Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Кафедра мережних технологій факультету інформатики

«Розробка системи ранжування авторів за публікаціями та цитуванням (фронтенд, ASP.NET)»

Текстова частина до курсової роботи
за спеціальністю «Комп'ютерні науки» 122

Керівник курсової роботи
старший викладач Сініцина Р.Б.
_____
(підпис)
"____" _____ 2021 р.
Виконала студентка Шекета К.О.
"____" _____ 2021 р.

Київ 2021

Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Кафедра мультимедійних систем факультету інформатики

ЗАТВЕРДЖУЮ

старший викладач

_____ Сініцина Р.Б.

(підпис)

„____”_____2021 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

на курсову роботу

студентці Шекеті Катерині

4-го курсу факультету інформатики

ТЕМА: Розробка системи ранжування авторів за публікаціями та цитуванням (фронтенд, ASP.NET)

Вихідні дані:

-

-

Зміст ТЧ до курсової роботи:

Вступ
Анотація
1. Аналіз предметної області
2. Вибір програмного забезпечення
3. Реалізація практичної частини
Висновки
Список літератури
Додатки (за необхідністю)

Дата видачі „___” _____ 2021 р.

Керівник _____

(підпис)

Завдання отримав _____

(підпис)

Календарний план виконання курсової роботи

Тема: Розробка системи ранжування авторів за публікаціями та цитуванням (фронтенд, ASP.NET)

№ п/п	Назва етапу курсового проекту (роботи)	Термін виконання етапу	Примітка
1.	Отримання завдання на курсову роботу	листопад 2020 р.	
2.	Огляд літератури за темою роботи	листопад - грудень 2020 р.	
3.	Вибір програмного забезпечення для роботи	січень 2021 р.	
4.	Створення веб-застосунку (практичної частини роботи)	січень - березень 2021 р.	
5.	Написання текстової частини роботи	березень - квітень 2021 р.	
6.	Створення слайдів для доповіді та написання доповіді	квітень 2021 р.	
7.	Надання роботи керівнику для перевірки	квітень 2021 р.	
8.	Корегування роботи за результатами перевірки керівником	квітень 2021 р.	
9.	Остаточне оформлення пояснювальної роботи та слайдів	квітень 2021 р.	
10.	Захист курсової роботи	19 квітня 2021 р.	

Студентка Шекета К.О._____
Керівник Сініцина Р.Б._____
“_____” _____

Зміст

Вступ

Наука відрізняється від інших видів діяльності людини (мистецтво, релігія, мораль тощо) головним чином тим, що в ній існують визначені стандарти для характеристики прогресу. Науку також можна розглядати як спільну діяльність поколінь дослідників, які вносять свій вклад у її розвиток. Щоб внести цей вклад як учений, необхідно отримати визнання своїх колег шляхом проведення деякого дослідження чи експерименту і подальшої публікації дослідження у науковому журналі.

На момент 2021 року існує близько 30 000 різних наукових журналів, у яких щороку в середньому публікується близько двох мільйонів наукових статей. Деякі з журналів є більш авторитетними, аніж інші, і в наукових колах це береться до уваги. Зважаючи на велику кількість наукових робіт, які публікуються, дуже критично брати до уваги якість цих робіт. Деякі наукові роботи можуть виявитися роботами некомпетентних шарлатанів, які ведуть в оману і не дають жодного поштовху для наукового прогресу. Для такого послідовного процесу, як розвиток науки, необхідно керуватися чіткими критеріями для визначення дійсно корисних робіт і вчених, компетентності яких можна довіряти. Одними з таких критеріїв є кількість цитувань робіт автора та загальна кількість його публікацій. Чим їх більше, тим авторитетнішим є дослідження і його автор.

Для цього існують спеціальні сервіси для авторів наукових публікацій. На цих сервісах розміщуються дані про статті і авторів: назва статті, імена авторів, місце роботи авторів, ключові слова, резюме статті, вказується джерело фінансування, список використаної літератури, а також в яких інших публікаціях було процитовану роботу і скільки разів або інші показники впливовості автора.

Мета даної роботи розробити фронтенд для системи ранжування авторів за публікаціями та цитуваннями, яка, використовуючи загальноприйняті

бібліометричні показники вченого: індекс Гірша, загальна кількість публікацій, загальна кількість цитувань — проводить ранжування авторів.

Анотація

Курсова робота присвячена створенню фронтенд частини для веб-застосунку системи ранжування авторів за публікаціями та цитуваннями з використанням веб-технологій HTML, CSS та JavaScript-бібліотеки React.

1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1. Огляд основних понять

Бібліометрія — поняття, введене англійським вченим А. Прічардом, яке означає застосування математичних та статистичних методів до вивчення друкованих видань. Традиційні базові бібліометричні показники можна умовно поділити на дві основні групи – показники вагомості журналу (наприклад, імпакт-фактор) та показники оцінки публікаційній діяльності вченого (наприклад, індекс Гірша) [1].

Індекс Гірша, також відомий як h-індекс — кількісна статистична характеристика продуктивності вченого. Цей індекс запропонував американський фізик Хорхе Гірш. Він заснований на кількості публікацій вченого і на кількості цитувань його публікацій за весь період наукової діяльності. За допомогою цього показника можливо більш точно оцінити наукову продуктивність дослідника, на відміну від загального числа публікацій або цитувань. Цей індекс вираховується таким чином: учений має індекс h, якщо h з його N статей цитуються як мінімум h раз кожна. [2]

Алгоритми ранжування — особливі методи фільтрації результатів запиту. Вони визначають порядок і вид показу інформації згідно із запитом. [3]

Фронтенд веб-розробка — представлення даних у вигляді графічного інтерфейсу за допомогою HTML, CSS та JavaScript таким чином, щоб користувачі могли взаємодіяти з цими даними.

1.2. Аналіз наявних сервісів

Microsoft Academic — пошукова система наукових робіт. На цьому сервісі можна здійснювати пошук авторів за їхніми іменами, організаціями, в яких вони працюють. Також можна отримати інформацію про загальну кількість цитувань і публікацій конкретного автора.

Сервіс також на головній стоірнці надає список авторів, відсортованих за рейтингом помітності (saliency rating). Причина для використання цієї метрики

замість традиційних пояснюється тим, що традиційні методи, включаючи h-, g-індекс та еквівалентні, розглядають кожне цитування як рівне та постійне. З іншого боку, помітність бере до уваги кожне цитування на основі факторів джерел, що цитують, а також репутацію та скільки років існує кожне цитування.

Щоб переглянути рейтинг авторів за іншими показниками, необхідно перейти у спеціальний розділ, де у випадаючому списку можна обрати індекс, за яким ранжується список.

Веб-застосунок, який розробляється у даній роботі так само надаватиме можливість для різноманітних критеріїв ранжування. На етапі початкової розробки ранжування можна здійснити за традиційними h-індексом, кількістю цитувань або публікацій. Але інтерфейс також буде надавати можливість розширити цей список.

