ПСКП

Лабораторная работа №1

1. Дайте определение понятию “Интернет”

   Информационно-коммуникационная сеть, объединяющая компьютеры для хранения и передачи информации. (Сеть на основе TCP/IP)

2. Дайте определение понятию “Служба Интернет”

   Подсистема, предоставляющая услуги пользователям Интернета.

3. Дайте определение понятию “Узел сети Интернет”

   Устройство, имеющее IP-адрес и подключенное к сети Интернет.

4. Дайте определение понятию “клиент-серверное приложение”

   Приложение (программа), состоящее из двух компонент — клиента и сервера. Клиент и сервер взаимодействуют между собой в соответствии с заданными правилами (протоколами). Инициатором соединения всегда является клиент.

5. Дайте определение понятию “сетевой протокол”

   Набор определённых правил или соглашений, который определяет передачу данных между программами.

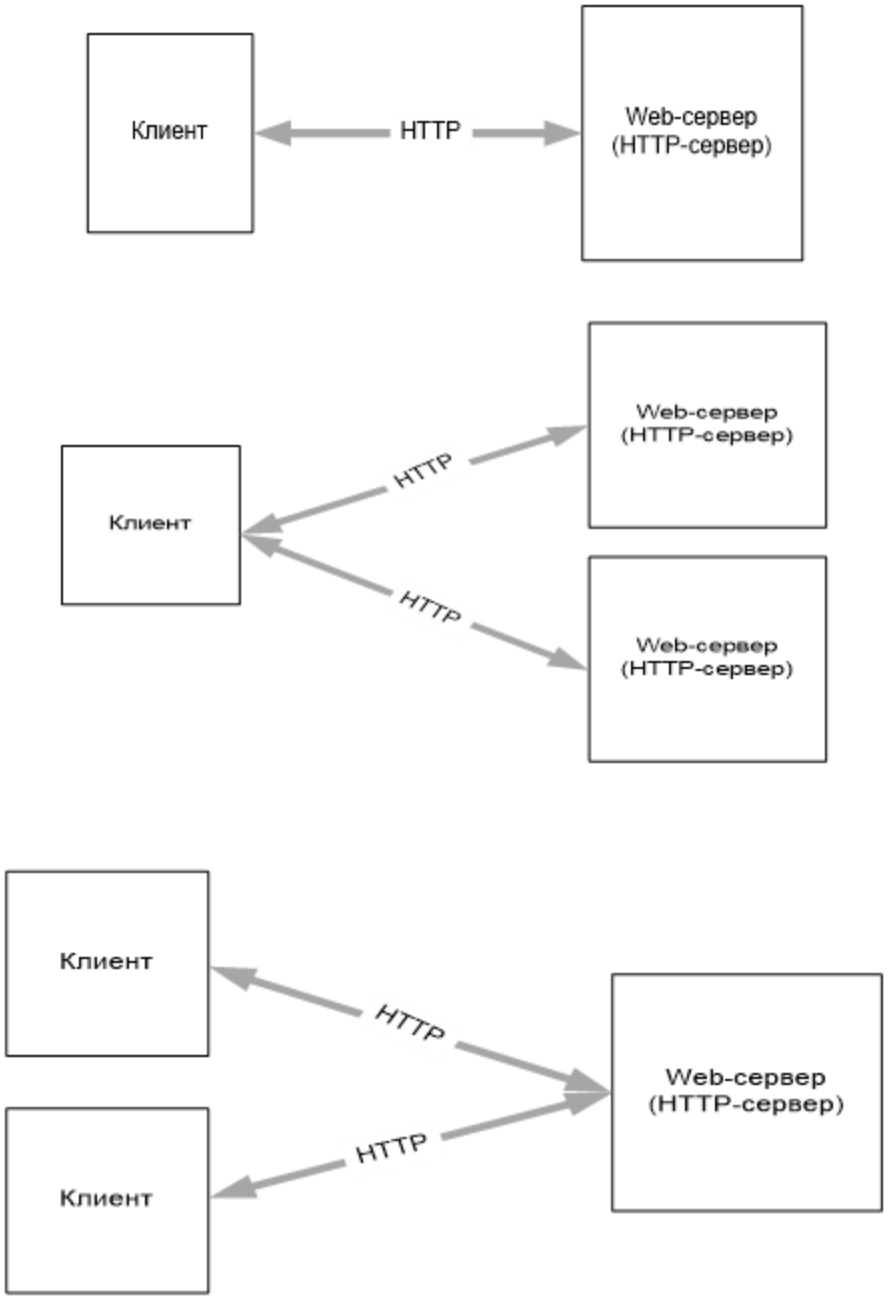
6. Перечислите основные свойства протокола HTTP
    a. разные версии;
    b. два типа абонентов: клиент и сервер;
    c. два типа сообщений: request и response;
    d. на один request всегда один response; (обратное тоже верно)
    e. для адресации используется URI или URL.
7. Перечислите состав информации, пересылаемой в HTTP-запросе
    a. метод;
    b. URI;
    c. версия протокола;
    d. заголовки;
    e. параметры;
    f. расширение.
8. Перечислите состав информации, пересылаемой в HTTP-ответе
    a. версия протокола;
    b. код состояния;
    c. пояснение к коду состояния;
    d. заголовки;
    e. расширение.
9. Дайте определение понятию “web-приложение”

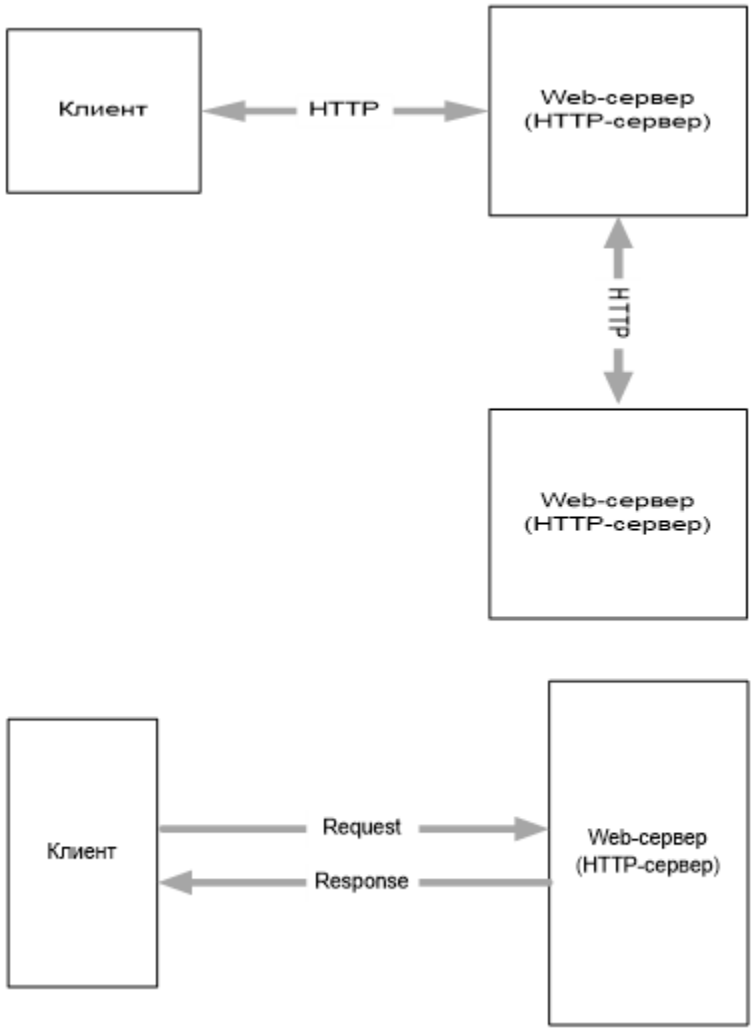
    Клиент-серверное приложение, у которого клиент и сервер взаимодействуют по протоколу HTTP.
10. Дайте определение понятию “frontend” и ”backend”

    Клиентская и серверная часть приложения.
11. Дайте определение понятию “кроссплатформенное приложение”

Приложение, способное работать на более чем одной программно-аппаратной платформе.

12. Изобразите и поясните общую схему web-приложения





13. Назовите основные технологии разработки серверных кроссплатформенных приложений

    Java, PHP, JS/NodeJS/NestJS, C++, Go, Ruby и так далее.

