Хмельницький національний університет
Факультет інформаційних технологій
Кафедра телекомунікацій, медійних та інтелектуальних технологій

ДИПЛОМНА РОБОТА МАГІСТРА

Багатокритеріальна модель формування оптимального штату виконавців-розробників ІТ-компанії

Назва теми

Галузь знань 11 - Математика та статистика

Спеціальність 113 - Прикладна математика

Шифр ДРПМ.2021/077.01.03.00

Виконав:
студентка 2 курсу, група ПМм-20-1 _____ О.О. Шевчук
Підпис Ініціали, прізвище

Керівник:
канд.техн.наук, доцент _____ І.В. Драч
Підпис, дата Ініціали, прізвище

До захисту допускаю:
Зав. кафедри ТМІТ д-р.техн.наук, доцент _____ С.К. Підченко
Підпис, дата Ініціали, прізвище

8 12 2021 р.

Хмельницький 2021

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет: ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
Кафедра: ТЕЛЕКОМУНІКАЦІЙ, МЕДІЙНИХ ТА ІНТЕЛЕКТУАЛЬНИХ ТЕХНОЛОГІЙ
Освітній рівень: МАГІСТР
Галузь знань: 11 МАТЕМАТИКА ТА СТАТИСТИКА
Спеціальність: 113 ПРИКЛАДНА МАТЕМАТИКА
Освітня програма: ОСВІТНЬО-ПРОФЕСІЙНА

ЗАТВЕРДЖУЮ
Зав. кафедри ТМІТ
_____ Підченко С.К.
« 01 » 09 2021 р.

ЗАВДАННЯ
НА ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ)

Шевчук Ольга Олександрівна
Прізвище, ім'я, по батькові студента

1. Тема проекту (роботи) Багатокритеріальна модель формування оптимального штату виконавців-розробників ІТ-компанії

Керівник проекту (роботи) Драч Ілона Володимирівна, к.т.н доцент
Прізвище, ім'я, по батькові, науковий ступінь, вчене звання

Затверджена наказом ректора університету від 25.08.2021 р. № 102

2. Строк подання студентом проекту (роботи) на кафедру 01.12.2021 р.

3. Вихідні дані до проекту (роботи). Наукові джерела з питань моделювання формування оптимального штату виконавців-розробників ІТ-компанії

4. Зміст пояснювальної записки (перелік питань, які потрібно розробити). Виконати аналіз сучасного стану проблем моделювання в області розподілу праці; ознайомитися з теорією прийняття рішень при нечіткому відношенні переваги на множині альтернатив та методами експертних оцінок; побудувати математичну модель формування оптимального штату виконавців-розробників ІТ-компанії; на основі побудованої моделі розробити застосунок для системи Бітрікс24.

5. Перелік графічного матеріалу (із зазначенням обов'язкових креслень).

6. Консультанти розділів дипломного проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийня
Розділ 1			
Розділ 2			
Розділ 3			

7. Дата видачі завдання «03» вересня 2021 р.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів (розділів) дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Затвердження теми науковим керівником	01.09.2020 – 02.09.2020	Виконано
2	Збір та аналіз існуючих підходів та систем в області розподілу праці	03.09.2020 –08.09.2020	Виконано
3	Розробка 1 розділу написання ДРМ	09.09.2020 – 20.09.2020	Виконано
4	Ознайомлення з теорією прийняття рішень при нечіткому відношенні переваги на множині альтернатив та методами експертних оцінок	21.09.2020 – 27.09.2020	Виконано
5	Розробка 2 розділу написання ДРМ	28.09.2020 – 7.10.2020	Виконано
6	Побудова математичної моделі формування оптимального штату виконавців-розробників ІТ-компанії	08.10.2020 – 13.10.2020	Виконано
7	Розробка 3 розділу написання ДРМ	14.10.2020 – 20.10.2020	Виконано
8	Розробка застосунка для системи Бітрікс24 на основі побудованої моделі	21.10.2021 – 27.10.2021	Виконано
9	Розробка 4 розділу написання ДРМ	28.10.2021 – 05.11.2021	Виконано
10	Написання вступу, висновків, формування переліку джерел посилання та додатків	06.11.2020 – 08.11.2020	Виконано
11	Попередній захист дипломної роботи	09.11.2020 –10.11.2020	Виконано
12	Подача роботи на: кафедру, антиплагіат, рецензування, нормоконтроль	12.11.2020 – 3.12.2020	Виконано
13	Захист дипломної роботи	4.12.2020 –15.12.2020	Виконано

Студент Підпис О. О. Шевчук

Керівник проекту (роботи) Підпис І. В. Драч

АНОТАЦІЯ

Тема дипломної роботи: Багатокритеріальна модель формування оптимального штату виконавців-розробників ІТ-компанії.
Автор роботи: Шевчук Ольга Олександрівна
Керівник роботи: Драч Ілона Володимирівна.
Загальний обсяг роботи: 78 сторінок, 1 додаток, 20 посилань

ПРИЙНЯТТЯ РІШЕНЬ, БАГАТОКРИТЕРІАЛЬНА МОДЕЛЬ, МАТЕМАТИЧНЕ МОДЕЛЮВАННЯ ФОРМУВАННЯ ОПТИМАЛЬНОГО ШТАТУ ВИКОНАВЦІВ-РОЗРОБНИКІВ ІТ-КОМПАНІЇ, НЕЧІТКЕ ВІДНОШЕННЯ ПЕРЕВАГИ, МНОЖИНА АЛЬТЕРНАТИВ.

Побудовано багатокритегіальну модель формування оптимального штату виконавців-розробників ІТ-компанії. На основі моделі було розроблено програмний застосунок для системи Бітрікс24, який дозволяє швидко та обґрунтовано визначити найкращих претендентів для виконання завдання.

ANNOTATION

Thesis topic: Multi-criteria model for an optimal staff formation of developers in an IT-company
Author of the work: Olha Shevchuk
Mentor: Ilona Drach.
Total volume of work: 78 pages, 1 appendices, 20 references.

DECISION MAKING, MULTICRITERIAL MODEL, MATHEMATICAL MODELING FOR AN OPTIMAL STAFF FORMATION OF DEVELOPERS IN AN IT-COMPANY, FUZZY RATIO OF ADVANTAGES, PLURAL OF ALTERNATIVES.

A multi-criteria model for an optimal staff formation of developers has been built. Based on the model, a software application was developed for the Bitrix24 system, which allows you to quickly and reasonably determine the best contenders for the task.

ЗМІСТ

ВСТУП

Актуальність теми. Одна з найважливіших проблем у різних сферах людської діяльності – покращення управління. Ефективний менеджмент заснований на оптимальному використанні ресурсів, а також комплексній грамотній оцінки організації. Розвиток інформаційних технологій призвело до створення безлічі програмних продуктів, призначених для автоматизації діяльності кожної компанії, повністю або частково усуваючи людський фактор. Якщо на автоматизованому виробництві ефективність обладнання можна розрахувати, то в компаніях, де ефективність роботи характеризується використанням людських ресурсів, все складніше. [1]

Якщо проаналізувати питання про умови, за яких може бути досягнуто максимальної продуктивності, очевидно, що це можливо тільки при правильному розподілі внутрішніх завдань відповідно до можливостей працівників. Однією з особливостей, яка не дозволяє якісно розподіляти завдання між співробітниками у існуючому ПЗ, є складність їх адаптації до специфіки галузі.[1]

В умовах постійного розвитку галузі ІКТ, гострої конкуренції західних і місцевих роботодавців за висококваліфікованих співробітників роботодавець зацікавлений у найбільш ефективному використанні наявних людських ресурсів.[2]

Найбільш поширений варіант найму виконавця - це розподіл завдань керівником відділу або проєкту, в якому співробітник бере участь, на особистий вибір, часто суб'єктивний. Тоді рішення керівника не завжди буде правильним та найефективнішим.[2]

Метою дипломної роботи є побудова математичної моделі формування оптимального штату виконавців-розробників ІТ-компанії та розробка на її основі програмного застосунка для системи Бітрікс24.

Досягнення поставленої мети здійснюється шляхом розв’язання наступних основних завдань:

– зібрати та проаналізувати існуючі підходи та системи в області розподілу праці;

– детально ознайомитися з теорією прийняття рішень при нечіткому відношенні переваги на множині альтернатив та методами експертних оцінок;

– побудувати математичну модель формування оптимального штату виконавців-розробників ІТ-компанії;

– побудовану модель взяти за основу розробки застосунка для системи Бітрікс24 для формування оптимального штату виконавців-розробників ІТ-компанії.

У якості *об'єкта дослідження* обрано структуру виконуваних завдань та якісні характеристики виконавців-розробників ІТ-компанії ПП «Авіві» для формування оптимального призначення для виконання завдання.

Предметом дослідження є моделювання прийняття управлінських рішень на основі нечіткого відношення переваги.

Науково-практична новизна результатів дипломної роботи: побудовану багатокритеріальну модель взято за основу розробки застосунка для системи Бітрікс24 для формування оптимального штату виконавців-розробників ІТ-компанії. Розроблений застосунок допоможе приймати оптимальні рішення про призначення завдань виконавцям-розробникам, тим самим прискорити цей процес та мінімізувати перевантаженість програмістів.

Основна частина складається з чотирьох розділів. Перший розділ роботи присвячений аналізу вже існуючих підходів та систем в області розподілу праці.

У другому розділі розглянуті питання прийняття рішень при нечіткому відношенні переваги на множині альтернатив.

У третьому розділі описано побудову математичної моделі формування оптимального штату виконавців-розробників ІТ-компанії. Для побудови моделі використовуються принципи прийняття рішень при нечіткому відношенні переваги на множині альтернатив.

У четвертому розділі представлено основні моменти розробки програмного застосунка для системи Бітрікс24. Також проведено практичну апробацію та перевірку коректності отриманих результатів.

Публікації та апробація результатів дослідження.

За темою роботи опубліковано статтю (Додаток А) –

Шевчук О. О. Методи прийняття рішень в умовах нечіткої інформації в задачах розподілення робіт між працівниками // Актуальні проблеми комп'ютерних наук. Збірник наукових праць за матеріалами XIII всеукраїнської науково-практичної конференції «Актуальні проблеми комп'ютерних наук АПКН-2021» – Хмельницький: ХНУ, 2021 – С. 274-277.

1 ІСНУЮЧІ ПІДХОДИ ТА СИСТЕМИ В ОБЛАСТІ РОЗПОДІЛУ ПРАЦІ

1.1 АНАЛІЗ ІСНУЮЧИХ ПІДХОДІВ

Сьогоднішній надлишок кваліфікованих фахівців робить дедалі актуальнішим питання ефективності управління персоналом. В умовах високої конкуренції співробітники є ключовим ресурсом успіху компанії на ринку праці.[3]

Методи та підходи найму все ще погано вивчені, тому виникають проблеми при прийнятті кадрових рішень. На результат кадрових рішень впливають індивідуальні характеристики претендента на посаду, зокрема мотивація задля досягнення успіху у роботі [4]. Для прийняття подібного рішення використовується один із таких аналітичних методів:

– багатокритеріальний метод;
– раціональний метод;
– кластерний метод.

Рекомендується використовувати багатокритеріальний підхід для підтримки прийняття кадрових рішень, оскільки він є найпростішим і найбільш розвиненим [5]. При багатокритеріальному підході кожна окрема альтернатива може бути оцінена певним числом і порівняння альтернатив зводиться до порівняння відповідних чисел. Саме тому розглядається ця модель [4]. На рисунку 1.1 показано покрокову діаграму підтримки прийняття кадрових рішень.

Характерною рисою багатокритеріального підходу є пошук рішень за умов невизначеності в дискретній чи неперервній множині альтернатив. Існує класифікація за змістом та типом отримуваної інформації. Інформаційне наповнення використовує інформацію про переваги на множині критеріїв та про наслідки альтернатив [6].

Рисунок 1.1 – Процедура прийняття рішення

Типи використовуваної інформації та відповідні процеси прийняття рішень (таблиця 1.1).

Найбільш перспективними з них є декомпозиційні методи теорії очікуваної корисності, методи аналізу ієрархій та теорії нечітких множин.

Таблиця 1.1 – Методи прийняття рішень

Типи інформації	Метод прийняття
Відсутність інформації про вподобання; кількісна і/або інтервальна інформація про наслідки. Якісна інформація про вподобання та кількісна про наслідки Якісна (порядкова) інформація про вподобання та наслідки Кількісна інформація про вподобання та наслідки	Метод з дискретизацією невизначеності Стохастичне домінування Методи прийняття рішень в умовах ризику та невизначеності на основі глобальних критеріїв Метод аналізу ієрархій Метод теорії нечітких множин Метод практичного прийняття рішень Метод вибору статистично ненадійних рішень Метод кривих байдужості для прийняття рішень в умовах ризику та невизначеності Метод дерев рішень Декомпозиційні методи теорії очікуваної корисності

Перспективними ці методи є тому, що якнайкраще задовольнять вимогам універсальності.[7]

Комплексний проєкт - це проєкт по розробці, створенню, впровадженню та супроводу великих прикладних інформаційних систем [7].

Високий рівень компетенції учасників складних проєктів – один із найважливіших факторів успішної реалізації та здачі проєкту замовнику [8].

Компетенція – це раціональне поєднання навичок, особистісних характеристик та мотивації співробітників компанії, які розглядаються в часовому інтервалі (рис. 1.2) [7].

Складність у тому, що комплексний проєкт завжди обмежений у часі. Тому дуже важливо ефективно розподілити обов'язки між учасниками проєкту.[7]

Рисунок 1.2 – Зміст і сутність поняття «компетенція»

Чинники, що впливають на процес прийняття рішення при такому розподілі[7]:

- Особистісні оцінки керівника. Суб'єктивна думка про пріоритетні завдання.
- Ризики та невизначеності. Чинники довкілля
- Час.
- Вартість інформації. Витрати на інформацію мають бути збалансовані доходами від використання та впровадження
- Взаємозв'язок рішень. Системний підхід.

Розподіляючи обов'язки, керівник проєкту збирає групу людей, щоб вибрати тих, хто підходить для успішної реалізації проєкту. Відбір проводиться за допомогою комп'ютерної техніки. Зокрема, це відноситься до порівняльного аналізу різних методів та моделей. Процес прийняття рішень можна розглядати як типове завдання, яке має певний алгоритм, що полягає у виборі рішення з

наявних, яке буде найефективнішим при розподілі обов'язків між учасниками комплексного проєкту.[7]

Усі методи прийняття рішень можна поділити на дві групи: формалізовані та неформалізовані. При вирішенні добре структурованих та частково структурованих завдань використовуються формалізовані методи оцінки, вибору та обґрунтування варіанта вирішення. Неформалізовані - для складних, частково структурованих та неструктурованих завдань при створенні варіантів рішень, їх аналізі та оцінці, виборі та обґрунтуванні найкращого рішення [7].

Формалізовані групи методів засновані на алгоритмі, що оптимізує час та зусилля. Виділимо ключові етапи прийняття рішення [7]:

- виділення пріоритетів.
- планування роботи та результатів.
- концентрація на конкретному етапі.
- систематизація результатів.
- оцінка ефективності результатів.

Формалізована група включає методи обґрунтування та вибору кращих рішень та включає в себе[7]:

- економіко-математичні моделі та методи (ЕММ), що формалізують взаємозв'язки між процесами та явищами;
- системний аналіз, що дозволяє виявити взаємодії елементів системи та стратегію їх розвитку;
- Експертні оцінки та судження, що дозволяють кваліфікованим фахівцям оцінити важливість подій, явищ, факторів, прогнози розвитку систем та підсистем, взаємозв'язок між детермінованими та імовірнісними факторами.

У більшості компаній менеджери всіх рівнів та співробітники відділу кадрів беруть участь в оцінці ефективності праці персоналу. Керівники та працівники відділів кадрів повинні володіти сучасними методами оцінки працівників [7].

Як правило, атестація працівників повинна включати такі заходи:

- чітке формулювання вимог та стандартів для конкретної посади;
- створення системи критеріїв оцінки рівня кваліфікації працівника, яка спрямована на відповідність вимогам посади;
- комплексна (кількісна та якісна) оцінка роботи працівника
- оцінка відповідності навичок працівника вимогам посади(визначення ступеня близькості вимог роботи з рівнем компетенції виконавця);
- створення системи, що зв'язує результати оцінки праці працівників із системою винагород за працю, тобто з визначенням заробітної плати, розміру премій, пільг і т. п.;
- створення механізму, що пов'язує результати оцінки співробітників із системою просування (кар'єрного зростання) та розвитку співробітників у конкретній компанії;
- створення механізму прив'язки результатів атестації персоналу до системи підвищення кваліфікації та перепідготовки кадрів [7].

Найчастіше використовувані методи оцінки - графічна шкала оцінки; метод альтернативного ранжування; метод попарного порівняння; метод примусового розподілу; метод критичних випадків; рейтингові шкали, прив'язані до якості працівника; метод управління за цілями Розглянемо кілька методів оцінки ефективності роботи, які добре себе зарекомендували: метод попарного порівняння та угорський метод.[7]

Метод парного порівняння: згідно з параметрами, вибраними для оцінки компетенцій, співробітник порівнюється з іншим працівником, який працює у парі. На рисунку 1.3 показаний приклад попарного порівняння п'яти працівників відділу. При порівнянні «+» отримує кращий співробітник з пари. На рисунку 1.3 показано, що Марія отримує вищі бали за якість роботи, а Артур - вищі бали за творчість [7].

Характеристика «Якість роботи»						Характеристика «Творчість»					
Робітник, якого ранжують						Робітник, якого ранжують					
У порівнянні з	A Артур	B Марія	C Ігор	D Діана	E Іван	У порівнянні з	A Артур	B Марія	C Ігор	D Діана	E Іван
A Артур		+	+	-	-	A Артур		-	-	-	-
B Марія	-		-	-	-	B Марія	+		-	+	+
C Ігор	-	+		+	-	C Ігор	+	-	+		+
D Діана	+	+	-		+	D Діана	+	-	+		-
E Іван	+	+	+	-		E Іван	+	-	-	+	
Найвищий рейтинг тут у Марії						Найвищий рейтинг тут у Артура					

Рисунок 1.3 – Ранжування учасників за методом попарного порівняння

Задача про призначення (угорський метод). Задачу про призначення можна сформулювати наступним чином: є n виконавців і n робіт, задано $c_{ij}(i,j = 1..n)$ - ефективність виконання кожної роботи кожним виконавцем (таблиця, в якій містяться чисел, яка характеризує ефективність, називається $n_x n$- або n^2-матрицею). Задача полягає в тому, щоб дати кожному виконавцю одну і тільки одну роботу таким чином, щоб оптимізувати задану функцію ефективності [9].

Математична модель задачі про призначення [9]:

$$C = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} \to min,$$

$$\begin{cases} \sum_{j=1}^{n} x_{ij} = 1\,, & i = \overline{1,n}, \\ \sum_{i=1}^{n} x_{ij} = 1\,, & j = \overline{1,n}, \\ x_{ij} \in \{0.1\}, & i,j = \overline{1,n}. \end{cases} \tag{1.1}$$

У більшості випадків алгоритм вирішення такої задачі заснований на угорському методі, який включає наступний принцип: оптимальність вирішення

задачі про призначення не порушується, якщо елементи рядка/стовпця зменшуються або збільшуються на одне й те саме число.[9]

Рішення вважається оптимальним, якщо всі витрати $c_{ij}^{*} \geq 0, (i = 1..m; j = 1..n)$, змінені таким чином, і можна відшукати такий набір x_{ij}, який буде задовольняти функції (1.2) [9].

$$\sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}^{*} x_{ij} = 0 \qquad (1.2)$$

Кроки реалізації алгоритму

Крок 1. Отримання нулів у кожному рядку.

Виберемо невеликий елемент у кожному рядку та запишемо його значення у правий стовпець. Віднімемо мінімальні елементи з відповідних рядків. Перехід до кроку 2.[9]

Крок 2. Отримання нулів у кожному стовпці

У таблиці знаходимо мінімальні значення у кожному стовпці і записуємо їх у нижній рядок. Віднімемо мінімальні елементи з відповідних стовпців. Перехід до кроку 3.[9]

Крок 3. Пошук оптимального рішення

Зробимо призначення. Для цього подивимось на рядок, що містить найменше число нулів. Позначимо один з нулів у цьому рядку та закреслимо всі інші нулі у цьому рядку та стовпці, що містить зазначений нуль. Аналогічні операції виконуються послідовно в кожному рядку. Якщо призначення, яке отримане з усіма заданими нулями, є повним (число заданих нулів дорівнює n), то рішення є оптимальним. Інакше переходимо до кроку 4.[9]

Крок 4. Пошук мінімального набору рядків та стовпців, що містить усі нулі. Слід відзначити[9]:

1) усі рядки, що не містять зазначеного нуля

2) всі стовпці, що містять перекреслений нуль хоча б в одній із зазначених рядків;

3) всі рядки, що містять зазначені нулі хоча б в одному із зазначених стовпців.

Кроки 2 і 3 повторюються почергово, поки є що позначати. Потім неохідно закреслити кожен непозначений рядок і кожен позначений стовпець.[9]

Мета цього кроку - провести мінімальну кількість горизонтальних та вертикальних ліній, що перетинають усі нулі хоча б один раз.

Крок 5. Зміна місцями нулів [9].