1.3. Постановка задачі

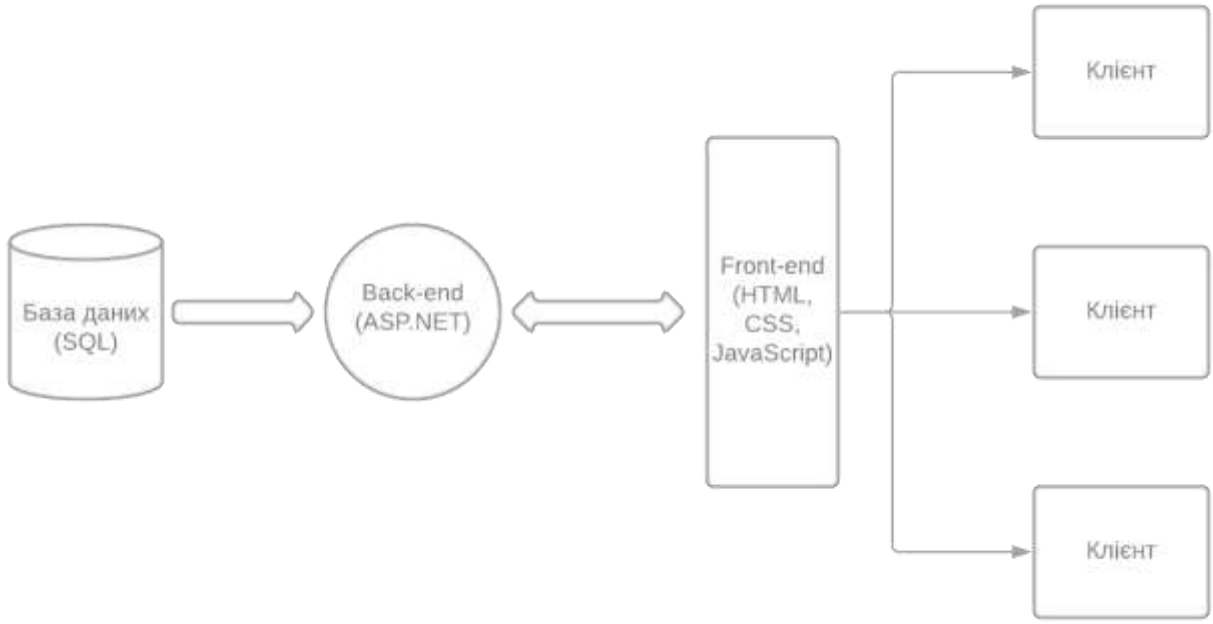


Рис. 1.1.1

Метою розробки є фронтенд (рис. 1.1.1) для веб-застосунку системи ранжування авторів за публікаціями та цитуваннями, в якому користувачам буде зручно здійснювати пошук автора за його ім'ям або назвою організації, в якій він працює, а також бачити відсортований список авторів за кількістю публікацій,

цитувань і деяким додатковим критерієм (наприклад, індексом Гірша). Use-case діаграма наведена в додатку А.

1.3.1. Що повинно бути у веб-застосунку

На головній сторінці користувач може бачити таблицю авторів, в якій зазначено ім'я автора, організація, в якій він працює, кількість цитат та публікацій, а також значення деякої обраної метрики після операції ранжування. В таблиці є можливість відображення даних у відсортованому порядку за кількістю публікацій, цитат або деяким індексом у порядку зростання або спадання. Над таблицею є поле для пошуку за ім'ям автора та/або організацією, в якій він працює. Результат пошуку відображається у таблиці. Також користувач може перейти на сторінку конкретного автора, де він зможе побачити основні дані про автора (ім'я, прізвище, організація, кількість цитувань) і таблицю з його публікаціями. Вигляд веб-застосунку є максимально простий і зрозумілий. Нижче наведено схему розміщення основних компонентів (рис. 1.3.1.1)

Рис. 1.3.1.1

2. ВИБІР ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1. React

Для розробки фронтенд частини веб-застосунку було обрано JavaScript-бібліотеку React.

React — популярна JavaScript-бібліотека для створення користувацьких інтерфейсів. Вона розробляється Facebook, Instagram і спільнотою розробників. Перший випуск відбувся в 2013 році. Головний принцип полягає у створенні простих незалежних компонентів, які зберігають і керують власним станом. За допомогою них створюються складні інтерфейси. [4]

Керування станом, починаючи з версії React 16.8, можна здійснювати за допомогою Hooks API [5]. React Hooks (далі — “гуки”) — це вбудовані функції, які дозволяють працювати зі станом функціональних компонентів. Загалом презентували 10 різних видів “гуків”, найчастіше з яких у даній роботі використовуються 2:

- useState() — дозволяє React обробляти або змінювати стан усередині функціональних компонентів без необхідності перетворювати його на компонент класу. useState()-“гук” повинен отримати початкове значення стану і повертає масив з двома змінними. Перший елемент масиву це саме значення стану, другий елемент — функція, яка оновлює значення стану.
- useEffect() — “гук”, який приймає функцію з ефективним кодом. У функціональних компонентах ефекти, такі як мутації, підписки, таймери, логування та інші ефекти, забороняється розміщувати всередині функціонального компонента, оскільки це може призвести до великої кількості невідповідностей при рендері інтерфейсу, а також до заплутаних помилок. useEffect() також використовується для отримання даних з зовнішнього API.

Головна перевага React у тому, що це бібліотека з простої міграцією між версіями. У цієї бібліотеки також є потужна підтримка спільноти, яка створює

компоненти повторного використання, які можна використовувати у проєктах. Ще однією особливістю цієї бібліотеки є те, що вона використовує віртуальний DOM (Document Object Model), тому замість повторного рендеру цілої сторінки, як це відбувається у звичайному DOM, React оновлює лише ті об’єкти, які змінилися, таким чином економлячи час та ресурси, які в іншому випадку витрачаються складними маніпуляціями DOM.

Отже, вибір React можна обгрунтувати його популярністю, використанням віртуального DOM і наявністю багатої бібліотеки компонентів.

2.2. Axios

Бібліотека, за допомогою якої можна створювати http-запити до зовнішніх ресурсів.[6] У таких випадках популярним рішенням також є Fetch API, але у Axios є ряд переваг:

- здатність працювати у старих браузерах.
- автоматичне переведення отриманих даних в рядок.
- простий і лаконічний синтаксис.

