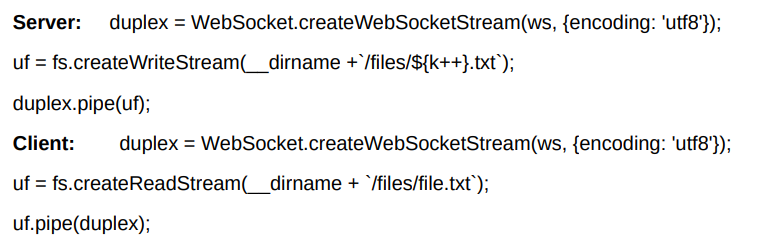
1. **Поясните понятие «TCP-порт».**

Числовой идентификатор от 1 до 65535, указывает, какой программе предназначается пакет. Каждый открытый порт – это установленное соединение между двумя точками – соответствующей программой и узлом соединения. TCP-порт — для обмена пакетами между приложениями

1. **Поясните понятие «сетевой сокет».**

IP + port. Сокет идентифицирует прикладной процесс в сети TCP/IP

1. **Поясните понятие «WebSocket».**
2. **Поясните процедуру установки соединения между WS-сервером и WS-клиентом.**
3. **Поясните понятие «широковещательное сообщение».**
4. **Поясните принцип организации потокового ввода/вывода через WS-соединение.**



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

Способ проверки доступности и живости соединения между клиентом и сервером WebSocket.

* Сервер или клиент посылает специальное сообщение ping.
* Другой участник должен ответить pong.  
  Это помогает обнаруживать "мертвые" соединения и поддерживать их активными.

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.

1. **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.

1. **Поясните принцип работы RPC-механизма, предоставляемого пакетом rpc-websockets.**
2. server.register(‘rpc-method’, <callback>);
3. если protected, то server.setAuth(login, password)
4. client.call(‘rpc-method’, params).then(res => sout(res));
5. если protected, то client.login().then(call());

В rpc-websockets клиент отправляет запрос на вызов определённой процедуры с параметрами через WebSocket. Сервер получает запрос, выполняет соответствующую функцию и возвращает результат обратно клиенту по тому же соединению. Таким образом, упрощается двунаправленное взаимодействие с удалёнными методами через WebSocket.

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

Publisher генерирует события – server.emit(‘event’) (у нас через консоль)

Subscriber подписывается – client.subscribe(‘event’)

**Subscriber/Publisher** (подписчик/издатель) — это модель взаимодействия, где издатель отправляет сообщения в определённую тему (topic), не зная, кто эти сообщения будет получать, а подписчики получают только те сообщения, на которые они подписались. Это обеспечивает асинхронную и масштабируемую рассылку сообщений.

1. **Поясните принцип действия и назначение механизма уведомлений.**

Клиент может отправлять серверу уведомления: client.notify(‘A’) (он кидает)

Сервер их обрабатывает: server.register(‘A’, <callback>)

**Механизм уведомлений** — это способ информирования пользователя или системы о наступлении определённых событий в реальном времени.

* **Назначение:** уведомления используются для мгновенного оповещения о новых данных, изменениях состояния, ошибках или важных событиях без необходимости клиенту постоянно опрашивать сервер (polling). Это снижает нагрузку и ускоряет реакцию системы.
* **Как работает:** сервер отслеживает события и при их наступлении посылает уведомление клиенту через открытое соединение (например, WebSocket). Клиент получает уведомление и может отобразить его пользователю или обработать автоматически.
* **Примеры:** уведомления о новых сообщениях в мессенджере, системные оповещения, обновления данных в реальном времени, напоминания.