Візьмемо найменше число клітин, де немає прямих ліній. Віднімемо його від кожного числа в не викреслених стовпцях і додамо до кожного числа у викреслених рядках. Ця операція не змінює оптимального рішення, після чого весь цикл розрахунку повторюється з кроку 3.[9]

Використовуючи цей алгоритм, всі учасники набувають оптимальних часових значень для реалізації проєкту всіма учасниками (таблиця 1.2) [9].

Таблиця 1.2 – Задача про призначення

Учасник проєкту, *i*	Час виконання *i*-тим учасником *j*-того етапу комплексного проєкту			
	1	2	3	4
1	0	2	4	2
2	0	0	4	0
3	0	1	0	1
4	4	0	0	0

За допомогою алгоритму було отримано матрицю значень для кожного учасника з певним часом на виконання свого етапу комплексного проєкту [9].

Термін виконання всіх етапів проєкту, отже, і всього комплексного проєкту, розраховується за формулою (1.3) [9]:

$$T = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij} \rightarrow min, \qquad (1.3)$$

Підставивши значення з матриці, отримали час, що дорівнює 17 місяцям [9]:

$$T = 3x_1 + 5x_1 + 2x_1 + 7x_1 = 17. \quad (1.4)$$

Таким чином, вибір проєктного рішення зводиться до відшукання альтернативи з найбільшим або найменшим значенням критеріїв.[9]

Розглянуті методи підходять для ситуацій, коли необхідно розподілити обов'язки між групою осіб, у яких кандидати порівнюються відповідно до запропонованих критеріїв. Оцініть час, необхідний для завершення комплексного проєкту з цією групою учасників, на основі матриці значень та передбачаючи успіх проєкту у встановлені терміни.[9]

1.2 АНАЛІЗ ІСНУЮЧИХ СИСТЕМ

Існують такі системи підтримки прийняття рішень при розподілі працівників:

а) «Імператор» – універсальна система підтримки прийняття рішень у багатьох сферах діяльності. Вона допоможе вирішити нагальні проблеми, такі як розподіл ресурсів і вибір найкращого рішення [10].

В основі системи «Імператор» використовується метод аналізу ієрархій. Цей метод дуже схожий на людське мислення і узагальнює найпопулярніші підходи поширених експертних систем. Розробники системи удосконалили методи аналізу ієрархій, тим самим виділивши її серед аналогів та вразивши користувачів функціоналом[10]:

1) створення графічної схеми завдання.
2) збір інформації від експертів
3) оцінка та зниження рівня суперечливої інформації
4) обрахування рейтингу альтернативних рішень

5) ведення тематичного каталогу проєктів, якій включає моделі для рейтингування і набори даних.
6) перевірка стійкості рейтингу.
7) визначення ключових факторів
8) створення та аналіз динамічних сценаріїв розвитку ситуації.
9) вирішення питання відновлення ситуації на підставі відомих рейтингів.
10) дослідження та аналіз проблеми за допомогою кількох моделей
11) моделювання дії випадкових факторів
12) робота з Microsoft Excel для експорту та імпорту моделей.
13) створення докладних звітів в Microsoft Word.

"Імператор" може аналізувати та синтезувати процес прийняття рішень, враховуючи «людський фактор». Експерт може надати як якісну та і об'єктивно кількісну інформацію.[10]

Система має простий у використанні інтерфейс та довідку, яка докладно пояснює, як працювати з програмою, моделювати та проектувати (рис. 1.5) [10]

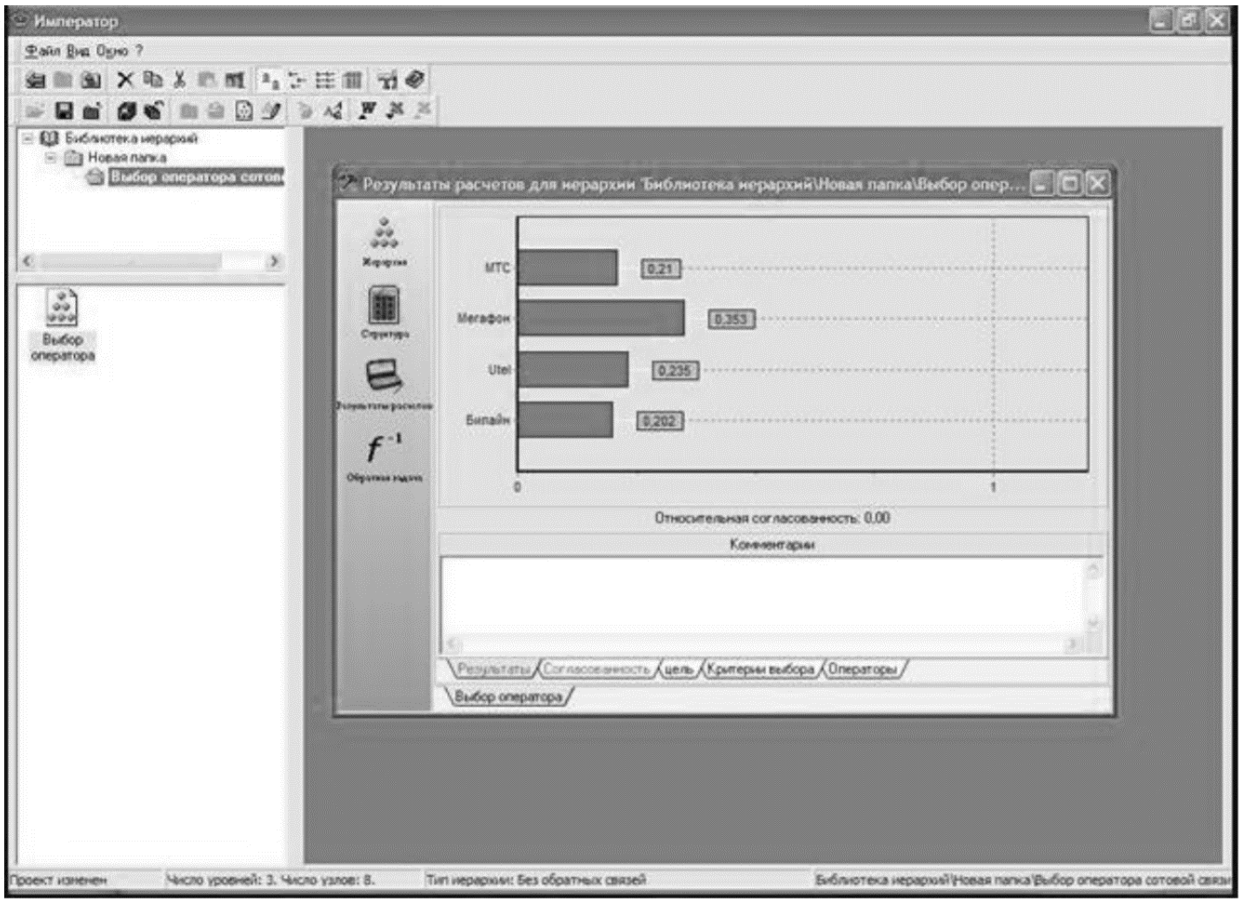


Рисунок 1.5 - Інтерфейс системи «Імператор»

"Імператор" має два мовних режима: російський та англійський. Гнучкість системи ґрунтується на принципах, закладених в основу. Вона має систематичні, регламентовані підходи осмисленого прийняття рішень, тому це корисно для [10]:

1) соціологів;
2) науковців;
3) політиків;
4) працівників управління;
5) військових;
6) працівників охорони здоров'я;
7) фінансистів;
8) оцінювачів;
9) психологів;
10) економістів;

11) працівників соціальної сфери;
12) консультантів.

Використовується для вирішення багатьох важливих задач [10]:

1) передбачення варіантів еволюції ситуації;
2) розробка стратегії для досягнення бажаного майбутнього;
3) розробка технологій управління;
4) дослідження та аналіз систем та можливих сценаріїв їх розвитку.
5) аналізування ризиків;
6) прийняття рішень щодо працівників;
7) вирішення конфліктних ситуацій
8) аналіз проєктів за багатьма критеріями
9) розподілення коштів між проєктами
10) дослідження та побудова черг
11) створення ретингів клієнтів, кандидатів, проєктів та ін.

"Імператор" допоможе компанії покращити процеси аналізу та вирішення проблем та приймати обґрунтовані рішення. Придбання системи "Імператор" допоможе підвищити якість прийняття рішень [10].

б) T-CHOICE 1.0 - діалогова система, спрямовану підтримку прийняття рішень у різних галузях людської діяльності. Ця система може стати внагоді усім кому потрібно приймати обґрунтовані рішення.

Пошук рішення в системі відбувається з використанням табличного методу. Вихідні дані представляються у табличному вигляді, де рядки представляють альтернативи, а стовпці - критерії, на яких ґрунтується прийняття рішення. Особа, яка приймає рішення, має відсортувати значення у кожному стовпці та визначити обмеження для кожного критерію. Прийнятними вважаються всі альтернативи, які не перевищують зазначену межу [11].

в) MPRIORITY 1.0 - діалогова система, спрямовану підтримку процесу прийняття рішень у різних сферах людської діяльності. Вона може бути

серйозною підтримкою усім, кому необхідно приймати раціональні обґрунтовані рішення [12].

Програма використовує Метод Аналізу Ієрархій, основною метою якого є вирішення слабоструктурованих задач прийняття рішень.

Структура задачі прийняття рішення в Методі Аналізу Ієрархій ієрархічна. На вершині ієрархії знаходиться основна мета, нижче – підцілі і нижче – альтернативи. Альтернативи ранжуються шляхом попарного зважування за інтуїтивною шкалою якості. [12].

Система MPRIORITY відрізняється від аналогів тим, що має діалоговий інтерфейс, адаптований під особливості МАІ, що дозволяє отримувати максимально повну інформацію про проведені попарні порівнянь і усувати можливі невідповідності в матрицях. Механізм шаблонів дозволяє користувачеві адаптувати програмну систему під себе [12].

Приклади задач прийняття рішення, для яких можна використовувати "MPRIORITY" [12]:

1) вибір керівником компанії майбутнього ділового партнера;
2) раціональний розподіл доходів підприємства за видами діяльності;
3) відбирати найкращих кандидатів на посади в компанії;
4) оцінка роботи персоналу;
5) підбір програмного забезпечення для потреб бізнесу;
6) оцінка культурних цінностей (картини, скульптури та ін.);
7) вибір найкращої стратегії;
8) вибір найкращої конструкції (варіанти) технічного вироби;
9) покупка квартири, дачі, ділянки, автомобіля;
10) вибір майбутнього навчального закладу для дитини;
11) вибір майбутнього робочого місця.

г) OPTIMUM 1.0 – діалогова система, орієнтована на вирішення практичних завдань глобальної оптимізації. Програмна система може бути застосована у разі, коли процес прийняття рішень може бути зведений до побудови та оптимізації деякої функції [11].

Багато завдань прийняття рішень зведено до обчислення цільової функції з подальшим знаходженням її оптимального значення. Діалогова система "OPTIMUM" базується на двох методах оптимізації [11]:

1) основному - адаптивному випадковому пошуку;
2) додатковому - детермінованому методі Хука-Джівса, що дозволяє користувачеві виконати уточнення будь-якого поточного рішення.

У процесі проектування системи "OPTIMUM" враховувалися характерні особливості діалогу [11]:

1) Спосіб подання користувачем завдання оптимізації.

Це можливість, що дозволяє користувачеві представити своє завдання оптимізації для подальшого її вирішення на ЕОМ. Різні програмні системи допускають застосування своєї (вбудованої) мови програмування, яку ще потрібно вивчити. "OPTIMUM" дозволяє застосовувати будь-яку існуючу мову програмування, здатну створити виконуваний програмний модуль (exe-файл) [11].

2) Різні засоби діалогу "людина-комп'ютер", що дозволяють користувачеві:
   – виконувати параметризацію пошуку;
   – стежити за процесом пошуку оптимального рішення;
   – включатися у пошуковий процес і, при необхідності, змінювати його.

Перераховані системи неідеальні. Для даної компанії потрібна розробка своєї вузьконаправленої системи, щоб вона зберігала свою швидкодію за рахунок відсутності непотрібних шаблонів прийняття рішення.[11]

2 БАГАТОКРИТЕРІАЛЬНИЙ ВИБІР АЛЬТЕРНАТИВ НА ОСНОВІ НЕЧІТКОГО ВІДНОШЕННЯ ПЕРЕВАГИ

2.1 ВИБІР НА ОСНОВІ ВІДНОШЕННЯ ПЕРЕВАГИ НА МНОЖИНІ АЛЬТЕРНАТИВ

Процес прийняття рішення щодо вибору найкращої (найраціональнішої) альтернативи з універсального набору альтернатив X може відбуватися з різним ступенем проінформованості ОПР.[13]

Якщо інформацію про реальну ситуацію, з якої порівнюються різні альтернативи, можна оформити як допоміжну функцію, перед нами проблема НМП. Однак такий опис інформації не завжди можливий. Більш універсальний опис інформації як співвідношення використовується у різних альтернативах.[13]

Розглянемо цей взаємозв'язок та його властивості. Припустимо, що на основі інформації, отриманої з ОПР, вводиться чітка кореляція між перевагами без рядка R на безлічі дозволених альтернатив X. Це означає. що будь-яка пара альтернатив *(x. y)* може бути виражена одним з таких тверджень;[13]

x не гірше за y записується $x \succcurlyeq y$ або $(x, y) \in R$;

y не гірше за x, тобто. $y \succcurlyeq x$ або $(y, x) \in R$;

x і y не можна порівняти, тобто. $(y, x) \notin R$, $(x, y) \notin R$.

Така інформація дозволяє звузити клас раціональних виборів, щоб увімкнути лише альтернативи, над якими не домінує жодна з альтернатив X.[13]

Для визначення недомінованих альтернатив вводиться коефіцієнт строгої переваги R_S, що відповідає коефіцієнту недомінованої переваги R, а також коефіцієнт байдужості R_J.[13]

Ми говоримо, що альтернатива x строго краща, ніж альтернатива *y*, коли і $x \succcurlyeq y$, *і* $y \succcurlyeq x$ відповідно $(x, y) \in R$, $(y, x) \notin R$. Множина всіх таких пар *(x, y)* називається выдношенням строгої переваги R_S на множині X[13]

Для компактності запису відношення R_S скористаємося визначенням відношення R^{-1} до R. Строге відношення переваг R_S, за його визначенням, записується таким чином:[13]

$$R_S = R \backslash R^{-1} \tag{2.1}$$

Якщо $(x, y) \in R_S$„ то кажуть, що альтернатива x домінує над альтернативою y (ми пишемо $x \succ y$). Альтернатива $x \in X$ називається недомінованою на множині X із заданим значенням R_S, якщо $(y, x) \notin R_S$ для будь-якої альтернативи $y \in X$. Іншими словами, якщо x – недомінована альтернатива, то альтернативи у у множині X, яка домінує над x, немає. Тому вибір недомінованих альтернатив можна вважати раціональним у задачі ухвалення рішення.[13]

Таким чином, інформація у вигляді відношення переваг R дозволяє звузити клас раціональних виборів X до підмножини недомінованих альтернатив $X_{нд}$ у вигляді[13]:

$$X_{\text{нд}} = \{x : x \in X, (y, x) \notin R \backslash R^{-1}, \forall y \in X\} \tag{2.2}$$

Якщо інформації, доступної ОПР як відношення переваг, недостатньо для вибору між альтернативами x і y, то між цими альтернативами існує співвідношення R_J («байдужості»). Як уже говорилося, існує відношення байдужості R_J між альтернативами x і y тоді і тільки тоді, коли відношення переваги $x \succcurlyeq y$ та переваги $y \succcurlyeq x$ виконуються або не виконуються одночасно. З цього визначення випливає, що R_J можна записати як[13]:

$$R_J = \{X \times X\} \backslash \{R \cup R^{-1}\} \cup \{R \cap R^{-1}\} \tag{2.3}$$

Розглядається метод прийняття рішення, що передбачає побудову безлічі недомінованих альтернатив на основі нечіткого відношення переваги. Знаходиться згортка щодо переваги за критеріями, що мають різні ваги. [13]

Короткі відомості про метод. Нехай задано безліч альтернатив *X* й кожна альтернатива характеризується декількома ознаками з номерами *j=1, m*. Інформація про попарне порівняння альтернатив до кожної з ознак *j* представлена у формі відношення переваги R_j. Таким чином, є *m* відношень переваги R_j на множині X. Потрібно за цією інформацією вибрати альтернативу з множини $\{X, R_j, \ldots, R_m\}$.[13]

Дамо деякі визначення.

Визначення 2.1. Нечітким відношенням *R* на множині *X* називається нечітка підмножина декартового добутку *X*×*X*, що характеризується функцією приналежності $\mu_R: X \times X \to [0,1]$. Значення $\mu_R(x, y)$ цієї функції сприймається як ступінь виконання відношення *xRy*. [13]

Визначення 2.2. Нечітким відношенням нестрогої переваги на X називається будь-яке задане на цій множині рефлексивне нечітке відношення. [13]

Визначення 2.3. Функція приналежності відношення строгої переваги[13]

$$\mu_R^S(x,y) = \begin{cases} \mu_R(x,y) - \mu_R(y,x), \text{якщо } \mu_R(x,y) \geq \mu_R(y,x), \\ 0, \text{у протилежному разі.} \end{cases} \quad (2.4)$$

Визначення 2.4. Нехай *X* – множина альтернатив і μ_R – задане на ньому нечітке відношення переваги. Нечітка підмножина недомінованих альтернатив множини (X, μ_R) описується функцією приналежності[13]

$$\mu_R^{\text{нд}}(x) = 1 - \sup_{y \in X} \mu_R^S(y,x), x \in X. \quad (2.5)$$

Користуючись визначеннями 2.3 і 2.4, можна показати, що [13]

$$\mu_R^{\text{нд}}(x) = 1 - sup[\mu_R(y,x) - \mu_R(x,y)] \quad (2.6)$$

Визначення 2.5. Чітко недомінованими називаються альтернативи, для яких, $\mu_R^{\text{нд}}(x) = 1$ а безліч таких альтернатив [13]

$$X^{\text{чнд}} = \{x : x \in X \ \text{ та } \mu_R^{\text{нд}}(x) = 1\} \tag{2.7}$$

Визначення 2.6.,Носієм нечіткої множини А з функцією приналежності μ_{A} є множина[13]

$$\{x : x \in X \ \text{ та } \mu_{\text{A}}(x) > 1\}. \tag{2.8}$$

Розглянемо найпростішу ситуацію, коли відношення R_j описуються заданими функціями корисності $f_j : X \to R$. Значення функції f_j можна розуміти як числову оцінку альтернативи X за ознакою *j*. Альтернатива з більшою оцінкою $f_j(x)$ вважається кращою за ознакою *j*. Завдання полягає в тому, щоб вибрати альтернативу, яка має найбільші оцінки за всіма ознаками. Раціональним у цьому разі природно вважати вибір альтернативи $x_0 \in X$, що володіє властивістю[13]:

$$f_j(y) \geq f_j(x_0), j = \overline{1,m} \Rightarrow f_j(y) = f_j(x_0), j = \overline{1,m} \tag{2.9}$$

Такі альтернативи називають ефективними; рішенням цього завдання вибору є множина всіх ефективних альтернатив. Кожна з функції описує звичайне відношення переваги X наступного виду[13]:

$$R_j = \{(x,y) | x, y \in X, f_j(x) \geq f_j(y), \}. \tag{2.10}$$

Нехай $Q_1 = \bigcap_{i=1}^{m} R_i$. Тоді множина всіх ефективних альтернатив у множині (X, Q_1) збігається з множиною ефективних альтернатив для набору функцій $f_j, j = \overline{1,m}$. [13]

Таким чином, для знаходження множини ефективних альтернатив, можна замість набору відношень $R_j j = \overline{1,m}$, використовувати перетин цих відношень Q_1 і знайти иножину недомінованих альтернатив на множині (X, Q_1). [13]

Представимо тепер перетин відношень R_j в дещо іншій формі. Нехай

$$\mu_j(x,y) = \begin{cases} 1, \text{якщо } (x,y) \in R_i; \\ 0, \text{якщо } (x,y) \notin R_i. \end{cases} \quad (2.11)$$

- функція приналежності R_j. [13]

Тоді перетину цих множин відповідає функція приналежності

$$\mu_{Q_1}(x,y) = \min\{\mu_1(x,y), \mu_2(x,y), .., \mu_m(x,y)\}, \quad (2.12)$$

аналогічна згортанню критеріїв виду

$$F(x) = \min_i \omega_j \, f_j(x), \quad (2.13)$$

застосовуваній у багатокритеріальних завданнях прийняття рішень. [13] Числа ω_j, у згортці являють собою коефіцієнти відносної переваги критеріїв, що розглядаються. В згортці відношень[13]

$$\omega_j = 1, j = \overline{1,m}\,. \quad (2.14)$$

До цього часу передбачалося, що всі задані відношення однаково важливо враховувати під час виборів альтернатив. Якщо відношення різняться за важливістю, тобто розрізняються за важливістю відповідні ознаки, за якими потрібно порівнювати альтернативи, то в згортці можна використовувати різні за значенням коефіцієнти . Однак при цьому необхідно розглянути ширший клас нечітких відношень. Іншими словами, у визначенні функцій належності числа 0

і 1 слід розуміти не як значення булевої змінної, що свідчить про належність елемента множині, а як крайні точки одиничного інтервалу можливих значень ступеня належності. [13]

В результаті згортки вихідних відношень R_j з коефіцієнтами ω_j такими, що

$$\sum_{j=1}^{m} \omega_j = 1, \omega_j \geq 0, j = \overline{1,m}\,, \tag{2.15}$$

- отримуємо функцію приналежності виду:

$$\mu_{Q_1}(x,y) = \min\{\omega_1\mu_1(x,y), \omega_2\mu_2(x,y), .., \omega_n\mu_n(x,y)\}. \tag{2.16}$$

тобто функцію приналежності нечіткого відношення переваги. [13] Неважко побачити, що це відношення переваги не рефлексивне і не дозволяє врахувати відмінності у відносній важливості заданих відношень.

