**1)Какие бывают уровни клиент серверной архитектуры?**

**Двух-уровневая архитектура** клиент-серверного взаимодействия, заключается в том, что обработка запроса происходит на одном сервере без обращения к другим серверам в процессе этой обработки.

**Многоуровневая архитектура** (**трехуровневая архитектура)** это самая распространенная архитектура взаимодействия в интернете. Она появилась, когда серверную часть двухуровневой разделили на две части: слой логики и слой данных.

**Слой клиента** — отвечает за взаимодействие с пользователем. Этот слой не должен содержать бизнес-логики и не должен хранить критически важные данные.

Однако какая-то логика тут все же есть. Во-первых, это взаимодействие с пользователем через интерфейс, валидация вводимых им данных, работа с локальными файлами. Еще сюда можно отнести все, что касается авторизации пользователя и шифрование данных при работе с сервером.

Во-вторых, это несложная бизнес логика. Например, прислал интернет-магазин список товаров, на стороне клиента мы их можем отсортировать, отфильтровать. И примитивное хранение данных тут тоже есть: кэширование, куки залогиненого пользователя и тому подобное.

**Слой бизнес-логики** располагается на втором уровне, на нем сосредоточена бо́льшая часть бизнес-логики. Вне его остаются только фрагменты, экспортируемые на клиента, а также элементы логики, погруженные в базу данных (хранимые процедуры и триггеры).

**Слой данных** обеспечивает хранение данных и выносится на отдельный уровень, реализуется, как правило, средствами систем управления базами данных (СУБД), подключение к этому компоненту обеспечивается только с уровня сервера приложений.

**2)Горячий резерв серверов? холодный резерв серверов? Балансировщики**

[Существует](http://iaas-blog.it-grad.ru/katastrofoustoychivoe-reshenie-s-ispolzovaniem-iaas/) так называемое катастрофоустойчивое решение – допустимый масштаб аварии. DRS-системы могут восстановиться после крупного отказа сразу нескольких дата-центров, не парализовав работу сервисов на длительное время.

«Холодный» резерв – для репликации и резервного копирования любых данных в удаленное облачное хранилище.  Это часто низко-производительные серверы, которые заказываются и настраиваются уже после возникновения аварии. «Подъем» этой инфраструктуры может тянуться несколько дней или недель и зависит от поставщиков, транспорта и навыков персонала.

«Теплый резерв» – это более слабые серверы в минимально необходимом количестве для запуска критически важных систем; они подключены, активированы и всегда готовы к переводу нагрузки. Для запуска такой системы требуется не более одного дня.

«Горячий» резерв – решение с максимальным уровнем отказоустойчивости информационных систем или отдельных решений. Это серверы, производительность которых соответствует серверам основной площадки; в этом случае все данные реплицируются регулярно и на постоянной основе. Так как есть готовая инфраструктура, каналы, программное обеспечение, и все это собирается воедино автоматически, запуск такой системы происходит в пределах одного часа (часто меньше).

**Балансировщик нагрузки** – это устройство, которое находится между клиентом и группой серверов, который обеспечивает одинаковое использование всех серверов ресурсов, предотвращает перегрузки.

**3)Прокси-сервер** — промежуточный сервер в компьютерных сетях, выполняющий роль посредника между пользователем и целевым сервером, позволяющий клиентам как выполнять косвенные запросы к другим сетевым службам, так и получать ответы.

Позволяет скрывать реальные данные пользователя, например, IP-адрес и местоположение, подменять их своими. Ограничивать трафик от компьютера пользователя или отслеживать его — такое применяется в компаниях. Обходить блокировки по IP-адресу или по географическому положению.

**4) API (Application Programming Interface** - интерфейс программных приложений) — это установка функций и правил позволяющая взаимодействовать между программным обеспечением, которое предоставляет API и другими программными компонентами. В Веб разработке, под API обычно подразумевают набор стандартных методов, свойств, событий и URL ссылок для взаимодействия с Веб контентом.

Представляет собой передачу данных по определенному запросу со стороны клиента или другого приложения. Например: гугл календарь, погодное приложение, кнопки авторизации с помощью соц.сетей.

