

Міністерство освіти та науки України  
КРИВОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ  
Кафедра моделювання та програмного забезпечення

## **Практична робота № 2**

з дисципліни «Сучасні технології Internet-програмування»

студент групи ІПЗ-22-1

Ющенко М.О.

Перевірили викладачі:

Трачук А.А.

Кривий Ріг

2025 р.

## Дослідження роботи протоколу HTTP

**Мета:** навчитися працювати з HTTP-запитами та відповідями, аналізувати їхні заголовки та коди стану, а також здійснювати роботу з REST API.

### Хід роботи

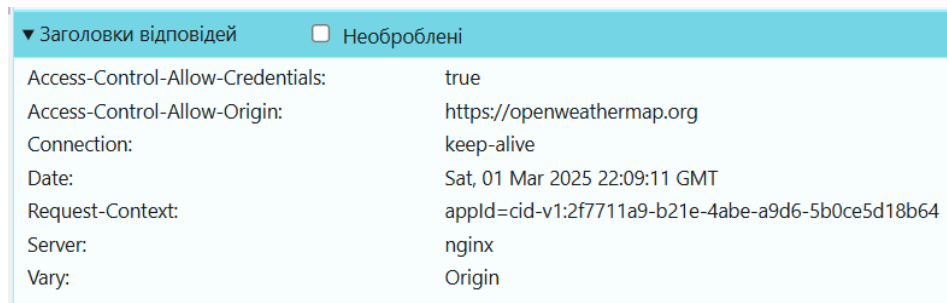
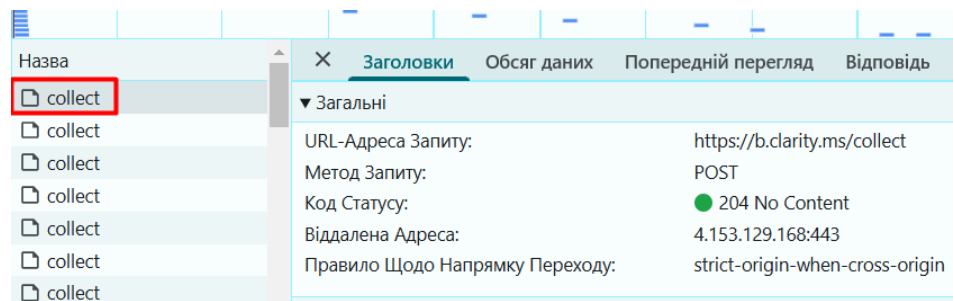
#### **Завдання 1: Аналіз HTTP-запиту через браузер. (Варіант №20).**

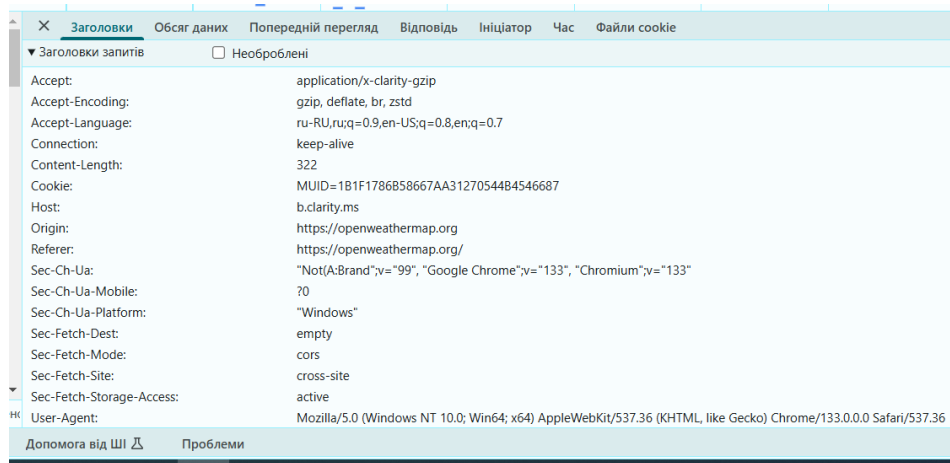
Варіант 20-29: <https://openweathermap.org>