Введемо згортку вихідних відношень іншого виду[13]:

$$\mu_{Q_1}(x,y) = \sum_{j=1}^{m} \omega_j\mu_j(x,y). \tag{2.17}$$

і розглянемо її використання у сформульованій задачі раціонального вибору альтернатив. Зауважимо, що результуюче нечітке відношення μ_{Q_2}, отримане в результаті згортки вихідних звичайних відношень R_j, рефлексивне, так як рефлексивні вихідні відношення R_j. [13]

Нехай, як і раніше, всі вихідні відношення переваги однакові за важливістю. Побудуємо нечітку підмножину недомінованих альтернатив множини $\{X, \mu_Q\}$, користуючись визначеннями, наведеними раніше[13]:

$$\mu_{Q_2}^{\text{нд}}(x) = 1 - \frac{1}{m} \sup_{y \in X} \left\{ \sum_{j=1}^{m} \mu_j(y,x) - \mu_j(x,y) \right\}, x \in X. \qquad (2.18)$$

Позначимо через $X_1^{\text{чнд}}$ підмножину чітко недомінованих альтернатив множини (X, μ_{Q_1}) ($X_1^{\text{чнд}}$ – множина ефективних альтернатив) і через $X_2^{\text{чнд}}$ – відповідну множину (X, μ_{Q_2}). Покажемо, що[13]

$$X_2^{\text{чнд}} \subseteq X_1^{\text{чнд}}. \qquad (2.19)$$

Розглянемо властивості альтернатив з множини – носія $\mu_{Q_2}^{\text{нд}}$. Функція $\mu_{Q_2}^{\text{нд}}(x)$ набуває лише значення виду де k – натуральне число, $k < m$.. Нехай для деякої альтернативи $\mu_{Q_2}^{\text{нд}}(x') = \frac{k}{m}$. Відповідно це означає, що

$$\sup_{y} \left\{ \sum_{j=1}^{m} \mu_j(y,x') - \mu_j(x',y) \right\} = m - k, \qquad (2.20)$$

або

$$\left\{ \sum_{j=1}^{m} \mu_j(y,x') - \mu_j(x',y) \right\} \leq m - k, \qquad (2.21)$$

за будь-якого $y \in X$. [13]

Оскільки члени суми приймають лише значення 0 і 1 то різниця між числом членів цієї суми, рівних +1, і числом членів, рівних -1, не перевищує (1-k) при будь-якому $y \in X$. Цей факт можна пояснити так. Нехай $p(y,x)$ - число функцій f_j із заданого набору, по кожній з яких альтернатива y строго краще x, і $q(y,x)$ - число функцій по яких x строго краще y. Якщо:

$$\mu_{Q_2}^{\text{нд}}(x') = \frac{k}{m}, \tag{2.22}$$

то

$$p(y, x') - q(y, x') \leq m - k, \tag{2.23}$$

за будь-якого $y \in X$. [13]

Таким чином, функція $\mu_{Q_2}^{\text{нд}}$ упорядковує альтернативи за ступенем їхньої недомінованості. Наприклад, якщо $\mu_{Q_2}^{\text{нд}}(x_0) = \frac{3}{4}$, тобто $m - k = 1$, і деяка альтернатива $y \in X$ строго краща за альтернативу x_0 за будь-якими двома критеріями, то не менше ніж по одному з інших x_0 строго краще y. Якщо взяти перетин множин $X_1^{\text{чнд}}$ і $\mu_{Q_2}^{\text{нд}}$, то отримаємо відповідне впорядкування на множині ефективних альтернатив, користуючись яким можна здійснити вибір серед них. Якщо ж у згортці коефіцієнти ω_j неоднакові, то кожна з введених вище характеристик $p(y, x')$ і $q(y, x)$ буде представляти собою не число відповідних критеріїв, а їх сумарну важливість. [13]

Застосування згортки вихідних звичайних відношень переваги, в задачі прийняття рішень по набору функцій дозволяє отримати додаткову інформацію про відносний ступінь недомінованості ефективних альтернатив і тим самим звузити клас раціональних виборів до множини[13]:

$$X^{\text{чнд}} = \left\{ x | x \in X, \mu_{Q_2}^{\text{нд}}(x) = \sup_{x' \in X_2^{\text{чнд}}} \mu_{Q_2}^{\text{нд}}(x') \right\} \tag{2.24}$$

У загальній задачі, коли на множині альтернатив задані нечіткі відношення переваги R_j, $j = \overline{1,m}$ та коефіцієнти ω_j відносної важливості цих співвідносин, можна чинити аналогічним чином. [13]

Процедура розв'язання задачі вибору.

1. Будуємо нечітке відношення Q_1 (перетин вихідних відношень) [13]:

$$\mu_{Q_1}(x,y) = \min\{\mu_1(x,y), \mu_2(x,y), .., \mu_m(x,y)\}. \quad (2.25)$$

і визначаємо нечітку підмножину недомінованих альтернатив у множині (X, μ_{Q_1}) :

$$\mu_{Q_1}^{\text{нд}}(x) = 1 - \sup_{y \in X}\{\mu_{Q_1}(y,x) - \mu_{Q_1}(x,y)\}. \quad (2.26)$$

2. Будуємо нечітке відношення Q_2[13]:

$$\mu_{Q_2}(x,y) = \sum_{j=1}^{m} \omega_j \mu_j(x,y), \quad (2.27)$$

і визначаємо нечітку підмножину недомінованих альтернатив у множині (X, μ_{Q_2}):

$$\mu_{Q_2}^{\text{нд}}(x) = 1 - \sup_{y \in X}(\mu_{Q_2}(y,x) - \mu_{Q_2}(x,y)). \quad (2.28)$$

Ця функція впорядковує альтернативи за ступенем їхньої недомінованості.

3. Знаходимо перетин множин $\mu_{Q_1}^{\text{нд}}$ і $\mu_{Q_2}^{\text{нд}}$[13]:

$$\mu_{\text{нд}}(x) = \min\{\mu_{Q_1}^{\text{нд}}(x), \mu_{Q_2}^{\text{нд}}(x)\}. \quad (2.29)$$

4. Раціональним вважаємо вибір альтернатив з множини[13]:

$$X^{\text{нд}} = \left\{ x | x \in X, \mu^{\text{нд}}(x) = \sup_{x' \in X} \mu^{\text{нд}}(x') \right\}. \quad (2.30)$$

Найбільш раціональним слід вважати вибір альтернативи з множини $X^{\text{нд}}$, яка має максимальний ступінь недомінованості.

2.2 МЕТОДИ ЕКСПЕРТНИХ ОЦІНОК

Методи експертної оцінки є частиною великої галузі теорії прийняття рішень, а експертна оцінка як така є процедурою отримання оцінки проблеми з урахуванням думок фахівців (експертів) для подальшого прийняття рішення (вибору).[14]

У разі надзвичайної складності проблеми, її новизни, нестачі інформації, неможливості математичної формалізації процесу вирішення необхідно звернутися до рекомендацій грамотних фахівців, які досконало знають проблему – експертів. Їхнє вирішення проблеми, аргументація, формування кількісних оцінок, розробка останніх формальними методами називається методом експертних оцінок.[14]

Є дві групи експертних оцінок[14]:

- Індивідуальні оцінки ґрунтуються на використанні думок окремих експертів, незалежних один від одного.
- Колективні оцінки ґрунтуються на використанні оцінок.

Методи вимірювання об'єктів

Ранг - це розташування об'єктів у порядку зростання чи зменшення деяких внутрішніх властивостей. Ранг дозволяє вибрати найважливіший фактор із безлічі розглянутих факторів.[14]

Парне порівння визначає перевагу об'єктів при порівнянні всіх можливих пар. Тут необов'язково, як при рангуванні, впорядковувати всі об'єкти. Необхідно виділити найбільш значущий об'єкт у кожній парі або встановити їх схожість.[14]

Пряма оцінка. Часто буває бажано не тільки відсортувати (рангувати об'єкти аналізу), а й визначити, наскільки один фактор значніший за інші. У цьому випадку серія змін властивостей об'єкта розбивається на окремі діапазони, кожному з яких надається певна оцінка (бал), наприклад, від 0 до 10. Тому метод прямої оцінки також називають бальним методом.[14]

Простий метод класифікації вимагає, щоб кожного експерта просили розташувати характеристики у порядку переваги.[14]

Метод встановлення ваги (a_{ij})

Всім знакам присвоюються вагові коефіцієнти так, щоб сума коефіцієнтів дорівнювала фіксованому числу (наприклад, одиниці, десяти або сотні).

Найзначніший з усіх символів зважується за фіксованим числом, проте всі символи зважуються з його часткою.

Метод послідовного порівняння виглядає так[14]:

- експерти впорядковують усі характеристики у порядку зменшення важливості: A1> A2>…> An;
- привласнюється значення один першій характеристиці: A1 = 1, а вагові коефіцієнти присвоюються у вигляді часток один до інших характеристик;
- порівнюються значення першої характеристики із сумою всіх наступних характеристик.

При порівнянні пар необов'язково сортувати усі позиції як у рейтингах; необхідно визначити важливіший об'єкт у кожній парі або визначити їх рівність. Попарні порівняння можна проводити з великою кількістю піддослідних, а також у тих випадках, коли різницю між піддослідними настільки незначні, що їх класифікація практично недоцільна.[14]

Зазвичай при використанні цього методу створюється матриця n×n, де n - кількість об'єктів, що порівнюються.[14]

Безпосередня оцінка. Часто необхідно не тільки спланувати (організувати елементи аналізу), а й з'ясувати цінність чого-небудь. У цьому випадку більшість змін властивостей об’єкта поділяються на різні простори, кожне з яких дає певну оцінку (бал), наприклад від 0 до 10. Таким чином, метод безпосередньої оцінки, також називають методом балів.[14]

Для аналізу результатів використовуються різні математичні методи. І його можна налаштувати та розширити відповідно до типу проєкту та бажаних результатів.[14]

Формування загальної оцінки

Нехай група експертів оцінили річ, то x_j — це j-й бал експерта, а m — кількість експертів. [14]

Для формування загальної оцінки групи експертів зазвичай використовуються середні величини. Медіана, на якій базується порівняння, означає, що кількість більших оцінок дорівнює кількості менших оцінок. [14]

Визначення відносної ваги об'єкта

Іноді виникає потреба визначити важливість конкретного фактору (об'єкту) з точки зору будь-якого критерія. У цьому випадку кажуть, що вага кожного компонента повинна бути визначена. [14]

Існує велика кількість можливих методів обробки оцінок. Крім того, результати можуть включати кілька алгоритмів зв'язаних між собою. Наприклад, алгоритм, який використовується для порівняння коефіцієнта кваліфікації експерта, впливає на статистичну значущість експерта тощо. [14]

У випадку участі в опитуванні декількох експертів розходження в їх оцінках неминучі, однак величина цього розходження має велике значення. Якщо відповіді експертів добре збігаються, можна сказати, що групова оцінка є достатньо надійна. [14]

Для аналізу розкиду і узгодження оцінок використовуються статистичні характеристики – міри розкиду або статистична варіація. [14]

Способи визначення міри розкиду[14]:

– варіаційний розмах:

$$R = x_{max} - x_{min} \tag{2.31}$$

– середнє лінійне відхилення.

$$\text{a} = \frac{1}{n}\sum_{i=1}^{n} |x_i - \overline{x}| \tag{2.32}$$

– середньоквадратичне відхилення:

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (x_i - \overline{x})^2} \tag{2.33}$$

– дисперсія.

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n} (x_i - \overline{x})^2 \tag{2.34}$$

– коефіцієнт рангової кореляції Спірмена.

$$\rho = 1 - \frac{6\sum_{i=1}^{n}\left(x_{ij} - x_{ik}\right)^2}{n(n^2-1)} = 1 - \frac{6\sum_{i=1}^{n} d_i^2}{n(n^2-1)} \tag{2.35}$$

Коефіцієнт (величина ρ) може змінюватися від -1 до +1. При ідеальному збігу оцінок коефіцієнт дорівнює 1. Значення коефіцієнта -1 вказує на найбільшу різницю в думках експертів. [14]

X_{ij} – ранг (важливість), присвоєний j-им експертом і-му об’єкту, x_{ik} – ранг, призначений k-им експертом і-му об’єкту, і d_i – різниця між рангами присвоєними і-му об'єкту. [14]

Кажучи про досягнення консенсусу серед експертів, важливо зазначити, що рангування не передбачає (або не завжди передбачає) відстань. Тобто для одного експерта A> B> C означає, що A >> B> C, а для іншого A> B >> C. І будь-які кореляції і розрахунки середніх оцінок у цій ситуації не допоможуть. В узгодженні може допомогти обрахунок індексу згоди. Подібно до кількості суперечливих замкнутих ланцюжків думок експертів (перший рахує, що А більша за В, другий, що В більше С, а третій, що С більше А) до кількості усіх подібних ланцюжків. [14]

Рейтинги зазвичай базуються на деякій ймовірносній моделі, тому потрібно ретельно враховувати область їх можливого застосування. [14]

3. МАТЕМАТИЧНА МОДЕЛЬ ФОРМУВАННЯ ОПТИМАЛЬНОГО ШТАТУ ВИКОНАВЦІВ-РОЗРОБНИКІВ ІТ-КОМПАНІЇ

Для призначення програміста на виконання певної задачі менеджеру необхідно врахувати багато факторів, щоб робота була виконана вчасно та якісно.

Дослідження проводилось у співпраці з однією з ІТ-компаній міста Хмельницького ПП «Авіві».

3.1 КРИТЕРІЇ

Було здійснено опитування менеджерів для визначення основних критеріїв, які впливають на вибір програміста.

Для дослідження було обрано 4 критерії та визначено вагу кожного:

- завантаженість програміста – вага 0.1,
- рівень компетенції – вага 0.5,
- спеціалізація – вага 0.2,
- обізнаність з проєктом – вага 0.2.

Завантаженість програміста

Для початку потрібно визначити, як навантажуються розробники.

Менеджери формують завдання для розробників. Вони можуть бути невеликі та прості або досить об’ємні та складні. Також кожне завдання має певний термін виконання. Кожному розробнику можуть призначити декілька завдань будь-якої складності та з різними термінами, а також за потреби визначити пріоритетність у виконанні.

За таких умов завантаженість буде визначатися як сумарний час, який лишився на виконання усіх завдань розробника та кращим претендентом на виконання нового завдання буде вважатися той розробник, у якого цей показник буде менший.

Рівень компетенції

Існують різні підходи до визначення рівнів Junior/Middle/Senior. Часто прив'язують цю градацію з кількістю років досвіду – це найпростіший, зрозумілий і неправильніший спосіб. У деяких компаніях, особливо великих або тих, які мають складну предметну область, до цих рівнів прив'язуються великі списки конкретних технічних компетенцій, але в кожної компанії вони свої [15]. У *Авіві* на перше місце ставлять рівень самостійності та завдань, які розробник може виконувати.

Junior-розробник може вирішувати лише невеликі завдання, нескладні та чітко поставлені. Він має мало знань, багато питань та для нього важлива постійна робота з наставником чи у добрій команді.[15]

Middle-розробник має достатньо досвіду, за порадами звертається рідко, багато знає сам або може самостійно розібратися. Завдання не потрібно пояснювати детально, достатньо адекватно позначити мету. Він володіє своєю сферою відповідальності, розуміє контекст і здатний приймати рішення щодо реалізації з урахуванням як технічних нюансів, так і погляду з боку бізнесу та користувачів. Це рівень більшості програмістів.[15]

Senior-розробник сам може розповісти, що потрібно робити і чому. Він знає свою частину проєкту, розуміє та формує напрямок її розвитку. Він здатний пам'ятати великі та складні завдання, працювати на високому рівні абстракції та враховувати неявні взаємодії та наслідки. Зазвичай, на цьому рівні виникає спеціалізація — хтось займається складним проектуванням, хтось знає глибокі особливості величезної кількості інструментів та фреймворків, хтось може організувати роботу над великим завданням.[15]

Рівень компетенції буде вимірюватися на проміжку від 0 до 1. Значення 0 буде відповідати Junior рівню, 0,5 – Middle та 1 - Senior. А кращим буде вважатися значення, яке відповідає мінімальному рівню навичок необхідних для виконання завдання.

Спеціалізація

Веб-розробка має тенденцію розбиватися на три основні концентрації[16]:

- Front-end,
- Back-end,
- Full-stack.

Розробники Front-end відповідають за ефективне впровадження візуальних компонентів на веб-сайті. Вони також виконують важливі завдання з розробки веб-сайту, такі як навігацію, кнопки і таке інше, що допомагає покращити загальну видимість веб-сайту. Для цього часто використовуються HTML, JavaScript і CSS, щоб забезпечити безперебійну роботу веб-сайту. Це дозволяє користувачам вільно та комфортно взаємодіяти з веб-сайтом.[17]

Технології, необхідні для Front-end розробки[17]:

- HTML (Hyper Text Markup Language) — це мова, яка використовується у всесвітній мережі. Це стандартна мова форматування тексту, яка використовується для створення та відображення сторінок в Інтернеті. Файли HTML складаються з двох речей: вмісту та тегів, які форматують його для належного відображення на сторінках.
- CSS — це каскадні таблиці стилів. Це мова стилів, яка досить проста для елементів HTML. Він популярний у веб-дизайні, його застосування також поширене в XHTML.
- JavaScript - відкритий вихідний код і найпопулярніша мова сценаріїв на стороні клієнта, яка підтримується багатьма браузерами.

JavaScript використовується для покращення взаємодії веб-сайту з користувачем.

Ролі та відповідальність Front-end розробників[17]:

- Визначає компоненти на сторінці за допомогою HTML
- Зробіть їх привабливим за допомогою CSS.
- Працює над інтерактивністю з JavaScript.
- Підвищує продуктивність за допомогою фреймворків.
- Він відповідає за виконання таких завдань, як контроль версій, автоматизація, системи керування вмістом тощо.
- Аналізує ефективність веб-сторінки на стороні клієнта, щоб краще зрозуміти споживчий досвід.

Back-end розробники працюють над розробкою на стороні сервера. Вони зосереджені на базах даних, сценаріях та архітектурі веб-сайту.[17]

Ці фахівці також стежать за тим, як працює сайт, як вносити зміни та оновлювати всі закулісні функції.

Технології, необхідні для Back-end розробки[17]:

- Мови веб-розробки
- База даних і кеш
- Сервер
- API (REST & SOAP)

а) Мови веб-розробки:

Backend розробка включає одну серверну мову програмування, таку як Java, Python, Ruby, .Net тощо.[17]

PHP: PHP — це мова сценаріїв на стороні сервера, призначена для веб-розробки. Оскільки PHP-код пишеться і виконується на стороні сервера, його називають мовою сценаріїв на стороні сервера.[17]

C++: C++ — це об'єктно-орієнтована мова програмування загального призначення. Він широко використовується як серверна мова програмування.

Java: Java — це об'єктно-орієнтована мова програмування на основі класів для створення веб- та настільних додатків. Він дуже масштабований. Компоненти Java легко доступні для використання.[17]

Python: Python – це об’єктно-орієнтована мова програмування, яка дозволяє швидко працювати та ефективніше інтегрувати системи.[17]

Node.js: Node.JS — це кросплатформне середовище виконання з відкритим вихідним кодом, яке використовується для розробки веб-додатків на стороні сервера. Програми Node.JS написані на JavaScript і працюють на широкому спектрі операційних систем.[17]

SASS: це надійна, зріла та надійна мова розширення CSS. Він використовується для розширення існуючої функціональності CSS, включаючи все, від змінних, успадкування та вкладення з легкістю.[17]

б) База даних і кеш:

Внутрішня розробка також включає в себе різні технології СУБД як важливі серверні бази даних, такі як MySQL , MongoDB , Oracle, SQLServer і Redis.[17]

в) Сервер:

Бажано включати серверні технології сайту, такі як Apache , Nginx, сервери IIS, Microsoft IIS тощо. Знання Linux також допомагає в адмініструванні серверів.[17]

г) API (REST & SOAP):

API також важливий для Backend розробки. Важливими є технічні знання щодо створення та використання сервісів REST та SOAP.[17]

Також [17]:

1) Досвід роботи з такими фреймворками, як Django для Python, Laravel для PHP тощо.
2) Вміння писати якісні модульні тести.
3) Технічне знання алгоритмів і структур даних також є важливою потребою для будь-якого професійного розробника серверної частини.

4) Здатний знати відмінності між кількома платформами доставки, такими як мобільні та настільні.
5) Бажано базові технічні знання інтерфейсу, такі як HTML і CSS
6) Повинен мати технічні знання з управління сесіями в середовищі розподіленого сервера.

Ролі та обов'язки Back-end розробників[17]:

1) Back-end розробникам потрібно динамічно створювати компоненти та вміст сторінки на будь-якому веб-сервері.
2) Створення веб-сторінок за допомогою програмування на мовах PHP, Java, JavaScript, Perl, Python і Ruby.
3) Робота з контролем версій програмного забезпечення за допомогою таких технологій, як CVS, Git або SVN.
4) Back-end розробники повинні розуміти цілі веб-сайту та знаходити ефективні рішення.
5) Керування ресурсами API, які працюють на різних пристроях.
6) Back-end розробники може брати участь в архітектурі системи та аналізі науки про дані.
7) Back-end розробники відповідають за організацію логіки системи, яка працює на різних пристроях.
8) Back-end розробники також потрібно створити фреймворки або архітектуру.