3. РЕАЛІЗАЦІЯ ПРАКТИЧНОЇ ЧАСТИНИ

3.1. Структура проєкту

Проєкт складається з React-компонентів, які знаходяться у директорії ClientApp усього проєкту включно з бекендом. Загалом структура клієнтської частини проєкту зображено на рисунку 3.1.1:

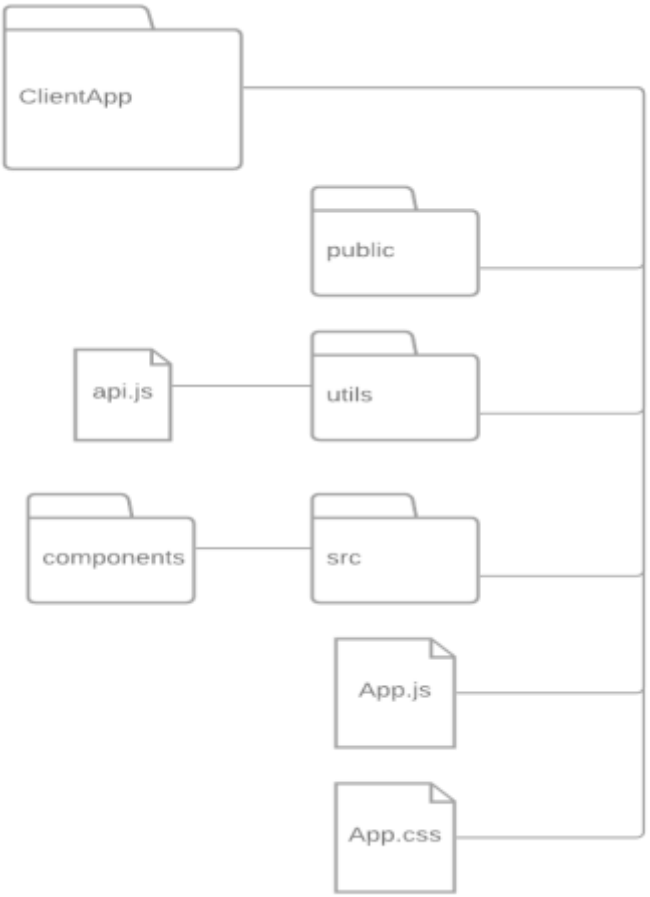


Рис. 3.1.1

Дамо опис кожному компоненту схеми:

- основна директорія для клієнтської частини проєкту ClientApp. В ній знаходяться:
  - директорія public
  - директорія src. У директорії src знаходяться усі React-компоненти
  - директорія utils. Тут знаходиться спеціальний скрипт api.js для взаємодії за допомоги axios з api бекенд-серверу.

- файли App.js та App.css. App.js є найголовнішим React-компонентом, який слугує контейнером для решти всіх компонентів. App.css є відповідним файлом стилів.

3.2. Робота з даними

Дані отримуються з серверу у вигляді JSON. GET- та POST- запити до серверу здійснюються за допомогою axios.

З серверу можна отримати масив з об’єктами, які є репрезентацією авторів. Ці об’єкти мають такий вигляд:

```
author {
    id,
    msId,
    citationsCount,
    estimatedCitationsCount,
    fullName,
    publicationsCount,
    affiliationMsId,
    affiliationName,
    papers
}
```

Рис. 3.2.1

де:

- id: id в базі даних
- msId: Microsoft-id,
- citationsCount: кількість цитувань автора,
- estimatedCitationsCount: приблизна кількість цитувань автора,
- fullName: повне ім’я автора,
- publicationsCount: кількість публікацій автора,
- affiliationMsId: Microsoft-id організації, де працює автор,
- affiliationName: назва організації, де працює автор,
- papers: роботи автора

3.3. React-компоненти

У директорії src знаходиться папка components. Компонентом називається клас або функція JavaScript, яка може приймати на вхід дані, тобто властивості (props) і повертати елемент React, який описує, як повинен виглядати цей

компонент в інтерфейсі користувача. Схема цієї директорії і компонентів наведена у Додатку Б. Далі дано пояснення до кожного компонента.

В проєкті є такі компоненти:

- AuthorPage — директорія з компонентами для відображення персональної сторінки автора. Компоненти:
  - AuthorData — компонент, props у якого виступає author (автор). Цей компонент представляє собою список з даними про цього автора.
  - JournalsTable — компонент, props у якого виступають papers (роботи конкретного автора). Цей компонент відображає дані у вигляді таблиці з інформацією про журналами і опублікованих автором в них статей.
  - AuthorPage — компонент, який слугує контейнером для AuthorData та JournalsTable. У цьому компоненті здійснюється GET-запит на отримання даних про автора за id і отримання масиву з публікаціями автора за id, які потім відповідно відображатимуться в компонентах AuthorData та JournalsTable.

    Стани цього компоненту визначенні за допомогою “гука” useState():
    - papersList — стан, в якому зберігається масив з об’єктами, які представляють собою інформацію про публікації автора. Початковим станом є пустий масив ([])
    - authorData — стан, в якому зберігається об’єкт з інформацією про автора. Початковим станом є об’єкт:

```
{ “id”: id,
  “msId”: id,
  “citationsCount”: 0,
  “estimatedCitationsCount”: 0,
  “fullName”: “”,
  “publicationsCount”: 0,
  “affiliationMsId”: 0,
  “affiliationName”: “”,
  “papers”: null }
```

- AuthorsTable — директорія з компонентами для відображення таблиці з авторами. Компоненти:
  - AuthorsTable — компонент, в якому здійснюються GET-запити до серверу для отримання масиву з авторами, які потім відображаються у таблиці. Дані відображаються по-сторінково. Їхнє розбиття на сторінки було реалізовано на бекенд-частині, тому сторінка вказується у запиті.

    Стани цього компоненту визначенні за допомогою “гука” useState():
    - arrows — стан стрілок для сортування.
    - showLoader — стан, який має булеве значення і визначає, чи показувати процес завантаження. Початкове значення false.
    - authorsList — стан, в якому зберігається список авторів відповідної сторінки і пошукових запитів відсортованих за вказаним типом і порядком сортування. Початкове значення пустий масив ([]).
    - totalPgs — стан, в якому зберігається загальна кількість сторінок. Початкове значення 1.
    - activePage — стан, в якому зберігається значення сторінки, на яку перейшов користувач.
    - sortingType — стан, у якому зберігається номер для типу сортування даних. Це значення потім використовується при отриманні масиву з авторами. Початкове значення число “0”.
    - sortingOrder — стан, у якому зберігається номер для порядку сортування даних. Це значення потім використовується при отриманні масиву з авторами. Початкове значення число “0”.
    - searchName — стан, у якому зберігається ім’я, за яким здійснюється пошук. Це значення використовується в запиті при отриманні масиву з авторами. Початкове значення порожній рядок (“”).

- searchAffiliation — стан, у якому зберігається назва організації, де працює/ють автор/и, які цікавлять користувача. Це значення використовується в запиті при отриманні масиву з авторами. Початкове значення порожній рядок (“”).

- DropDownMenu — компонент, у якого в якості props виступає {title, items}, де title — назва для випадаючого меню, items — елементи випадаючого меню. Сам компонент являє собою випадаюче меню з елементами, при натисканні на яких виконується деяка задана в item дія.

• Home — компонент, який слугує контейнером для AuthorsTable та NavBar.

• NavBar — компонент навігаційного меню.

3.4. Функціональність веб-застосунку

3.4.1. Головна сторінка

На головній сторінці (рис. 3.4.1.1) відображаються таблиця з авторами та навігаційне меню, а також поля для пошуку автора. Також є випадаючий список з варіантами для ранжування. На даному етапі розробки було реалізоване ранжування за Гіршем.

