**HTTP:**

HTTP Request method:

Get – для получение данных с сервера

Post – когда мы отправляем данные на сервер

Put – когда мы хотим обновить данные

Delete – когда мы хотим удалить данные

Head – такой же как и Get, только в случаи с Get нам в ответ приходит тело, а в случаи Head нам придет только метаданные.

Connect – используется, когда нам нужно установить соединение для определённого ресурса

Options – используется, когда мы хотим получить с сервера какие-то настройки

Trace – когда нужно проверить связь с ресурсом

PATCH – Обновляет только конкретные данные, т.e. те данные которые мы ему предоставим.

Разница между PUT и PATCH

Основное различие между PUT и PATCH в REST API заключается в том,

что PUT обрабатывает обновления, заменяя весь объект,

в то время как PATCH обновляет только те поля, которые вы ему предоставляете.

Идемпотентный метод – метод, которые при повторном запросе вернет один и тот же результат. К идемпотентный метод относится – GET, PUT, DELETE, HEAD.

**HTTPS:**

Https – является улучшенной версией протокола HTTP, которая поддерживает шифрования в целях повышения безопасности.

Если в случаи HTTP порт по умолчанию 8080, то в случаи с HTTPS порт по умолчанию 443.

Безопасность задается с помощью сертификатов. В данный момент протокол SSL не поддерживается, вместо него теперь TLS (на практике, протокол TSL называют еще SSL/TLS).

**Общие принципы работы SSL/TLS:**

Функции, которые выполняет TLS: шифрование, проверка целостности сообщений, аутентификация.

**Генерация ключа:**

**Публичный ключ/Открытый —** это ключ, который шифрует.

**Приватный ключ/Закрытый —** это ключ, который дешифрует.

На сервере по умолчанию хранится открытый ключ и закрытый ключ, но отдает он только открытый ключ.

**Как это работает под капотом:**

Клиент, обращаясь к серверу должен установить TLS соединение и сообщить ему версию TLS протокола, а также, методы шифрования с которыми клиент может работать. Когда сервер получил эту информацию он ее обрабатывает (выбирает самый безопасный метод шифрования и самую безопасную версию TLS) и отправляет эту информацию клиенту, а также присылает свой сертификат и открытый ключ. Клиент, получив эти данные, обращается в центр сертификации, дабы проверить подлинность сертификата не просрочен ли он и т.д. Когда все эти фазы пройдены, клиент генерирует ключи. С помощью открытого ключа, клиент шифрует и отправляет данные серверу, сервер в свою очередь дешифрует эти данные. Само общение происходит по сеансовому ключу.