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

Розробка системи ранжування редакторів за відгуками авторів (фронтенд, Angular)

Текстова частина до курсової роботи
за спеціальністю „Інженерія програмного забезпечення” 121

Керівник курсової роботи
к.ф.-м.н., Сініцина Р.Б.
(прізвище та ініціали)
_____
(підпис)
“____” _____ 2020 р.
Виконала студентка
Черепина Є.О.
_____
(прізвище та ініціали)
“____” _____ 2020 р.

Київ 2020

Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Кафедра мережних технологій факультету інформатики

ЗАТВЕРДЖУЮ

Зав. Кафедри мережних технологій,
проф.,д.ф.-м.н.

_____ Г.І. Малашонок
(підпис)

“____” _____ 2020 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ
на курсову роботу

Студентці Черепиній Єлизаветі Олексіївні факультету інформатики 4 курсу
ТЕМА: Розробка системи ранжування редакторів за відгуками авторів (фронтенд, Angular)

Вихідні дані:
Клієнтська частина системи ранжування редакторів за критеріями
Зміст ТЧ до курсової роботи:
Вступ
Частина 1: Аналіз предметної області
Частина 2: Теоретичні відомості
Частина 3: Опис реалізації програмного продукту
Висновки
Список використаної літератури
Додатки

Дата видачі „___” _____ 2019 р. Керівник _____
(підпис)
Завдання отримала _____
(підпис)

Зміст

Анотація

Дана робота спрямована на розробку системи ранжування редакторів за відгуками авторів, яка може бути впроваджена в системи у сфері видавничих послуг. Розробка полягала у реалізації фронтенд частини для підключення API ранжування до системи.

Для реалізації роботи використовувався фреймворк Angular та бібліотека UI компонентів Angular Material.

Для реалізації класів було використано мову програмування TypeScript, а для графічного відображення - Angular Material, HTML5, CSS.

Вступ

Рейтингова оцінка є дуже актуальним способом оцінювання, оскільки дає змогу майже одразу отримати об’єктивний фідбек щодо об’єкту оцінювання. Це особливо важливо для оцінки роботи людей. Зокрема – редакторів текстів певного сервісу. Створення системи для ранжування редакторів мало б велике практичне значення, оскільки це дало б змогу формувати рейтингову оцінку редакторів за різними критеріями та аналізувати їхню ефективність роботи та активність.

Мета цієї роботи – дослідити, які є можливі варіанти оцінювання роботи редакторів та варіанти формування моделей ранжування.

Об’єктом дослідження є системи, які вже мають функціонал для обрахування рейтингової оцінки та системи ранжування.

Предмет дослідження – область ранжування об’єктів та надання їм рейтингової оцінки, їх способи , а також область редагування робіт.

Спираючись на отримані результати можна буде сформувати основні критерії оцінки для ранжування редакторів, сформувати власну математичну модель ранжування та реалізувати систему для ранжування редакторів.

Робота складається зі вступу, трьох розділів, висновків, списку використаних джерел та додатків.

РОЗДІЛ 1: Аналіз предметної області. Постановка завдання курсової роботи

1.1 Аналіз сучасного стану питання та обґрунтування теми

На сьогоднішній день ранжування та формування рейтинговою оцінки присутні у всіх сферах життя. Ми ранжуємо підприємства, університети, міста за різними критеріями. Так ми можемо більш об’єктивно оцінити стан питання для кожного об’єкта ранжування. Ранжування – це своєрідний спосіб побудувати рейтингову оцінку, із врахуванням різних критеріїв, в яких враховуються ранги або вагові коефіцієнти кожного з критеріїв.

Вважається, що ранжування є більш об’єктивним методом оцінки, оскільки результат залежить від важливості кожного критерія для певної задачі. Саме тому системи ранжування впроваджуються у різних компаніях, підприємствах, тощо.

Предметна область дослідження – область взаємодії авторів та редакторів, а саме редагування робіт за різною тематикою. Наразі існує багато платних систем, де автори можуть знайти собі редактора для своєї роботи. Такі системи є свого роду marketplace -ми, що робить контролювання взаємодії учасників системи вкрай важливим.

Редагування текстів існує давно, люди зазвичай розділяють етапи реалізації думки «на папері» та вичитування тексту, зокрема діляться відповідальністю з іншими людьми, в нашому випадку- редакторами.

Адміністратору системи вкрай необхідно перевіряти активність редакторів, які є у системі. Оскільки у таких системах основний дохід йде від авторів - їхня думка має бути в пріоритеті. Зазвичай у подібних системах є можливість для авторів оцінити виконану роботу редактором.

А для того, щоб правильно використати цю інформацію, сформувати рейтингову оцінку та оцінити, як працюють редактори потрібна саме модель ранжування.

Інформації щодо того, чи використовують подібні системи у своїй роботі моделі ранжування не було знайдено у відкритому доступі. Саме тому, можна зробити припущення, що розробка подібної системи та подальше її впровадження на ресурси для редагування текстів буде актуальною.