У користувача є можливість:

- переглянути список авторів по-сторінково.
- здійснити пошук автора/-ів за ім’ям, прізвищем або назвою організації, де він/вони працюють.
- переглянути список усіх наявних авторів або тих, які відповідають параметрам пошуку, відсортованими в порядку спадання/зростання кількості цитат.
- переглянути список усіх наявних авторів або тих, які відповідають параметрам пошуку, відсортованими в порядку спадання/зростання кількості публікацій.

- здійснити ранжування за обраним у випадаючому меню індексом і переглянути список усіх наявних авторів або тих, які відповідають параметрам пошуку, відсортованими в порядку спадання/зростання значення обраного індекса.
- натиснути на ім’я автора, що цікавить, і перейти на його сторінку, де можна переглянути список його опублікованих робіт.

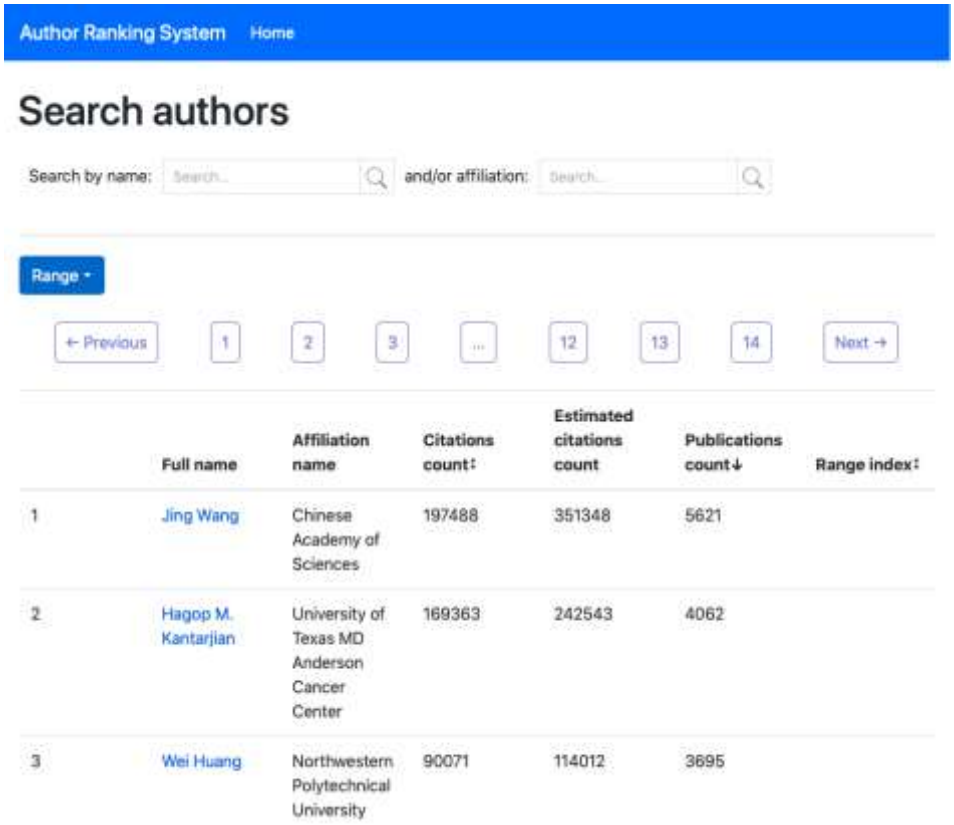


Рис. 3.4.1.1

Список авторів відображається у компоненті AuthorsTable, а з серверу отримується за допомогою GET-запиту

“…/authors/all?page=&name=&affiliation=&orderType=&sortingType=”

Щоб відображався рейтинг автора, у спеціальній функції rangeList() (рис. 3.4.1.2) до кожного елемента додається поле rate, значення якого дорівнює місцю в рейтингу.

```
const rangeList = (list) => {
    for (let i = 0; i < list.length; i++){
        list[i].rate = i+1;
    }
    return list;

}
```

Рис. 3.4.1.2

Розглянемо цей запит більш детально. Значення параметру page вказує на сторінку, за якою хочемо отримати авторів. Поділ даних на сторінки відбувається на сервері. Відповідно до значень параметрів name та affiliation повертаються автори, які мають відповідне ім’я та працюють у потрібній організації. Якщо значеннями цих параметрів є пустий рядок “”, то повертаються усі автори. Значення orderType та sortingType визначають порядок та тип сортування відповідно. Значеннями для sortingType можуть бути: “1” — за кількістю публікацій, “2” — за кількістю цитувань, “3” — за значенням обраного індексу після ранжування за ним. Значеннями для orderType можуть бути: “0” — за спаданням, “1” — за зростанням. Реалізація в інтерфейсі цих функцій наведено в Додатку В.

Цей запит викликається асинхронною функцією fetchAuthors() (рис. 3.4.1.3), якій в якості параметрів необхідно передати значення для сторінки, імені та організації, за якими здійснюється пошук, тип для сортування та його порядку. На виході функція повертає масив об’єктів з потрібними авторами.

```
const fetchAuthors = async (page, name, affil, order, sortType) => {

let res = await api.get( url: "/authors/all?page="+page
    +"&name="+name
    +"&affiliation="+affil
    +"&orderType="+order
    +"&sortingType="+sortType);

res = res.data;
return await res;
}
```

Рис. 3.4.1.3

Щоб результат запиту відобразився на сторінці, необхідно змінити деякі стани компоненту і здійснити рендер. Це відбувається за допомогою “гука” useEffect() (рис. 3.4.1.4)

```
useEffect( effect: () =>{
    const getAuthors = async () => {
        const data = await fetchAuthors(activePage, searchName, searchAffiliation, sortingOrder, sortingType);
        setAuthorsList(data.authors);
        setTotalPgs(data.totalPages);

    }

    getAuthors();

}, deps: [activePage, searchName, searchAffiliation,sortingOrder, sortingType, showLoader])
```

Рис. 3.4.1.4

За допомогою функції map, у таблиці дані про кожного автора з масиву authorsList відображається окремим рядком. Якщо масив порожній (дані досі не отримано) відображається індикатор завантаження. Код реалізації наведено в Додатку Г.

Ранжування відбувається при виклику функції rangeAuthors(). (рис. 3.4.1.5)

```
const rangeAuthors = async (rangeType) => {

    setShowLoader( value: true);
    let payload = { rankingType: '1' };
    try{
        await api.post( url: 'authors/rank?rankingType='+rangeType, payload)
            .then(function(res : AxiosResponse<any> ){
                console.log("ranged in secs:", res)
                setShowLoader( value: false);
            });
    }
    catch(exception){
        console.log(exception)
    }

}
```

Рис. 3.4.1.5

В цій функції відбувається надсилання POST-запиту на сервер, для якого вказується rangeType. Наприклад, якщо значенням для rangeType є “1”, то ранжування відбувається за h-індексом.

3.4.2. Сторінка автора

На сторінці автора відображається загальна інформація про автора, кількість його публікацій та таблиця з його публікаціями, де зазначена їхня назва, назва журналу, в якому вони опубліковані та кількість цитувань. Реалізація в інтерфейсі цих функцій наведено в Додатку Д.