Різниця між Front-end і Back-end розробниками [17](табл. 3.1):

Таблиця 3.1 – Різниця між Front-end і Back-end розробниками.

Параметр	Front-end	Back-end
Визначення	Front-end розробники відповідають за ефективне впровадження візуальних компонентів на веб-сайті.	Back-end розробники працюють над розробкою на стороні сервера. Вони зосереджені на базах даних, сценаріях, архітектурі веб-сайту.

Набори навичок	Мови, з якими повинен бути знайомий веб-розробник, це HTML, CSS та JavaScript.	Внутрішні мови програмування, такі як PHP, Java, .Net та знання бази даних, сервера, API тощо.
Команда	Front-end розробники створюють зовнішній вигляд веб-сайту, беручи дані користувачів і змінюючи їх шляхом тестування.	Back-end розробники розробляють додаток, який підтримує інтерфейс. Вони також повинні забезпечувати підтримку, безпеку та керування вмістом.
Автономний сервіс	Послугу Front-end розробки не можна запропонувати самостійно.	Back-end розробка може бути запропонована як самостійна послуга в BaaS (Back-end as a service).
Мета	Front-end розробники повинні переконатися, що веб-сайт доступний для всіх користувачів. Він реагує у всіх режимах перегляду – на мобільних і настільних ПК.	Команді Back-end розробників може знадобитися створити додаток навколо інтерфейсу та підтримувати його. Крім того, вони повинні переконатися, що веб-сайт відкривається та працює належним чином.
Повинен мати	Вони повинні розуміти, як ефективно працювати над дизайном та UI/UX веб-сайту чи програми.	Back-end розробники повинні вміти реалізовувати алгоритми та вирішувати системні проблеми.
Середня зарплата	104 405 доларів США на рік	120 798 доларів США на рік
Найпопулярніші інструменти	jQuery та HTML5	MySQL і PHP.

Full-stack розробники працюють, подібно до Back-end розробників але вони також можуть вільно виконувати роботу Front-end розробників.[16]

Спеціалізація буде вимірюватися на проміжку від 0 до 1. Front-end буде відповідати значення 0, Back-end – 1 та Full-stack - 0,5. Кращим значення буде вважатися Full-stack або те, яке відповідає вимогам завдання.

Обізнаність з проєктом

Навіть найдосвідченіший програміст витрачає час на те, щоб розібратися з новим для нього проєктом. Тому задля економії часу та сил розробників при виборі виконавця задачі менеджер надасть перевагу тому, хто вже працював над цим проєктом.

Обізнаність з проєктом буде вимірюватися на проміжку від 0 до 1. Для даного критерія значення буде визначатися шляхом перевірки, чи виступає розробник у ролі відповідального або співвиконавця у будь-якому завданні проєкту. У позитивному випадку критерій буде приймати значення 1, у негативному – 0. Кращим значенням буде вважатися 1.

3.2 ПОБУДОВА МОДЕЛІ

Нехай менеджер має обрати, якого програміста призначити для виконання наступного завдання: Внести правки по дизайну сайту. Проєкт ПП «Еко». На виконання завдання виділяється 6 годин. Рівень компетенції, необхідний для виконання завдання – Middle. Спеціалізація – Front-end.

Обрати виконавця завдання менеджер може серед наступних розробників:

Таблиця 3.2 – Альтернативи (x_i) та значення критеріїв (R_j)

	Ім’я та прізвище	Рівень компетенції	Спеціалізація	Проєкти	Залишок часу
x_1	Антон Броварчук	Junior	Front-end	ПрАТ "Мульти" ПрАТ "Вектор"	3:00
x_2	Тамара Васильєва	Junior	Back-end	ПрАТ "Мегамакс"	5:00

x_3	Любов Васильєва	Junior	Full-stack	ПАТ "Планета-Дизайн"	13:00
x_4	Тарас Захарчук	Middle	Front-end	ТОВ "Вектор-Стиль"	5:00
x_5	Ярослав Іванченко	Middle	Back-end	Едельвейс	10:00
x_6	Ілля Мельниченко	Middle	Full-stack	Альфа-Сервіс	20:00
x_7	Віра Середа	Senior	Front-end	ПрАТ "Майнер-Дизайн", ПрАТ "Вектор"	5:10
x_8	Руслан Таращук	Senior	Back-end	ПП "Еко"	20:00
x_9	Роман Шевченко	Senior	Full-stack	ТОВ "Мега"	30:00

Альтернативи будуть оцінюватися за наступними критеріями:

- R_1 - Рівень компетенції,
- R_2 - Спеціалізація,
- R_3 - Обізнаність з проєктом,
- R_4 - Завантаженість програміста.

Запишемо значення критеріїв для кожної альтернативи у числову оцінку:

Таблиця 3.3 – Числова оцінка критеріїв

	R_1	R_2	R_3	R_4
x_1	0	0	0	3:00
x_2	0	1	0	5:00
x_3	0	0,5	0	13:00
x_4	0,5	0	0	5:00
x_5	0,5	1	0	10:00
x_6	0,5	0,5	0	20:00
x_7	1	0	0	5:10
x_8	1	1	1	20:00
x_9	1	0,5	0	30:00

Встановимо відношення переваги критеріїв на множині альтернатив:

– $R_1: x_1 \approx x_2, x_2 \approx x_3, x_3 \prec x_4, x_4 \approx x_5, x_5 \approx x_6, x_6 \succcurlyeq x_7, x_7 \approx x_8,$
$x_8 \approx x_9, x_1 \prec x_9,$

– $R_2: x_1 \succ x_2, x_1 \approx x_3, x_3 \approx x_4, x_2 \approx x_5, x_4 \approx x_6, x_6 \approx x_7, x_2 \approx x_8,$
$x_7 \approx x_9, x_1 \approx x_9,$

– $R_3: x_1 \approx x_2, x_2 \approx x_3, x_3 \approx x_4, x_4 \approx x_5, x_5 \approx x_6, x_6 \approx x_7, x_7 \prec x_8,$
$x_8 \succ x_9, x_1 \approx x_9,$

– $R_4: x_1 \succ x_2, x_4 \succ x_7, x_2 \approx x_4, x_3 \prec x_5, x_4 \succcurlyeq x_7, x_5 \prec x_7, x_6 \prec x_3,$
$x_8 \approx x_6, x_8 \succ x_9.$

Мета – знайти найкраще компромісне рішення за сукупністю критеріїв, використовуючи згортки Q_1, Q_2. При цьому для Q_2 взяти ваги критеріїв ω_1=0.5, ω_2=0.2, ω_3=0.2, ω_4=0.1.

Побудуємо матрицю відношення R_1. Вважатимемо всі відношення транзитивними.

Таблиця 3.4 – Матриця відношення R_1.

$x_i \setminus x_j$	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9
x_1	1	1	1	0	0	0	0	0	0
x_2	1	1	1	0	0	0	0	0	0
x_3	1	1	1	0	0	0	0	0	0
x_4	1	1	1	1	1	1	1	1	1
x_5	1	1	1	1	1	1	1	1	1
x_6	1	1	1	1	1	1	1	1	1
x_7	1	1	1	0	0	0	1	1	1
x_8	1	1	1	0	0	0	1	1	1
x_9	1	1	1	0	0	0	1	1	1

Аналогічно будуємо матриці відношень R_2, R_3, R_4.

Таблиця 3.5 – Матриця відношення R_2.

$x_i \setminus x_j$	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9
x_1	1	1	1	1	1	1	1	1	1
x_2	0	1	0	0	1	0	0	1	0
x_3	1	1	1	1	1	1	1	1	1
x_4	1	1	1	1	1	1	1	1	1
x_5	0	1	0	0	1	0	0	1	0
x_6	1	1	1	1	1	1	1	1	1
x_7	1	1	1	1	1	1	1	1	1
x_8	0	1	0	0	1	0	0	1	0
x_9	1	1	1	1	1	1	1	1	1

Таблиця 3.6 – Матриця відношення R_3.

$x_i \setminus x_j$	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9
x_1	1	1	1	1	1	1	1	0	1
x_2	1	1	1	1	1	1	1	0	1
x_3	1	1	1	1	1	1	1	0	1
x_4	1	1	1	1	1	1	1	0	1
x_5	1	1	1	1	1	1	1	0	1
x_6	1	1	1	1	1	1	1	0	1
x_7	1	1	1	1	1	1	1	0	1
x_8	1	1	1	1	1	1	1	1	1
x_9	1	1	1	1	1	1	1	0	1

Таблиця 3.7 – Матриця відношення R_4.

$x_i \setminus x_j$	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9
x_1	1	1	1	1	1	1	1	1	1
x_2	0	1	1	1	1	1	1	1	1
x_3	0	0	1	0	0	1	0	1	1
x_4	0	1	1	1	1	1	1	1	1
x_5	0	0	1	0	1	1	0	1	1

x_6	0	0	0	0	0	1	0	1	1
x_7	0	0	1	0	1	1	1	1	1
x_8	0	0	0	0	0	1	0	1	1
x_9	0	0	0	0	0	0	0	0	1

Будуємо згортку відношень $Q_1 = R_1 \cap R_2 \cap R_3 \cap R_4$.

Таблиця 3.8 – Згортка відношень Q_1.

$x_i \setminus x_j$	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9
x_1	1	1	1	0	0	0	0	0	0
x_2	0	1	0	0	0	0	0	0	0
x_3	0	0	1	0	0	0	0	0	0
x_4	0	1	1	1	1	1	1	0	1
x_5	0	0	0	0	1	0	0	0	0
x_6	0	0	0	0	0	1	0	0	1
x_7	0	0	1	0	0	0	1	0	1
x_8	0	0	0	0	0	0	0	1	0
x_9	0	0	0	0	0	0	0	0	1

Знаходимо множину недомінованих альтернатив:

$$\mu_{Q_1}^{\text{нд}}(x_1) = 1 - sup\{0-1; 0-1; 0-0; 0-0; 0-0; 0-0; 0-0; 0-0\} = 1;$$
$$\mu_{Q_1}^{\text{нд}}(x_2) = 1 - sup\{1-0; 0-0; 1-0; 0-0; 0-0; 0-0; 0-0; 0-0\} = 0;$$
$$\mu_{Q_1}^{\text{нд}}(x_3) = 1 - sup\{1-0; 0-0; 1-0; 0-0; 0-0; 1-0; 0-0; 0-0\} = 0;$$
$$\mu_{Q_1}^{\text{нд}}(x_4) = 1 - sup\{0-0; 0-1; 0-1; 0-1; 0-1; 0-1; 0-0; 0-1\} = 1;$$
$$\mu_{Q_1}^{\text{нд}}(x_5) = 1 - sup\{0-0; 0-0; 0-0; 1-0; 0-0; 0-0; 0-0; 0-1\} = 0;$$
$$\mu_{Q_1}^{\text{нд}}(x_6) = 1 - sup\{0-0; 0-0; 0-0; 1-0; 0-0; 0-0; 0-0; 0-1\} = 0;$$
$$\mu_{Q_1}^{\text{нд}}(x_7) = 1 - sup\{0-0; 0-0; 0-1; 1-0; 0-0; 0-0; 0-0; 0-1\} = 0;$$
$$\mu_{Q_1}^{\text{нд}}(x_8) = 1 - sup\{0-0; 0-0; 0-0; 0-0; 0-0; 0-0; 0-0; 0-0\} = 1;$$

$$\mu_{Q_1}^{\text{нд}}(x_9) = 1 - sup\{0-0; 0-0; 0-0; 1-0; 0-0; 1-0; 1-0; 0-0\} = 0.$$

Отже, $\mu_{Q_1}^{\text{нд}}(x) = [1; 0; 0; 1; 0; 0; 0; 1; 0]$.

Будуємо н.в.п. Q_2 (адитивну згортку відношень R_j), (2.39)

Його функція належності:

Таблиця 3.9 – Згортка відношень Q_2.

$x_i \setminus x_j$	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9
x_1	1	1	1	0,5	0,5	0,5	0,5	0,3	0,5
x_2	0,7	1	0,8	0,3	0,5	0,3	0,3	0,3	0,3
x_3	0,9	0,9	1	0,4	0,4	0,5	0,4	0,3	0,5
x_4	0,9	1	1	1	1	1	1	0,8	1
x_5	0,7	0,9	0,8	0,7	1	0,8	0,7	0,8	0,8
x_6	0,9	0,9	0,9	0,9	0,9	1	0,9	0,8	1
x_7	0,9	0,9	1	0,4	0,5	0,5	1	0,8	1
x_8	0,7	0,9	0,7	0,2	0,4	0,3	0,7	1	0,8
x_9	0,9	0,9	0,9	0,4	0,4	0,4	0,9	0,7	1

Знаходимо підмножину недомінованих альтернатив для відношення Q_2:

$$\mu_{Q_2}^{\text{нд}}(x_1) = 1 - sup\begin{Bmatrix} 0.7-1; 0.9-1; 0.9-0.5; 0.7-0.5; \\ 0.9-0.5; 0.9-0.5; 0.7-0.3; 0.9-0.5 \end{Bmatrix} = 0.6;$$

$$\mu_{Q_2}^{\text{нд}}(x_2) = 1 - sup\begin{Bmatrix} 1-0.7; 0.9-0.8; 1-0.3; 0.9-0.5; \\ 0.9-0.3; 0.9-0.3; 0.9-0.3; 0.9-0.3 \end{Bmatrix} = 0.3;$$

$$\mu_{Q_2}^{\text{нд}}(x_3) = 1 - sup\begin{Bmatrix} 1-0.9; 0.8-0.9; 1-0.4; 0.8-0.4; \\ 0.9-0.5; 1-0.4; 0.7-0.3; 0.9-0.5 \end{Bmatrix} = 0.4;$$

$$\mu_{Q_2}^{\text{нд}}(x_4) = 1 - sup\begin{Bmatrix} 0.5-0.9; 0.3-1; 0.4-1; 0.7-1; \\ 0.9-1; 0.4-1; 0.2-0.8; 0.4-1 \end{Bmatrix} = 1;$$

$$\mu_{Q_2}^{\text{нд}}(x_5) = 1 - sup\begin{Bmatrix} 0.5-0.7; 0.5-0.9; 0.4-0.8; 1-0.7; \\ 0.9-0.8; 0.5-0.7; 0.4-0.8; 0.4-0.8 \end{Bmatrix} = 0.7;$$

$$\mu_{Q_2}^{\text{нд}}(x_6) = 1 - sup\begin{Bmatrix} 0.5-0.9; 0.3-0.9; 0.5-0.9; 1-0.9; \\ 0.8-0.9; 0.5-0.9; 0.3-0.8; 0.4-1 \end{Bmatrix} = 0.9;$$

$$\mu_{Q_2}^{\text{нд}}(x_7) = 1 - sup\begin{Bmatrix} 0.5-0.9; 0.3-0.9; 0.3-1; 1-0.4; \\ 0.7-0.5; 0.9-0.5; 0.7-0.8; 0.9-1 \end{Bmatrix} = 0.4;$$

$$\mu_{Q_2}^{\text{нд}}(x_8) = 1 - sup\begin{Bmatrix} 0.3-0.7; 0.3-0.9; 0.3-0.7; 0.8-0.2; \\ 0.8-0.4; 0.8-0.3; 0.8-0.7; 0.7-0.8 \end{Bmatrix} = 0.4;$$

$$\mu_{Q_2}^{\text{нд}}(x_9) = 1 - sup\begin{Bmatrix} 0.5-0.9; 0.3-0.9; 0.5-0.9; 1-0.4; \\ 0.8-0.4; 1-0.4; 1-0.9; 0.8-0.7 \end{Bmatrix} = 0.4.$$

Отже, $\mu_{Q_2}^{\text{нд}}(x) = [0.6; 0.3; 0.4; 1; 0.7; 0.9; 0.4; 0.4; 0.4]$.

Знаходимо перетин множин $Q_1^{\text{нд}}, Q_2^{\text{нд}}$ і обчислюємо функцію належності результуючої підмножини $Q_{\text{нд}} = Q_1^{\text{нд}} \cap Q_2^{\text{нд}}$. Її функція належності дорівнює $\mu_Q^{\text{нд}}(x_i) = \min\{\mu_{Q_1}^{\text{нд}}(x_i), \mu_{Q_2}^{\text{нд}}(x_i)\}$;

$$\mu_Q^{\text{нд}}(x_i) = [0.6; 0; 0; 1; 0; 0; 0; 0.4; 0];$$

Найбільший ступінь недомінованості має альтернатива x_4.

Отже, на роль виконавця поставленої задачі найкраще підходить Тарас Захарчук – програміст Middle рівня, що спеціалізується на Front-end розробці, не виконував задачі проєкту ПП «Еко», проте має досить малу завантаженість, порівняно з іншими.

Також можна виділити альтернативи x_1 та x_8. Це розробники Антон Броварчук та Руслан Таращук відповідно. Антон Броварчук спеціалізується на Front-end розробці, має найменшу завантаженість, проте не виконував задачі проєкту ПП «Еко» та має недостатній рівень навичок – Junior. Руслан Таращук – розробник Senior рівня, мав справу з проєктом ПП «Еко», однак має велику завантаженість та спеціалізується на Back-end розробці.

4 РОЗРОБКА ЗАСТОСУНКА ДЛЯ ФОРМУВАННЯ ОПТИМАЛЬНОГО ШТАТУ ВИКОНАВЦІВ-РОЗРОБНИКІВ ІТ-КОМПАНІЇ

4.1 РОЗРОБКА ЗАСТОСУНКА

Побудувавши математичну модель вибору програміста для вирішення задачі, було розпочато розробку програмного рішення для перевірки ефективності моделі на практиці.

Програмний застосунок було написано під «Бітрікс24» – сервіс для керування бізнесом, який на сьогоднішній день користується великим попитом. Сервіс має дві версії: хмарну та коробкову. На відміну від хмарної, коробкову версію встановлюють на власний сервер, що дає можливість кастомізувати функціонал під індивідуальні потреби. Аби якомога більша кількість користувачів змогла скористатися застосунком, було прийнято рішення створити програму-інтеграцію, яку можна встановлювати як на коробковій версії сервісу, так і на хмарній.

Під локальними програмами розуміються програми, які описуються і додаються безпосередньо на конкретний Бітрікс24. Адміністратор порталу вирішує, які права надати такому додатку, як він називатиметься в інтерфейсі Бітрікс24. Саме в цьому суть терміну "локальний" (при цьому технології, на яких розроблено конкретну програму, цілком можуть бути серверними).[18]

В основі дизайну застосунка було використано безкоштовний відкритий HTML, CSS і JS фреймворк «Bootstrap». Функціонування клієнтської частини забезпечує HTML, CSS, JavaScript, jQuery, а серверної – PHP.

Для написання програмного застосунка було використано відкритий REST API. REST – концепція (архітектура) для організації взаємодії між незалежними об'єктами (додатками) у вигляді протоколу HTTP. API – інтерфейс взаємодії з будь-яким об'єктом (програмою, додатком), що включає набір правил, які дозволяють одному додатку спілкуватися з іншим. Ці «правила» можуть включати операції створення, читання, оновлення та видалення. REST API

дозволяє застосунку взаємодіяти з одним або декількома різними програмами, використовуючи концепції REST.[19]

Програма використовує базовий SDK як клас CRest для виконання запитів та розширення токенів аутентифікації. Стандартно бібліотека CRest працює під користувачем, який встановив програму. Але нам необхідно виконувати запит до порталу Бітрікс24 під користувачем, чиї токени прийшли на сторінку в POST. Для цього було успадковано методи класу CRest [20](рис. 4.1). Повний текст програмного коду можна побачити в додатку Б.

```
require_once(__DIR__ . '/crest.php');
class CRestCurrent extends CRest
{
    protected static $dataExt = [];
    protected static function getSettingData()
    {
        $return = static::expandData(file_get_contents(__DIR__ . '/settings.json'));
        if(is_array($return))
        {
            if(!empty(static::$dataExt))
            {
                $return['access_token'] = htmlspecialchars(static::$dataExt['AUTH_ID']);
                $return['domain'] = htmlspecialchars(static::$dataExt['DOMAIN']);
                $return['refresh_token'] = htmlspecialchars(static::$dataExt['REFRESH_ID']);
                $return['application_token'] = htmlspecialchars(static::$dataExt['APP_SID']);
            }
            else
            {
                $return['access_token'] = htmlspecialchars($_REQUEST['AUTH_ID']);
                $return['domain'] = htmlspecialchars($_REQUEST['DOMAIN']);
                $return['refresh_token'] = htmlspecialchars($_REQUEST['REFRESH_ID']);
                $return['application_token'] = htmlspecialchars($_REQUEST['APP_SID']);
            }
        }
        return $return;
    }
    public static function setDataExt($data)
    {
        static::$dataExt = $data;
    }
}
```

Рисунок 4.1 – Клас CRestCurrent.

У картці користувача було створено два додаткових користувацьких поля типу «Список»: «Рівень компетенції» та «Спеціалізація» з назвами UF_USR_1636807293198 та UF_USR_1636807393886 відповідно. Поля заповнюються відповідно до рівня компетенції та спеціалізації кожного розробника (рис. 4.2).