1.2 Огляд існуючих аналогів розробки

Прямих аналогів розробки під час аналізу предметної області не було знайдено, саме тому було проаналізовано системи, що виконають схожі завдання, але належать до іншої предметної області. Серед них розглянемо IMDb (Internet Movie Database) – найбільшу базу даних про кінематограф. Основну функцію, яку буде розглянуто в цій системі - формування рейтингу фільмів.

IMBd формує різні рейтинги фільмів за кількістю та якістю голосів. Дана система враховує не тільки негативний або позитивний відгук відвідувача під час обрахування рейтингової оцінки, але й їх співвідношення. Це дає змогу формувати «об’єктивні» рейтингові оцінки для фільмів. Окрім цього, враховуються голоси лише користувачів, які зареєстровані в системі та голосують постійно.

Щоб уникнути «накручування» рейтингової оцінки, оцінка самого фільму та оцінка, яка бере участь у формуванні рейтингу, наприклад, «250 найкращих фільмів всіх часів», буде відрізнятися.

Для формування рейтингу «250 найкращих фільмів всіх часів» IMBd використовує справжню оцінку Баєса, про це сказано безпосередньо на одній сторінці з самим рейтингом з описанням даних, які використовуються.

Ще одним прикладом ранжування загалом - світовий рейтинг університетів «QS World University Rankings». Цей рейтинг використовує експертні дані у 2011 від вчених та академіків та передбачає оцінку наукової діяльності університету, виходячи з кількості цитувань статей вузів в базі даних Scopus, співвідношення кількості викладачів та студентів, кількості міжнародних співробітників та студентів.

Methodology of QS World University Rankings[23]

Indicator	Weighting	Elaboration
Academic peer review	• 40%	Based on an internal global academic survey
Faculty/Student ratio	• 20%	A measurement of teaching commitment
Citations per faculty	• 20%	A measurement of research impact
Employer reputation	• 10%	Based on a survey on graduate employers
International student ratio	• 5%	A measurement of the diversity of the student community
International staff ratio	• 5%	A measurement of the diversity of the academic staff

Рис. 1 : Методологія QS World University Rankings

Фактично, як ми можемо побачити у даній таблиці, визначено критерії та їх вагу у відсотковому відношенні відповідно.

В обох наведених прикладах, задання критеріїв, їхнє оголошення дає більш об’єктивну оцінку, та психологічно вселяє більше впевненості у її достовірності.

1.3Постановка задачі

Після проведеного дослідження та з огляду на важливість

ранжування у формуванні рейтингової оцінки, було сформовано задачу для розробки продукта.

Основна задача - розробити модель ранжування редакторів та реалізувати фронтенд частину для демонстрації роботи API ранжування, з можливістю редагувати пріоритетність критеріїв та додавати додаткові. Для цього було визначено послідовність дій :

1. Визначити критерії оцінки редакторів авторами.
2. Визначити додаткові критерії оцінювання редакторів.
3. Сформувати модель обрахування рейтингової оцінки
4. Продумати дизайн системи для ранжування
5. Розробити систему для відображення результатів ранжування та кастомізації моделі ранжування, де реалізувати :
    a. Надання пріоритетності основних критеріїв ранжування
    b. Додавання додаткових критеріїв для формування оцінки з можливістю задати їм пріоритетність
    c. Можливість обрання галузі та підгалузей, в межах якої(их) повинно відбуватися ранжування
6. Протестувати програму

РОЗДІЛ 2: Теоретичні відомості

Для розробки моделі ранжування редакторів за відгуками авторів було умовно виділено 2 області дослідження. Зокрема, це дослідження, за якими критеріями оцінюється робота редакторів, та варіанти формування рейтингової оцінки.

Оскільки прямих аналогів розробки системи не було знайдено, було проаналізовано оцінювання роботи редакторів в інших системах.

Це допомогло виділити 3 основних критерії оцінки :

- Якість роботи
- Комунікація під час роботи
- Швидкість виконання роботи

Критерій «Якість роботи» є комплексним, його не можна оцінити загально. Для вирішення цього питання було проаналізовано ресурси копірайтингу та редагування наукових робіт. Дані ресурси виділяли саме критерії оцінки тексту, серед яких було обрано наступні для впровадження у системі :

- дотримання граматичних норм
- відповідність тексту галузі та підгалузі
- збереження смислової цілісності
- зв’язність тексту
- структурованість тексту
- інформаційна насиченість

Дані критерії оцінки було обрано як підкритерії критерія «Якість роботи». Передбачається, що автор виставляє оцінку від 1 до 5 за кожним підкритерієм для оцінювання роботи редактора, а система, використовуючи геометричне середнє, вираховує оцінку загального критерія «Якість роботи». Оскільки, ми не можемо обрати пріоритетність для кожного з

підкритеріїв, геометричне середнє обрахує приблизну середню оцінку серед усіх підкритеріїв.

Наступним критерієм, який було виділено для формування оцінки редактора – «Комунікація під час роботи». Даний критерій - це суб'єктивна оцінка самого автора, наскільки гарна була взаємодія з боку редактора для вирішення питань щодо роботи. Вимірюється від 1 до 5.

Останнім критерієм було виділено «Швидкість виконання», який є комплексним та складається з кількох складових.