14. Поясните понятие “асинхронная операция”

    Операция, в которой выполнение осуществляется в 2 фазы: заявка на исполнение и получение результата.

15. Поясните принцип выполнения асинхронного запроса с помощью объекта XMLHTTPRequest и Fetch

    Оба способа позволяют делать запрос, но fetch возвращает Promise, что позволяет избавиться от большого количества коллбэков.

16. Поясните основного назначение сервера NODE.JS

    Позволяет клиентам устанавливать соединение с сервером, который запущен на текущем устройстве.

17. Перечислите основные свойства NODE.JS
    a. Оптимальная скорость работы приложение;
    b. Кроссплатформенность;
    c. Большое количество бесплатных инструментов
    d. Открытый исходный код
    e. Основан на Chrome V8
    f. Среда (контейнер) исполнения приложений на JavaScript
    g. Высокая масштабируемость
    h. Скорость
    i. Асинхронный
    j. Однопоточный

Лабораторная работа №2

1. Перечислите основные свойства сервера NODE.JS
    a. Основан на Chrome V8
    b. Среда (контейнер) исполнения приложений на JavaScript
    c. Высокая масштабируемость
    d. Скорость
    e. Асинхронный
    f. Однопоточный
2. Что такое npm?

    Node Package Manager — менеджер пакетов, входящий в состав Node.js.

3. Поясните назначение HTTP-заголовка Content-Type.

Content-Type сообщает тип передаваемого контента.

4. Поясните назначение функции require.

   Загрузка модуля в проект (импорт, подключение).

5. Поясните понятие «Модуль Node.js».

   Модуль — текстовый файл, содержащий код на языке JS.

6. Поясните понятие «Node.js built-in modules».

   В Node.js каждый файл js представляет собой модуль. Это сделано для того, чтобы структурировать и делать независимые части.

7. Какой модуль NODE.JS обеспечивает работу с протоколом HTTP?

   http

8. Какой модуль NODE.JS обеспечивает работу с файловой системой?

   fs

Лабораторная работа №3

1. Перечислите основные свойства глобальные объекты и поясните их предназначение

   Глобальные объекты доступны во всех модулях, можем использовать без включения. *В Node.js объекты — модули, функции, строки и, собственно, объекты.*

   - __dirname
   - __filename
   - global
   - process
   - require

2. Поясните понятие «асинхронная функция».

   Асинхронная функция, которая выполняется в неблокирующем режиме (не блокирует основной поток приложения).

3. Поясните понятие стандартные «системные потоки».
    1. stdin — поток ввода;
    2. stdout — поток вывода;
    3. stderr — поток ошибок.
4. Поясните назначение функций `process.nextTick()`, `setImmediate()`, поясните в чем разница.

    Понимание метода `process.nextTick()` — всякий раз, когда инициализируется новая очередь операций, мы можем думать об этом как о новом тике. Метод `process.nextTick()` добавляет функцию обратного вызова в начало следующей очереди событий. Следует отметить, что в начале программы метод `process.nextTick()` вызывается в первый раз перед обработкой цикла событий.

    Понимание метода `setImmdeiate()` — всякий раз, когда мы вызываем метод `setImmediate()`, его функция обратного вызова помещается в фазу проверки следующей очереди событий. Здесь следует отметить небольшую деталь: метод `setImmediate()` вызывается на этапе опроса, а его функции обратного вызова вызываются на этапе проверки.

process.nextTick()	setImmediate()
`process.nextTick()` используется для планирования функции обратного вызова, которая будет вызываться на следующей итерации Event Loop.	Метод `setImmediate()` используется для выполнения функции сразу после завершения текущего цикла событий.
Его преимущество в том, что у него нет времени на обратный вызов.	Метод `setImmediate()` обрабатывается только на этапе обработчика проверки цикла событий (event loop).
Функция `process.nextTick()` определен в Node.js Event Loop.	Обычно он находится в модуле «Timers».
Синтаксис: `process.nextTick(callback)`;	Синтаксис: `setImmediate(callback)`;
Преимущество этого заключается в том, что он может вызвать голодание ввода-вывода. (It has a benefit that it has the potential to cause I/O starvation)	Его возвращаемое значение является уникальным идентификатором таймера, который можно использовать в другом вызове функции.

Лабораторная работа №4

1. Дайте пояснению понятию «событие программного объекта».

   Событие программного объекта — это процесс перехода объекта из одного состояние в другое.

2. Объясните механизм генерации и обработки событий в C#.

   Создается делегат, объявляется событие ключевым словом event типа делегата, добавление обработчика +=, вызов события: событие() или событие?.Invoke

3. Поясните как самостоятельно реализовать механизм генерации и обработки событий на JS или C++.

   js: new Event() или new CustomEvent()

   `addEventListener('name', callback)` — подписка на событие.

   `dispatchEvent()` — вызов события.

   c++: Артёма спросите

4. Какой встроенный механизм используется в Node.js для генерации и обработки событий. Поясните принцип его работы.

   Модуль ‘event’ класс eventEmitter.

Лабораторная работа №5

1. `setInterval` — принимает параметром функцию, которая будет выполняться бесконечное количество раз с заданным интервалом в миллисекундах, сам интервал передается вторым параметром.
2. `setTimeout` — возвращает объект Timeout , который можно использовать в качестве ссылки на тайм-аут, который был установлен.
3. `ref` — события**,** генерируемые таймером будут учитываться при завершении работы Node.js.
4. `unref` — события**,** генерируемые таймером не будут учитываться при завершении работы Node.js.

Лабораторная работа №7

1. Поясните аббревиатуру MIME, какой организаций поддерживается допустимые MIME, в каких компонентах запросов и ответов используется MIME.

   MIME — **Multipurpose Internet Mail Extensions (многоцелевое расширения интернет-почты) -** стандарт, описывающий передачу различных типов данных по электронной почте, а также, в общем случае, спецификация для кодирования информации и форматирования сообщений таким образом, чтобы их можно было пересылать по Интернету. **Медиа тип** (так же известный как **Multipurpose Internet Mail Extensions**
   или **MIME тип)** является стандартом, который описывает природу и формат документа, файла или набора байтов. Организация: **World Mime Organisation**.

2. Перечислите теги HTML, интерпретация которых приводит к HTTP-запросам.

   frame, audio, a, form, video, script, link

3. Перечислите способы выполнения HTTP-запросов из JS-сценария.

   https://medium.com/nuances-of-programming/сравниваем-различные-способы-выполнения-http-запросов-в-javascript-a0c80b89d690

   fetch, XMLHttpRequest, httpclient…

4. Поясните понятие «параметризованный модуль».

   Параметризованный модуль - модуль с параметрами

Лабораторная работа №8

1. Поясните назначение заголовка **ContentType.**

   Заголовок-сущность **Content-Type** используется для того, чтобы определить MIME тип ресурса.

   В ответах сервера заголовок Content-Type сообщает клиенту, какой будет тип передаваемого контента. В некоторых случаях браузеры пытаются сами определить MIME тип передаваемого контента, но их реакция может быть неадекватной. Чтобы предотвратить такие ситуации, Вы можете установить в заголовке X-Content-Type-Options значение nosniff.

2. Поясните назначение заголовка **Accept.**

HTTP заголовок запроса Accept указывает, какие типы контента, выраженные как MIME типы, клиент может понять. Используя согласование контента, сервер затем выбирает одно из предложений, использует его и информирует клиента о своем выборе с помощью заголовка ответа Content-Type.

3. Для чего используется значение **Multipart/formdata** заголовка **ContentType.**

   Тип содержимого multipart/form-data — это составной тип содержимого, чаще всего использующийся для отправки HTML-форм с бинарными (не-ASCII) данными методом POST протокола HTTP. Указывается в поле заголовка Content-Type (тип содержимого) и следует правилам для составных MIME-данных в соответствии с RFC 2045. Для форм, не имеющих больших бинарных (не-ASCII) данных, может использоваться тип содержимого application/x-www-form-urlencoded.

4. Как с помощью тега **form**, обеспечить значение **Multipart/formdata** заголовка **ContentType.**

   ```
   <form action="login.php" method="post"
       enctype="multipart/form-data">
   </form>
   ```

5. Какое значение заголовка **ContentType** отправляется тегом **form** в запросе по умолчанию.

   application/x-www-form-urlencoded – по умолчанию

6. Где и в каком формате передаются параметры в GET-запросе?