Простыми словами, API – это посредник среди программ, задающий правила «составления» (например схема Клієнт кафе ― програма 1, кухар ― програма 2, офіціант ― API)

API обмениваются данными и функциями, а для этого требуются четкие протоколы и архитектуры – правила, по которым будет работать API. Смысл игры не меняется, но структура разнится: RPC, REST и SOAP. Их можно назвать "форматами", каждый из которых имеет уникальные характеристики и используется для разных целей.

**5) Что такое REST API (RESTful)**

Это репрезентативная передача состояния – осуществляет реализацию клиента и сервера независимо друг от друга.

REST используются для быстрого обмена простыми параметрами, в том числе теми, из которых состоят базы данных. Поэтому REST API отлично подходит для взаимодействия огромных баз данных. Эти характеристики делают REST популярным для публичных API, например для мобильных приложений.

Model:

**Client** -- (method URI HTTP+headers, body)**>> API (REST API)** –(http request)>>**SERVER**

**Client** << (JSON, XML, HTML) **-- API (REST API)** << (http response)--**SERVER**

Технологию REST API применяют везде, где пользователю сайта или веб-приложения нужно предоставить данные с сервера. Например, при нажатии иконки с видео на видеохостинге REST API проводит операции и запускает ролик с сервера в браузере.

**6) Требования к архитектуре Rest**

**- отделение клиента от сервера** ( код запросов остается на стороне клиента, а код для доступа к данным остается на стороне сервера )

**- отсутствие записи состояния клиента (Stateless).**Сервер не должен хранить информацию о состоянии (проведенных операций) клиента. Каждый запрос от клиента должен содержать только ту информацию, которая нужна для получения данных от сервера.

**- кэшируемость.** В данных запроса должно быть указано, нужно ли кэшировать данные (сохранять в специальном буфере для частых запросов). Если такое указание есть, клиент получит право обращаться к этому буферу при необходимости.

**единство интерфейса (Uniform Interface).** Все данные должны запрашиваться через один URL-адрес стандартными протоколами, например, HTTP. Это упрощает архитектуру сайта или приложения и делает взаимодействие с сервером понятнее.

**многоуровневость системы (Layered System).**В RESTful сервера могут располагаться на разных уровнях, при этом каждый сервер взаимодействует только с ближайшими уровнями и не связан запросами с другими.

**предоставление кода по запросу (Code on Demand).**Серверы могут отправлять клиенту код (например, скрипт для запуска видео). Так общий код приложения или сайта становится сложнее только при необходимости.

**начало от нуля (Starting with the Null Style).**Клиент знает только одну точку входа на сервер. Дальнейшие возможности по взаимодействию обеспечиваются сервером.

**7) Что такое SOAP**

Это [протокол](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%82%D0%B5%D0%B2%D0%BE%D0%B9_%D0%BF%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB) обмена структурированными сообщениями в [распределённой вычислительной среде](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D1%81%D0%BF%D1%80%D0%B5%D0%B4%D0%B5%D0%BB%D1%91%D0%BD%D0%BD%D0%B0%D1%8F_%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B8%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D1%81%D1%80%D0%B5%D0%B4%D0%B0), по которому веб-сервисы взаимодействуют друг с другом или с клиентом.

Для общения используется файлы в формате XML иначе сервер вернет ошибку.

SOAP более безопасный чем REST.

**8) Чем REST отличается от SOAP**

**REST** - **это архитектурный стиль** где REST предоставляет набор требований, которым нужно следовать, чтобы предоставить веб-службу RESTful. REST поддерживает несколько форматов файлов XML, JSON, TXT, CSV, HTML, но только с одним протоколом http/https.

Архитектурный стиль для операций по работе с сервером, который взаимодействует при помощи http методов (GET, POST, PUT, DELETE).

Считается более простой альтернативой SOAP. Каждая единица информации для этого вида API ― уникальный URL-адрес, который можно запросить.