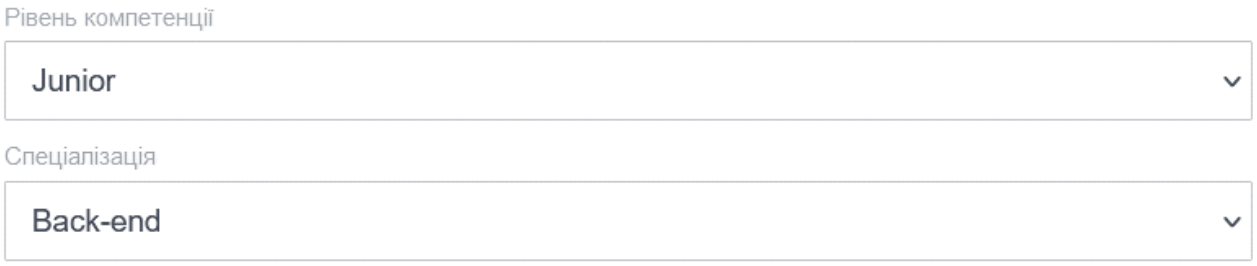


Рисунок 4.2 – Додаткові користувацькі поля «Рівень компетенції» та «Спеціалізація».

Робимо вибірку усіх розробників (сортуємо їх за полем «Рівень компетенції» за зростанням, обираємо поля «ID», «Ім’я», «Прізвище», «Рівень компетенції» та «Спеціалізація»), задач (які знаходяться у незавершених статусах) та проєктів (рис. 4.3).

```
require_once(__DIR__.'/crestcurrent.php');
$users = CRestCurrent::call(
    "user.get",
    array(
        "SORT" => "UF_USR_1636807293198",
        "ORDER" => "ASC",
        "FILTER" => array("UF_DEPARTMENT" => array(3))
        "SELECT" => array("ID", "NAME", "LAST_NAME", "UF_USR_1636807293198",
                "UF_USR_1636807393886")
    ),
    false,
    true
);
$tasks = CRestCurrent::call(
    "tasks.task.list",
    array(
        "FILTER" => array('STATUS' => array(1, 2, 3)),
    ),
    false,
    true
);
$groups = CRestCurrent::call(
    "sonet_group.get",
    array(),
    false,
    true
);
```

Рисунок 4.3 – Вибірка розробників, задач та проєктів

Формуємо масив $userArray (рис. 4.4).

```
$usersID = array();
$userArray = array();
$level_of_competence = array(44 => 0, 46 => 0.5, 48 => 1);
$level_of_competence_names = array(44 => 'Junior', 46 => 'Middle', 48 => 'Senior');
$specialization = array(50 => 0, 54 => 0.5, 52 => 1);
$specialization_names = array(50 => 'Front-end', 54 => 'Full-stack', 52 => 'Back-end');
foreach($users['result'] as $user){
    $fullName = $user['NAME']." ".$user['LAST_NAME'];
    $fullNameArr = str_split($fullName);
    $fullNameArr = array_pad($fullNameArr, 38, " ");
    $userArray['USERS'][$user['ID']]['FULL_NAME'] = implode("", $fullNameArr);
    $userArray['USERS'][$user['ID']]['ID'] = $user['ID'];
    $userArray['USERS'][$user['ID']]['TIME_LEFT'] = 0;
    $userArray['USERS'][$user['ID']]['LEVEL_OF_COMPETENCE'] =
        $level_of_competence_names[$user['UF_USR_1636807293198']];
    $userArray['USERS'][$user['ID']]['LEVEL_OF_COMPETENCE_VALUE'] =
        $level_of_competence[$user['UF_USR_1636807293198']];
    $userArray['USERS'][$user['ID']]['SPECIALIZATION'] =
        $specialization_names[$user['UF_USR_1636807393886']];
    $userArray['USERS'][$user['ID']]['SPECIALIZATION_VALUE'] =
        $specialization[$user['UF_USR_1636807393886']];
    $usersID[] = $user['ID'];
}
foreach ($groups['result'] as $group) {
    $userArray['GROUPS'][$group['ID']]['NAME'] = $group['NAME'];
}
foreach ($tasks['result']['tasks'] as $task) {
    array_push($task['accomplices'], $task['responsibleId']);
    $userArray['TASKS'][$task['id']]['USERS_ID'] = $task['accomplices'];
    unset($user);
    foreach ($task['accomplices'] as $user) {
        $userArray['USERS'][$user]['GROUPS'][] = $task['groupId'];
    }
    unset($user);
    $userArray['TASKS'][$task['id']]['TIME_ESTIMATE'] = $task['timeEstimate'];
    $userArray['TASKS'][$task['id']]['TIME_ELAPSED'] = 0;
}
```

Рисунок 4.4 – Формування масиву $userArray.

Робимо вибірку витраченого часу по задачам (сортуємо за датою створення за зростанням, фільтруємо за користувачами, від імен яких були зроблені записи

про витрачений час, обираємо поля «Секунди» та «ID задачі») та доповнюємо масив $userArray (рис. 4.5).

```
$elapsed = CRest::call(
    "task.elapseditem.getlist",
    array(
        "ORDER" => array("CREATED_DATE" => "asc"),
        "FILTER" => array("USER_ID" => $usersID),
        "SELECT" => array("SECONDS", "TASK_ID")
     )
);
foreach ($elapsed['result'] as $value) {
    if(!empty($userArray['TASKS'][$value['TASK_ID']])){
        $userArray['TASKS'][$value['TASK_ID']]['TIME_ELAPSED'] += $value['SECONDS'];
    }
}
foreach ($userArray['TASKS'] as &$task) {
    $task['TIME_LEFT'] = $task['TIME_ESTIMATE'] - $task['TIME_ELAPSED'];
    foreach ($task['USERS_ID'] as $user) {
        $userArray['USERS'][$user]['TIME_LEFT'] += $task['TIME_LEFT'];
    }
}
```

Рисунок 4.5 – Витрачений час.

Вигляд встановленого застосунка зображено на рисунку 4.6. Пункти списку «Проєкт» формуються з масиву $userArray['GROUPS'].

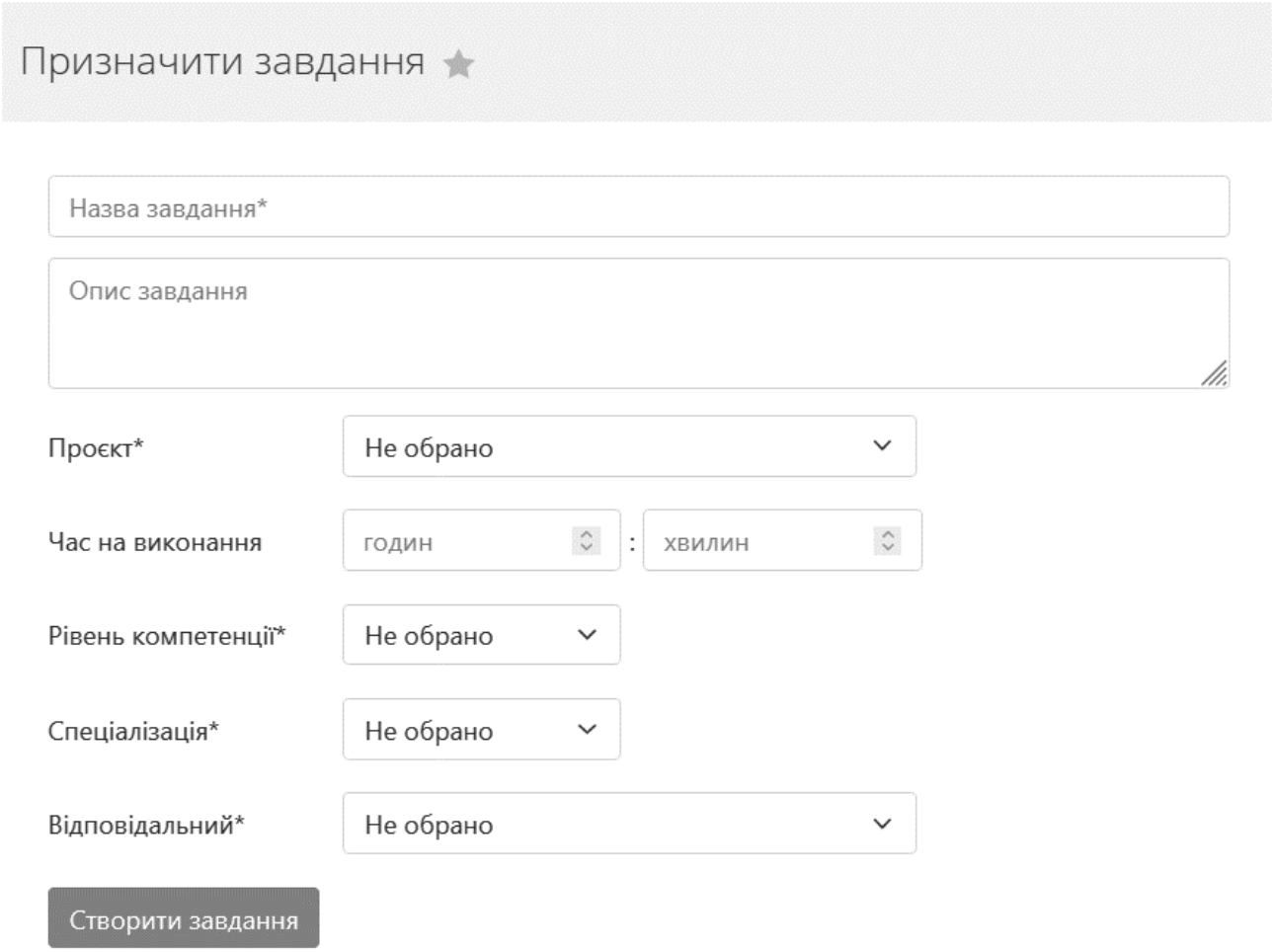


Рисунок 4.6 – Форма створення завдання.

Для створення завдання обов’язковими до заповнення є поля «Назва завдання», «Проєкт», «Рівень компетенції», «Спеціалізація» та «Відповідальний». Обрати відповідального можливо лише у разі заповнення полів «Проєкт», «Рівень компетенції» та «Спеціалізація», тоді список розробників сортується наступним чином: вище відображаються найкращі кандидати, а нижче усі інші.

Визначення оптимальних кандидатів відбувається за алгоритмом, описаним у 3 розділі.

Події фокусування на списку розробників було призначено обробник подій (рис. 4.7):

```
$(document).ready(function () {
    $(document).on('focus', '#users', selectFocus);
});
```

Рисунок 4.7 – Призначення обробника подій.

Обробник має наступну послідовність дій:

1) записує у змінні значення полів «Проєкт», «Рівень компетенції» та «Спеціалізація» (рис. 4.8);

```
var selectFocus = function(event) {
    var serializeArray = $("#newTask").serializeArray();
    var group = serializeArray[2].value;
    var level_of_competence = serializeArray[5].value;
    var specialization = serializeArray[6].value;

    . . . . . .

}
```

Рисунок 4.8 – Створення та заповнення змінних.

2) якщо усі необхідні поля заповнені, записується масив розробників, створюються масиви для подальших обрахунків, далі пункт 3, у протилежному випадку користувачу виводиться прохання заповнити потрібне поле (рис. 4.9);

```
var selectFocus = function(event) {

    .  .  .  .  .  .  .  .  .  .  .  .  .  .  .

    if(group.length!=0 && level_of_competence.length!=0 && specialization.length!=0){
        var userArray = Object.values(<?php echo json_encode($userArray['USERS']);?>);
        var r1 = [], r2 = [], r3 = [], r4 = [];

    .  .  .  .  .  .  .  .  .  .  .  .  .  .  .

    }
    else if(group.length==0){
        $("select[name=group]").focus();
        alert("Вкажіть проект.");
    }
    else if(level_of_competence.length==0){
        $("select[name=level_of_competence]").focus();
        alert("Вкажіть рівень компетенції.");
    }
    else if(specialization.length==0){
        $("select[name=specialization]").focus();
        alert("Вкажіть спеціалізацію.");
    }
}
```

Рисунок 4.9 – Розгалуження.

3) формує масиви r1, r2, r3 та r4, які відповідають матрицям відношення R_1, R_2, R_3 та R_4 відповідно (рис. 4.10-4.11);

```
if(group.length!=0 && level_of_competence.length!=0 && specialization.length!=0){

    .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .

    for (var i = 0; i < userArray.length; i++) {
        tempArr = Array(userArray.length);
        for (var j = 0; j < userArray.length; j++) {
            var iLOCV = userArray[i].LEVEL_OF_COMPETENCE_VALUE;
            var jLOCV = userArray[j].LEVEL_OF_COMPETENCE_VALUE;
            switch (level_of_competence) {
                case '0':
                    tempArr[j] = (iLOCV <= jLOCV) ? 1 : 0;
                    break;
                case '0.5':
                    if(iLOCV == 0.5) iLOCV += 1;
                    if(jLOCV == 0.5) jLOCV += 1;
                    tempArr[j] = (iLOCV >= jLOCV) ? 1 : 0;
                    break;
                case '1':
                    tempArr[j] = (iLOCV >= jLOCV) ? 1 : 0;
                    break;
            }
        }
        r1.splice(i, 0, tempArr);
    }
    for (var i = 0; i < userArray.length; i++) {
        tempArr = Array(userArray.length);
        for (var j = 0; j < userArray.length; j++) {
            var iSPV = userArray[i].SPECIALIZATION_VALUE;
            var jSPV = userArray[j].SPECIALIZATION_VALUE;
            switch (specialization) {
                case '0':
                    if(iSPV == 0) iSPV = 0.5;
                    if(jSPV == 0) jSPV = 0.5;
                    tempArr[j] = (iSPV <= jSPV) ? 1 : 0;
                    break;
                case '0.5':
                    if(iSPV == 1) iSPV = 0;
                    if(jSPV == 1) jSPV = 0;
                    tempArr[j] = (iSPV >= jSPV) ? 1 : 0;
                    break;
                case '1':
                    if(iSPV == 1) iSPV = 0.5;
                    if(jSPV == 1) jSPV = 0.5;
                    tempArr[j] = (iSPV >= jSPV) ? 1 : 0;
                    break;
            }
        }
        r2.splice(i, 0, tempArr);
    }
    .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .

}
```

Рисунок 4.10 – Формування масивів r1 та r2.

```
if(group.length!=0 && level_of_competence.length!=0 && specialization.length!=0){

    .  .  .  .  .  .  .  .  .  .  .  .  .  .  .

    for (var i = 0; i < userArray.length; i++) {
        tempArr = Array(userArray.length);
        for (var j = 0; j < userArray.length; j++) {
            var iGR = userArray[i].GROUPS;
            var jGR = userArray[j].GROUPS;
            iGR = iGR.includes(group) ? 1 : 0;
            jGR = jGR.includes(group) ? 1 : 0;
            tempArr[j] = (iGR >= jGR) ? 1 : 0;
        }
        r3.splice(i, 0, tempArr);
    }
    for (var i = 0; i < userArray.length; i++) {
        tempArr = Array(userArray.length);
        for (var j = 0; j < userArray.length; j++) {
            var iTL = userArray[i].TIME_LEFT;
            var jTL = userArray[j].TIME_LEFT;
            tempArr[j] = (iTL <= jTL) ? 1 : 0;
        }
        r4.splice(i, 0, tempArr);
    }

    .  .  .  .  .  .  .  .  .  .  .  .  .  .  .

}
```

Рисунок 4.11 – Формування масивів r3 та r4.

4) формує масиви q1 та mQ1, які відповідають згортці відношень Q_1 та множині недомінованих альтернатив $\mu_{Q_1}^{\text{нд}}(x)$ відповідно (рис. 4.12);

```
if(group.length!=0 && level_of_competence.length!=0 && specialization.length!=0){

    . . . . . .

    var q1 = [];
    for (var i = 0; i < userArray.length; i++) {
        tempArr = Array(userArray.length);
        for (var j = 0; j < userArray.length; j++) {
            tempArr[j] = Math.min(r1[i][j], r2[i][j], r3[i][j], r4[i][j]);
        }
        q1.splice(i, 0, tempArr);
    }
    var mQ1 = Array(userArray.length);
    for (var i = 0; i < userArray.length; i++) {
        var sup = 0;
        for (var j = 0; j < userArray.length; j++) {
            sup = (q1[j][i] - q1[i][j]) > sup ? q1[j][i] - q1[i][j] : sup;
        }
        mQ1[i] = 1 - sup;
    }

    . . . . . .

}
```

Рисунок 4.12 – Формування масивів q1 та mQ1.

5) формує масиви q2 та mQ2, які відповідають згортці відношень Q_2 та підмножині недомінованих альтернатив $\mu_{Q_2}^{\text{нд}}(x)$ відповідно (рис. 4.13);

```
if(group.length!=0 && level_of_competence.length!=0 && specialization.length!=0){

    .  .  .  .  .  .  .  .  .  .  .  .  .  .  .

    var q2 = [];
    for (var i = 0; i < userArray.length; i++) {
        tempArr = Array(userArray.length);
        for (var j = 0; j < userArray.length; j++) {
            tempArr[j] = r1[i][j]*0.5 + r2[i][j]*0.2 + r3[i][j]*0.2 + r4[i][j]*0.1;
        }
        q2.splice(i, 0, tempArr);
    }
    var mQ2 = Array(userArray.length);
    for (var i = 0; i < userArray.length; i++) {
        var sup = 0;
        for (var j = 0; j < userArray.length; j++) {
            sup = (q2[j][i] - q2[i][j]) > sup ? q2[j][i] - q2[i][j] : sup;
        }
        mQ2[i] = 1 - sup;
    }

    .  .  .  .  .  .  .  .  .  .  .  .  .  .  .

}
```

Рисунок 4.13 – Формування масивів q2 та mQ2.

6) формує масиви mQ, який відповідає функції належності перетину множин $Q_1^{\text{нд}}$ та $Q_2^{\text{нд}}$, створює масив sortUsersObj з даними про розробників, строки початку та кінця групи оптимальних кандидатів та сортує масив sortUsersObj за спаданням значення ступеня недомінованості (рис. 4.14);

```
if(group.length!=0 && level_of_competence.length!=0 && specialization.length!=0){

    .  .  .  .  .  .  .  .  .  .  .  .  .  .  .

    var mQ = Array(userArray.length);
    var sortUsersObj = Array(userArray.length);
    var startBestsGroup = "";
    var endBestsGroup = "";
    for (var i = 0; i < userArray.length; i++) {
        mQ[i] = Math.min(mQ1[i], mQ2[i]);
        if(mQ[i]>0 && startBestsGroup == "" && endBestsGroup == ""){
            startBestsGroup = "<option hidden>Не обрано</option>
                <optgroup style='background-color: #5555; padding: 5px' label='Найкращий вибір'>";
            endBestsGroup = "</optgroup>";
        }
        sortUsersObj[i] = {mQ: mQ[i], userArray: userArray[i]};
    }
    sortUsersObj.sort(function(a, b) {
        return b.mQ - a.mQ;
    });

    .  .  .  .  .  .  .  .  .  .  .  .  .  .  .

}
```

Рисунок 4.14 – Формування масиву mQ, створення та сорування масиву sortUsersObj.

7) створює та встановлює новий відсортований список відповідальних (рис. 4.15).

```
if(group.length!=0 && level_of_competence.length!=0 && specialization.length!=0){

    .  .  .  .  .  .  .  .  .  .  .  .  .  .  .

    var newOpt = "";
    sortUsersObj.forEach(function(user){
        if(user.mQ>0 && (
              String(user.userArray.SPECIALIZATION_VALUE) == specialization ||
              user.userArray.SPECIALIZATION_VALUE == 0.5
        )){
            startBestsGroup += "<option style='background-color: #5555; padding: 5px' value="
                  +user.userArray.ID+">"+user.userArray.FULL_NAME
                  +user.userArray.LEVEL_OF_COMPETENCE+"   "
                  +user.userArray.SPECIALIZATION+"</option>";
        }
        else{
            newOpt += "<option value="+user.userArray.ID+">"
                  +user.userArray.FULL_NAME+user.userArray.LEVEL_OF_COMPETENCE
                  +"   "+user.userArray.SPECIALIZATION+"</option>";
        }
    });
    $("#users")[0].innerHTML = startBestsGroup + endBestsGroup + newOpt;
}
```

Рисунок 4.15 – Новий відсортований список відповідальних.

У результаті роботи обробника користувачу відображається відсортований список відповідальних із зазначенням оптимальних кандидатів.

Після заповнення усіх необхідних полів форми (рис. 4.6) є можливість одразу створити задачу на порталі, натиснувши кнопку «Створити завдання». При відправці данних форми спрацьовує скрипт, який зупиняє перезавантаження сторінки, формує масив данних, які відправляє методом POST за допомогою функції $.ajax() у файл task_add.php. Якщо у відповідь прийшло ID, поля форми очищаються та користувачу виводиться повідомлення, що задачу створено. Якщо повертається «error», користувач побачить повідомлення «Задачу не створено.». (рис. 4.16)

```
function newTask(formID){
    event.preventDefault();
    var request = jQuery.param(<?php echo json_encode($_REQUEST);?>);
    $.ajax({
        url:     "task_add.php",
        type:    "POST",
        dataType: "html",
        data: $("#"+formID).serialize()+"&"+request,
        success: function(response) {
            if(response != "error"){
                $("#"+formID)[0].reset();
                alert("Створено нову задачу з ID "+response);
            }
            else alert("Задачу не створено.");
        }
    });
}
```

Рисунок 4.16 – Скрипт відправки данних форми.