Типи HTTP-запитів, які виконуються:

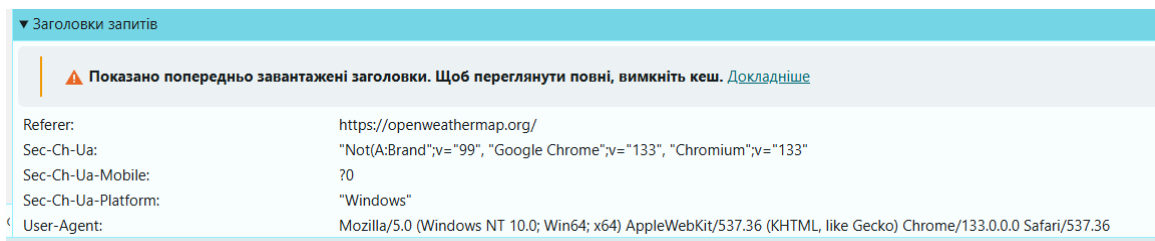
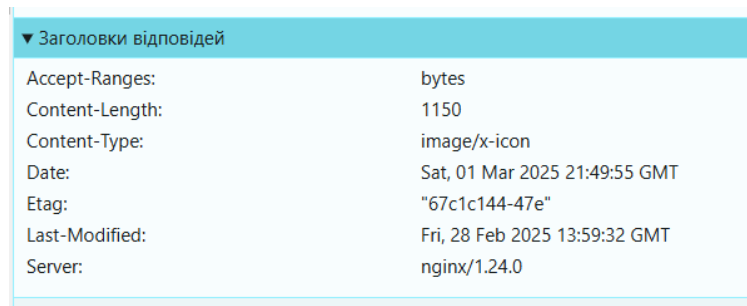
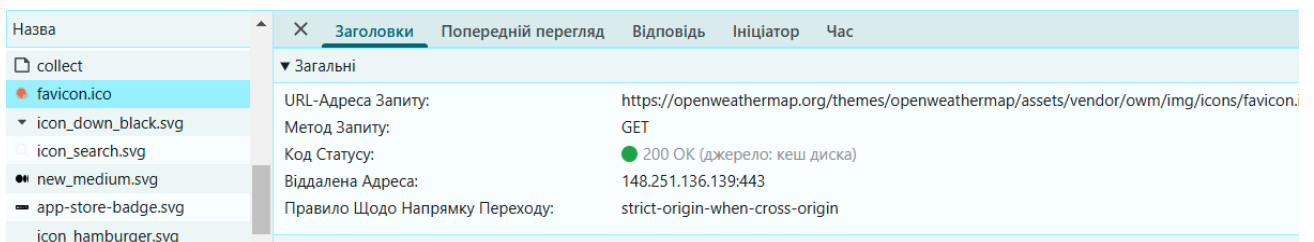
Проаналізувавши всі запити на сайті, я дійшов висновку, що присутні 2 запити: POST та GET. Нижче наведемо приклад для наглядності запитів та відповідей.

Приклад запиту POST:





Приклад запиту GET:



Завдання 2: Надсилання запиту через Postman:

Варіант 16-30: <https://api.spacexdata.com/v4/launches>

GET:

