясните понятие «асинхронная операция».**

Асинхронная операция - это операция, которая выполняется независимо от основного потока выполнения программы. Вместо блокирования выполнения программы до завершения операции, асинхронные операции позволяют продолжить выполнение других задач

1. **Поясните основное назначение NODE.JS.**

Node.js - это среда выполнения JavaScript, которая позволяет запускать JavaScript на сервере, в отличие от браузера, где JavaScript обычно выполняется.

1. **Перечислите основные свойства NODE.JS.**

Основные свойства Node.js:

a. Асинхронность: Node.js построен на асинхронной модели выполнения, что позволяет обрабатывать множество запросов без блокировки исполнения других операций. Это особенно полезно для обработки большого количества одновременных подключений.

b. Однопоточность: Node.js работает в одном потоке, что обеспечивает легковесность и высокую производительность для обработки сетевых запросов.

c. Использование JavaScript: Node.js позволяет использовать один и тот же язык программирования (JavaScript) как на сервере, так и на клиенте, что упрощает совместимость и разработку.

d. Модульность: Node.js поддерживает модульность через систему модулей CommonJS, что позволяет разрабатывать и использовать множество независимых модулей для упрощения разработки.

e. Коммуникация по событиям: Node.js использует событийную модель, позволяя создавать обратные вызовы и реагировать на события, такие как запросы HTTP, без блокировки.

f. Обширная библиотека: Node.js имеет широкую экосистему библиотек и модулей, что делает его подходящим для разнообразных типов приложений и задач.

g. Высокая производительность: Благодаря асинхронной природе и эффективному механизму ввода/вывода (I/O), Node.js обеспечивает высокую производительность для серверных приложений.