У файлі task_add.php отримані данні формуються у масив $arFields (рис. 4.17), який передається в якості параметра у функцію створення завдання (рис. 4.18).

```
require_once(__DIR__.'/crestcurrent.php');
if (!empty($_POST)) {
    $level_of_competence = array(0 => 44, 0.5 => 46, 1 => 48);
    $specialization = array(0 => 50, 0.5 => 54, 1 => 52);
    if(!empty($_POST['task_title']))
        $arFields['TITLE'] = $_POST['task_title'];
    if(!empty($_POST['task_body']))
        $arFields['DESCRIPTION'] = $_POST['task_body'];
    if(!empty($_POST['group']))
        $arFields['GROUP_ID'] = intval($_POST['group']);
    if(!empty($_POST['level_of_competence']))
        $arFields['UF_USR_1636807293198'] =
            $level_of_competence[intval($_POST['level_of_competence'])];
    if(!empty($_POST['specialization']))
        $arFields['UF_USR_1636807393886'] = $specialization[intval($_POST['specialization'])];
    if(!empty($_POST['responsible']))
        $arFields['RESPONSIBLE_ID'] = intval($_POST['responsible']);
    if(!empty($_POST['time_estimate_hour']) || !empty($_POST['time_estimate_min'])){
        if(empty($_POST['time_estimate_hour'])) $_POST['time_estimate_hour'] = 0;
        if(empty($_POST['time_estimate_min'])) $_POST['time_estimate_min'] = 0;
        $arFields['TIME_ESTIMATE'] =
            intval($_POST['time_estimate_hour'])*3600+intval($_POST['time_estimate_min'])*60;
        $arFields['ALLOW_TIME_TRACKING'] = 'Y';
    }

    . . . . . .

}
```

Рисунок 4.17 – Формування масиву $arFields.

```
if (!empty($_POST)) {

    . . . . . .

    $tasks = CRestCurrent::call(
        "tasks.task.add",
        array(
            "fields" => $arFields
        ),
        false,
        true
    );

    . . . . . .

}
```

Рисунок 4.18 – Функція створення завдання.

Якщо завдання було створено, у відповідь повертається його ID, яке передається у відповідь функції $.ajax(). У разі, якщо завдання не будо створено, функції повертається «error». (рис. 4.19)

```
if (!empty($_POST)) {

    . . . . . .

    if(!empty($tasks['result'])){
        $response = intval($tasks['result']['task']['id']);
    }
    else{
        $response = "error";
    }
    echo json_encode($response);
}
```

Рисунок 4.19 – Формування відповіді функції $.ajax().

4.2 ПЕРЕВІРКА РОБОТИ ЗАСТОСУНКА

Для наочності, використано такі ж самі дані, як у розділі 3.

Створено список розробників та заповнено поля «Рівень компетенції» та «Спеціалізація» (рис. 4.20). Також створено проєкти та задачі (рис. 4.21). Слід зазначити, що Антон Броварчук є співвиконавцем у задачі з ID = 20.

ID	Ім'я та прізвище	Рівень компетенції	Спеціалізація
4	Ілля Мельниченко Не в мережі	Middle	Full-stack
6	Ярослав Іванченко Не в мережі	Middle	Back-end
8	Тарас Захарчук Не в мережі	Middle	Front-end
10	Руслан Таращук Не в мережі	Senior	Back-end
12	Антон Броварчук Не в мережі	Junior	Front-end
14	Роман Шевченко Не в мережі	Senior	Full-stack
16	Тамара Васильєва Не в мережі	Junior	Back-end
18	Любов Васильєва Не в мережі	Junior	Full-stack
20	Віра Середа Не в мережі	Senior	Front-end

Рисунок 4.20 – Список розробників.

ID	Відповідальний	Проєкт	Назва
20	Віра Середа	ПрАТ "Вектор"	Змінити колір 00:00 / 01:00
18	Антон Броварчук	ПрАТ "Мульти"	Доробити дизайн сайту 08:00 / 10:00
16	Тамара Васильєва	ПрАТ "Мегамакс"	Доробити функціонал сайту 10:00 / 15:00
14	Любов Васильєва	ПАТ "Планета-Дизайн"	Доробити сайт 17:00 / 30:00
12	Тарас Захарчук	ТОВ "Вектор-Стиль"	Створити дизайн сайту 13:00 / 18:00
10	Ярослав Іванченко	Едельвейс	Створити функціонал сайту 20:00 / 30:00
8	Ілля Мельниченко	Альфа-Сервіс	Створити сайт 35:00 / 55:00
6	Віра Середа	ПрАТ "Майнер-Дизайн"	Створити дизайн сайту 16:00 / 20:10
4	Руслан Таращук	ПП "Еко"	Створити функціонал сайту 20:00 / 40:00
2	Роман Шевченко	ТОВ "Мега"	Створити сайт 20:00 / 50:00

Рисунок 4.21 – Список завдань.

Переходимо до застосунка. Заповнюємо поля «Назва завдання», «Проєкт», «Рівень компетенції» та «Спеціалізація» наступним чином (рис. 4.22):

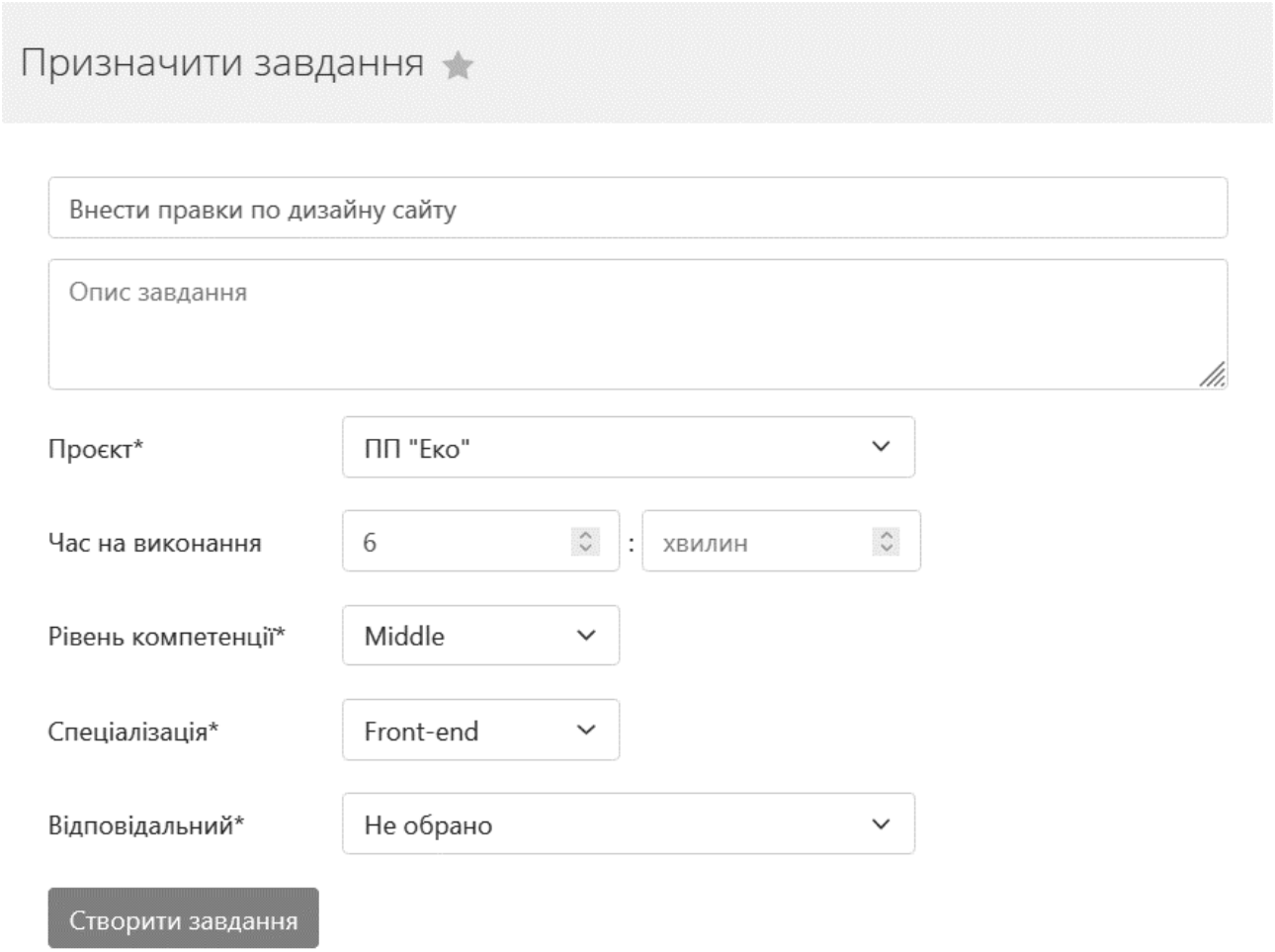


Рисунок 4.22 – Заповнені поля.

Переходимо до поля «Відповідальний». Бачимо, що сформувався список з виділеною групою «Найкращий вибір», до якої входять два розробника: Тарас Захарчук та Антон Броварчук. Також третім у списку є Руслан Таращук, але його не включено в групу через непідходящу спеціалізацію. (рис. 4.23)

Можна побачити, что результат роботи скрипта збігається з результатом підрахунків у 3 розділі.

Обираємо відповідальним Тараса Захарчука та натискаємо «Створити завдання». Отримуємо повідомлення, що створено задачу з ID = 38 (рис. 4.24).

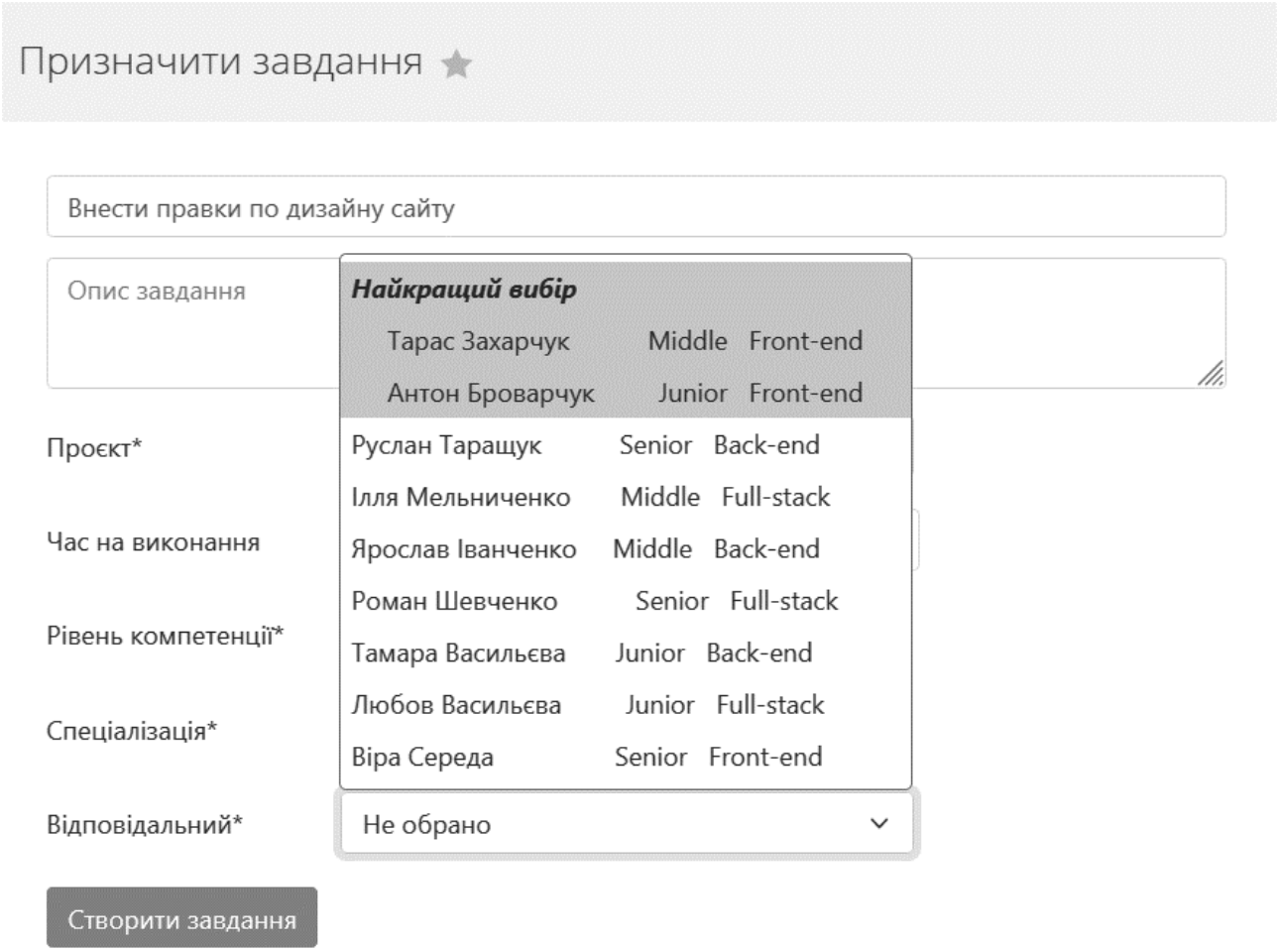


Рисунок 4.23 – Результат роботи скрипта.

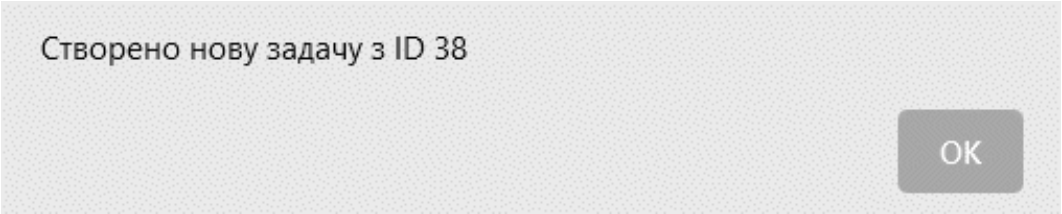


Рисунок 4.23 – Результат роботи скрипта.

Перевіривши список задач, бачимо нове завдання з ID = 38 та усіма даними з полів форми (рис. 4.24).

ID	Відповідальний	Проєкт	Назва
38	Тарас Захарчук	ПП "Еко"	Внести правки по дизайну сайту 00:00 / 06:00
20	Віра Середа	ПрАТ "Вектор"	Змінити колір 00:00 / 01:00
18	Антон Броварчук	ПрАТ "Мульти"	Доробити дизайн сайту 08:00 / 10:00
16	Тамара Васильєва	ПрАТ "Мегамакс"	Доробити функціонал сайту 10:00 / 15:00
14	Любов Васильєва	ПАТ "Планета-Дизайн"	Доробити сайт 17:00 / 30:00
12	Тарас Захарчук	ТОВ "Вектор-Стиль"	Створити дизайн сайту 13:00 / 18:00
10	Ярослав Іванченко	Едельвейс	Створити функціонал сайту 20:00 / 30:00
8	Ілля Мельниченко	Альфа-Сервіс	Створити сайт 35:00 / 55:00
6	Віра Середа	ПрАТ "Майнер-Дизайн"	Створити дизайн сайту 16:00 / 20:10
4	Руслан Таращук	ПП "Еко"	Створити функціонал сайту 20:00 / 40:00
2	Роман Шевченко	ТОВ "Мега"	Створити сайт 20:00 / 50:00

Рисунок 4.23 – Список завдань з новою задачею.

ВИСНОВКИ

Прийняття рішень - це процес, який має місце у всіх управлінських діях: планування, організація, мотивація та контроль. Рішення є основою, джерелом цих процесів в управлінні. Через прийняття рішень визначаються цілі та завдання, способи їх досягнення та виміру. У сучасному бізнесі та в сучасній літературі все більше уваги приділяється способам та методам прийняття рішення.

Необхідність прийняття рішення виникає у ситуації, коли стоїть завдання вибору найоптимальнішого варіанта як мінімум із двох і більше можливих варіантів.

Менеджери IT-компаній стикаються з прийняттям рішення кожен раз, коли необхідно призначити виконавця завдання. Цей вибір в подальшому впливає на швидкість та якість виконання завдання, а також на авторитет та доходи компанії загалом. Тому так важливо обрати найбільш підходящого розробника на роль виконавця конкретного завдання.

При написанні дипломної роботи було зібрано та проаналізовано вже існуючі підходи та системи в області розподілу праці, проте кожна система має певні недоліки та досить різноманітний функціонал, що впливає на продуктивність, тому було прийнято рішення розробити власну вузьконаправлену систему, яка буде зберігати свою швидкодію за рахунок відсутності непотрібних шаблонів прийняття рішення.

Математичну модель було побудовано за принципами прийняття рішень при нечіткому відношенні переваги на множині альтернатив.

У співпраці з однією з IT-компаній міста Хмельницького ПП «Авіві» було визначено основні критерії, які впливають на вибір виконавця, а саме:

- завантаженість програміста,
- рівень компетенції,
- спеціалізація,
- обізнаність з проєктом.

Для кожного критерія була визначена своя оціночна шкала.

Було побудовано математичну модель та проаналізовано результат виконання. Отримана альтернатива дійсно найкраще відповідала вимогам.

На основі отриманої моделі було розроблено програмний застосунок для системи Бітрікс24, проведено апробацію та перевірку коректності отриманих результатів. Розробники, виділені в результаті роботи скрипта, також найкраще серед усіх підходили на роль виконавця.

Розроблений програмний застосунок буде корисний менеджерам проєктів IT-компаній тим, що дозволяє швидко та обґрунтовано визначити найкращих претендентів для виконання завдання та у той самий час залишає право остаточного вибору за менеджером.

Даний програмний застосунок в подальшому може бути допрацьований під потреби конкретної компанії. До прикладу, додати у форму стандартні та/або користувацькі поля, додати або змінити критерії та їх ваги, додати функціонал створення нового проєкту.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Єрьоміна І. І., Лисанов Д. М. Математическая модель оптимизации процесса распределения задач и трудовых ресурсов на предприятии/ American Scientific Journal. 2020. №42. С. 65-71.
2. Хабибрахманов Р. Г., Файзрахманов Р. А. Оптимизация деятельности ит-службы предприятия путем эффективного распределения задач / Вестник Марийского государственного технического университета. 1(8) 2010. С. 31-35.
3. Ломазов В. А., Прокушев Я. Е. Решение задачи экономичного многокритериального выбора на основе метода анализа иерархий. 2010. т. 7. № 14-1-1. С. 128-131.
4. Ломазов В. А., Прокушев Я. Е. Инструментальная поддержка принятия решений при отборе и оценке персонала с учетом мотивации. 2013. 67. № 8 (28). С. 31.
5. Курсы по программированию 1С 8.3 Евгения Гилева и Насипова Фарита: Базовый курс – Начало работы с платформой «1С: Предприятие 8» [Відеозапис] / реж. Гилев Е., Фарит Н.; в ролях: Евгений Гилев и Насипов Фарит; - http://xn----1-bedvffifm4g.xn--p1ai/ - 2012-2016.
6. Маклаков С. В. Моделирование бизнес-процессов с AllFusion Process Modeler / С. В. Маклаков. – М.: Диалог-МИФИ, 2014. – 224 с.
7. Абакумов В. В. Учебник по дисциплине: "Менеджмент" / В.В. Абакумов, А.А. Голубев, В.П. Кустарев, В.И. Подлесных – М.: Бизнес-пресса. 2016. С. 175.
8. Ломазов В. А., Прокушев Я. Е. Процедура поддержки принятия кадровых решений с учетом мотивации работников. 2014. №4 С. 2-10.
9. Дубейковскій В.І. Эффективное моделирование с CA ERwin Process Modeler и AllFusion Process Modeler / В.И. Дубейковский. – М.: Диалог-МИФИ, 2013. С. 384.

10. Франчайзинг 1С [Електронний ресурс] / Власник: ООО "1С" - Режим доступу: http://www.1c.ru/rus/firm1c/franch.htm
11. Новікова Н. І. Выбор стратегии с помощью метода анализа иерархий. 2008. № S1. С. 162-164.
12. Романова Ю. Д. Информационные технологии в менеджменте (управлении): учебник и практикум /под общ. ред. Ю. Д. Романовой. – М.: Видавництво Юрайт, 2014. С. 478. – Серія: Бакалавр. Базовий курс.
13. Зайченко Ю.П. Дослідження операцій [Текст] : підруч. для студ. вищ. навч. закл., що навч. за напрям. "Комп’ютерні науки" та "Прикладна математика" / Ю. П. Зайченко. - 7-ме вид., перероб. та доп. - К. : Слово, 2006. - 816 с. : іл., табл. - Бібліогр.: с. 812-816. - ISBN 966-8407-64-4.
14. Методы экспертных оценок. URL: https://habr.com/ru/post/189626/
15. Витвицкая А. Отличия junior, middle и senior разработчиков — объясняют эксперты. URL: https://tproger.ru/experts/junior-middle-senior-developers-differences/
16. Michael Wales 3 Web Dev Careers Decoded: Front-End vs Back-End vs Full Stack. URL: https://www.udacity.com/blog/2020/12/front-end-vs-back-end-vs-full-stack-web-developers.html
17. Matthew Martin Frontend Developer vs Backend Developer: Key Differences. URL: https://www.guru99.com/front-end-vs-back-end-developers.html
18. Локальные приложения. Быстрый старт. Курс «Приложения Битрикс24.Маркет» URL: https://dev.1c-bitrix.ru/learning/course/index.php?COURSE_ID=99&CHAPTER_ID=08593&LESSON_PATH=8771.8583.8593
19. Простой REST API в PHP - Пошаговое руководство. URL: https://only-to-top.ru/blog/programming/2019-11-06-rest-api-php.html
20. Документация по REST. Работа с SDK CRest в контексте пользователя. URL: https://dev.1c-bitrix.ru/rest_help/rest_sum/rest_user.php

ДОДАТОК А

(обов’язковий)

Шевчук О.О. Методи прийняття рішень в умовах нечіткої інформації в задачах розподілення робіт між працівниками // Актуальні проблеми комп’ютерних наук. Збірник наукових праць за матеріалами XIII всеукраїнської науково-практичної конференції «Актуальні проблеми комп’ютерних наук АПКН-2021» – Хмельницький: ХНУ, 2021 – С. 274-277.