GET https://api.spacexdata.c

No environment

https://api.spacexdata.com/v4/launches

SaveShare

GEThttps://api.spacexdata.com/v4/launchesSend

ParamsAuthorizationHeaders (7)BodyScriptsSettings

Cookies

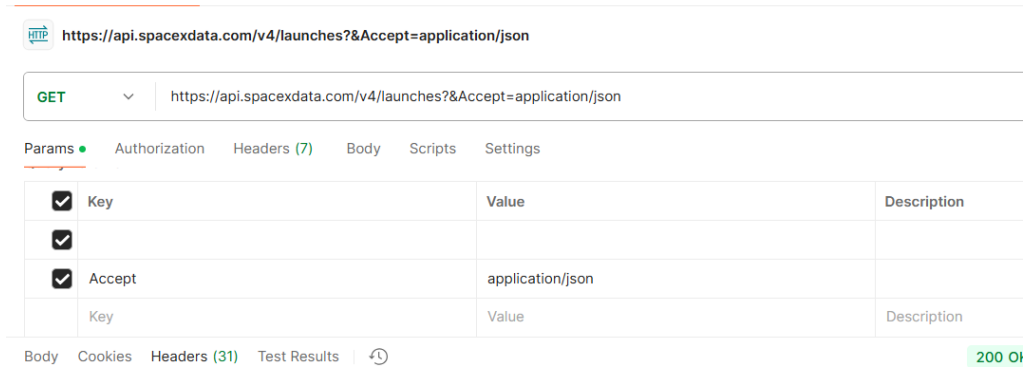
BodyCookiesHeaders (31)Test Results

200 OK1.46 s427.35 KB

JSONPreviewVisualize

```
1  [
2    {
3      "fairings": {
4        "reused": false,
5        "recovery_attempt": false,
6        "recovered": false,
7        "ships": []
8      },
9      "links": {
10       "patch": {
11         "small": "https://images2.imgbox.com/94/f2/NN6Ph45r_o.png",
12         "large": "https://images2.imgbox.com/5b/02/QcxHUb5V_o.png"
13       },
14       "reddit": {
15         "campaign": null,
16         "launch": null,
17         "media": null,
18         "recovery": null
19       },
20       "flickr": {
21         "small": [],
22         "original": []
23       }
24     }
25   ]
```

Додамо заголовок Асцепт: application/json і знову надішліть запит.



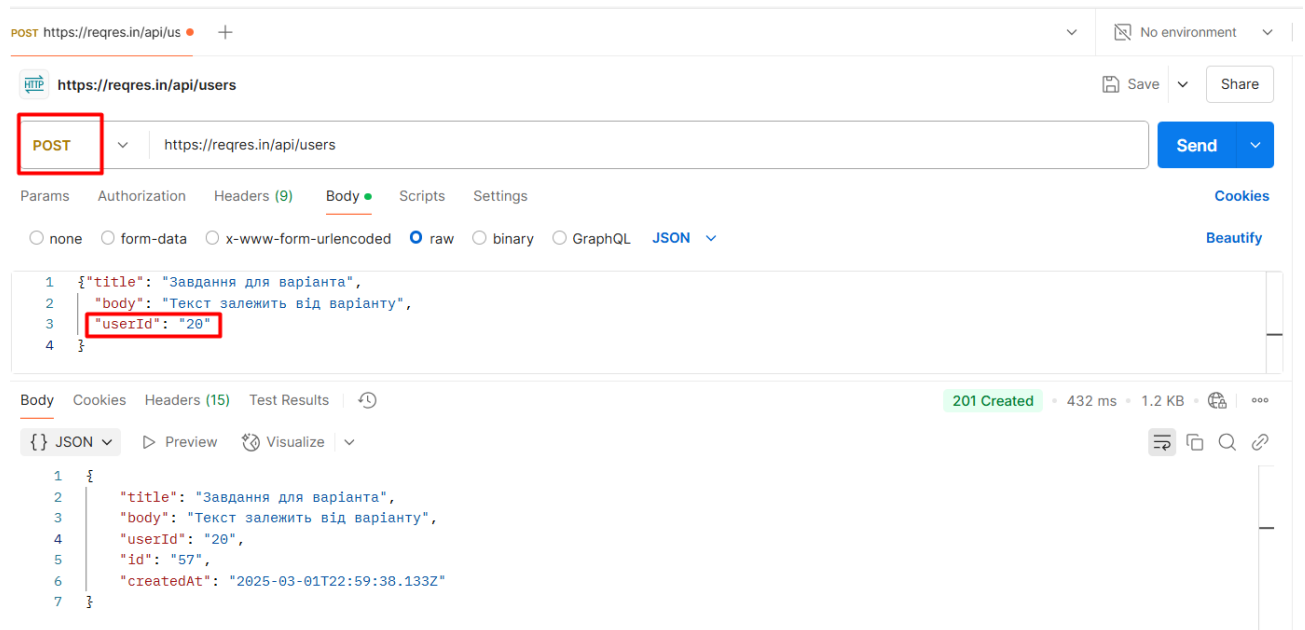
Відповідь не змінилася.