   В Query Params в виде ключ-значение.

   Не существует конкретной максимальной величины GET-запроса.

   Фактически в одном таком запросе должно быть не больше 5 параметров, иначе каждый из них будет сложно контролировать со стороны сервера и браузера. Если нужно передать большое количество информации, рекомендуется использовать метод POST.

   - Могут кэшироваться
   - Остаются в истории браузера
   - Могут быть/стать "закладкой"

- Не должны использоваться для передачи паролей и всего такого
- Имеют ограничение по длине (URL и в некоторых браузерах свои заморочки)

7. Где и в каком формате передаются параметры в POST-запросе?

   Значения отправляются в тело запроса в формате, указанном типом содержимого.

   - Никогда не кэшируются
   - Не остаются в истории браузера
   - Не могут быть/стать "закладкой"
   - Не имеют таких ограничений по длине (обычно в браузерах и на web серверах есть ограничение по умолчанию)

8. Поясните понятие **JSON**?

   JSON (JavaScript Object Notation) - простой формат обмена данными, удобный для чтения и написания как человеком, так и компьютером. Он основан на подмножестве языка программирования JavaScript. JSON - текстовый формат, полностью независимый от языка реализации.

   JSON основан на двух структурах данных:

   - Коллекция пар ключ/значение. В разных языках, эта концепция реализована как объект, запись, структура, словарь, хэш, именованный список или ассоциативный массив.
   - Упорядоченный список значений. В большинстве языков это реализовано как массив, вектор, список или последовательность.

9. Поясните понятие **XML**?

   XML - это расширяемый язык разметки (Extensible Markup Language), разработанный специально для размещения информации в World Wide Web, наряду с HTML, который давно стал стандартным языком создания Web-страниц. В отличие от HTML, вместо использования ограниченного набора определённых элементов вы имеете возможность создавать ваши собственные элементы и присваивать им любые имена по вашему выбору.

   XML решает ряд проблем, которые не решает HTML, например:

- Представление документов любого (не только текстового) типа, например, музыки, математических уравнений и т.д.
- Сортировка, фильтрация и поиск информации.
- Представление информации в структурированном (иерархическом) виде.

Лабораторная работа №9

1. Чем отличается передача параметров в GET и POST запросах.

   Запрос GET передает данные в URL в виде пар "имя-значение" (другими словами, через ссылку), а **запрос POST передает данные в теле запроса**. Это различие определяет свойства методов и ситуации, подходящие для использования того или иного HTTP метода.

   https://guruweba.com/html/metody-get-i-post-ispolzovanie-i-otlichiya/#:~:text=Основное отличие метода GET от,подробно показано в примерах ниже).

2. Поясните структуру http-запроса с вложенным файлом (multipart/form-data).

   Строка статуса, Заголовок, Тело(файл).

   Первая строка в HTTP ответе – это строка состояния, иначе Status-Line. Она состоит из версии протокола HTTP, числового кода состояния HTTP сервера и поясняющей фразы. Окончание строки состояния в HTTP ответе является символ CRLF.

   Поля заголовка HTTP ответа необходимы серверу для того, чтобы передать дополнительную информацию клиенту, которая не может быть помещена в строку состояния. Поля заголовка HTTP ответа помогают клиенту правильно обработать HTTP сообщение сервера. Так же поля заголовка HTTP ответа могут содержать дополнительную информацию о сервере и о дальнейшем доступе к ресурсу, указанному в URI (URI в HTTP).

   Тело HTTP сообщения необязательная составляющая HTTP сообщения. Тело HTTP сообщения используется для передачи тела объекта запроса или тела объекта ответа. Между телом объекта и телом HTTP сообщения есть разница только в том случае, когда используется кодирование передачи, кодирование передачи указывается специальным полем Transfer-Encoding. Данное поле

является частью HTTP сообщения, а не передаваемого объекта, поэтому его может удалить любая из общающихся сторон. Присутствие тела HTTP сообщения обозначается заголовками Content-Length и Transfer-Encoding.

Лабораторная работа №10

1. Поясните разницу между полудуплексным и дуплексным каналами данными.

   Реализующее **дуплексный** способ связи устройство может в любой момент времени и передавать, и принимать информацию. Передача и приём ведутся устройством одновременно по двум физически разделённым каналам связи Пример дуплексной связи — разговор двух людей (корреспондентов) по городскому телефону: каждый из говорящих в один момент времени может и говорить, и слушать своего корреспондента.

   Реализующее **полудуплексный** (англ. half-duplex) способ связи устройство в один момент времени может либо передавать, либо принимать информацию. Пример полудуплексной связи — разговор по рации.

2. Какой тип канала применяется HTTP-протоколом.

   Полудуплексный.

3. Поясните понятие **WebSocket**.

   WebSocket — протокол связи поверх TCP-соединения, предназначенный для обмена сообщениями между браузером и веб-сервером в режиме реального времени.

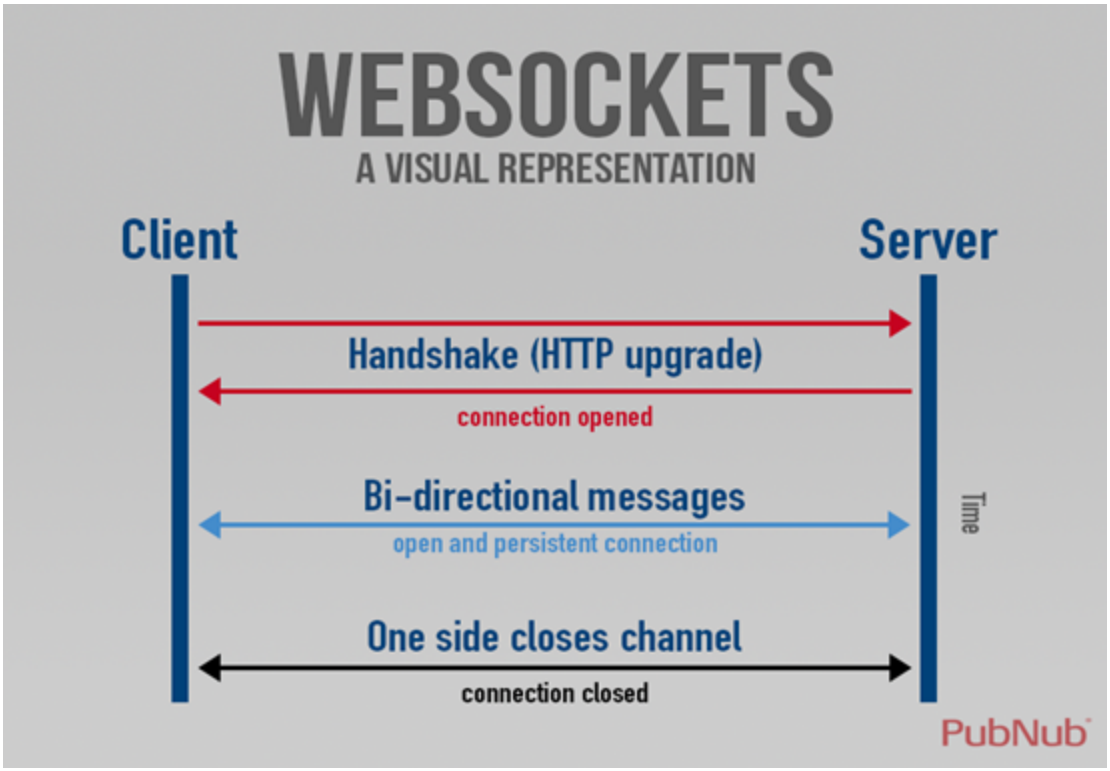
   HTTP и WebSocket — это очень разные протоколы, в которых используются различные подходы к обмену данными. HTTP основан на модели «запрос — ответ»: сервер отправляет клиенту некие данные после того, как они будут запрошены. В случае с WebSocket всё устроено иначе. А именно:

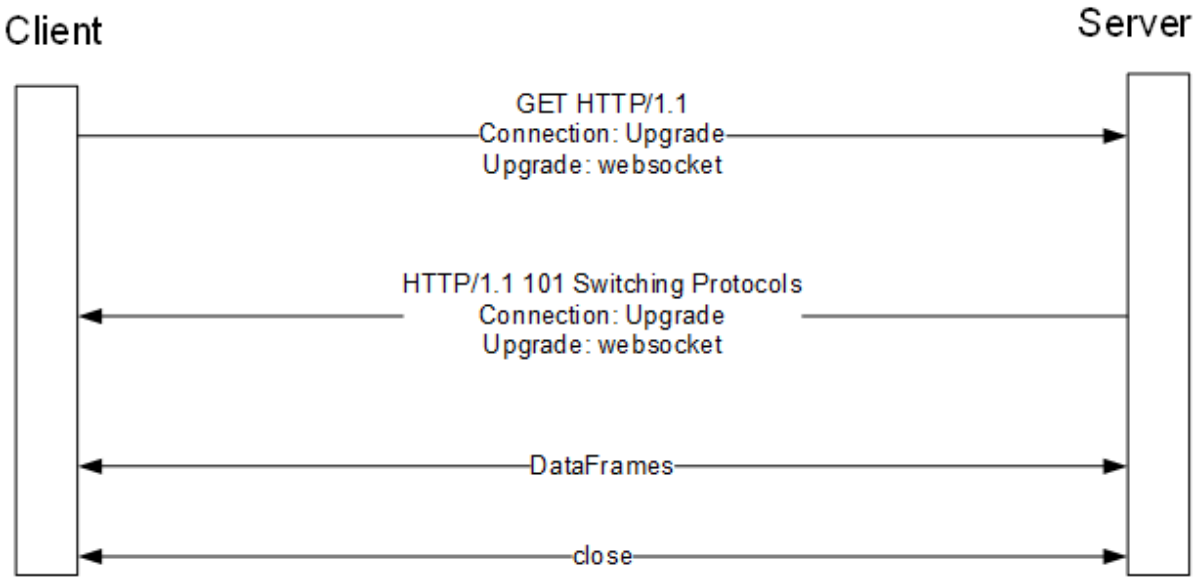
   - Сервер может отправлять сообщения клиенту по своей инициативе, не дожидаясь поступления запроса от клиента.
   - Клиент и сервер могут обмениваться данными одновременно.
   - При передаче сообщения используется крайне малый объём служебных данных. Это, в частности, ведёт к низким задержкам при передаче данных.

4. Какой тип канала применяется WebSocket-протоколом.

   Дуплексный.

5. Расскажите процедуру WebSocket-рукопожатия.





6. Поясните понятия «широковещательное сообщение» (broadcast), «широковещательный сервер».

   Широковещательный сообщения используются для отправки пакетов всем узлам в сети.

   Широковещательный адрес — условный адрес, который используется для передачи широковещательных пакетов в компьютерных сетях.

Лабораторная работа №11

1. Поясните понятие «TCP-порт».

   TCP-порт — для обмена пакетами между приложениями

2. Поясните понятие «сетевой сокет».

   Совокупность IP-адреса и номера порта называется сокетом. Сокет однозначно идентифицирует прикладной процесс в сети TCP/IP

3. Поясните понятие «WebSocket».

   WebSocket представляет собой альтернативу HTTP, его можно применять для организации обмена данными в веб-приложениях. Этот протокол позволяет создавать долгоживущие двунаправленные каналы связи между клиентом и сервером. После установления соединения канал связи остаётся открытым, что даёт в распоряжение приложения очень быстрое соединение, характеризующееся низкими задержками и небольшой дополнительной нагрузкой на систему.

   Протокол WebSocket поддерживают все современные браузеры.

   HTTP и WebSocket — это очень разные протоколы, в которых используются различные подходы к обмену данными. HTTP основан на модели «запрос — ответ»: сервер отправляет клиенту некие данные после того, как они будут запрошены. В случае с WebSocket всё устроено иначе. А именно:

   - Сервер может отправлять сообщения клиенту по своей инициативе, не дожидаясь поступления запроса от клиента.
   - Клиент и сервер могут обмениваться данными одновременно.

- При передаче сообщения используется крайне малый объём служебных данных. Это, в частности, ведёт к низким задержкам при передаче данных.

Протокол WebSocket очень хорошо подходит для организации связи в режиме реального времени по каналам, которые долго остаются открытыми. HTTP, в свою очередь, отлично подходит для организации эпизодических сеансов связи, инициируемых клиентом. В то же время надо отметить, что, с точки зрения программирования, реализовать обмен данными по протоколу HTTP гораздо проще, чем по протоколу WebSocket.

4. Поясните процедуру установки соединения между WS-сервером и WS-клиентом.

   Server: new WebSocket.Server({port: 5000, host: HOST, path: '/download'});

   Client: new WebSocket('ws:/localhost:5000/download');

5. Поясните понятие «широковещательное сообщение».

   Широковещательные сообщения - тип сообщений, не имеющих адресата. Рассылаются на все устройства, находящиеся в сети.

6. Поясните принцип организации потокового ввода/вывода через WS-соединение.

   Server: duplex = WebSocket.createWebSocketStream(ws, {encoding: 'utf8'});

   uf = fs.createWriteStream(__dirname +`/files/${k++}.txt`);

   duplex.pipe(uf);

   Client: duplex = WebSocket.createWebSocketStream(ws, {encoding: 'utf8'});

   uf = fs.createReadStream(__dirname + `/files/file.txt`);

   uf.pipe(duplex);

7. Поясните принцип действия и назначение механизма «ping/pong».

   В протокол встроена проверка связи при помощи управляющих фреймов типа PING и PONG.

   Тот, кто хочет проверить соединение, отправляет фрейм PING с произвольным телом. Его получатель должен в разумное время ответить фреймом PONG с тем же телом.

Эта функциональность встроена в браузерную реализацию, так что браузер ответит на PING сервера, но управлять ей из JavaScript нельзя.

Иначе говоря, сервер всегда знает, жив ли посетитель или у него проблема с сетью.

8. Поясните аббревиатуру «RPC».

   Remote Procedure Call — вызов удалённых процедур.

   Приложение предоставляет доступ к части своей функциональности посредством удаленного вызова процедуры. Взаимодействие между приложениями осуществляется синхронно в режиме реального времени.

9. Поясните принцип работы RPC-механизма, предоставляемого пакетом rpc-websockets.

   Для работы RPC-протокола необходимо выполнение следующих условий:

   1. Уникальная идентификации всех удаленно вызываемых процедур на данном хосте. RPC-запросы содержат три поля идентификаторов - номер удаленной программы (сервиса), номер версии удаленной программы и номер удаленной процедуры указанной программы. Номер программы назначается производителем сервиса, номер процедуры указывает на конкретную функцию данного сервиса
   2. Идентификация версии RPC-протокола. RPC-сообщения содержат поле версии RPC-протокола. Она используется для согласования форматов передаваемых параметров при работе клиента с различными версиями RPC.
   3. Предоставление механизмов аутентификации клиента на сервере. RPC-протокол обеспечивает процедуру аутентификации клиента в сервисе, и, в случае необходимости, при каждом запросе или отправке ответа клиенту. Кроме того, RPC позволяет использовать различные дополнительные механизмы безопасности.

   RPC может использовать четыре типа механизмов аутентификации:

   - AUTH_NULL - без использования аутентификации
   - AUTH_UNIX - аутентификация по стандарту UNIX

- AUTH_SHORT - аутентификация по стандарту UNIX с собственной структурой кодирования
- AUTH_DES - аутентификация по стандарту DES

1. Идентификация сообщений ответа на соответствующие запросы. Ответные сообщения RPC содержат идентификатор запроса, на основании которого они были построены. Этот идентификатор можно назвать идентификатором транзакции вызова RPC. Данный механизм особенно необходим при работе в асинхронном режиме и при выполнении последовательности из нескольких RPC-вызовов.
2. Идентификация ошибок работы протокола. Все сетевые или серверные ошибки имеют уникальные идентификаторы, по которым каждый из участников соединения может определить причину сбоя в работе.

10. Поясните принцип действия и назначение механизма «subscriber/publisher».

    Издатель отправляет сообщение в топик (тема) откуда оно рассылается всем подписчикам

11. Поясните принцип действия и назначение механизма уведомлений.