Перша складова - суб'єктивна оцінка автора, наскільки швидко, за його думкою, було виконано роботу. Вимірюється від 1 до 5.

Друга складова – оцінка швидкості виконання роботи системою. На вхід подаються дані про обсяг роботи та час , який було витрачено на виконання роботи. Оскільки важко серед усіх можливих результатів достовірно сказати, який є гарним, а який ні, було вирішено взяти еталоне значення швидкості, та порівнювати з ним. Було досліджено, що середній час вичитування та редагування тексту становить 15-20 сторінок на день. Все, що більше або дорівнює цій швидкості набуває значення 1, а все що менше – 0. Таким чином, редактори, які швидко виконували роботи матимуть вищу рейтингову оцінку.

Третя складова – коефіцієнт галузі. Дана складова є опціональною. Її дефолтне значення– 0. При включенні складової до ранжування для кожної галузі встановлюється свій ваговий коефіцієнт, що впливає на оцінку за роботи різних галузей.

Як було сказано, критерій «Швидкість виконання» є комплексним, його обрахування відбувається за формулою :

$V = (Rate * (V_{system} + k_{area})$,

де *Rate – оцінка від автора, [1;5];*

V – швидкість, обрахована системою, {0;1};

k_{area} *– ваговий коефіцієнт галузі;*

Для кожного з трьох основних критеріїв встановлюються вагові коефіцієнти, що дає можливість задати їм пріоритетність та відранжувати редакторів за нею. Вираховуються вагові коефіцієнти за формулою :

$$k_i = \frac{h_i}{\sum_{i=1}^{m} h_i}$$

де h_i – ранг, який отримав і-ий критерій

m – кількість критеріїв

Для отримання рейтингової оцінки побудуємо її у вигляді зваженої суми локальних критеріїв :

$$R_j = \sum_{i=1}^{m} k_i X_{ij}$$

де k_i – ваговий коефіцієнт

X_{ij} – і-й критерій j-го редактора

Застосувавши формулу, ми отримаємо зважену суму всіх критеріїв. Але це буде не об’єктивна оцінка.

Розглянемо два випадки. Припустимо, що є один редактор, який має лише 1 виставлену оцінку, яка дорівнює 5, тоді його середній рейтинг буде 5. У іншого редактора – 20 різних виставлених оцінок та його середній рейтинг – 4,9. Другий редактор матиме більш об’єктивну оцінку, аніж перший, оскільки його оцінили в 20 раз більше людей. Саме тому, під час формування рейтингової оцінки важливо враховувати не тільки середню

оцінку, а й кількість кожної отриманої. Для вирішення цього завдання було проаналізовано декілька методів для реалізації зваженого рейтингу.

Як було зазначено під час аналізу конкурентів, один із найвідоміших ресурсів оцінювання фільмів IMDb для обрахування рейтингової оцінки використовує формулу на основі справжньої оцінки Баєса :

$W = \frac{RV+CM}{V+M}$, *де*

V – кількість відданих голосів за фільм,

M – мінімальна кількість голосів для включення в рейтинг,

R – середня оцінка фільму (за десятибальною шкалою),

C – середня оцінка серед усіх фільмів.

Потрібно зауважити, що результат – є зваженим арифметичним середнім середньої оцінки фільму та середньої оцінки серед усіх фільмів із вектором ваг (V,M). За таким обрахуванням рейтинг фільмів, які мають досить маленьку кількість оцінок, буде зважений в бік *C*, водночас рейтинг фільмів з великою кількістю оцінок – в бік *R*.

Для ранжування редакторів в системі для слідкування їх успішності буде неправильно задавати мінімальну кількість голосів для включення в рейтинг (приклад - IMDb на грудень 2012 року встановило M = 25,000), оскільки тоді результати ранжування матимуть недостовірний результат. Середня оцінка серед усіх фільмів також задається експертами та може набувати різних значень.

Зважаючи на факт, що ми не можемо передбачити кількість редакторів, яких потрібно буде ранжувати, та визначати середню оцінку серед усіх фільмів, робимо висновок, що справжня оцінка Баєса не підходить для впровадження у системі ранжування редакторів.

Один з інших варіантів сортування – нижня границя довірчого інтервалу Уілсона, який базується на позитивних та негативних відгуках.

Довірчий інтервал - це інтервал, у межах якого з заданою довірчою імовірністю можна чекати значення оцінюваної (шуканої) випадкової величини.

Ідея полягає в тому, щоб розглянути набір рейтингів об'єктів ранжування як статистичну вибірку гіпотетичного набору рейтингів об'єктів, а потім використовувати цей показник. Нам потрібно збалансувати частку позитивних оцінок із невизначеністю невеликої кількості спостережень. Довірчий інтервал розглядає біноміальний розподіл для підрахунку рейтингу.

$$\left(\hat{p} + \frac{z^2_{\alpha/2}}{2n} \pm z_{\alpha/2}\sqrt{\left[\hat{p}(1-\hat{p}) + z^2_{\alpha/2}/4n\right]/n}\right) / \left(1 + z^2_{\alpha/2}/n\right)$$

Рис. 2: Нижня границя інтервалу Уілсона

Варто зауважити, що у формулі враховуються позитивний або негативний відгук, але формулу можна також використовувати для К-того рейтингу, задавши інтервали , що відповідали б за негативні та позитивні оцінки.