### Завдання 3: Відправка POST-запиту:

Варіант 16-30: <https://reqres.in/api/users>

Запит POST:

Введемо номер свого варіанта та зафіксуємо відповідь.



### Завдання 4: Створення простого HTTP-сервера:

Варіант № 20: Повертає HTML із вбудованим стилем.

Код у файлі server.js:

```

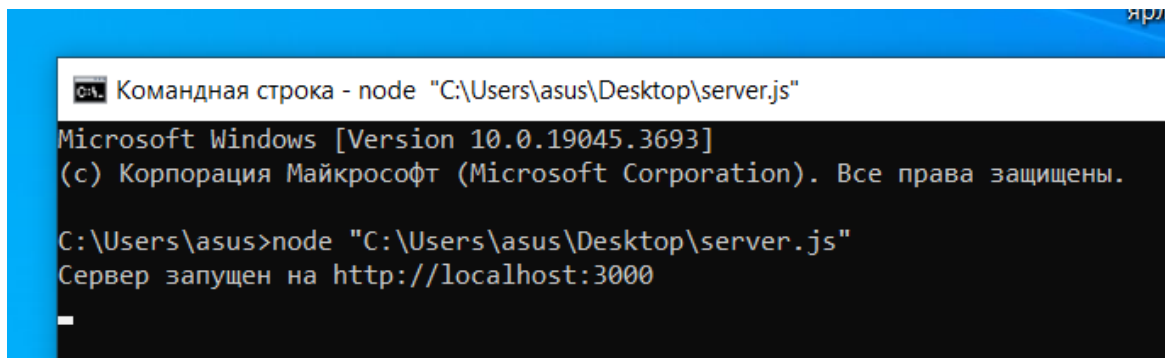
const http = require('http');

const server = http.createServer((req, res) => {
  res.writeHead(200, { 'Content-Type': 'text/html; charset=utf-8' });
  res.end(`
    <!DOCTYPE html>
    <html lang="ru">
    <head>
      <meta charset="UTF-8">
      <meta name="viewport" content="width=device-width, initial-scale=1.0">
      <title>SERVER</title>
      <style>
        body {
          font-family: Arial, sans-serif;
          background-color: #f4f4f4;
          text-align: center;
          padding: 50px;
        }
        h1 {
          color: #333;
        }
        p {
          font-size: 18px;
        }
      </style>
    </head>
    <body>
      <h1>Добро пожаловать на сервер Node.js</h1>
      <p>Этот сервер возвращает HTML со встроенными стилями.</p>
    </body>
    </html>
  `);
});

const PORT = 3000;
server.listen(PORT, () => {
  console.log(`Сервер запущен на http://localhost:${PORT}`);
});