    Создание-подписка-отправка-уведомление

Лабораторная работа №12

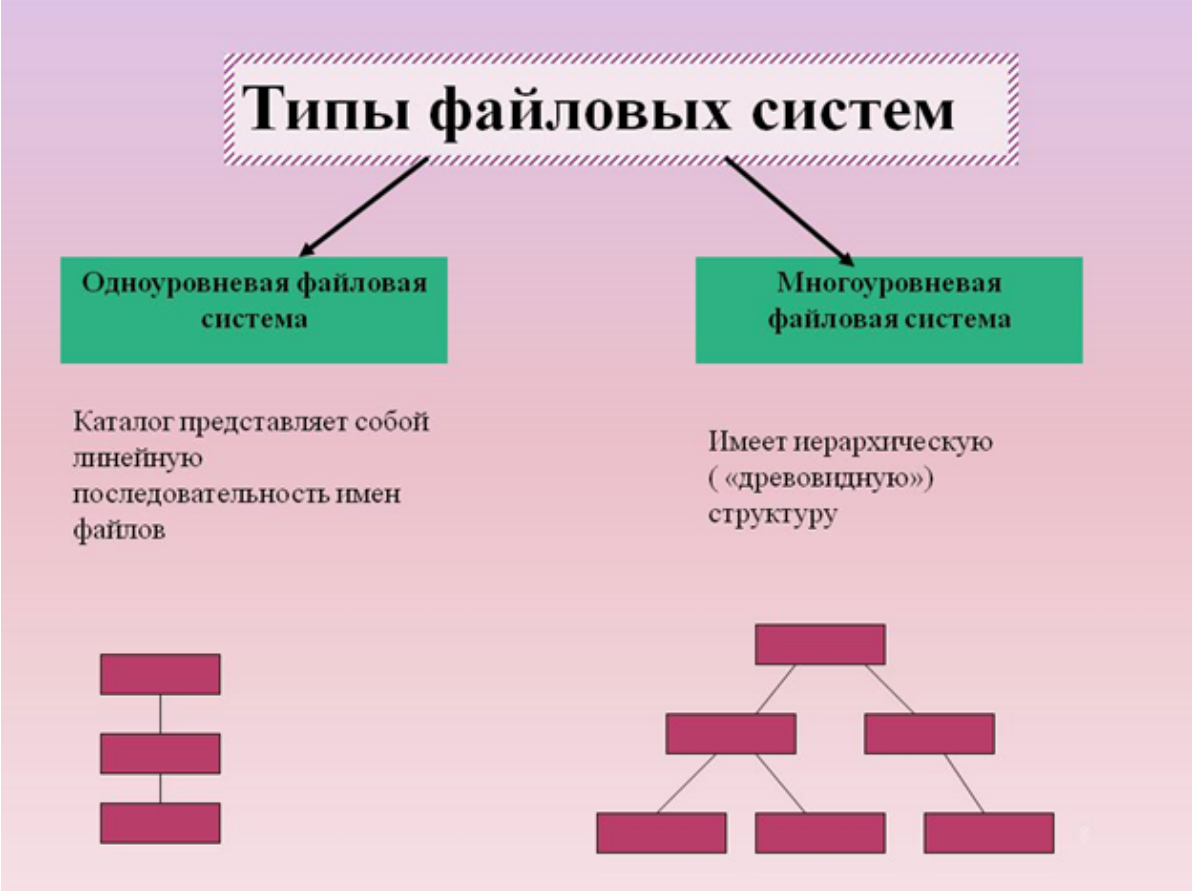
1. Поясните понятие «файл».

   Файл (англ. file) — именованная область данных на носителе информации.

2. Поясните понятие «файловая система».

   Фа́йловая систе́ма (англ. file system) — порядок, определяющий способ организации, хранения и именования данных на носителях информации в компьютерах, а также в другом электронном оборудовании

3. Перечислите типы файловых систем для различных ОС.



4. Поясните понятие «поток данных».

   Поток данных (англ. stream) в программировании — абстракция, используемая для чтения или записи файлов, сокетов и т. п. в единой манере. Потоки являются удобным унифицированным программным интерфейсом для чтения или записи файлов, сокетов и передачи данных между процессами

5. Поясните понятие «системные потоки данных».

   Пото́к выполне́ния (тред; от англ. *thread* — нить) — наименьшая единица обработки, исполнение которой может быть назначено ядром операционной системы.

6. Перечислите типы потоков данных, поддерживаемых Node.js, и поясните их назначение.

   - Readable — поток, который предоставляет данные на чтение;
   - Writable — поток, в который данные можно записывать;

- Duplex — поток, из которого можно как читать данные (Readable), так и записывать в него (Writable), при этом процесс чтения и записи происходит независимо друг от друга;
- Transform — разновидность Duplex потоков, которые могут изменять данные при их записи и чтении в/из потока (чаще используется как промежуточное звено в цепочке передачи данных).

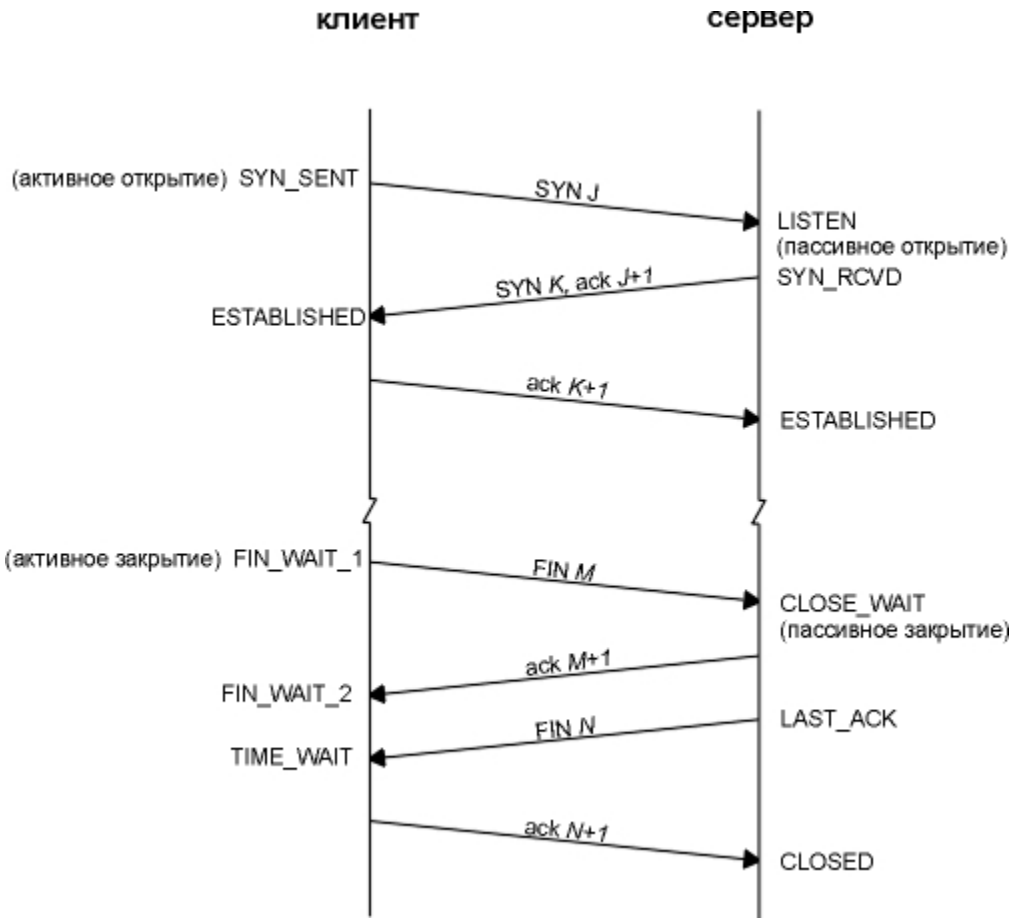
Лабораторная работа №13

1. Поясните основные свойства протокола TCP.
    a. **Надёжность доставки данных**. Получатель подтверждает получение каждого пакета данных. Для этого каждый октет нумеруется. Если отправитель не получает подтверждения, то он отправляет данные повторно.
    b. **Управление потоком данных**. Получатель регулирует поток поступающих данных. Это достигается отправкой фрейма вместе с каждым подтверждением. Окно определяет кол-во данных, которое получатель готов принять. Отправитель не высылает данных больше, чем допускается окном. Также протокол TCP содержит ф-ию проталкивания.
    c. **Разделение каналов**. Для того чтобы множество приложений могли использовать возможности TCP, используется механизм сокетов.
    d. **Работа с соединениями.**До того, как начать обмен данными, стороны устанавливают соединения, при этом в памяти каждого хоста создаётся структура – блок управления передачей, в котором хранятся сокеты сторон, участвующих в соединении, адреса буферов, размеры окон, последовательные номера, различные флаги и некоторая служебная инф-ия. Весь этот набор данных и образует соединение. Каждое соединение уникальным образом идентифицируется парой сокетов.
    e. **Двунаправленный обмен д-ми**. Приложение передает д-е в виде непрерывного потока актетов. Модуль TCP самостоятельно осущ-т сегментацию и буферизацию передаваемых д-х. В случае необход-ти избегать буферизации, возможность использ-я ф-ции проталкивания.
2. Поясните процедуры установки и закрытия TCP-соединения.

