Web-програмування

Лабораторна робота №9 (частина 2)
Робота з Fetch API, URL API, History API.

Мета: створити сторінку пошуку друзів у соціальних мережах з картками користувачів, пошуком, сортуванням та фільтрацією. Набути практичних навичок роботи з Fetch API.

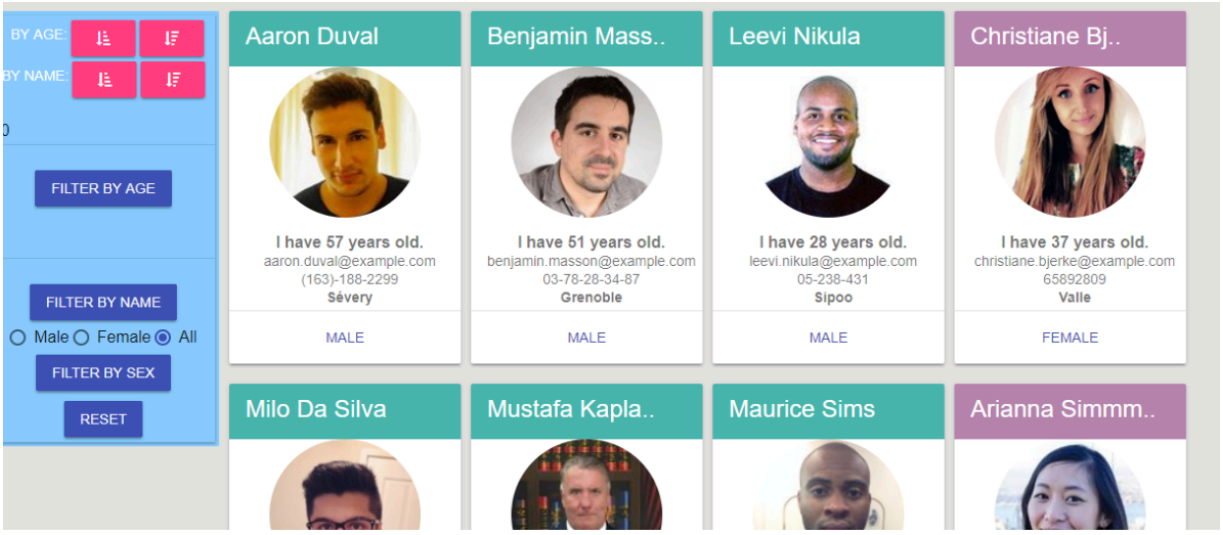
Вимоги та самостійна робота:

1. Randomuser api with awesome simple examples and docs - ви можете використовувати один із цих API для отримання даних
2. Network requests - https://uk.javascript.info/network
3. URL API - URL API;
4. History API - https://developer.mozilla.org/en-US/docs/Web/API/History_API
5. Методи масивів - https://uk.javascript.info/array-methods
6. Array - https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array

Практичні завдання:

Створіть невеличку сторінку пошуку друзів у соціальних мережах з картками користувачів, пошуком, сортуванням та фільтрацією їх за віком, прізвищем, ім'ям, або чим завгодно, за допомогою Random User API.

Це приклад. Ви повинні стилізувати свою програму так, як хочете. Зробіть це особистим. Зробіть це красивим.



Технічні вимоги:

- Під час заходу на сторінку, показати юзеру форму авторизації/реєстрації з л.р. 9 (частина 1). Зробити фейкову авторизацію юзера на сторінці, записавши його у LocalStorage (в реальності це мав би бути запит на сервер). Logout очищає дані з LocalStorage і знову показує юзеру форму авторизації/реєстрації.
- Вивести список карток користувачів на сторінці, картки повинні бути гарно стилізовані з фотографією користувача, ім'ям, віком, номером телефону, можливо, з іншою інформацією користувача, яку ви бачите в інформаційній картці користувача;
- Пошук із застосуванням debounce: ввівши ім’я в поле пошуку, і одразу побачити фільтрацію на сторінці;
- Сортування: додайте можливість сортувати картки за іменем/віком, за алфавітом (A-Z/Z-A) та за датою реєстрації;
- Фільтрація: додайте можливість фільтрувати друзів за віком, іменами, місцем розташування та електронною поштою;
- Додати стильні іконки та крутий дизайн з тінями, градієнтами.
- Створить гарний дизайн, який підходить для мобільних пристроїв (responsive/fluid/elastic тощо);
- Реалізувати підтримку адресного рядка, що означає, що користувач повинен бачити стан фільтрів і сортування в URL;
- Додайте довантаження контенту при прокручуванні (скролу).

Контрольні питання:

1. Що таке API, і яка його роль у розробці веб-сайтів?Які типи даних можна отримати через API для веб-сайту?
2. Що таке мережевий запит і які його типи існують?
3. Які переваги має використання Fetch API порівняно з іншими методами зробити мережеві запити?
4. Як ви виконуєте асинхронний мережевий запит за допомогою Fetch API?
5. Що таке URL API, і яка його роль у веб-розробці?
6. Які основні методи доступу до URL API?
7. Що таке History API і для чого він використовується в веб-розробці?
8. Як використовувати History API для зміни URL без перезавантаження сторінки?
9. Які методи і події History API можуть бути корисними для покращення користувацького досвіду?