На цій сторінці користувач може:

- переглянути основну інформацію про автора.
- переглянути список публікацій автора.
- повернутися на домашню сторінку через навігаційне меню.

У цьому компоненті використовується змінна “id”, яка отримує значення параметру маршрута (route parameter) і відповідає значенню id автора, який переглядається на сторінці (рис. 3.4.2.1).

```
let { id } = useParams();
```

Рис. 3.4.2.1

Інформація про автора відображається списком атрибутів, дані отримуються за допомогою GET-запиту у асинхронній функції fetchAuthor() (рис. 3.4.2.2), яка повертає об’єкт, що репрезентує автора.

```
const fetchAuthor = async (a_id) => {
    let res = await api.get( url: "authors/"+a_id);
    res = res.data;
    return await res;
}
```

Рис. 3.4.2.2

Інформація про публікації отримуються у функції fetchPapers() (рис. 3.4.2.3), в якій надсилається запит, що повертає публікації автора за його id.

```
const fetchPapers = async (a_id) => {
    let res = await api.get( url: "papers/byAuthor/"+a_id);
    res = res.data;
    return await res;
}
```

Рис. 3.4.2.3

Щоб інформація відобразилася, необхідно використати “гук” useEffect(), в якому викликаємо функції для отримання даних та змінюємо відповідними функціями стани authorData та papersList (рис. 3.4.2.4).

```
useEffect( effect: () =>{
    const getAuthor = async (a_id) => {
        const data = await fetchAuthor(a_id);
        setAuthorData(data);
    }

    const getPapers = async (a_id) => {
        const data = await fetchPapers(a_id);
        setPapersList(data.papers);

    }

    getAuthor(id);
    getPapers(id);

}, deps: [])
```

Рис. 3.4.2.4

Висновки

У курсовій роботі було розроблено фронтенд для системи ранжування авторів за публікаціями та цитуванням за допомогою JavaScript-бібліотеки React. Використовувалися основні особливості даної бібліотеки, як “гуки” і компоненти, а також використовувалася бібліотека для HTTP-запитів axios.

В результаті було отримано фронтенд для веб-застосунку, який відображав усю необхідну для користувача інофрмацію, включно зі списком авторів, відсортованими за кількістю публікацій, кількістю цитат або за обраною бібліометричною метрикою, як h-індекс. Користувач також міг переглядати сторінку окремогоо атвора, де міг побачити таблицю з публікаціями цього автора.

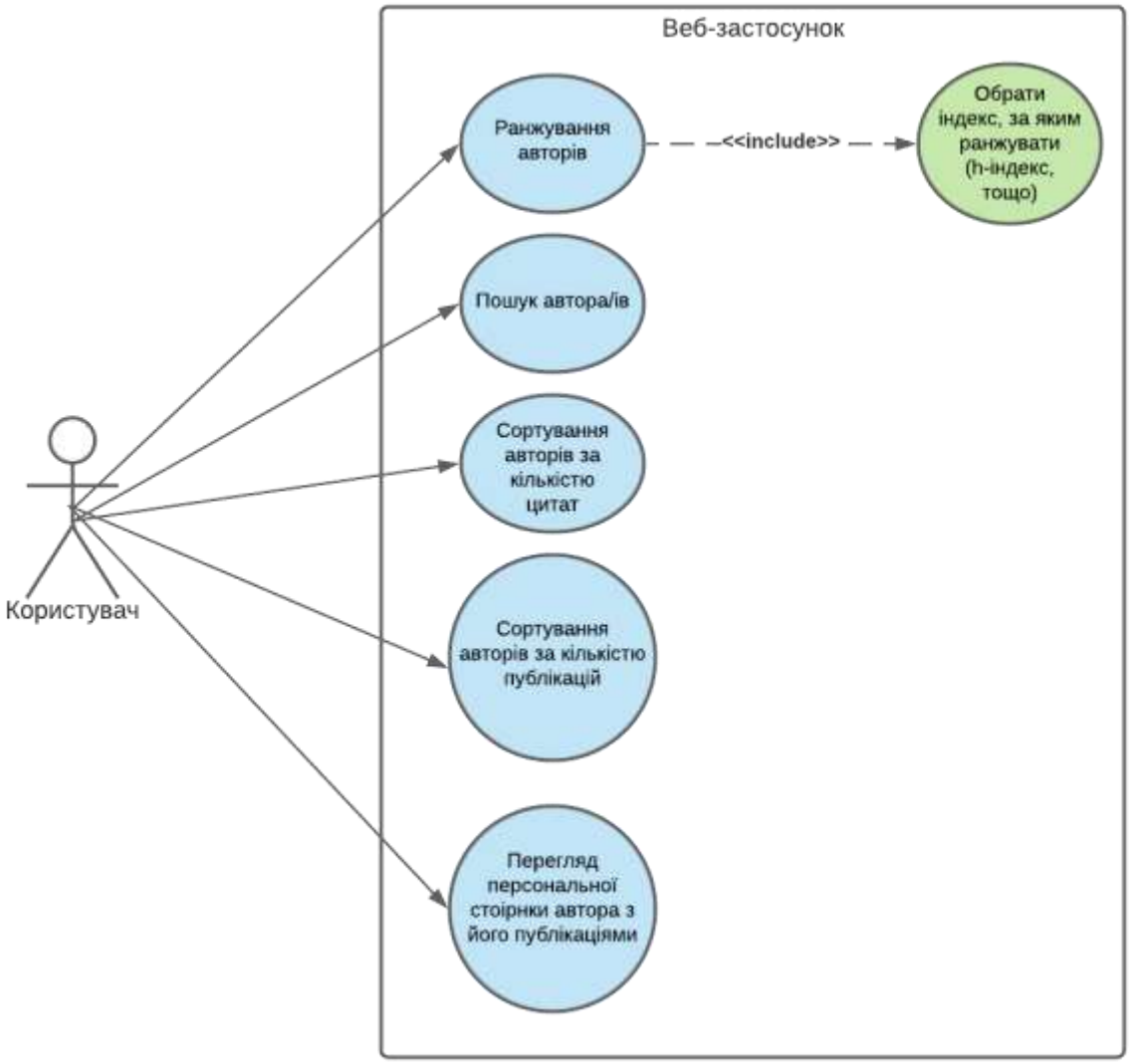
JavaScript-бібліотека React є дуже популярною і цікавою технологією для веб-розробки. Рівень моїх знань цієї технології знаходяться на стадії початківця, але я б хотіла поглибити свої знання цієї технології, щоб надалі застосовувати більш ефективні рішення.