Проблема даного методу обрахунку полягає у тому, що цей інтервал надає значення 0 як для об'єктів, які ще не оцінювалися, так і для об'єктів, які мають лише негативні відгуки. Оскільки система ранжування редакторів повинна розрізняти не оцінених редакторів та погано оцінених редакторів, можна зробити висновок, що довірчий інтервал Уілсона не підходить.

Для сортування редакторів із врахуванням кількості оцінок було обрано метод, схожий на довірчий інтервал Уілсона та пов’язаний зі справжньою оцінкою Баєса– нижня границя нормального наближення імовірного інтервалу Баєса для середнього рейтингу.

Імовірний інтервал – це інтервал для визначення апостерірного розподілу ймовірності. Він є аналогічним довірчому інтервалу у частотній статистиці, хоча й мають певні відмінності. Імовірні інтервали трактують свої межі як фіксовані, а параметр оцінювання як випадкову змінну. Довірчі – навпаки, трактують межі як випадкові змінні, а параметр – фіксований.

$$s(n_1, \dots, n_k) = \sum_{k=1}^{K} s_k \frac{n_k + 1}{N + K} - z_{\alpha/2} \sqrt{\left(\left(\sum_{k=1}^{K} s_k^2 \frac{n_k + 1}{N + K} \right) - \left(\sum_{k=1}^{K} s_k \frac{n_k + 1}{N + K} \right)^2 \right) / (N + K + 1)}$$

Рис. 3 : Баєсове наближення

де $z\alpha/2$ - це $1-\alpha/21-\alpha/2$ квантиль для нормального розподілу.

Під час підрахування тестових значень, було зроблено висновок, що нижня границя нормального наближення імовірного інтервалу Баєса розрізняє об’єкти, які ще не мають оцінку та ті, що мають погану оцінку, що є важливим фактором для ранжування, оскільки система ранжування редакторів – форма слідкування за їх активністю та повинна розрізняти останніх.

Отже, за допомогою даної формули, ми зможемо обрахувати рейтингову оцінку редакторів, яка буде спиратися на кількісне відношення оцінок серед усіх оцінок редактора.

Таким чином, ми отримаємо математичну модель ранжування реакторів.

РОЗДІЛ 3: Опис реалізації програмного продукта

3.1 Аналіз технічного завдання

Конкретизуємо поставлену нами задачу. Передбачається, що в даній роботі розглядається лише відображення існуючих даних, тобто вважаємо, що є певна система, де відбувається оцінювання редакторів авторами. Ми працюємо лише з даними, які нам було надано.

Основна задача – розробити кастомізовану систему формування моделі ранжування редакторів, яка на вході мала б критерії оцінки, а на виході – відранжований список редакторів за заданими критеріями.

Дана система має бути прикладом реалізації клієнтської частини для підключення API ранжування.

Система повинна мати панель критеріїв, список редакторів з оцінками та панель пошуку.

Серед основних можливостей система повинна мати у панелі критеріїв перелік основних критеріїв оцінювання, можливістю додати додаткові критерії, та виставити пріоритетність як для обов’язкових так і для необов’язкових критеріїв. Пріоритетність повинна виставлятися за допомогою так званих списків пріоритетності. Система повинна передбачати можливість встановити однаковий пріоритет для декількох критеріїв. Окрім цього, панель повинна мати перелік усіх галузей та підгалузей, представлений у вигляді дерева для можливості обрання галузей, серед яких виконувати ранжування.

Система повинна мати панель пошукового запиту для можливості пошуку серед колекції редакторів.

Інформація про редакторів повинна бути представлена у вигляді таблиці. Окрім цього, повинна бути пагінація сторінок, з можливістю

перейти на наступну сторінку пагінації та/ або змінити параметри пагінації – скільки редакторів відображати на 1 сторінці.

При зміні будь-якого з критеріїв система повинна автоматично оновлювати таблицю з новим рейтинговими оцінками.

