Лабораторна робота №4: Створення примітивного кешуючого проксі-сервера на базі модуля node:http

Мета роботи

- Опанувати вбудований модуль node:http
- На практиці перевірити знання про протокол http, структуру запиту та відповіді
- Навчитися застосовувати відомі патерни асинхронного програмування
- Застосувати програму curl та браузер для тестування проксі сервера

Теоретичний матеріал

- Лекція 5 - Практика програмування з Node.js
- Лекція 6 - Обробка подій Node.js і асинхронне виконання
- Лекція 7 - Протокол HTTP. Інструменти налагодження HTTP запитів. Простий сервер на Node.js

Вступ

Проксі-сервер - це сервер посередник між клієнтом та іншим сервером. Усі запити до іншого серверу ідуть через нього, і таким чином він отримує можливість змінювати як запит, так і відповідь. Кешуючий проксі-сервер використовує кеш (для того, щоб обмежити трафік до іншого серверу, та/або приквидшити відповідь) зберігаючи відповіді до попередніх запитів. Коли запит роблять вперше, то проксі-сервер бачить, що у кеші немає збереженої відповіді для такого запиту. Тому він перенаправляє запит на інший сервер, а коли відповідь приходить, то він: 1) Спочатку зберігає відповідь у кеш; 2) Далі пересилає отриману відповідь клієнту. Наступного разу, для того ж самого запиту він бачить, що у кеші є вже збережена відповідь. Тому він читає цю готову відповідь і надсилає її одразу клієнту, оминаючи цим запит до іншого сервера.

Для лабораторної роботи, ми кешуватимемо картинки з сайту https://http.cat, на якому для кожного з статусних кодів HTTP можна отримати жартівливу картинку кота. Наш проксі сервер має кешувати картинки на диск, і надсилати їх клієнту для кожного з повторних запитів. Для отримання картинки, передайте стаусний код HTTP у шляху URL https://http.cat

Підготовка до роботи

- Створіть **новий репозиторій з назвою bc2024-4**
- Встановіть ім'я користувача у форматі **`Ваше Ім'я bc2024-4`** та вашу електронну пошту для коміту з допомогою команди git config
- **Створіть файл package.json** у кореневій теці вашого репозиторію. Створіть головний файл програми. Встановіть пакети Commander.js і superagent локально. Встановіть пакет nodemon локально як залежність лише для розробки. Закомітьте файли package.json та головний файл програми.

Частина 1 - параметри командного рядка та Веб сервер

- Ваша програма має приймати наступні аргументи (використайте `Commander.js`):
  - `-h, --host` - (обов'язковий параметр) адреса сервера
  - `-p, --port` - (обов'язковий параметр) порт сервера
  - `-c, --cache` - (обов'язковий параметр) шлях до директорії, яка міститиме закешовані файли
- З допомогою модуля http запустіть веб сервер. Значення параметрів `--host, --port` мають бути передані у метод http.Server.listen()
- Для того, щоб сервер автоматично перезапускався кожного разу, коли ви змінюєте програму, запустіть сервер з допомогою nodemon
- Програма має виводити помилку, якщо не задано обов'язковий параметр.
- Виконайте вимоги цієї частини та закомітте результат.

Частина 2 - читання, запис та видалення файлів з кешу

- Використовуйте асинхронні виклики **`fs.promise.readFile`** та **`fs.promise.writeFile`** для читання та запису файлів (модуль fs Node.js) у цій частині.
- Кожен з http запитів приймає назву http коду, для якого ми демонструємо картинку, яку передають у шляху URL (наприклад, /200 означає картинку для коду 200)
- Після реалізації одного з методів, зробіть коміт зі змінами перез тим, як починати працювати над наступним. Протестуйте роботу запиту перед тим як робити коміт.

- Сервер має відповідати на запити з наступними методи HTTP:
    - **GET** - отримати картинку з кешу на диску, яка відповідає заданому коду HTTP. Картинка має бути у тілі відповіді.
    - **PUT** - записати картинку, яка відповідає заданому коду HTTP, у кеш на диск або замінити існуючу. Картинка яку зберігають має міститися у тілі запиту.
    - **DELETE** - видалити картинку, яка відповідає заданому коду HTTP, з кешу
- Якщо використано будь який інший метод, то сервер має повернути відповідь зі статусним кодом 405 (Method not allowed)
- Якщо картинку не знайдено у кеші, сервер має повернути відповідь з кодом **404 (Not Found)**
- Для кожного успішного запиту відповідь має містити код **200 (OK)**, крім методу PUT, який має повертати код **201 (Created)**
- Відповідь яка містить картинку, має мати хедер **Content-Type** зі значенням **image/jpeg**

Частина 3 - реалізація запиту на сервер http.cat

- Додайте перевірку, яка буде надсилати запит на отримання картинки з сайту https://http.cat у випадку, коли кеш не має такої картинки.
    - Для запиту підключіть і використайте модуль superagent
    - Для отримання картинки, передайте стаусний код HTTP у шляху URL https://http.cat
    - Якщо запит завершився помилкою, то проксі-сервер має повернути відповідь з кодом 404 (Not Found)
    - Якщо запит був успішний, збережіть картинку у кеш, так щоб наступного разу проксі-сервер брав картинку з кешу без запиту на сервер
- Протестуйте роботу проксі сервера на відповідність вимогам. Закомітьте результат роботи.

Результат:

- як мінімум 5 комітів виконані з командого рядка
- репозиторій має містити лише файл package.json та головний файл програми
- сервер має відповідати функціональним вимогам, які описані у ході лабораторної роботи

Максимальна оцінка: 6 балів