Список використаних джерел

1. Scientific Progress [Електронний ресурс] // Stanford Encyclopedia of Philosophy — 2002 — Посилання: https://plato.stanford.edu/entries/scientific-progress/
2. Бібліометрія та альтернативні метрики [Електронний ресурс] / Олександр Жабін // Центр досліджень соціальних комунікацій — Посилання: http://nbuviap.gov.ua/index.php?option=com_content&view=article&id=3148:bibliometriya-ta-alternativni-metriki&catid=81&Itemid=415
3. Наукометричні показники [Електронний ресурс] // webometr.kpi.ua — Посилання на статтю: https://webometr.kpi.ua/citation-data
4. Факторы ранжирования в поисковых системах [Електронний ресурс] / Ніколай Шмичков // SEOQUICK — 2018 — Посилання на статтю: https://seoquick.com.ua/search-engine-rankings/
5. React: Making faster, smoother UIs for data-driven Web apps [Електронний ресурс] / Пол Крілл // InfoWorld — 2014 — Посилання на статтю: https://www.infoworld.com/article/2608181/react--making-faster--smoother-uis-for-data-driven-web-apps.html
6. Getting Started With The React Hooks API [Електронний ресурс] / Шедрак Акінтайо // Smashing Magazine — 2020 — Посилання: https://www.smashingmagazine.com/2020/04/react-hooks-api-guide/
7. Route Params [Електронний ресурс] / Алекс Сірс // scotch.io fun and practical web development tutorials — Посилання: https://scotch.io/courses/using-react-router-4/route-params
8. Working with Axios in React [Електронний ресурс] / Чезаре Феррарі // DEV community — 2019 — Посилання: https://dev.to/cesareferrari/working-with-axios-in-react-540c