3.2 Обґрунтування алгоритму й структури програми

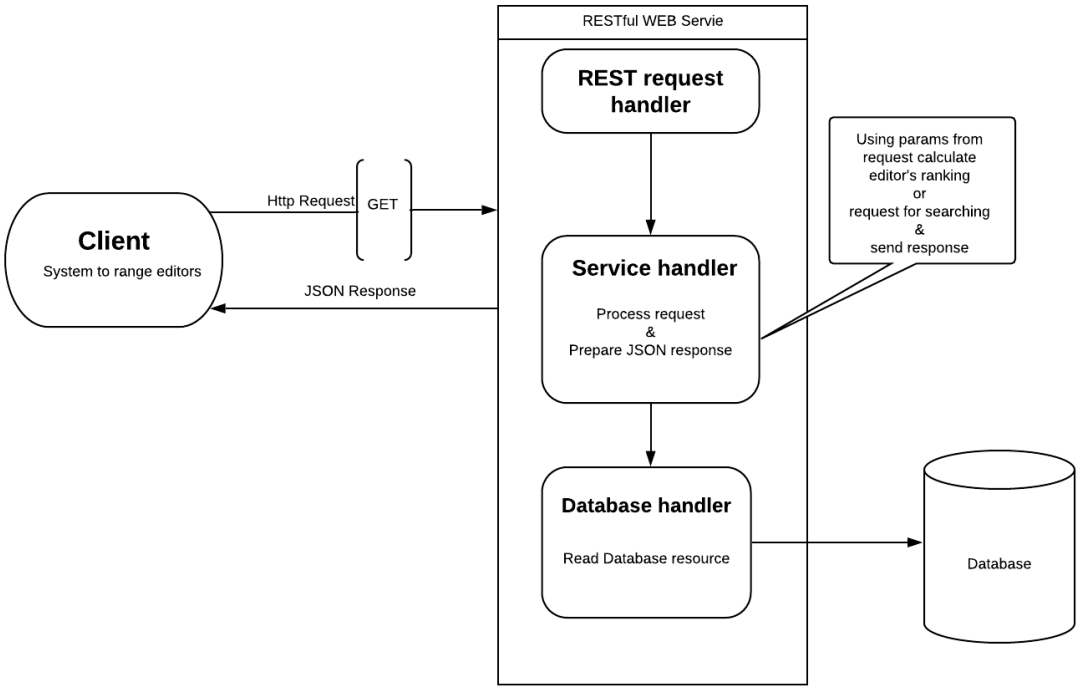

Рис. 4 : Структура роботи системи

Метод передачі та отримання даних, який представлений на діаграмі є найнадійнішим , оскільки важливо розділяти фронтенд та бекенд частину системи. Окрім цього, API надасть можливість надавати результати обчислення різним системам, просто підключивши його. Як приклад підключення даного API є розроблена клієнтська частина в даній курсовій роботі.

3.3 Обґрунтування вибору засобів розробки

Основним засобом розробки було обрано фреймворк з відкритим кодом Angular 9, оскільки він має багато переваг.

По-перше, будь-який застосунок на Angular свою певну логіку, а саме дерево компонентів. Це дозволяє відокремлювати частини коду. Така архітектура є самодостатньою, саме тому компоненти можна використовувати в інших застосунках, або впроваджувати готові рішення в тій же системі, але в іншому місці.

Angular – це оптимальний варіант для створення SPA (single-page apps), оскільки браузер перезавантажує не всі сторінку, а лише окремі компоненти, з якими взаємодіє користувач.

Однією з концепцій фреймворку є Dependency Injection – шаблон проектування для обмежування сильного зв’язування. Компоненти як класи, в яких є UI та сервіси – як класи, які містять аплікаційну логіку. Ми пишемо клас сервісу, а фреймворк вставляє екземпляр цього класу в компонент.

Насамкінець, Angular – це повноцінний фреймворк, із підтриманням швидкого вимальовування сторінок, навігацію, різні готові компоненти, DI, тощо. Це дає йому велику перевагу серед інших існуючих фреймворків.

Під час виконання роботи також використовувалась бібліотека Angular Material ,що містить в собі готові компоненти, розроблені відповідно до специфікації Google Material Design. За допомогою неї були створені всі основні компоненти в розробленій системі. Це дуже зручно, оскільки це зменшує кількість коду, та збільшує його чіткість.

Середовище розробки – VS Code , було обрано для простої взаємодії бекенда та фронтенда, оскільки середовище є багатомовне і підтримує всі використані засоби розробки.

3.4 Опис розробки програми

Для реалізації поставленої задачі було розроблені компоненти, моделі та сервіси.

Спочатку було створено компоненти, кожен з яких – це набір файлів для роботи з певною структурою програми, зокрема туди входять html-складова , складова стилів та клас компонента, де описані методи для його роботи.

Після того, як були визначені основні компоненти було створено моделі. Під час взаємодії клієнтської частини з API, а саме відправка запитів та отримання відповіді , дані, що надійшли потрібно приводити до певного типу. Саме для цього створюється інтерфейс з описом властивостей об’єкту.

Для взаємодії з API було створено сервіси – в них описується методи для звернення до серверу із запитом отримання даних.

3.5 Створення об’єктів і розробка головної програми

Серед компонентів було створено наступні :

- *«criteria-panel»*

Цей компонент є відповідальним за панель критеріїв ранжування та є основним у створенні моделі ранжування. У цьому компоненті створюються основні критерії для ранжування редакторів та реалізовується їх функціонал.

Зокрема, це можливість задати пріоритетність для 3-х основних критеріїв ранжування. Цю можливість було реалізовано за допомогою DropLists, що дає змогу користувачу просто розставити критерії із певним пріоритетом, просто перетягуючи їх. З боку системи приймається виставлений ранг критерія та обраховується за ним ваговий коефіцієнт. За

таким же методом реалізовано надання пріоритетності галузям, якщо користувач хоче додати їх до ранжування.