```

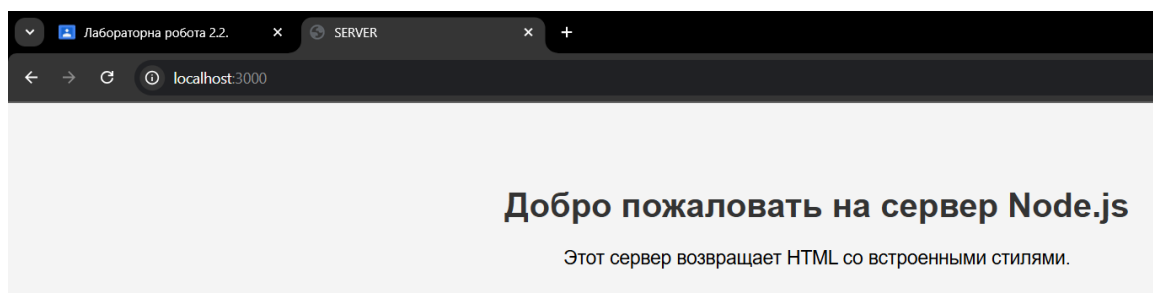
**Запуск сервера:**



```
Командная строка - node "C:\Users\asus\Desktop\server.js"
Microsoft Windows [Version 10.0.19045.3693]
(c) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

C:\Users\asus>node "C:\Users\asus\Desktop\server.js"
Сервер запущен на http://localhost:3000
```

## Скріншот роботи:



## Контрольні запитання:

### 1. Що таке HTTP і які його основні методи?

HTTP (HyperText Transfer Protocol) — це протокол передачі гіпертексту, який використовується для комунікації між веб-браузером і сервером. Основні методи:

- **GET** – отримання даних із сервера.
- **POST** – надсилання даних на сервер для створення ресурсу.
- **PUT** – оновлення або створення ресурсу на сервері.
- **PATCH** – часткове оновлення ресурсу.
- **DELETE** – видалення ресурсу.
- **HEAD** – отримання тільки заголовків відповіді без тіла.
- **OPTIONS** – запит на доступні методи для ресурсу.

### 2. Для чого використовуються заголовки HTTP?

Заголовки HTTP містять мета-інформацію про запит або відповідь. Вони використовуються для:

- **Ідентифікації клієнта та сервера** (User-Agent, Host, Referer).
- **Автентифікації та авторизації** (Authorization, WWW-Authenticate).
- **Керування кешуванням** (Cache-Control, Expires, ETag).
- **Визначення типу контенту** (Content-Type, Content-Length).
- **Захисту та безпеки** (Strict-Transport-Security, Content-Security-Policy).

3. Які групи статусних кодів ви знаєте?

Статусні коди HTTP поділяються на 5 основних груп:

- **1xx (Інформаційні)** – запит отримано, продовження обробки (100 Continue).
- **2xx (Успішні)** – запит виконано успішно (200 OK, 201 Created).
- **3xx (Перенаправлення)** – потрібно виконати додаткові дії (301 Moved Permanently, 302 Found).
- **4xx (Помилки клієнта)** – проблема на стороні клієнта (400 Bad Request, 404 Not Found).
- **5xx (Помилки сервера)** – проблема на сервері (500 Internal Server Error, 503 Service Unavailable).

4. Що таке REST API та його основні принципи?

REST (Representational State Transfer) — це архітектурний стиль для створення веб-сервісів, який базується на стандартах HTTP. Основні принципи:

- **Клієнт-серверна архітектура** – розподіл клієнтської та серверної логіки.
- **Відсутність стану (stateless)** – кожен запит повинен містити всю необхідну інформацію.
- **Кешування** – можливість використовувати кеш для зменшення навантаження.
- **Єдиний інтерфейс** – стандартизовані HTTP методи (GET, POST, PUT, DELETE).
- **Шарова архітектура** – можливість використання проксі та балансувальників навантаження.

5. Як HTTPS захищає передачу даних у мережі?



HTTPS (HyperText Transfer Protocol Secure) – це розширення HTTP, яке використовує **SSL/TLS** для захисту даних. Воно забезпечує:

- **Шифрування** – всі передані дані зашифровані та не можуть бути прочитані третіми особами.
- **Цілісність даних** – дані не можуть бути змінені під час передачі.
- **Автентифікацію** – сервер (а іноді й клієнт) ідентифікуються через SSL-сертифікати.