1) rest использует разные форматы файла, а soap только xml

2) rest работает только с протоколами http/https. soap работает с разными протоколами : использоваться с протоколами SMTP, FTP, [HTTP](https://blog.skillfactory.ru/glossary/http/), HTTPS

3)rest это архитектурный стиль, но в котором нет огромного количества правил, есть только рекомендации по созданию приложений, а soap это протокол который сильно ограничен теми правилами которые ему предъявляется

4)rest может быть закэширован. soap нет

5)в случае rest это простота, скорость, расширяемость, нет привязки к правилам

6)в случае soap больше безопасности, и проще для тестирования, разработка дольше и энергозатратна

7) Soap проще тестировать потому что есть строгие правила создания приложений по soap-WSDL документ-общедоступен, в случае rest из-за рекомендаций, а не правил создания ресурсов-нет чётких правил, по этому документация и наполнение приложений/ресурсов -индивидуальны, документация может быть даже не в общем доступе.

**9) Что такое HTTP, HTTPS**

**Http/https**- это протоколы прикладного уровня, https- секьюрный, то есть более безопасный и зашифрованный

-**Hyper text transfer protocol** предназначен для передачи файлов между клиентом и сервером. По правилам этого протокола, сервер может отвечать только после того, как получит запрос от клиента.

**Hyper text transfer protocol Secure** это безопасный протокол передачи данных, который поддерживает шифрование посредством криптографических протоколов SSL и TLS, и является расширенной версией протокола HTTP работает поверх протоколов TCP/IP.

**10) Какие методы HTTP**

**- GET запрос информации (безопасный)**

**- HEAD запрашивает информацию как GET, только без тела ответа (безопасный)**

- **POST отправка данных на сервер (не безопасный)**

- **PUT обновление ранее отправленных данных (не безопасный)**

- **DELETE удаление данных (не безопасный)**

- PATCH частичное изменение данных (не безопасный)

- TRACE вызов возвращаемого тестового сообщения

**-** OPTIONS описание параметров соединения с ресурсом (безопасный)

- CONNECT установление связи с сервером

Что такое CRUD

Чем GET отличается от POST

Чем POST отличается от PUT

Чем PUT отличается от PUTCH

Какие методы бывают идемпотентные

Какие методы есть безопасные

В чем разница между безопасностью и идемпатентностью?

Можно ли в POST передать данные и через URL и через Body

Можно ли с помощью URLa передать данные на сервер?

Какое ограничение есть у URLa по количеству символов

В чем разница между URI, URL, URN ?

Из чего состоит путь запроса?

Из чего состоит запрос HTTP - реквест

Из чего состоят ответы HTTP -респонс

Что такое JSON и XML

Каким форматом данных могут быть ключи в JSON

Каким форматом данных могут быть значения в JSON

Что такое статус код сервера

Какие ты знаешь?

Что такое IP адрес

Что такое порт

Что такое хост?

Чем web service отличается от web server ?

Что такое толстый клиент?

Что такое тонкий клиент?

Что такое теплый клиент

Что такое холодный клиент

Что такое куки

Что такое кеш

**Что такое WSDL**

Язык описания веб-сервисов и доступа к ним, основанный на языке XML.- описание операций в soap что умеет какой-то ресурс в SOAP.

**Что такое WADL**

Машинно-читаемое [XML](https://ru.wikipedia.org/wiki/XML)-описание для [web](https://ru.wikipedia.org/wiki/World_Wide_Web" \o "World Wide Web)-приложений [HTTP](https://ru.wikipedia.org/wiki/Hypertext_Transfer_Protocol) (как правило, [веб-сервисы](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%B5%D1%80%D0%B2%D0%B8%D1%81) [REST](https://ru.wikipedia.org/wiki/Representational_State_Transfer)). Аналог [WSDL](https://ru.wikipedia.org/wiki/WSDL) для SOAP.

Что такое Ad point

Что такое атака MAN in the Midl

Что такое и какая разница, Идентификация, Аутентификация, Авторизация

Какие ты знаешь Headers в Request

Какие ты знаешь Headers в Responce

Какие стореджи браузера ты знаешь?

В чем разница между сешин сторедж и локал сторедж?

Что такое валидация и верификация

Что такое HTML/CSS/JavaScript?

Какую структуру имеет веб-страница?

Зачем чистить кэш?

Для чего в веб-страницах используют JavaScript?

Что такое AJAX?

Что такое адаптивная и респонсивная верстка?

Как выполнить Debug страницы в браузере?

Что такое WebSocket и как проверить обрыв соединения?

SSL и TLS - это?

TCP/IP - это

Уровни TCP/IP

Уровни OSI

Что такое FTP?