Ще одною складовою панелі критеріїв є дерево галузей. За допомогою нього користувач може обрати, в якій галузі та/або підгалузі ранжувати редакторів. Дерево галузей винесено в окремий компонент *«area-tree»*.

- *«area-tree»*

У цьому компоненті реалізовується можливість переглядання та обрання галузей та підгалузей у вигляді Tree Checkbox нод-галзей/підгалузей. У класі «*area-tree*» описані методи для перевірки, які ноди було позначено у цьому дереві, а також перевизначено представлення даних за допомогою моделі, що повертаються за запитом з сервера.

- *«editors-list»*

Компонент *«editors-list»* містить у собі реалізацію представлення інформації про редакторів. Після кастомізації моделі ранжування (обрання критеріїв, галузі, тощо) приходить відранжований список редакторів із вказаним рейтингом. Дані представляються у вигляді таблиці, що містить основну інформацію про редактора, зокрема, це ім’я , e-mail, галузі, у яких працює редактор, його рейтинг. Таблиця має пагінатор та можливість пошуку серед колекції.

Серед моделей було створено :

- *«editors_areas»*

Містить у собі інтерфейс представлення редакторів

- *«areas»*

Містить у собі інтерфейс представлення галузей з підгалузями

Серед сервісів було створено :

- *«editors.service»*

Описує методи із запитами до серверу для передачі параметрів оцінювання та отримання списку редакторів

- *«areas.service»*

Описує методи із запитами для отримання переліку всіх галузей та підгалузей

3.6 Опис файлів даних та інтерфейсу програми

Як джерело вхідних даних система отримає відповідь на Http запити, які описані у сервісах програми. Дані приходять у вигляді Json-об’єктів. Для роботи з ними використовуються моделі, за допомогою яких Json-об’єкт приводиться до інтерфейсу моделі.


```
{
    "idArea": 1,
    "areaTitle": "Business and Economics",
    "subareas": [
        {
            "idSubarea": 1,
            "subareaTitle": "Business Studies and\nManagement",
            "fkArea": 0,
            "area": null,
            "subSubareas": [
                {
                    "idSsubarea": 1,
                    "ssubareaTitle": "Marketing",
                    "fkSubarea": 0,
                    "subareasIdSubareaNavigation": null,
                    "editorsHasSubSubareas": []
                },
                {
                    "idSsubarea": 2,
                    "ssubareaTitle": "Business Studies",
                    "fkSubarea": 0,
                    "subareasIdSubareaNavigation": null,
                    "editorsHasSubSubareas": []
                }
            ]
        },
        {
            "idSubarea": 2,
            "subareaTitle": "Economics",
            "fkArea": 0,
            "area": null,
            "subSubareas": []
        }
    ]
}
```


Рис. 5 : Приклад отриманих даних

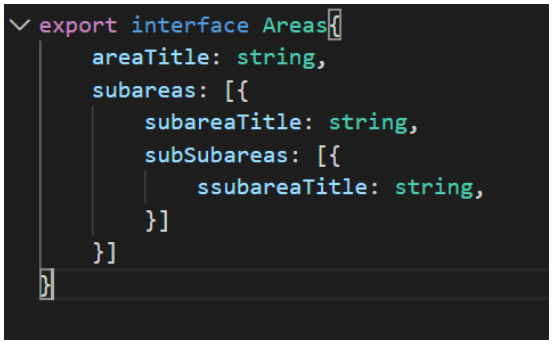

Рис. 6 : Приклад інтерфейсу для роботи з отриманими даними

Висновки

У даній роботі було реалізовано всі пункти поставленого завдання – від дослідження до розробки системи.

Було проаналізовано багато способів обрахування рейтингової оцінки для ранжування редакторів та обрання критеріїв. В цьому допомогли зокрема схожі системи та їхній підхід до вирішення завдання.

Це допомогло визначитися з критеріями оцінки редакторів , як обов’язкових так і додаткових. Було розглянуто також варіанти формування рейтингової оцінки. Зокрема, як враховувати не тільки саму оцінку, а й її кількісне відношення серед усіх інших оцінок. На основі цих даних було розроблено математичну модель формування рейтингової оцінки редакторів.

Був продуманий простий інтерфейс відображення та роботи з даними та розроблена сама система. Вона підтримує можливість кастомізації критеріїв, зокрема надання обов’язковим критеріям пріоритетність, можливість врахування додаткового критерія (галузі), та обрання, серед яких галузей потрібно ранжувати редакторів.

Оскільки, наразі редагування текстів, можливо поєднане з перекладом, має величезний попит на ринку, будуть користуватися попитом і системи для контролю самих редакторів. Це ще раз підтверджує необхідність розробки даного типу системи.