Чтобы установить TCP соединение, выполняются следующие шаги:

1. Запрашивающая сторона (которая, как правило, называется клиент) отправляет SYN сегмент, указывая номер порта сервера, к которому клиент хочет подсоединиться, и исходный номер последовательности клиента. Это сегмент номер 1.
2. Сервер отвечает своим сегментом SYN (установить соединение между компьютерами), содержащим исходный номер последовательности сервера (сегмент 2). Сервер также подтверждает приход SYN клиента с использованием ACK (ISN клиента плюс один). На SYN используется один номер последовательности.
3. Клиент должен подтвердить приход SYN от сервера с использованием ACK (ISN сервера плюс один, сегмент 3).

Этих трех сегментов достаточно для установления соединения. Часто это называется тройным рукопожатием (three-way handshake).



Для того чтобы установить соединение, необходимо 3 сегмента, а для того чтобы разорвать - 4. Это объясняется тем, что TCP соединение может быть в наполовину закрытом состоянии. Так как TCP соединение полнодуплексное (данные могут передвигаться в каждом направлении независимо от другого направления), каждое направление должно быть закрыто независимо от другого. Правило заключается в том, что каждая сторона должна послать FIN (флаг окончания передачи данных), когда передача данных завершена. Когда TCP принимает FIN, он должен уведомить приложение, что удаленная сторона разрывает соединение и прекращает передачу данных в этом направлении. FIN обычно отправляется в результате того, что приложение было закрыто.

https://studfile.net/preview/4586678/page:3/

3. Поясните понятие «порт».

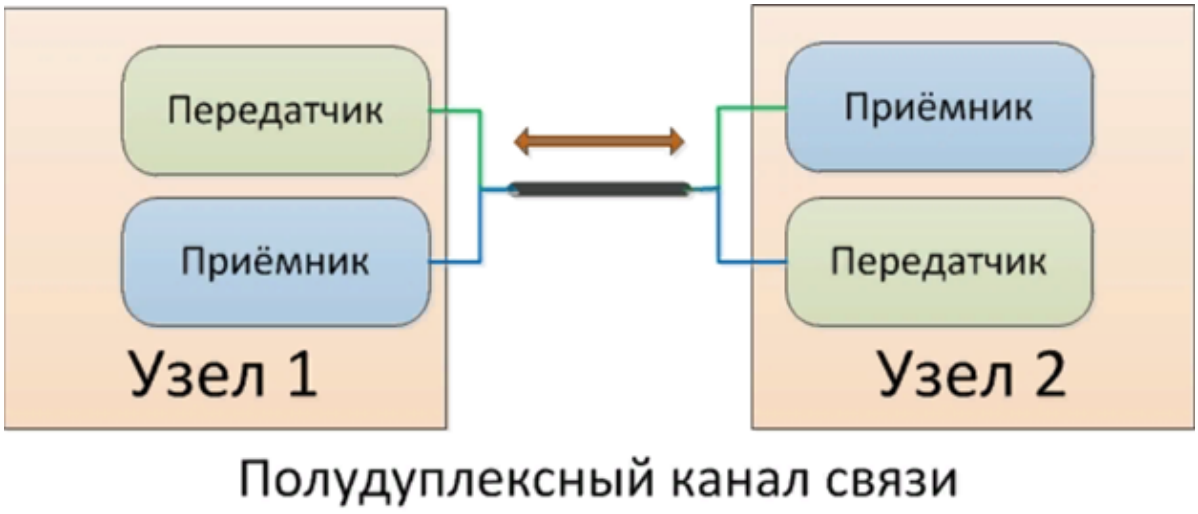
   Порт – номер процесса, который передает и получает данные.

4. Поясните понятие «сокет».

   Программный интерфейс для обеспечения обмена данными между процессами, состоит из ip и порта.

5. Поясните понятие «полудуплексный канал связи».

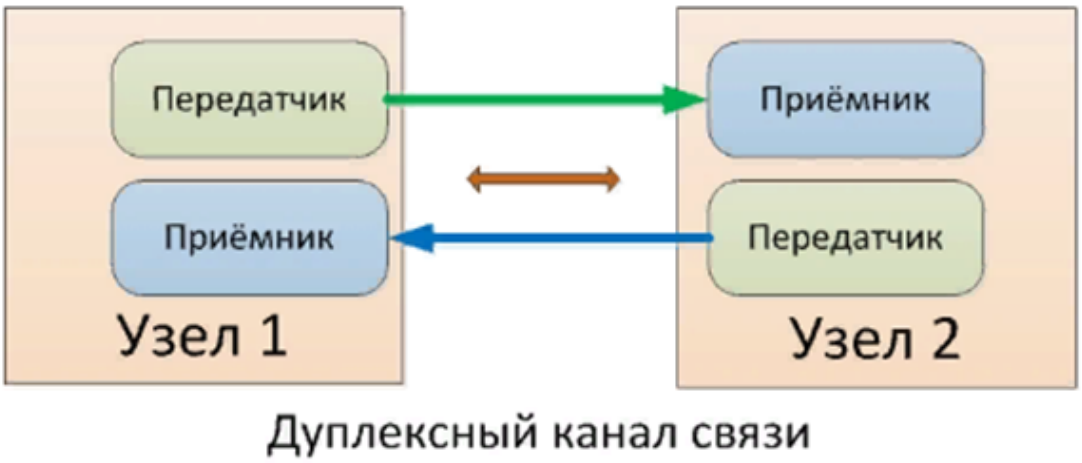
   При полудуплексном типе связи оба абонента имеют возможность принимать и передавать сообщения. Каждый узел имеет в своём составе и приёмник, и передатчик, но одновременно они работать не могут. В каждый момент времени канал связи образуют передатчик одного узла и приёмник другого.



Полудуплексный канал связи

6. Поясните понятие «дуплексный канал связи».

   По дуплексному каналу данные могут передаваться в обе стороны одновременно. Каждый из узлов связи имеет приёмник и передатчик. После установления связи передатчик первого абонента соединяется с приёмником второго и наоборот.



Дуплексный канал связи

7. Поясните отличие протокола UDP от TCP.

   - отсутствие механизмов обеспечения надежности: пакеты не упорядочиваются, и их прием не подтверждается;
   - отсутствие гарантий доставки: пакеты отправляются без гарантии доставки, поэтому процесс Прикладного уровня (программа пользователя) должен сам отслеживать и обеспечивать (если это необходимо повторную передачу);
   - отсутствие обработки соединений: каждый отправляемый или получаемый пакет является независимой единицей работы; UDP не имеет методов установления, управления и завершения соединения между отправителем и получателем данных;
   - UDP может по требованию вычислять контрольную сумму для пакета данных, но проверка соответствия контрольной суммы ложится на процесс Прикладного уровня;
   - отсутствие буферизации: UDP оперирует только одним пакетом, и вся работа по буферизации ложится на процесс Прикладного уровня;

- UDP не содержит средств, позволяющих разбивать сообщение на несколько пакетов (фрагментировать) – вся эта работа возложена на процесс Прикладного уровня.
  Ненадежные: UDP
  Надежные: TCP

Лабораторная работа №14

1. Перечислите параметры соединения с сервером БД.

   user - User name to use for authentication.

   password - Password to use for authentication.

   server - Server to connect to. You can use 'localhost\instance' to connect to named instance.

   port - Port to connect to (default: 1433). Don't set when connecting to named instance.

   domain - Once you set domain, driver will connect to SQL Server using domain login.

   database - Database to connect to (default: dependent on server configuration).

   connectionTimeout - Connection timeout in ms (default: 15000).

   requestTimeout - Request timeout in ms (default: 15000). NOTE: msnodesqlv8 driver doesn't support timeouts < 1 second. When passed via connection string, the key must be request timeout

   stream - Stream recordsets/rows instead of returning them all at once as an argument of callback (default: false). You can also enable streaming for each request independently (request.stream = true). Always set to true if you plan to work with large amount of rows.

   parseJSON - Parse JSON recordsets to JS objects (default: false). For more information please see section JSON support.

   pool.max - The maximum number of connections there can be in the pool (default: 10).

   pool.min - The minimum of connections there can be in the pool (default: 0).

pool.idleTimeoutMillis - The Number of milliseconds before closing an unused connection (default: 30000).

