**1-A. Принцип работы Интернета, используя уровни модели OSI**

В [концептуальную модель, используемую в Интернет](https://en.wikipedia.org/wiki/Internet_protocol_suite), (Internet Protocol Suite) входят четыре больших уровня. А именно:

1. Прикладной уровень
2. Транспортный уровень
3. Интернет уровень
4. Канальный уровень

HTTP (протокол передачи гипертекста) - это структура для хранения "данных". Он находится в самом верхнем уровне - прикладном уровне. Он перемещается по Интернету с помощью транспортного уровня. наиболее часто используемыми технологиями транспортного уровня являются [TCP](https://en.wikipedia.org/wiki/Transmission_Control_Protocol) и [UDP](https://en.wikipedia.org/wiki/User_Datagram_Protocol). TCP/UDP пакеты содержат в себе HTTP и перемещаются благодаря Интернету и канальному уровню. Чтобы лучше понять как работают сети и Интернет, изучите [7-уровневую модель OSI](https://en.wikipedia.org/wiki/OSI_model).

**1-B. URLы и ресурсы**

Доступ к HTTP обычно осуществляется с помощью HTTP URI. URI - это [Uniform Resource Identifier](https://en.wikipedia.org/wiki/Uniform_Resource_Identifier) (унифицированный идентификатор ресурса). URL - ([Uniform Resource Locator](https://en.wikipedia.org/wiki/Uniform_Resource_Locator) - единообразный локатор ресурса) - это особый тип URI, который используется для идентификации веб-адреса. Поэтому, когда мы говорим о ссылке, например, <https://www.instagram.com/>, мы говорим о HTTP ресурсе. Таким образом, HTTP URI, принимает форму HTTP URL.

Мы используем URLы для определения HTTP ресурсов доступных нам. Их назначение должно быть понятно из названия! Например, <https://www.instagram.com/accounts/login/> описывает ресурс, позволяющий пользователю войти в систему.

**1-C. HTTP запросы и ответы**

HTTP ресурсы предоставляются через **запросы** и **ответы**. HTTP **запрос** указывает - "дай мне это" или "сделай это " серверу. Сервер будет выполнять определенную последовательность процессов, основываясь на данных запроса, определяя какие функции должны быть вызваны и какие данные должны быть возвращены. Как только это процессы будут завершены, сервер посылает обратно HTTP **ответ**. В этом ответе сообщается - "я сделал это" или "вот <ресурс>, который был запрошен" кроме стандартных сообщений об успешном выполнении запроса или возникшей ошибки.

**1.4 HTTP методы**

Когда выполняется HTTP запрос, он содержит конкретный метод, определяющий действие, которое будет выполнено над ресурсом. Один и тот же ресурс, доступный по URL-адресу, может приминать HTTP запросы с различными типами методом и выполнять различные функции, в зависимости от указанного метода.

Наиболее важными HTTP методами являются GET и POST.

* GET: используется для получения данных. Например, для получения домашней страницы, изображений в Instagram
* POST: используется для отправки данных на сервер. Например, при входе в систему, загрузки изображений в Instagram

Назначение всех остальных методов довольно легко определить из названия. [Вы можете узнать о них больше здесь.](https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol#Request_methods)

Чтобы показать как HTTP ресурс может работать с различными типами HTTP запросов + различными комбинациями метода, мы будем использовать пример URL-адреса из Instagram:

https://www.instagram.com/media/<media-id>/likes

[Документацию к этому API можно найти здесь.](https://www.instagram.com/developer/endpoints/likes/)

Этот URL-адрес позволяет нам выполнять операции с лайками фото или видео пользователя. Выполняя HTTP запросы, используя разные методы, к этому URL-адресу можно осуществлять различные операции:

* Запрос с методом GET: возвратит список всех лайков для медиаресурса
* Запрос с методом POST: позволит текущему пользователю лайкнуть медиаресурс