Список використаної літератури

1. https://www.cwauthors.com/
2. https://studme.org/50989/menedzhment/metod_zadannogo_raspredeleniya
3. https://uk.wikipedia.org/wiki/%D0%91%D0%B0%D1%94%D1%81%D0%BE%D0%B2%D0%B0_%D0%BE%D1%86%D1%96%D0%BD%D0%BA%D0%B0
4. https://cyberleninka.ru/article/n/vektornyy-metod-reytingovoy-otsenki/viewer
5. https://www.etxt.ru/subscribes/kak-ocenit-tekst-6-obyazatelnyh-pokazateley-kachestva/
6. https://www.evanmiller.org/ranking-items-with-star-ratings.html#examples
7. https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/36265.pdf
8. https://fulmicoton.com/posts/bayesian_rating/
9. https://medium.com/tech-that-works/wilson-lower-bound-score-and-bayesian-approximation-for-k-star-scale-rating-to-rate-products-c67ec6e30060
10. https://www.evanmiller.org/how-not-to-sort-by-average-rating.html
11. https://bayes.wustl.edu/etj/articles/confidence.pdf
12. https://uk.m.wikipedia.org/wiki/%D0%86%D0%BC%D0%BE%D0%B2%D1%96%D1%80%D0%BD%D0%B8%D0%B9_%D1%96%D0%BD%D1%82%D0%B5%D1%80%D0%B2%D0%B0%D0%BB
13. https://uk.m.wikipedia.org/wiki/%D0%94%D0%BE%D0%B2%D1%96%D1%80%D1%87%D0%B8%D0%B9_%D1%96%D0%BD%D1%82%D0%B5%D1%80%D0%B2%D0%B0%D0%BB
14. https://www.sciencedirect.com/topics/mathematics/credible-interval
15. http://jmlr.csail.mit.edu/papers/volume12/weng11a/weng11a.pdf

16. http://docplayer.ru/47058887-Opredelenie-rangov-v-metode-ranzhirovaniya.html
17. https://en.wikipedia.org/wiki/QS_World_University_Rankings

Додаток А. Програмний код

Criteria-panel.component.html

```
<mat-drawer-container class="criteria-container">
  <mat-drawer class="criteria-sidenav" mode="side" opened fxLayout="column">
    <div fxLayout="column">
      <mat-card>
        <mat-label class="ranking-label"
        fxLayoutAlign="center" ><h2>Ranking Criteria</h2></mat-label>

      </mat-card>


        <mat-expansion-panel [expanded]="true">
          <mat-expansion-panel-header>
            <mat-panel-title>
              <div class="criteria-name" fxLayout="row" >
                <h2>Order Box</h2>
                <div class="info">
                  <mat-icon  matTooltip="Set a priority for the criteria">info_outline</mat-icon>
                </div>

              </div>
            </mat-panel-title>
          </mat-expansion-panel-header>
          <div cdkDropListGroup>
            <div class="example-container">

              <div *ngFor="let o_list of criteriaList; index as i; last as isLast" >
                <div class="example-container" >
                  <!-- <div *ngIf="isLast; else elseBlock"><h3>All</h3></div> -->
                  <!-- <ng-template #elseBlock><h3>{{i+1}}</h3></ng-template> -->
                  <h3>{{i+1}}</h3>
                  <!-- <div fxLayoutAlign="center"> -->
```

```
                    <div>
                      <div
                        cdkDropList
                        [cdkDropListData]="criteriaList[i]"
                        class="order-list"
                        (cdkDropListDropped)="dropCriteria($event)"

                        >
                        <!-- [cdkDropListEnterPredicate]="evenPredicate" -->
                        <!-- <div class="order-box" *ngFor="let key of objectKeys(o_list)" cdkDrag
>{{o_list[key]}}</div> -->
                        <div class="order-box" *ngFor="let item of o_list" cdkDrag >{{item}}</div>

                      </div>
                    </div>
                  </div>
                </div>
              </div>
            </mat-expansion-panel>


            <mat-expansion-panel [expanded]="true">
              <mat-expansion-panel-header>
                <mat-panel-title>
                  <div class="criteria-name"fxLayout="row" >
                    <h2 >Areas</h2>
                    <mat-icon matTooltip=
                    "Set checked to include in ranking.If checked - set the priority for the areas">
                      info_outline
                    </mat-icon>
                  </div>
                </mat-panel-title>
```

```
            </mat-expansion-panel-header>
            <mat-checkbox matTooltip="Include in Ranking" [(ngModel)]="checked"> Include
Areas</mat-checkbox>
            <div cdkDropListGroup>
              <div class="example-container">

                <div  >

                  <div class="example-container" >
                    <!-- <div *ngIf="isLast; else elseBlock"><h3>All</h3></div> -->
                    <!-- <ng-template #elseBlock><h3>{{i+1}}</h3></ng-template> -->
                    <!-- <div fxLayoutAlign="center"> -->

                    <div>
                      <div
                        cdkDropList
                        [cdkDropListData]="order_area_required"
                        class="order-list"
                        (cdkDropListDropped)="drop($event)"

                        >

                        <div class="order-box" *ngFor="let item of order_area_required; index as i"
cdkDrag >{{item.title}}</div>
                      </div>
                    </div>
                  </div>
                </div>
              </div>

            </mat-expansion-panel>
```

```
            <mat-expansion-panel [expanded]="true">
              <mat-expansion-panel-header>
                <mat-panel-title>
                  <div class="criteria-name" fxLayout="row" >
                    <h2>Rank in Area</h2>
                    <mat-icon matTooltip="Select areas in which to rank">info_outline</mat-icon>
                  </div>
                </mat-panel-title>
              </mat-expansion-panel-header>
              <app-aree-trea></app-aree-trea>
            </mat-expansion-panel>
            <mat-card ></mat-card>
            <mat-card ></mat-card>
          <mat-card class="fixed-button-card">
            <button class="btnRank" mat-stroked-button fxLayoutAlign="center"
(click)="onRankClick()">Rank</button>
          </mat-card>
        </div>
    </mat-drawer>
    <mat-drawer-content>
      <app-editors-list></app-editors-list>
    </mat-drawer-content>

</mat-drawer-container>
```