2. Перечислите SQL-операторов и операторы, входящие в эти группы.

   DML: Select, Insert, Delete, Update
   TCL: Commit, Rollback, Savepoint
   DDL: Create, Alter, Drop

   DCL: Grant, Revoke, Deny

3. Поясните понятие «результирующий набор».

   Отправленный набор данных, представляющий собой результат запроса к БД

4. Поясните понятия «транзакция», «фиксация транзакции», «откат транзакции». Как создать транзакцию с помощью пакета **mssql**.

   Транзакция – набор операторов, которые выполняются вместе или не выполняется вообще.
   Фиксация транзакции – сохранение результата транзакции
   Откат – возврат БД в состояние до выполнения транзакции

   Определяем функцию, вызываем пул, какие данные входные если они есть, сам запрос.

   return this.connectionPool.then(pool => pool.request().query('select * FROM FACULTY'))

5. Поясните понятие «пул соединений» и его назначение.

   Набор заранее открытых **соединений** с базой данных используемый для предоставления **соединения** в тот момент, когда оно требуется.

   Если менеджер подключений находит в пуле доступное подключение, которое в текущий момент не используется, то оно возвращается для использования. Если же доступного подключения нет, и максимальный размер пула еще не превышен (по умолчанию размер равен 100), то создается новое подключение. Если доступного подключения нет, но при этом превышен максимальный размер пула, то новое подключение добавляется в очередь и ожидает, пока в пуле не освободится место, и тогда оно станет доступным.

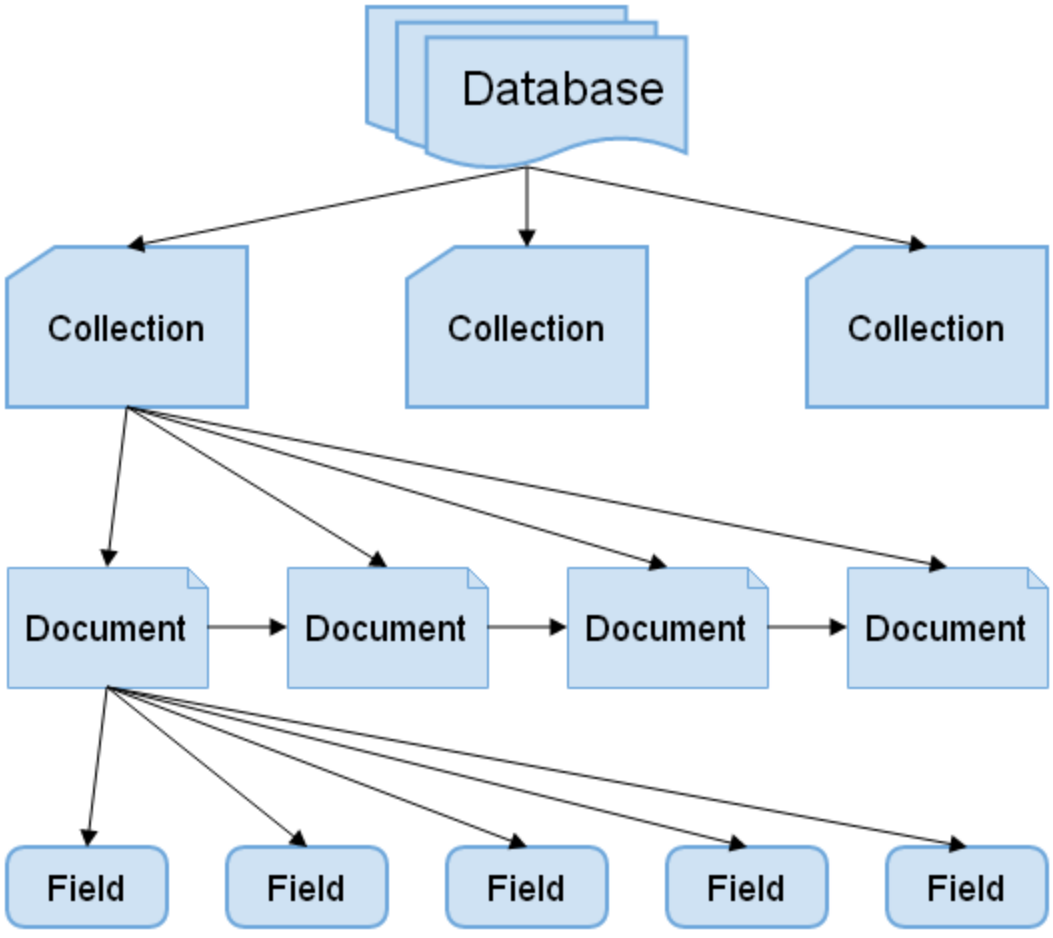
   Организатор пулов соединений удаляет соединение из пула после его простоя в течение примерно 4-8 минут или при определении им разрыва соединения с

сервером. Обратите внимание, что разорванное соединение можно определить только после попытки связи с сервером. При обнаружении соединения, которое больше не имеет связи с сервером, оно помечается как недействительное. Недействительные соединения удаляются из пула соединений только после их закрытия или возврата.

Если соединение закрывается, то оно освобождается.

Лабораторная работа №15

1. Опишите структуру БД в СУБД MongoDB.



2. Перечислите все функции API СУБД MongoDB с помощью которых можно извлечь данные из БД.

.find()- извлечь все документы из коллекции
.find(param)- извлечь документы по поиску
.findOne – извлечь один документ

3. Перечислите все функции API СУБД MongoDB с помощью которых можно добавить данные в БД.

   insertOne(): добавляет один документ

   insertMany(): добавляет несколько документов

   insert(): может добавлять как один, так и несколько документов

4. Перечислите все функции API СУБД MongoDB с помощью которых можно удалить данные в БД.

   db.collection.deleteMany()

   db.collection.deleteOne()

5. Перечислите все функции API СУБД MongoDB с помощью которых можно изменить данные в БД.

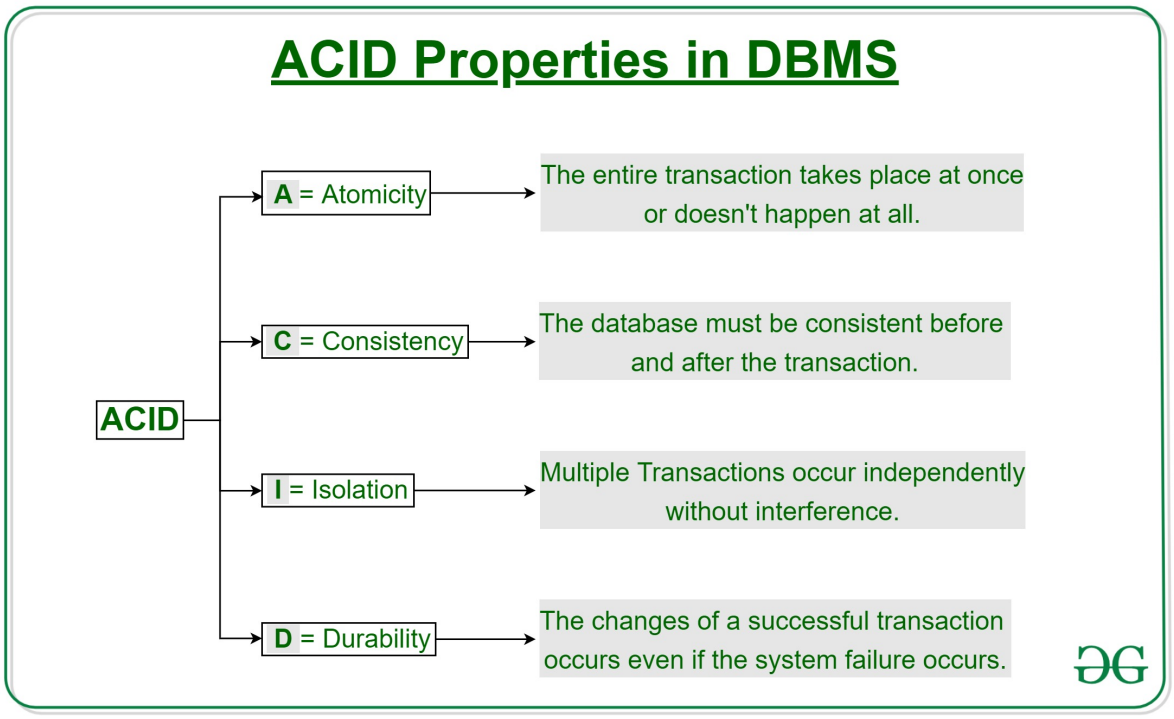
   db.collection.updateOne(<filter>, <update>, <options>)

   db.collection.updateMany(<filter>, <update>, <options>)

   db.collection.replaceOne(<filter>, <update>, <options>)