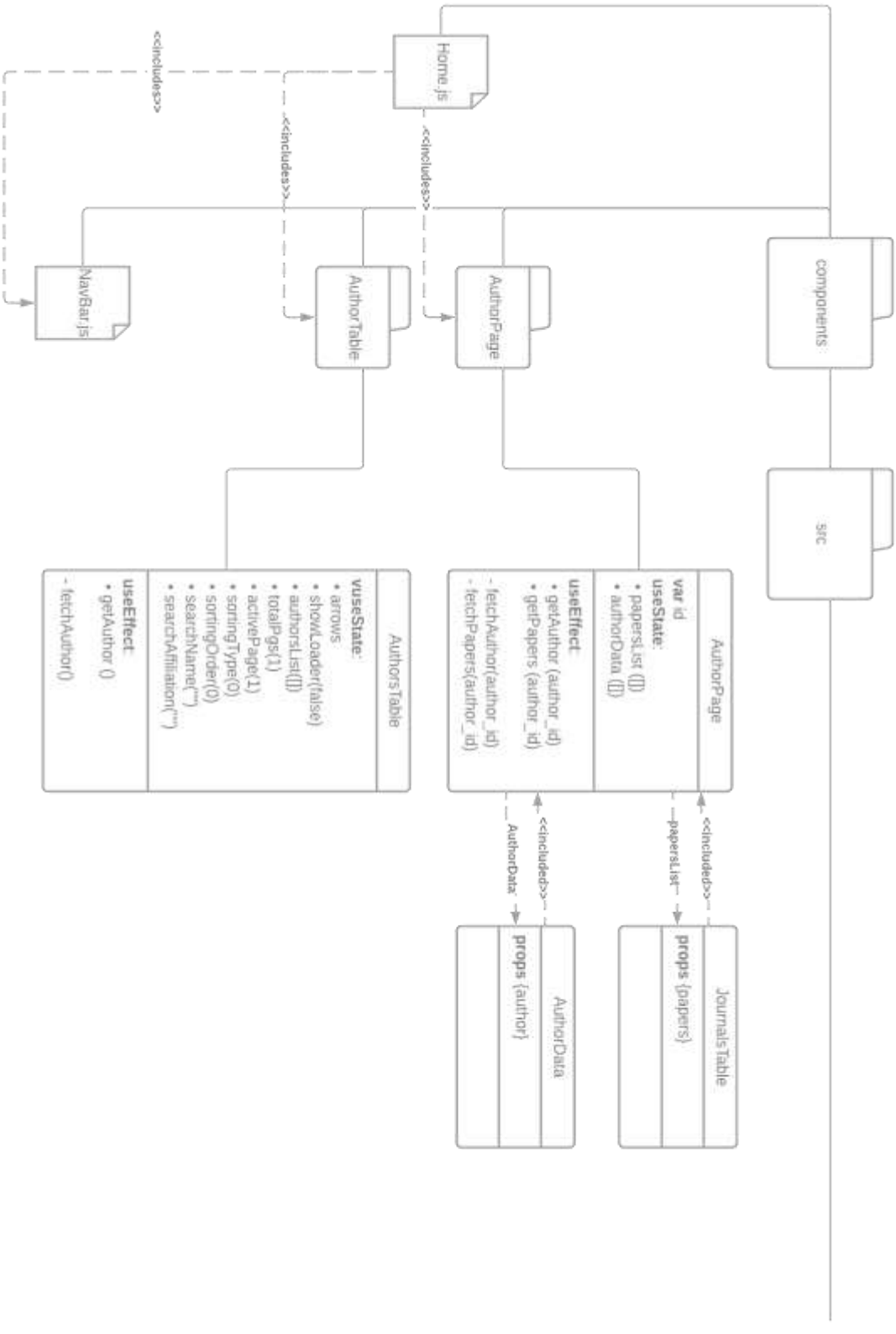
Додаток А

(обов’язковий)
Use-case діаграма веб-застосунку



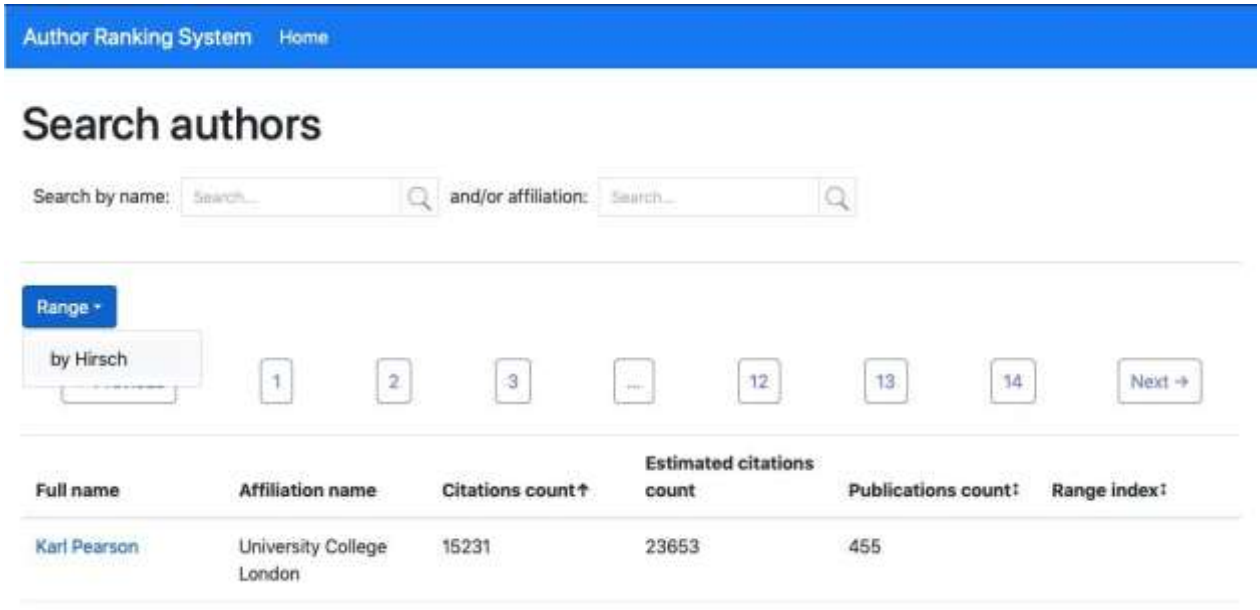
Додаток Б

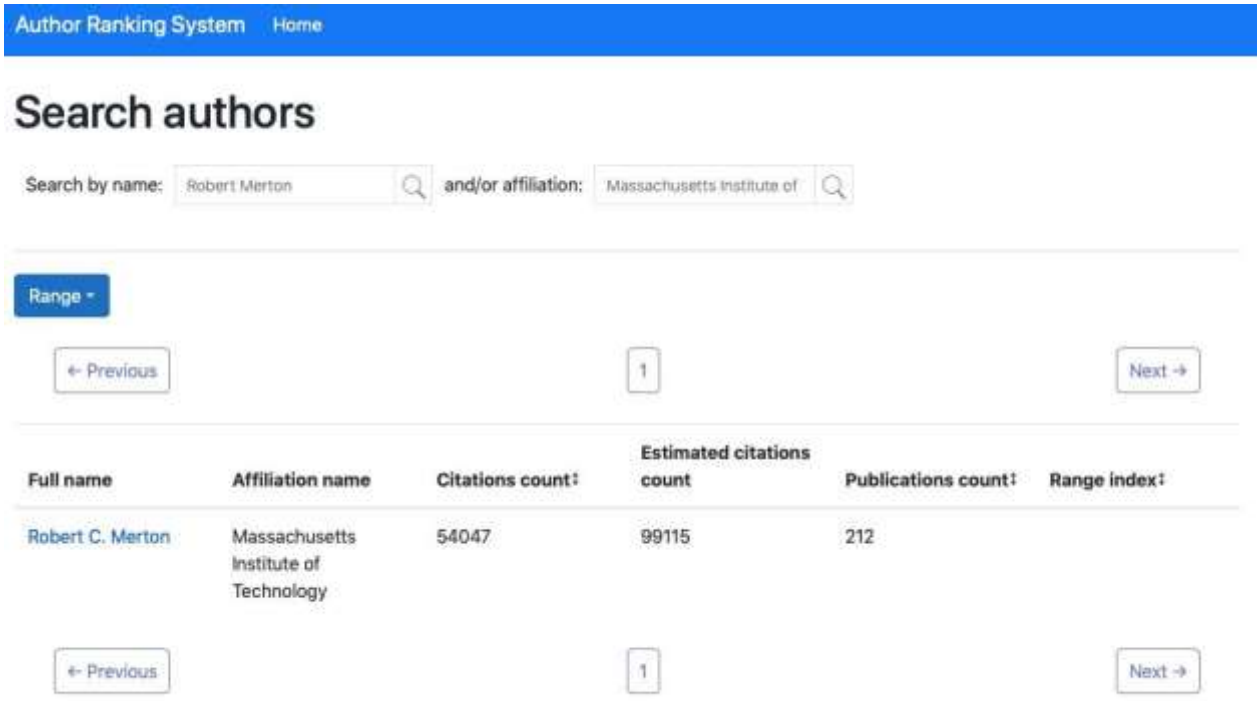
(обов'язковий)
Схема директорії src, всіх її компонентів та взаємозв'язків між ними



Додаток В

(обов'язковий)
Скріншоти інтерфейсу веб-застосунку при виконанні основних функцій на головній сторінці





← Previous 1 2 3 ... 12 13 14 Next →

Full name	Affiliation name	Citations count↑	Estimated citations count	Publications count↕	Range index↕
Karl Pearson	University College London	15231	23653	455	
Martin E. Hellman	Stanford University	20151	28689	74	
David Hilbert	University of Illinois at Chicago	21409	40451	224	
George B. Dantzig	Stanford University	22301	35906	182	
Stanford Moore	Rockefeller University	23833	37395	126	
Richard Courant	New York University	24887	42958	165	
William Shockley	Bell Labs	25911	43513	72	
William G. Cochran	Harvard University	27254	39391	99	
William H. Stein	Rockefeller University	28738	46282	109	
Alfred V. Aho	Columbia University	30468	55375	138	
N. E. Breslow	University of Washington	30539	54417	42	

← Previous 1 2 3 ... 12 13 14 Next →

Full name	Affiliation name	Citations count↓	Estimated citations count	Publications count↕	Range index↕
Douglas G. Altman	University of Oxford	445593	750494	975	
Eric S. Lander	Broad Institute	390600	520818	685	
Walter C. Willett	Harvard University	341132	515203	2412	
Bert Vogelstein	Johns Hopkins University School of Medicine	301825	425419	682	
Oliver H. Lowry	Washington University in St. Louis	286593	299063	194	
Ahmedin Jemal	American Cancer Society	280120	518975	545	
Shizuo Akira	Osaka University	273191	401475	1365	
Robert Langer	Massachusetts Institute of Technology	263214	368864	1781	
A. Lewis Farr	Washington University in St. Louis	259953	260274	2	
Rose J. Randall	Washington University in St. Louis	259255	259255	1	

← Previous 1 2 3 ... 12 13 14 Next →

Full name	Affiliation name	Citations count	Estimated citations count	Publications count↑	Range index
Rose J. Randall	Washington University in St. Louis	259255	259255	1	
Nira J. Rosebrough	Washington University in St. Louis	259255	259255	1	
George W. Snedecor		47895	68473	1	
A. Lewis Farr	Washington University in St. Louis	259953	260274	2	
Susan E B Folstein	NewYork-Presbyterian Hospital	66396	92028	3	
S. M. Sze	National Chiao Tung University	33347	62156	6	
Marion M. Bradford	University of Georgia	198966	235582	10	
Irene A. Stegun		67779	125863	16	
Sidney Siegel		34724	57244	17	
Alex Krizhevsky	Google	76921	203763	19	
Warren Gish	Washington University in St. Louis	95631	127782	28	

← Previous 1 2 3 ... 12 13 14 Next →

Full name	Affiliation name	Citations count	Estimated citations count	Publications count↓	Range index
Jing Wang	Chinese Academy of Sciences	197488	351348	5621	
Hagop M. Kantarjian	University of Texas MD Anderson Cancer Center	169363	242543	4062	
Wei Huang	Northwestern Polytechnical University	90071	114012	3695	
Gregory Y.H. Lip	University of Liverpool	132960	214452	3372	
Pete Smith	University of Aberdeen	116852	182549	3249	
Didier Raoult	Aix-Marseille University	120653	168699	3248	
David Price	University of Manchester	138351	324464	2998	
Patrick W. Serruys	Imperial College London	148325	215197	2973	
Derek Strom	University of Oregon	140360	371518	2957	

← Previous 1 2 3 ... 12 13 14 Next →

Full name	Affiliation name	Citations count↑	Estimated citations count	Publications count↕	Range index↕
Karl Pearson	University College London	15231	23653	455	Ranging...
Martin E. Hellman	Stanford University	20151	28689	74	Ranging...
David Hilbert	University of Illinois at Chicago	21409	40451	224	Ranging...
George B. Dantzig	Stanford University	22301	35906	182	Ranging...
Stanford Moore	Rockefeller University	23833	37395	126	Ranging...
Richard Courant	New York University	24887	42958	165	Ranging...