Editors.service

```
getEditorsWithParams(currentPage: number, editorsPerPage: number, criteria: Criteria){
    const json = JSON.stringify(criteria);

    const queryParams = `?page=${currentPage}&page-
size=${editorsPerPage}&criteria=${encodeURIComponent(json)}`;
    return this.http.get<{numPages : number, editors: EditorAreas[]}>(
      this.rootUrl +"/ranking" + queryParams
```

```
  );
 }
 getEditorsWithParamsSearch(currentPage: number, editorsPerPage: number, criteria: Criteria, query: string){
  const json = JSON.stringify(criteria);

  const queryParams = `?page=${currentPage}&page-size=${editorsPerPage}&criteria=${encodeURIComponent(json)}&query=${query}`;
  return this.http.get<{numPages : number, editors: EditorAreas[]}>(
   this.rootUrl +"/ranking" + queryParams
  );
 }
```

Criteria-panel.component.ts

```
calculateAreaWeight(index){
  let h_sum = 15;
  let priority = this.order_area_required.length - index;

  let r = priority / h_sum;
  return r;

 }

 getAreaByTitle(str: string, newIndex: number){
  for(let i=0; i<this.order_area_required.length; i++){
   if(this.order_area_required[i].title == str){
    this.order_area_required[i].k = this.calculateAreaWeight(newIndex);
   }
  }
 }

 dropCriteria(event: CdkDragDrop<string[]>) {
  if (event.previousContainer === event.container) {
   moveItemInArray(event.container.data, event.previousIndex, event.currentIndex);
```

```
  } else {
    transferArrayItem(event.previousContainer.data,
                      event.container.data,
                      event.previousIndex,
                      event.currentIndex);
  }
}

drop(event: CdkDragDrop<string[]>) {


  moveItemInArray(event.container.data, event.previousIndex, event.currentIndex);

  for(let i=0; i<event.container.data.length; i++){
    this.getAreaByTitle(event.container.data[i], i);
  }

}

createLists(){
  for (let i = 0; i < this.order_criteria_required.length; i++) {
    if(i==0){
      this.criteriaList[i] = this.order_criteria_required;
    }
    else{
      this.criteriaList[i] = [];
    }
  }

  this.areaList = this.order_area_required;
}
```

```
isEmpty(arr){
  return arr.length == 0;
}

calculateCriteriaWeigth(){
  let priority = this.criteriaList.length;
  let h_summ = 6;
  let sum = 1;
  let q_str = "Work Quality";
  let c_str = "Communication";
  let s_str = "Speed";


  for(let i=0; i<this.criteriaList.length; i++){
    let elem = this.criteriaList[i];
      if(elem.length===3){
        this.qk = this.sk = this.ck = sum/this.order_criteria_required.length;
        return;
      }
      else if(elem.length ===2 && i==0 &&
        (this.isEmpty(this.criteriaList[1]) || this.isEmpty(this.criteriaList[2]))
        || elem.length ===2 && i==1 && this.isEmpty(this.criteriaList[0])){
        if(elem.includes(q_str) && elem.includes(c_str)){
          this.qk = this.ck = sum/2-0.1;
          this.sk = sum/2-0.3;
          return;
        }
        else if(elem.includes(q_str) && elem.includes(s_str)){
          this.qk = this.sk = sum/2 - 0.1;
          this.ck = sum/2-0.3;
          return;
        }
```

```
  else if(elem.includes(c_str) && elem.includes(s_str)){
    this.ck = this.sk = sum/2-0.1;
    this.qk = sum/2-0.3;
    return;
  }
 }
 else if(elem.length ===2 && !this.isEmpty(this.criteriaList[0]) && i==1
        || elem.length ===2 && i==2){
  if(elem.includes(q_str) && elem.includes(c_str)){
    this.qk = this.ck = sum/4;
    this.sk = sum/2;
    return;
  }
  else if(elem.includes(q_str) && elem.includes(s_str)){
    this.qk = this.sk = sum/4;
    this.ck = sum/2;
    return;
  }
  else if(elem.includes(c_str) && elem.includes(s_str)){
    this.ck = this.sk = sum/4;
    this.qk = sum/2;
    return;
  }
 }
 else{
  for (let index = 0; index < elem.length; index++) {
    let e = elem[index];
    if(e===q_str){
      this.qk = priority/h_summ;
    }
    else if(e===c_str){
      this.ck = priority/h_summ;
```

```
        }
        else if (e===s_str){
          this.sk = priority/h_summ;
        }
       }

      priority--;
    }

}
```