Лабораторная работа 02

ПвИ

**Задание 01**

1. Разработайте серверное приложение **02-01**, которое на запрос [**http://localhost:5000**](http://localhost:5000)**/html** возвращает ответ, в теле которого содержится файл **index.html** (расположен в той же папке, что и скрипт **02-01.js**).
2. Файл **index.html** содержит статическую HTML-разметку с данными студента (факультет, курс, группа, фамилия, имя и отчество).
3. С помощью браузера проверьте работоспособность приложения.

**Задание 02**

1. Разработайте серверное приложение **02-02**, которое на GET-запрос [**http://localhost:5000**](http://localhost:5000)**/png** возвращает ответ, в теле которого содержится файл **pic.png** (расположен в той же папке, что и скрипт **02-02.js**).
2. Файл **pic.png** содержит любое изображение в формате png.
3. С помощью браузера проверьте работоспособность приложения.

**Задание 03**

1. Разработайте серверное приложение **02-03**, которое на GET-запрос [**http://localhost:5000/api/name**](http://localhost:5000/api/name)возвращает ответ, в теле которого содержится ваша фамилия имя и отчество в текстовом виде (text/plain).
2. C помощью браузера проверьте работоспособность приложения.
3. Проверьте работоспособность приложения с помощью **POSTMAN**.

**Задание 04**

1. Разработайте серверное приложение **02-04**, которое на запрос к [**http://localhost:5000/xmlhttprequest**](http://localhost:5000/xmlhttprequest)возвращает ответ, в теле которого содержится файл **xmlhttprequest.html** (расположен в той же папке, что и скрипт **02-04.js**).
2. Файл **xmlhttprequest.html** содержит HTML-разметку и JS.
3. С помощью объекта **XMLHTTPRequest** выполняется асинхронный GET-запрос к [**http://localhost:5000/api/name**](http://localhost:5000/api/name) иполученный ответ (фамилия, имя и отчество) динамически встраивается в HTML-разметку страницы.

**Задание 05**

1. Разработайте серверное приложение **02-05**, которое на запрос к [**http://localhost:5000/fetch**](http://localhost:5000/fetch)возвращает ответ, в теле которого содержится файл **fetch.html** (расположен в той же папке, что и скрипт **02-05.js**).
2. Файл **fetch.html** содержит HTML-разметку и JS.
3. С помощью объекта **fetch** выполняется асинхронный GET-запрос к [**http://localhost:5000/api/name**](http://localhost:5000/api/name)иполученный ответ (фамилия, имя и отчество) динамически встраивается в HTML-разметку страницы.

**Задание 06**

1. Разработайте серверное приложение **02-06**, которое на запрос к [**http://localhost:5000/jquery**](http://localhost:5000/jquery)возвращает ответ, в теле которого содержится файл **jquery.html** (расположен в той же папке, что и скрипт **02-06.js**).
2. Файл **jquery.html** содержит HTML-разметку и JS.
3. С помощью **jQuery.ajax** (функция библиотеки jQuery, обеспечьте в **jquery.html** соответствующий <script> c jQuery CDN: [https://code.jquery.com](https://code.jquery.com/)) выполняется асинхронный GET-запрос к [**http://localhost:5000/api/name**](http://localhost:5000/api/name)иполученный ответ (фамилия, имя и отчество) динамически встраивается в HTML-разметку страницы.

**Задание 07.** Ответьте на следующие вопросы.

1. Что такое **асинхронный запрос**?

Асинхронный запрос - это запрос, который выполняется в фоновом режиме без блокирования основного потока выполнения, позволяя приложению продолжать работу во время ожидания ответа.

1. Что такое **AJAX**?

AJAX (Asynchronous JavaScript and XML) - это технология для отправки и получения данных с сервера без перезагрузки всей страницы, обычно с использованием JavaScript и XML или других форматов данных.

1. Поясните назначение HTTP-заголовка **Content-Type**.

1. Тип данных, содержащийся в теле HTTP-запроса или HTTP-ответа.

HTTP-заголовок Content-Type определяет тип данных, передаваемых в HTTP-запросе или ответе, что позволяет получателю корректно интерпретировать содержимое, например, как HTML, JSON, или изображение.

1. Поясните назначение функции **require**.

1. Для загрузки модулей, библиотек, js-code в приложение Node.js(Синхронная, в отличии от export())

Функция require - это функция в Node.js, используемая для импорта внешних модулей или файлов в текущий скрипт, что позволяет использовать их функциональность.

1. Поясните понятие «**Модуль Node.js**».

Модуль Node.js - это независимая часть кода, представляющая собой закрытое пространство имен с определенным функционалом. Модули позволяют организовывать код в Node.js приложении, разделять его на логические компоненты и повторно использовать.

1. Поясните понятие «**Node.js built-in modules**» («**Node.js Core modules**»).

"Node.js built-in modules" или "Node.js Core modules" - это набор модулей, встроенных в Node.js по умолчанию, предоставляющих базовую функциональность для работы с файлами, сетью, системными ресурсами и другими основными задачами.

1. Какой модуль **NODE.JS** обеспечивают работу с протоколом HTTP, с файловой системой?

Модуль "http" обеспечивает работу с протоколом HTTP, а модуль "fs" - работу с файловой системой в Node.js.

1. Что такое **npm**?

npm (Node Package Manager) - это менеджер пакетов для Node.js, предназначенный для управления сторонними библиотеками и зависимостями, а также для публикации и обмена пакетами между разработчиками Node.js.