УДК 004.02

Шевчук О. О.

Хмельницький національний університет

МЕТОДИ ПРИЙНЯТТЯ РІШЕНЬ В УМОВАХ НЕЧІТКОЇ ІНФОРМАЦІЇ В ЗАДАЧАХ РОЗПОДІЛЕННЯ РОБІТ МІЖ ПРАЦІВНИКАМИ

У роботі розглядаються різні підходи та існуючі системи підтримки прийняття рішень в умовах нечіткої інформації..

The paper considers different approaches and existing decision support systems in the conditions of fuzzy information.

Однією з найбільш важливих проблем, що виникають в різних сферах людської діяльності, є проблема вдосконалення управління. Ефективне управління базується на оптимальному використанні ресурсів і грамотної комплексної оцінці організації [1].

Надлишок кваліфікованих фахівців сьогодні робить актуальним питання досягнення ефективності в управлінні персоналом. У досягненні успіху діяльності всієї компанії на ринку праці, персонал є основним ресурсом, в умовах високої конкуренції [2].

Методи і підходи роботи підбору персоналу, все ще недостатньо вивчені, тому існують проблеми, пов'язані з прийняттям кадрових рішень. На результат прийняття кадрових рішень впливають індивідуальні особливості претендента на посаду [3]. Приймаючи подібне рішення, використовується один з перерахованих методів аналізу:

- багатокритеріальний метод;
- раціональний метод;
- кластерний метод.

Для підтримки прийняття кадрових рішень доцільно використовувати багатокритеріальний метод, так як він є найпростішим і найбільш розвиненим [4]. Використовуючи багатокритеріальний метод, кожну окремо взяту альтернативу можна оцінити конкретним числом, а порівняння альтернатив звести до порівняння відповідних їм чисел.

Характерна риса багатокритеріального методу – пошук рішень в умовах невизначеності з дискретної або неперервної множини альтернатив. Існує класифікація, заснована на змісті та типі одержуваної інформації. В якості змісту інформації використовується інформація про переваги на множині критеріїв та про наслідки альтернатив [5].

Найбільш перспективними з них є декомпозиційні методи теорії очікуваної корисності, методи аналізу ієрархій і теорії нечітких множин.

Перспектива визначена тим, що ці методи найбільшою мірою задовольняють вимогам універсальності [6].

Всі методи прийняття рішень поділяються на дві групи: формалізовані і неформалізовані. При вирішенні добре структурованих і частково слабоструктурованих проблем для оцінки варіантів рішень, вибору і обґрунтування оптимального варіанту застосовують формалізовані методи. Для складних слабоструктурованих і неструктурованих проблем для генерування варіантів рішень, їх аналізу та оцінки, вибору і обґрунтування найкращого рішення - неформалізовані [6].

До формалізованої групи належать методи для обґрунтування і вибору оптимальних рішень і включають в себе [6]:

- економіко-математичні моделі і методи (ЕММ), що формалізують взаємозв'язки процесів і явищ;
- системний аналіз, що дозволяє виявити взаємодії складових частин систем, стратегію їх розвитку;
- експертні оцінки і судження, що дозволяють кваліфікованим фахівцям оцінити значимість подій, явищ, факторів, прогнози розвитку систем і підсистем, співвідношення детермінованих і імовірнісних факторів.

Розвиток інформаційних технологій викликав появу безлічі програмних продуктів, покликаних автоматизувати діяльність будь-якого підприємства шляхом повного або часткового усунення людського фактору. І якщо на автоматизованому виробництві продуктивність обладнання обчислювана, то в компаніях, де ефективність роботи характеризується використанням людських ресурсів, все не так просто. Якщо проаналізувати питання про те, за якої умови можна досягти максимальної продуктивності праці, то очевидно, що це можливо тільки при правильному розподілі внутрішніх завдань відповідно до можливостей персоналу. До особливостей, які не дозволяють якісно розподіляти завдання між співробітниками в існуючих програмних продуктах, відноситься складність адаптації їх до специфіки предметної області[1].

Існують наступні системи підтримки прийняття рішень при розподілі працівників:

а) «Імператор» – це універсальна система підтримки прийняття рішень в різних сферах діяльності. Вона допомагає при вирішенні найважливіших проблем, таких як розподіл ресурсів і вибір найбільш пріоритетного вирішення [7].

Основою системи «Імператор» є метод аналізу ієрархій. Дана методологія дуже схожа з людським мисленням і узагальнює підхід, застосовуваний в більшості поширених експертних систем. Розробники системи вдосконалили метод аналізу ієрархій і тим самим забезпечили їй відміну від своїх аналогів і враження користувачів її можливостями. Вона дозволяє [7]:

- створювати графічну схему проблеми;
- проводити збір даних від експертів;
- оцінювати і мінімізувати ступінь суперечливості даних;
- обчислювати рейтинг альтернативних рішень;

- вести тематичний каталог проектів, що включають моделі рейтингування і набори даних;
- дослідити стійкість рейтингу;
- виявляти істотні фактори;
- створювати і аналізувати динамічні сценарії розвитку ситуації;
- вирішувати завдання відновлення ситуацій по відомим рейтингам (так звані зворотні завдання);
- проводити аналіз проблеми на основі декількох моделей;
- моделювати вплив випадкових чинників;
- здійснювати роботу з Microsoft Excel з експорту та імпорту моделей;
- створювати докладні звіти в Microsoft Word.

«Імператор» дозволяє проводити аналіз і синтез для ситуацій прийняття рішень, з огляду на «людський фактор». Інформація, надана експертом, може бути як об'єктивною кількісною так і якісною [7].

б) T-CHOICE 1.0 – діалогова система, орієнтована на підтримку прийняття рішень в різних сферах людської діяльності. Вона може стати незамінним помічником для всіх, кому необхідно приймати обґрунтовані раціональні рішення [8].

Система, використовує табличний метод при пошуку рішення. Вихідні дані представлені у вигляді таблиці, де рядки це альтернативи, а стовпці відповідають критеріям, за якими приймається рішення. ЛПР повинен впорядкувати значення кожного стовпчика і призначити для кожного з критеріїв кордон, при якій перспективне безліч не було б порожнім і занадто великим по потужності. Прийнятними прийнято вважати все альтернативи, що не виходять за встановлену межу [8].

в) MPRIORITY 1.0 – діалогова система, орієнтована на підтримку прийняття рішень в різних сферах людської діяльності. Вона може стати незамінним помічником усім, хто бажає або змушений за родом своєї діяльності приймати обґрунтовані раціональні рішення [9].

Програмна система використовує Метод Аналізу Ієрархія (MAI). Основне призначення методу – рішення слабоструктурованих задач прийняття рішень [9].

Структура задачі прийняття рішення в MAI є ієрархією. У вершині ієрархії розташовується основна мета, нижче – підцілі і ще нижче - альтернативи. Альтернативи ранжуються за допомогою парного зважування щодо інтуїтивно обґрунтованої якісної шкали [9].

Приклади задач ПР для яких можливе застосування "MPRIORITY" [9]:

- вибір керівником фірми майбутнього ділового партнера;
- раціональний розподіл доходів підприємства по галузях;
- відбір кращих претендентів на робочі місця фірми;
- оцінка роботи персоналу фірми;
- вибір програмного забезпечення для потреб фірми;
- оцінка культурних цінностей (картин, скульптур і т.д.);
- вибір найкращої стратегії;

– вибір найкращої конструкції (варіанти) технічного вироби;
– покупка квартири, дачі, ділянки, автомобіля;
– вибір майбутнього навчального закладу для дитини;
– вибір майбутнього робочого місця.

г) OPTIMUM 1.0 – діалогова система, орієнтована на рішення практичних задач глобальної оптимізації. Програмна система може бути застосована в разі, коли процес прийняття рішень може бути зведений до побудови та оптимізації деякої функції [8].

Багато задач прийняття рішень зведені до обчислення цільової функції з подальшим знаходженням її оптимального значення. Діалогова система "OPTIMUM" базується на двох методах оптимізації [8]:

– основному – адаптивному випадковому пошуку;
– додатковому – детермінованому методі Хука-Джівса, що дозволяє користувачеві виконати уточнення будь-якого поточного рішення.

Перераховані системи мають певні недоліки та досить різноманітний функціонал, що впливає на продуктивність, тому планується розробка своєї вузьконаправленої системи, яка буде зберігати швидкодію за рахунок відсутності непотрібних шаблонів прийняття рішення.

Перелік посилань

1. Єрьоміна І. І., Лисанов Д. М. Математическая модель оптимизации процесса распределения задач и трудовых ресурсов на предприятии / American Scientific Journal. 2020. №42. С. 65-71.
2. Ломазов В. А., Прокушев Я. Е. Решение задачи экономичного многокритериального выбора на основе метода анализа иерархий. 2010. т. 7. № 14-1-1. С. 128-131.
3. Ломазов В. А., Прокушев Я. Е. Инструментальная поддержка принятия решений при отборе и оценке персонала с учетом мотивации. 2013. 67. № 8 (28). С. 31.
4. Курсы по программированию 1С 8.3 Евгения Гилева и Насипова Фарита: Базовый курс – Начало работы с платформой «1С: Предприятие 8» [Відеозапис] / реж. Гилев Е., Фарит Н.; в ролях: Евгений Гилев и Насипов Фарит; - http://xn----1-bedvffifm4g.xn--p1ai/ - 2012-2016.
5. Маклаков С. В. Моделирование бизнес-процессов с AllFusion Process Modeler / С. В. Маклаков. – М.: Диалог-МИФИ, 2014. – 224 с.
6. Абакумов В. В. Учебник по дисциплине: "Менеджмент" / В.В. Абакумов, А.А. Голубев, В.П. Кустарев, В.И. Подлесных – М.: Бизнес-пресса. 2016. С. 175.
7. Франчайзинг 1С [Електронний ресурс] / Власник: ООО "1С" - Режим доступу: http://www.1c.ru/rus/firm1c/franch.htm
8. Новікова Н. І. Выбор стратегии с помощью метода анализа иерархий. 2008. № S1. С. 162-164.
9. Романова Ю. Д. Информационные технологии в менеджменте (управлении): учебник и практикум / під заг. ред. Ю. Д. Романовой. – М.: Видавництво Юрайт, 2014. С. 478. – Серія: Бакалавр. Базовий курс.

ДОДАТОК Б

(обов’язковий)

Текст програмного коду

index.php:

```
<?php require_once(__DIR__.'/crestcurrent.php');
   $users = CRestCurrent::call(
      "user.get",
      array(
         "SORT"  => "UF_USR_1636807293198",
         "ORDER"  => "ASC",
         "FILTER"  => array("UF_DEPARTMENT"  => array(3)),
         "SELECT"  => array("ID",  "NAME",  "LAST_NAME",  "UF_USR_1636807293198",
"UF_USR_1636807393886")
      ),
      false,
      true
   );
   $tasks = CRestCurrent::call(
      "tasks.task.list",
      array(
         "FILTER"  => array('STATUS'  => array(1, 2, 3)),
      ),
      false,
      true
   );
   $groups = CRestCurrent::call(
      "sonet_group.get",
      array(),
      false,
      true
   );
   $usersID = array();
   $userArray = array();
   $level_of_competence = array(44 => 0, 46 => 0.5, 48 => 1);
   $level_of_competence_names = array(44 => 'Junior', 46 => 'Middle', 48 => 'Senior');
   $specialization = array(50 => 0, 54 => 0.5, 52 => 1);
   $specialization_names = array(50 => 'Front-end', 54 => 'Full-stack', 52 => 'Back-end');
   foreach($users['result'] as $user){
      $fullName = $user['NAME']." ".$user['LAST_NAME'];
      $fullNameArr = str_split($fullName);
      $fullNameArr = array_pad($fullNameArr, 38, " ");
      $userArray['USERS'][$user['ID']]['FULL_NAME']  = implode("", $fullNameArr);
      $userArray['USERS'][$user['ID']]['ID']  = $user['ID'];
      $userArray['USERS'][$user['ID']]['TIME_LEFT']  = 0;
      $userArray['USERS'][$user['ID']]['LEVEL_OF_COMPETENCE']  =
$level_of_competence_names[$user['UF_USR_1636807293198']];
      $userArray['USERS'][$user['ID']]['LEVEL_OF_COMPETENCE_VALUE']  =
$level_of_competence[$user['UF_USR_1636807293198']];
      $userArray['USERS'][$user['ID']]['SPECIALIZATION']  =
$specialization_names[$user['UF_USR_1636807393886']];
      $userArray['USERS'][$user['ID']]['SPECIALIZATION_VALUE']  =
$specialization[$user['UF_USR_1636807393886']];
      $usersID[] = $user['ID'];
   }

   foreach ($groups['result'] as $group) {
      $userArray['GROUPS'][$group['ID']]['NAME']  = $group['NAME'];
   }
```

```
    foreach ($tasks['result']['tasks'] as $task) {
        array_push($task['accomplices'], $task['responsibleId']);
        $userArray['TASKS'][$task['id']]['USERS_ID']  = $task['accomplices'];
        unset($user);
        foreach ($task['accomplices'] as $user) {
            $userArray['USERS'][$user]['GROUPS'][] = $task['groupId'];
        }
        unset($user);
        $userArray['TASKS'][$task['id']]['TIME_ESTIMATE']  = $task['timeEstimate'];
        $userArray['TASKS'][$task['id']]['TIME_ELAPSED']  = 0;
    }

    $elapsed = CRestCurrent::call(
        "task.elapseditem.getlist",
        array(
            "ORDER"  => array("CREATED_DATE"  => "asc"),
            "FILTER"  => array("USER_ID"  => $usersID),
            "SELECT"  => array("SECONDS",  "TASK_ID")
        )
    );
    foreach ($elapsed['result'] as $value) {
        if(!empty($userArray['TASKS'][$value['TASK_ID']])){
            $userArray['TASKS'][$value['TASK_ID']]['TIME_ELAPSED']  += $value['SECONDS'];
        }
    }
    foreach ($userArray['TASKS']  as &$task) {
        $task['TIME_LEFT']  = $task['TIME_ESTIMATE']  - $task['TIME_ELAPSED'];
        foreach ($task['USERS_ID'] as $user) {
            $userArray['USERS'][$user]['TIME_LEFT']  += $task['TIME_LEFT'];
        }
    } ?>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Quick start. Local server-side application with UI</title>
    <!-- подключення css-файла -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet"
integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3"
crossorigin="anonymous">
    <!-- подключення js-файла -->
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-
ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IlRH9sENBO0LRn5q+8nbTov4+1p"
crossorigin="anonymous"></script>
    <!-- подключення popper.js, необхідного для коректної работи деяких плагинів  Bootstrap -->
    <script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.10.2/dist/umd/popper.min.js" integrity="sha384-
7+zCNj/IqJ95wo16oMtfsKbZ9ccEh31eOz1HGyDuCQ6wgnyJNSYdrPa03rtR1zdB"
crossorigin="anonymous"></script>
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.min.js" integrity="sha384-
QJHtvGhmr9XOIpI6YVutG+2QOK9T+ZnN4kzFN1RtK3zEFEIsxhlmWl5/YESvpZ13"
crossorigin="anonymous"></script>
    <!-- подключення css-файла -->
    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/css/bootstrap.min.css"
    integrity="sha384-WskhaSGFgHYWDcbwN70/dfYBj47jz9qbsMId/iRN3ewGhXQFZCSftd1LZCfmhktB"
crossorigin="anonymous">
    <!-- подключення потрібної версії jQuery -->
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/2.2.0/jquery.min.js"></script>
</head>
<body>
    <script>
        var selectFocus = function(event) {
            var serializeArray = $('#newTask').serializeArray();
            var serialize = $('#newTask').serialize().split('&');
```

```
var group = serialize[2];
group = group.substr(group.indexOf("=")+1);
var level_of_competence = serialize[5];
level_of_competence = level_of_competence.substr(level_of_competence.indexOf("=")+1);
var specialization = serialize[6];
specialization = specialization.substr(specialization.indexOf("=")+1);
if(group.length!=0 && level_of_competence.length!=0 && specialization.length!=0){
    var userArray = Object.values(<?php echo json_encode($userArray['USERS']);?>);
    var r1 = [], r2 = [], r3 = [], r4 = [];
    for (var i = 0; i < userArray.length; i++) {
        tempArr = Array(userArray.length);
        for (var j = 0; j < userArray.length; j++) {
            var iLOCV = userArray[i].LEVEL_OF_COMPETENCE_VALUE;
            var jLOCV = userArray[j].LEVEL_OF_COMPETENCE_VALUE;
            switch (level_of_competence) {
                case '0':
                    tempArr[j] = (iLOCV <= jLOCV) ? 1 : 0;
                    break;
                case '0.5':
                    if(iLOCV == 0.5) iLOCV += 1;
                    if(jLOCV == 0.5) jLOCV += 1;
                    tempArr[j] = (iLOCV >= jLOCV) ? 1 : 0;
                    break;
                case '1':
                    tempArr[j] = (iLOCV >= jLOCV) ? 1 : 0;
                    break;
            }
        }
        r1.splice(i, 0, tempArr);
    }
    for (var i = 0; i < userArray.length; i++) {
        tempArr = Array(userArray.length);
        for (var j = 0; j < userArray.length; j++) {
            var iSPV = userArray[i].SPECIALIZATION_VALUE;
            var jSPV = userArray[j].SPECIALIZATION_VALUE;
            switch (specialization) {
                case '0':
                    if(iSPV == 0) iSPV = 0.5;
                    if(jSPV == 0) jSPV = 0.5;
                    tempArr[j] = (iSPV <= jSPV) ? 1 : 0;
                    break;
                case '0.5':
                    if(iSPV == 1) iSPV = 0;
                    if(jSPV == 1) jSPV = 0;
                    tempArr[j] = (iSPV >= jSPV) ? 1 : 0;
                    break;
                case '1':
                    if(iSPV == 1) iSPV = 0.5;
                    if(jSPV == 1) jSPV = 0.5;
                    tempArr[j] = (iSPV >= jSPV) ? 1 : 0;
                    break;
            }
        }
        r2.splice(i, 0, tempArr);
    }
    for (var i = 0; i < userArray.length; i++) {
        tempArr = Array(userArray.length);
        for (var j = 0; j < userArray.length; j++) {
            var iGR = userArray[i].GROUPS;
            var jGR = userArray[j].GROUPS;
            iGR = iGR.includes(group) ? 1 : 0;
            jGR = jGR.includes(group) ? 1 : 0;
            tempArr[j] = (iGR >= jGR) ? 1 : 0;
```

```
            }
            r3.splice(i, 0, tempArr);
        }
        for (var i = 0; i < userArray.length; i++) {
            tempArr = Array(userArray.length);
            for (var j = 0; j < userArray.length; j++) {
                var iTL = userArray[i].TIME_LEFT;
                var jTL = userArray[j].TIME_LEFT;
                tempArr[j] = (iTL <= jTL) ? 1 : 0;
            }
            r4.splice(i, 0, tempArr);
        }
        var q1 = [];
        for (var i = 0; i < userArray.length; i++) {
            tempArr = Array(userArray.length);
            for (var j = 0; j < userArray.length; j++) {
                tempArr[j] = Math.min(r1[i][j], r2[i][j], r3[i][j], r4[i][j]);
            }
            q1.splice(i, 0, tempArr);
        }
        var mQ1 = Array(userArray.length);
        for (var i = 0; i < userArray.length; i++) {
            var sup = 0;
            for (var j = 0; j < userArray.length; j++) {
                sup = (q1[j][i] - q1[i][j]) > sup ? q1[j][i] - q1[i][j] : sup;
            }
            mQ1[i] = 1 - sup;
        }
        var q2 = [];
        for (var i = 0; i < userArray.length; i++) {
            tempArr = Array(userArray.length);
            for (var j = 0; j < userArray.length; j++) {
                tempArr[j] = r1[i][j]*0.5 + r2[i][j]*0.2 + r3[i][j]*0.2 + r4[i][j]*0.1;
            }
            q2.splice(i, 0, tempArr);
        }
        var mQ2 = Array(userArray.length);
        for (var i = 0; i < userArray.length; i++) {
            var sup = 0;
            for (var j = 0; j < userArray.length; j++) {
                sup = (q2[j][i] - q2[i][j]) > sup ? q2[j][i] - q2[i][j] : sup;
            }
            mQ2[i] = 1 - sup;
        }
        var mQ = Array(userArray.length);
        var sortUsersObj = Array(userArray.length);
        var startBestsGroup = '';
        var endBestsGroup = '';
        for (var i = 0; i < userArray.length; i++) {
            mQ[i] = Math.min(mQ1[i], mQ2[i]);
            if(mQ[i]>0 && startBestsGroup == '' && endBestsGroup == ''){
                startBestsGroup = "<option hidden>Не обрано</option><optgroup style='background-color: #5555;
padding: 5px' label='Найкращий вибір'>";
                endBestsGroup = "</optgroup>";
            }
            sortUsersObj[i] = {mQ: mQ[i], userArray: userArray[i]};
        }
        sortUsersObj.sort(function(a, b) {
            return b.mQ - a.mQ;
        });

        var newOpt = '';
        sortUsersObj.forEach(function(user){
```

```
                if(user.mQ>0 && (String(user.userArray.SPECIALIZATION_VALUE)  == specialization ||
user.userArray.SPECIALIZATION_VALUE  == 0.5)){
                    startBestsGroup += "<option style='background-color: #5555; padding: 5px'
value="+user.userArray.ID+">"+user.userArray.FULL_NAME+user.userArray.LEVEL_OF_COMPETENCE+" 
;  "+user.userArray.SPECIALIZATION+"</option>";
                }
                else{
                    newOpt += "<option
value="+user.userArray.ID+">"+user.userArray.FULL_NAME+user.userArray.LEVEL_OF_COMPETENCE+" 
;  "+user.userArray.SPECIALIZATION+"</option>";
                }
            });
            $("#users")[0].innerHTML = startBestsGroup + endBestsGroup + newOpt;
        }
        else if(group.length==0){$("select[name=group]").focus(); alert("Вкажіть проєкт.");  }
        else if(level_of_competence.length==0){$("select[name=level_of_competence]").focus(); alert("Вкажіть
рівень компетенції.");}
        else if(specialization.length==0){$("select[name=specialization]").focus(); alert("Вкажіть спеціалізацію.");}
    }
    $(document).ready(function () {
        $(document).on('focus', '#users', selectFocus);
    });
    function newTask(formID){
        event.preventDefault();
        var request = jQuery.param(<?php echo json_encode($_REQUEST);?>);
        $.ajax({
            url:      "task_add.php",
            type:     "POST",
            dataType: "html",
            data: $("#"+formID).serialize()+"&"+request,
            success: function(response) {
                if(response != "error"){
                    $("#"+formID)[0].reset();
                    alert("Створено нову задачу з ID "+response);
                }
                else alert("Задачу не створено.");
            }
        });
    }
</script>
<form style="margin: 10px" class="row g-2 needs-validation" id="newTask"  method="post"
onsubmit="newTask('newTask');">
    <div class="input-group mb-1">
        <input class="form-control col-sm-8" type="text" name="task_title" placeholder="Назва завдання*" required>
    </div>
    <div class="input-group mb-1">
        <textarea class="form-control col-sm-8" name="task_body" placeholder="Опис завдання"></textarea>
    </div>
    <div class="form-row align-items-center mb-1">
        <div class="input-group-prepend col-sm-2">
            <span>Проєкт*</span>
        </div>
        <div class="input-group col-sm-4 my-1">
            <select class="form-select" name="group" required>
                <option value="">Не  обрано</option>
                <?foreach($userArray['GROUPS'] as $groupID => $group):?>
                    <option value="<?=$groupID?>">
                        <?=$group['NAME']?>
                    </option>
                <?endforeach;?>
            </select>
        </div>
    </div>
```

```
      <div class="form-row align-items-center mb-1">
        <div class="input-group-prepend col-sm-2">
          <span>Час на виконання</span>
        </div>
        <div class="col-sm-2 my-1">
          <input type="number" name="time_estimate_hour" min=0 class="form-control" placeholder="годин">
        </div>
          <span> : </span>
        <div class="col-sm-2 my-1">
          <input type="number" name="time_estimate_min" min=0 max=59 class="form-control"
placeholder="хвилин">
        </div>
      </div>
      <div class="form-row align-items-center mb-1">
        <div class="input-group-prepend col-sm-2">
          <span>Рівень компетенції*</span>
        </div>
        <div class="col-sm-2 my-1">
          <select class="form-select" name="level_of_competence" required>
            <option value="">Не обрано</option>
            <option value="0">Junior</option>
            <option value="0.5">Middle</option>
            <option value="1">Senior</option>
          </select>
        </div>
      </div>
      <div class="form-row align-items-center mb-1">
        <div class="input-group-prepend col-sm-2">
          <span>Спеціалізація*</span>
        </div>
        <div class="col-sm-2 my-1">
          <select class="form-select" name="specialization" required>
            <option value="">Не обрано</option>
            <option value="0">Front-end</option>
            <option value="1">Back-end</option>
            <option value="0.5">Full-stack</option>
          </select>
        </div>
      </div>
      <div class="form-row align-items-center mb-1">
        <div class="input-group-prepend col-sm-2">
          <span>Відповідальний*</span>
        </div>
        <div class="col-sm-4 my-1">
          <select class="form-select" name="responsible" id="users" title="Відповідальний" autocomplete="off"
required>
            <option selected hidden disabled>Не обрано</option>
          </select>
        </div>
      </div>
      <div class="form-row align-items-center mb-1">
        <div class="col-sm-auto my-1">
          <button type="submit" class="btn btn-primary">Створити завдання</button>
        </div>
      </div>
    </form>
</body>
</html>
```

