**Задание 02** Ответьте на следующие вопросы.

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

MIME — это стандарт, расширяющий возможности почтовых протоколов, позволяя пересылать не только текстовые сообщения, но и сложные данные, такие как изображения, видео, аудио и приложения. MIME-тип обозначает формат содержимого (например, text/html для HTML или application/json для JSON), определяя, как следует обрабатывать и отображать данные.

Multipurpose Internet Mail Extensions — многоцелев**ы**е расширения интернет-почты)

Поддержка и управление допустимыми MIME-типами осуществляются **Internet Assigned Numbers Authority (IANA)**. IANA ведёт реестр MIME-типов, добавляя новые по мере необходимости.

MIME-тип используется в компонентах HTTP-запросов и ответов:

* **Заголовок Content-Type** — указывает формат данных в теле запроса или ответа (например, Content-Type: text/html или Content-Type: application/json).
* **Заголовок Accept** — сообщает серверу, какие форматы данных клиент способен обрабатывать (например, Accept: application/json, text/html).

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

В HTML существуют теги, при интерпретации которых браузер отправляет HTTP-запросы к указанным в атрибутах ресурсам. К ним относятся:

* <img src="URL"> — загружает изображения.
* <script src="URL"> — загружает JavaScript-файлы.
* <link href="URL" rel="stylesheet"> — загружает CSS-стили.
* <iframe src="URL"> — загружает страницы во встроенный фрейм.
* <video src="URL">, <audio src="URL">, <source src="URL"> — загружают медиафайлы (видео, аудио).
* <object data="URL"> — загружает различные типы контента (например, PDF).
* <form action="URL" method="POST/GET"> — отправляет данные формы.

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

Из JavaScript можно отправлять HTTP-запросы следующими способами:

* **XMLHttpRequest** — классический метод, предоставляющий объект XMLHttpRequest для асинхронного обмена данными между клиентом и сервером (например, xhr.open('GET', url) и xhr.send()).
* **Fetch API** — современный и упрощённый способ, предоставляющий промисы для отправки запросов (например, fetch(url).then(response => response.json())).
* **AJAX** — технология асинхронного обмена данными, часто используемая с библиотекой jQuery ($.ajax({ url: "URL" })).
* **WebSockets** — для постоянного соединения с сервером и обмена сообщениями в реальном времени (например, new WebSocket("ws://example.com")).
* **Axios** — популярная JavaScript-библиотека для отправки HTTP-запросов с использованием промисов и упрощения работы с запросами (например, axios.get(url)).

1. Поясните понятие «параметризованный модуль».

В Node.js параметризованный модуль — это модуль, который можно настраивать с помощью внешних параметров, передаваемых при его инициализации или вызове. Это позволяет изменять его поведение в зависимости от условий, делая модуль более универсальным и многократно используемым.

**Зачем нужен параметризованный модуль?**

* **Гибкость и конфигурация**: один и тот же модуль может вести себя по-разному в зависимости от переданных параметров.
* **Многократное использование**: такие модули легко переиспользовать в различных частях приложения или даже в других проектах, просто передавая разные параметры.