← Previous 1 2 3 ... 12 13 14 Next →

Full name	Affiliation name	Citations count↕	Estimated citations count	Publications count↕	Range index↑
George W. Snedecor		47895	68473	1	0
Susan E B Folstein	NewYork–Presbyterian Hospital	66396	92028	3	1
Rose J. Randall	Washington University in St. Louis	259255	259255	1	1
Nira J. Rosebrough	Washington University in St. Louis	259255	259255	1	1
S. M. Sze	National Chiao Tung University	33347	62156	6	2
A. Lewis Farr	Washington University in St. Louis	259953	260274	2	2
Marion M. Bradford	University of Georgia	198966	235582	10	5

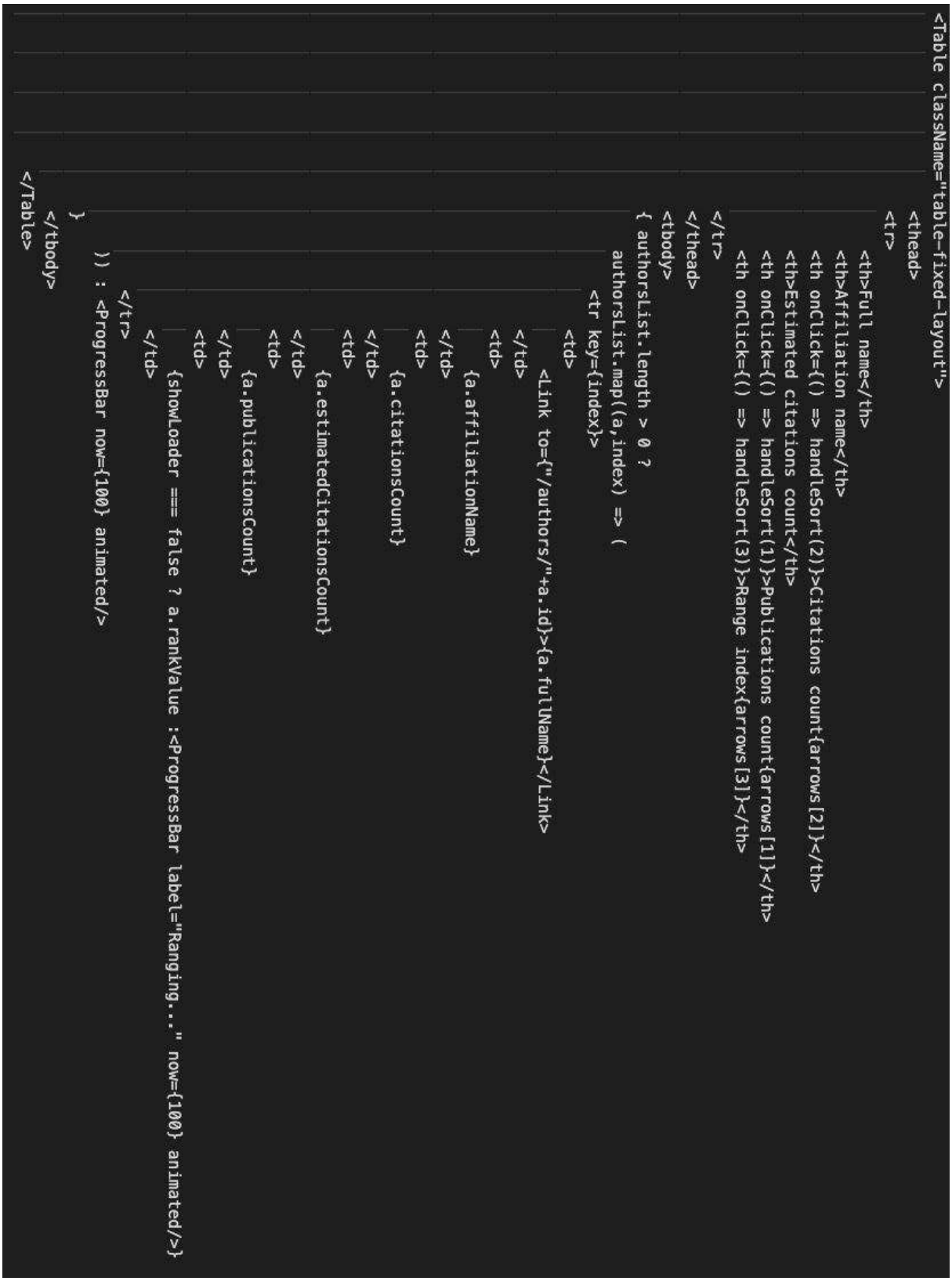
Range

← Previous 1 2 3 ... 12 13 14 Next →

Full name	Affiliation name	Citations count	Estimated citations count	Publications count	Range index
Walter C. Willett	Harvard University	341132	515203	2412	301
Graham A. Colditz	Washington University in St. Louis	214628	326931	1891	298
JoAnn E. Manson	Brigham and Women's Hospital	209851	316146	1961	283
Eric S. Lander	Broad Institute	390600	520818	685	259
Frank B. Hu	Harvard University	189617	282105	1799	259
Meir J. Stampfer	Harvard University	239078	360439	1461	255
Shizuo Akira	Osaka University	273191	401475	1365	243
Ronald C. Kessler	Harvard University	259090	414268	1425	239
Bert Vogelstein	Johns Hopkins	301825	425419	682	238

Додаток Г

(обов’язковий)
Код виведення таблиці з авторами



```
<Table className="table-fixed-layout">
    <thead>
      <tr>
        <th>Full name</th>
        <th>Affiliation name</th>
        <th onClick={() => handleSort(2)}>Citations count{arrows[2]}</th>
        <th>Estimated citations count</th>
        <th onClick={() => handleSort(1)}>Publications count{arrows[1]}</th>
        <th onClick={() => handleSort(3)}>Range index{arrows[3]}</th>
      </tr>
    </thead>
    <tbody>
    { authorsList.length > 0 ?
      authorsList.map((a,index) => (
        <tr key={index}>
          <td>
            <Link to={"/authors/"+a.id}>{a.fullName}</Link>
          </td>
          <td>
            {a.affiliationName}
          </td>
          <td>
            {a.citationsCount}
          </td>
          <td>
            {a.estimatedCitationsCount}
          </td>
          <td>
            {a.publicationsCount}
          </td>
          <td>
            {showLoader === false ? a.rankValue :<ProgressBar label="Ranging..." now={100} animated/>}
          </td>
        </tr>
    )) : <ProgressBar now={100} animated/>
    }
    </tbody>
</Table>
```

Додаток Д

(обов'язковий)
Скріншоти інтерфейсу сторінки автора