6. Какие бывают транзакции в MongoDB? Перечислите порядок их создания.

   ACID Transactions (A - атомарность, C - согласованность, I - изоляция, D - устойивость)



Естественно таких транзакций как в классических SQL решениях типа PostgreeSQL в MongoDB нет и наверно не может быть. А если появится, то это будет уже, скорее, реляционная база данных с полноценной нормализацией и контролем целостности. Поэтому, говоря о транзакциях в mongoDB, как правило, имеют в виду атомарные операции типа $set, применяемые в update() и findAndModify() в сочетании с уникальным индексом. А также двухфазный коммит, который распространен среди реляционных баз данных, если нужно обеспечить транзакции в пределах нескольких баз.

Уникальный индекс

Уникальный индекс в `mongodb` является причиной отклонить все документы, которые содержат повторяющиеся значения для индексированных полей.

Двухфазный коммит

https://habr.com/ru/post/259219/#5

https://www.mongodb.com/docs/manual/core/transactions/

https://www.digitalocean.com/community/tutorials/how-to-use-transactions-in-mongodb

Естественно таких транзакций как в классических SQL решениях типа PostgreeSQL

в MongoDB нет и наверно не может быть. А если появится, то это будет уже, скорее,
 реляционная база данных с полноценной нормализацией и контролем целостности.

Поэтому, говоря о транзакциях в mongoDB, как правило, имеют в виду атомарные операции типа $set, применяемые в update() и findAndModify() в сочетании с уникальным индексом. А также двухфазный коммит, который распространен среди реляционных баз данных, если нужно обеспечить транзакции в пределах нескольких баз.

Лабораторная работа №16

1. Поясните понятие **GraphQL**.

   GraphQL — это декларативный язык запросов, используемый клиентскими приложениями для работы с данными.
   GraphQL - это декларативный язык запросов и манипулирования данными с открытым исходным кодом для API, а также среда выполнения для выполнения запросов с существующими данными.

2. Поясните понятие схема **GraphQL**.

   Схема содержит в себе описания всех типов, полей и методов получения данных. Все типы в рамках GraphQL-схемы должны иметь уникальные имена. Не должно быть двух разных типов с одним именем.

3. Расшифруйте аббревиатуру **SDL GraphQL**.

   Для создания схем в GraphQL используется собственный язык Schema Definition Language (SDL).

4. Поясните понятие **resolver GraphQL**.

   Resolver или распознаватель — функция, которая возвращает данные для определённого поля. Resolver'ы возвращают данные того типа, который определён в схеме. Распознаватели могут быть асинхронными. С их помощью можно получать данные из REST API, базы данных или другого источника.

5. Поясните понятие **query GraphQL**.

   С помощью запросов GraphQL получает необходимые данные с сервера. Тип запроса Query в GraphQL — аналог GET в REST. ... Query описывает данные, которые необходимо получить с сервера.

6. Поясните понятие **mutation GraphQL**.

   В GraphQL изменения — способ модифицировать данные на сервере и получить обработанную информацию. Этот процесс можно рассматривать как аналогичный концепции CUD (Create, Update, Delete) в стандарте REST.

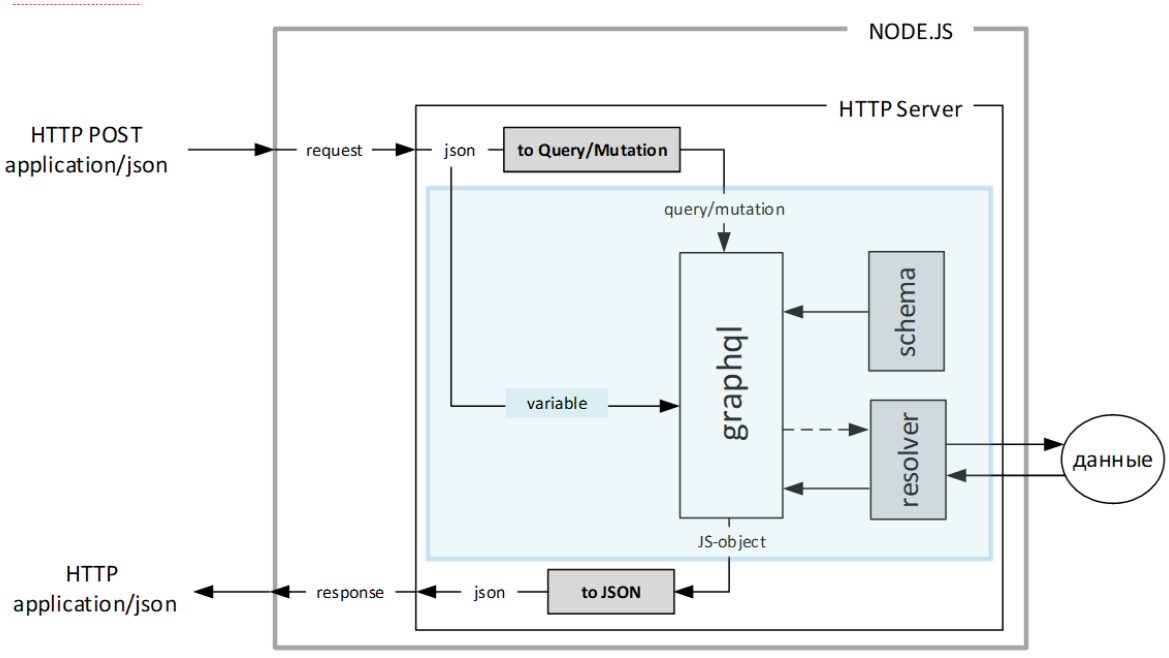
7. Поясните понятие **subscription GraphQL**.

   Подписки GraphQL - это способ передачи данных с сервера клиентам, которые выбирают прослушивание сообщений с сервера в режиме реального времени.

8. Поясните понятие **context GraphQL**.

   Контекст представляет из себя объект, хранящий информацию о подключении пользователя к базе данных.

9. Поясните схему работы модуля **graphql**



10. Поясните следующие компоненты **interface, enum, fragment, union схемы GraphQL.**

    Как и многие системы типов, GraphQL поддерживает интерфейсы. *Интерфейс*-это абстрактный тип, который включает в себя определенный набор полей, которые тип должен включать в себя для реализации интерфейса.

    Тип перечисления

    Также названный *Перечисления*, типы перечислений - это особый вид скаляра, который ограничен определенным набором допустимых значений. Это позволяет вам::

    fragment

    Допустим, у нас была относительно сложная страница в нашем приложении, которая позволяет нам смотреть на двух героев бок о бок вместе с их друзьями. Вы можете себе представить, что такой запрос может быстро усложниться, потому что нам нужно будет повторить поля по крайней мере один раз - по одному для каждой стороны сравнения.

    Вот почему GraphQL включает в себя многоразовые блоки, называемые *фрагментарный*. Фрагменты позволяют создавать наборы полей, а затем включать их в запросы там, где это необходимо. Вот пример того, как вы могли бы решить вышеприведенную ситуацию с помощью фрагментов:

    Union

    Типы объединения очень похожи на интерфейсы, но они не задают никаких общих полей между типами.