task_add.php:

```
<?require_once(__DIR__.'/crestcurrent.php');
if (!empty($_POST)) {
```

```
    $level_of_competence = array(0 => 44, 0.5 => 46, 1 => 48);
    $specialization = array(0 => 50, 0.5 => 54, 1 => 52);
    if(!empty($_POST['task_title']))
        $arFields['TITLE'] = $_POST['task_title'];
    if(!empty($_POST['task_body']))
        $arFields['DESCRIPTION'] = $_POST['task_body'];
    if(!empty($_POST['group']))
        $arFields['GROUP_ID'] = intval($_POST['group']);
    if(!empty($_POST['level_of_competence']))
        $arFields['UF_USR_1636807293198'] = $level_of_competence[intval($_POST['level_of_competence'])];
    if(!empty($_POST['specialization']))
        $arFields['UF_USR_1636807393886'] = $specialization[intval($_POST['specialization'])];
    if(!empty($_POST['responsible']))
        $arFields['RESPONSIBLE_ID'] = intval($_POST['responsible']);
    if(!empty($_POST['time_estimate_hour']) || !empty($_POST['time_estimate_min'])){
        if(empty($_POST['time_estimate_hour'])) $_POST['time_estimate_hour'] = 0;
        if(empty($_POST['time_estimate_min'])) $_POST['time_estimate_min'] = 0;
        $arFields['TIME_ESTIMATE'] =
intval($_POST['time_estimate_hour'])*3600+intval($_POST['time_estimate_min'])*60;
        $arFields['ALLOW_TIME_TRACKING'] = 'Y';
    }

    $tasks = CRestCurrent::call(
        "tasks.task.add",
        array(
            "fields" => $arFields
        ),
        false,
        true
    );
    if(!empty($tasks['result'])){
        $response = intval($tasks['result']['task']['id']);
    }
    else{
        $response = "error";
    }
    echo json_encode($response);
}?>
```

install.php:

```
<?php require_once (__DIR__.'/crest.php');
$result = CRest::installApp();
if($result['rest_only'] === false):?>
    <head>
        <script src="//api.bitrix24.com/api/v1/"></script>
        <?php if($result['install'] == true):?>
            <script>
                BX24.init(function(){
                    BX24.installFinish();
                });
            </script>
        <?php endif;?>
    </head>
    <body>
        <?php if($result['install'] == true):?>
            installation has been finished
        <?php else:?>
            installation error
        <?php endif;?>
    </body>
<?php endif;
```

crest.php:

```
<?php require_once (__DIR__.'/settings.php');
    class CRest
    {
        const BATCH_COUNT    = 50;
        const TYPE_TRANSPORT  = 'json';
        public static function installApp()
        {
            $result = [
                'rest_only' => true,
                'install' => false
            ];
            if($_REQUEST[ 'event' ] == 'ONAPPINSTALL'  && !empty($_REQUEST[ 'auth' ]))
            {
                $result['install'] = static::setAppSettings($_REQUEST[ 'auth' ], true);
            }
            elseif($_REQUEST['PLACEMENT'] == 'DEFAULT')
            {
                $result['rest_only'] = false;
                $result['install'] = static::setAppSettings(
                    [
                        'access_token' => htmlspecialchars($_REQUEST['AUTH_ID']),
                        'expires_in' => htmlspecialchars($_REQUEST['AUTH_EXPIRES']),
                        'application_token' => htmlspecialchars($_REQUEST['APP_SID']),
                        'refresh_token' => htmlspecialchars($_REQUEST['REFRESH_ID']),
                        'domain' => htmlspecialchars($_REQUEST['DOMAIN']),
                        'client_endpoint' => 'https://' . htmlspecialchars($_REQUEST['DOMAIN']) . '/rest/',
                    ],
                    true
                );
            }

            static::setLog(
                [
                    'request' => $_REQUEST,
                    'result' => $result
                ],
                'installApp'
            );
            return $result;
        }

        protected static function callCurl($arParams)
        {
            if(!function_exists('curl_init'))
            {
                return [
                    'error'            => 'error_php_lib_curl',
                    'error_information' => 'need install curl lib'
                ];
            }
            $arSettings = static::getAppSettings();
            if($arSettings !== false)
            {
                if(isset($arParams[ 'this_auth' ]) && $arParams[ 'this_auth' ] == 'Y')
                {
                    $url = 'https://oauth.bitrix.info/oauth/token/';
                }
                else
                {
                    $url = $arSettings[ "client_endpoint" ] . $arParams[ 'method' ] . '.' . static::TYPE_TRANSPORT;
                    if(empty($arSettings[ 'is_web_hook' ]) || $arSettings[ 'is_web_hook' ] != 'Y')
```

```
            {
                $arParams[ 'params' ][ 'auth' ] = $arSettings[ 'access_token' ];
            }
        }

        $sPostFields = http_build_query($arParams[ 'params' ]);

        try
        {
            $obCurl = curl_init();
            curl_setopt($obCurl, CURLOPT_URL, $url);
            curl_setopt($obCurl, CURLOPT_RETURNTRANSFER, true);
            curl_setopt($obCurl, CURLOPT_POSTREDIR, 10);
            curl_setopt($obCurl, CURLOPT_USERAGENT, 'Bitrix24 CRest PHP 1.36');
            if($sPostFields)
            {
                curl_setopt($obCurl, CURLOPT_POST, true);
                curl_setopt($obCurl, CURLOPT_POSTFIELDS, $sPostFields);
            }
            curl_setopt(
                $obCurl, CURLOPT_FOLLOWLOCATION, (isset($arParams[ 'followlocation' ]))
                ? $arParams[ 'followlocation' ] : 1
            );
            if(defined("C_REST_IGNORE_SSL") && C_REST_IGNORE_SSL === true)
            {
                curl_setopt($obCurl, CURLOPT_SSL_VERIFYPEER, false);
                curl_setopt($obCurl, CURLOPT_SSL_VERIFYHOST, false);
            }
            $out = curl_exec($obCurl);
            $info = curl_getinfo($obCurl);
            if(curl_errno($obCurl))
            {
                $info[ 'curl_error' ] = curl_error($obCurl);
            }
            if(static::TYPE_TRANSPORT == 'xml' && (!isset($arParams[ 'this_auth' ]) || $arParams[ 'this_auth' ] !=
'Y'))//auth only json support
            {
                $result = $out;
            }
            else
            {
                $result = static::expandData($out);
            }
            curl_close($obCurl);

            if(!empty($result[ 'error' ]))
            {
                if($result[ 'error' ] == 'expired_token' && empty($arParams[ 'this_auth' ]))
                {
                    $result = static::GetNewAuth($arParams);
                }
                else
                {
                    $arErrorInform = [
                        'expired_token'         => 'expired token, cant get new auth? Check access oauth server.',
                        'invalid_token'         => 'invalid token, need reinstall application',
                        'invalid_grant'         => 'invalid grant, check out define C_REST_CLIENT_SECRET or
C_REST_CLIENT_ID',
                        'invalid_client'        => 'invalid client, check out define C_REST_CLIENT_SECRET or
C_REST_CLIENT_ID',
                        'QUERY_LIMIT_EXCEEDED'    => 'Too many requests, maximum 2 query by second',
                        'ERROR_METHOD_NOT_FOUND' => 'Method not found! You can see the permissions of the
application: CRest::call(\'scope\')',
```

```
                    'NO_AUTH_FOUND'          => 'Some setup error b24, check in table "b_module_to_module" event
"OnRestCheckAuth"',
                    'INTERNAL_SERVER_ERROR'   => 'Server down, try later'
                ];
                if(!empty($arErrorInform[ $result[ 'error' ] ]))
                {
                    $result[ 'error_information' ] = $arErrorInform[ $result[ 'error' ] ];
                }
            }
        }
        if(!empty($info[ 'curl_error' ]))
        {
            $result[ 'error' ] = 'curl_error';
            $result[ 'error_information' ] = $info[ 'curl_error' ];
        }

        static::setLog(
            [
                'url'    => $url,
                'info'   => $info,
                'params' => $arParams,
                'result' => $result
            ],
            'callCurl'
        );

        return $result;
    }
    catch(Exception $e)
    {
        static::setLog(
            [
                'message' => $e->getMessage(),
                'code' => $e->getCode(),
                'trace' => $e->getTrace(),
                'params' => $arParams
            ],
            'exceptionCurl'
        );

        return [
            'error' => 'exception',
            'error_exception_code' => $e->getCode(),
            'error_information' => $e->getMessage(),
        ];
    }
}
else
{
    static::setLog(
        [
            'params' => $arParams
        ],
        'emptySetting'
    );
}

return [
    'error'             => 'no_install_app',
    'error_information' => 'error install app, pls install local application '
];
}
```

```
public static function call($method, $params = [])
{
   $arPost = [
      'method' => $method,
      'params' => $params
   ];
   if(defined('C_REST_CURRENT_ENCODING'))
   {
      $arPost[ 'params' ] = static::changeEncoding($arPost[ 'params' ]);
   }

   $result = static::callCurl($arPost);
   return $result;
}

public static function callBatch($arData, $halt = 0)
{
   $arResult = [];
   if(is_array($arData))
   {
      if(defined('C_REST_CURRENT_ENCODING'))
      {
         $arData = static::changeEncoding($arData);
      }
      $arDataRest = [];
      $i = 0;
      foreach($arData as $key => $data)
      {
         if(!empty($data[ 'method' ]))
         {
            $i++;
            if(static::BATCH_COUNT >= $i)
            {
               $arDataRest[ 'cmd' ][ $key ] = $data[ 'method' ];
               if(!empty($data[ 'params' ]))
               {
                  $arDataRest[ 'cmd' ][ $key ] .= '?' . http_build_query($data[ 'params' ]);
               }
            }
         }
      }
      if(!empty($arDataRest))
      {
         $arDataRest[ 'halt' ] = $halt;
         $arPost = [
            'method' => 'batch',
            'params' => $arDataRest
         ];
         $arResult = static::callCurl($arPost);
      }
   }
   return $arResult;
}

private static function GetNewAuth($arParams)
{
   $result = [];
   $arSettings = static::getAppSettings();
   if($arSettings !== false)
   {
      $arParamsAuth = [
         'this_auth' => 'Y',
         'params'    =>
```

```
            [
                'client_id'     => $arSettings[ 'C_REST_CLIENT_ID' ],
                'grant_type'    => 'refresh_token',
                'client_secret' => $arSettings[ 'C_REST_CLIENT_SECRET' ],
                'refresh_token' => $arSettings[ "refresh_token"],
            ]
        ];
        $newData = static::callCurl($arParamsAuth);
        if(isset($newData[ 'C_REST_CLIENT_ID' ]))
        {
            unset($newData[ 'C_REST_CLIENT_ID' ]);
        }
        if(isset($newData[ 'C_REST_CLIENT_SECRET' ]))
        {
            unset($newData[ 'C_REST_CLIENT_SECRET' ]);
        }
        if(isset($newData[ 'error' ]))
        {
            unset($newData[ 'error' ]);
        }
        if(static::setAppSettings($newData))
        {
            $arParams[ 'this_auth' ] = 'N';
            $result = static::callCurl($arParams);
        }
    }
    return $result;
}

private static function setAppSettings($arSettings, $isInstall = false)
{
    $return = false;
    if(is_array($arSettings))
    {
        $oldData = static::getAppSettings();
        if($isInstall != true && !empty($oldData) && is_array($oldData))
        {
            $arSettings = array_merge($oldData, $arSettings);
        }
        $return = static::setSettingData($arSettings);
    }
    return $return;
}

private static function getAppSettings()
{
    if(defined("C_REST_WEB_HOOK_URL") && !empty(C_REST_WEB_HOOK_URL))
    {
        $arData = [
            'client_endpoint' => C_REST_WEB_HOOK_URL,
            'is_web_hook'     => 'Y'
        ];
        $isCurrData = true;
    }
    else
    {
        $arData = static::getSettingData();
        $isCurrData = false;
        if(
            !empty($arData[ 'access_token' ]) &&
            !empty($arData[ 'domain' ]) &&
            !empty($arData[ 'refresh_token' ]) &&
            !empty($arData[ 'application_token' ]) &&
```

```
            !empty($arData[ 'client_endpoint' ])
        )
        {
            $isCurrData = true;
        }
    }

    return ($isCurrData) ? $arData : false;
}

protected static function getSettingData()
{
    $return = [];
    if(file_exists(__DIR__ . '/settings.json'))
    {
        $return = static::expandData(file_get_contents(__DIR__ . '/settings.json'));
        if(defined("C_REST_CLIENT_ID") && !empty(C_REST_CLIENT_ID))
        {
            $return['C_REST_CLIENT_ID'] = C_REST_CLIENT_ID;
        }
        if(defined("C_REST_CLIENT_SECRET") && !empty(C_REST_CLIENT_SECRET))
        {
            $return['C_REST_CLIENT_SECRET'] = C_REST_CLIENT_SECRET;
        }
    }
    return $return;
}

protected static function changeEncoding($data, $encoding = true)
{
    if(is_array($data))
    {
        $result = [];
        foreach ($data as $k => $item)
        {
            $k = static::changeEncoding($k, $encoding);
            $result[$k] = static::changeEncoding($item, $encoding);
        }
    }
    else
    {
        if($encoding)
        {
            $result = iconv(C_REST_CURRENT_ENCODING, "UTF-8//TRANSLIT", $data);
        }
        else
        {
            $result = iconv( "UTF-8",C_REST_CURRENT_ENCODING, $data);
        }
    }

    return $result;
}

protected static function wrapData($data, $debag = false)
{
    if(defined('C_REST_CURRENT_ENCODING'))
    {
        $data = static::changeEncoding($data, true);
    }
    $return = json_encode($data, JSON_HEX_TAG|JSON_HEX_AMP|JSON_HEX_APOS|JSON_HEX_QUOT);

    if($debag)
```

```
            {
                $e = json_last_error();
                if ($e != JSON_ERROR_NONE)
                {
                    if ($e == JSON_ERROR_UTF8)
                    {
                        return 'Failed encoding! Recommended \'UTF - 8\' or set define C_REST_CURRENT_ENCODING =
current site encoding for function iconv()';
                    }
                }
            }

            return $return;
        }

        protected static function expandData($data)
        {
            $return = json_decode($data, true);
            if(defined('C_REST_CURRENT_ENCODING'))
            {
                $return = static::changeEncoding($return, false);
            }
            return $return;
        }

        protected static function setSettingData($arSettings)
        {
            return  (boolean)file_put_contents(__DIR__ . '/settings.json', static::wrapData($arSettings));
        }

        public static function setLog($arData, $type = '')
        {
            $return = false;
            if(!defined("C_REST_BLOCK_LOG") || C_REST_BLOCK_LOG !== true)
            {
                if(defined("C_REST_LOGS_DIR"))
                {
                    $path = C_REST_LOGS_DIR;
                }
                else
                {
                    $path = __DIR__ . '/logs/';
                }
                $path .= date("Y-m-d/H")  . '/';

                if (!file_exists($path))
                {
                    @mkdir($path, 0775, true);
                }

                $path .= time() . '_' . $type . '_' . rand(1, 9999999) . 'log';
                if(!defined("C_REST_LOG_TYPE_DUMP") || C_REST_LOG_TYPE_DUMP !== true)
                {
                    $jsonLog = static::wrapData($arData);
                    if ($jsonLog === false)
                    {
                        $return = file_put_contents($path . '_backup.txt', var_export($arData, true));
                    }
                    else
                    {
                        $return = file_put_contents($path . '.json', $jsonLog);
                    }
                }
```

```
            else
            {
                $return = file_put_contents($path . '.txt', var_export($arData, true));
            }
        }
        return $return;
    }
}
```

crestcurrent.php:

```
<?php require_once(__DIR__ . '/crest.php');
class CRestCurrent extends CRest
{
    protected static $dataExt = [];
    protected static function getSettingData()
    {
        $return = static::expandData(file_get_contents(__DIR__ . '/settings.json'));
        if(is_array($return))
        {
            if(!empty(static::$dataExt))
            {
                $return['access_token'] = htmlspecialchars(static::$dataExt['AUTH_ID']);
                $return['domain'] = htmlspecialchars(static::$dataExt['DOMAIN']);
                $return['refresh_token'] = htmlspecialchars(static::$dataExt['REFRESH_ID']);
                $return['application_token'] = htmlspecialchars(static::$dataExt['APP_SID']);
            }
            else
            {
                $return['access_token'] = htmlspecialchars($_REQUEST['AUTH_ID']);
                $return['domain'] = htmlspecialchars($_REQUEST['DOMAIN']);
                $return['refresh_token'] = htmlspecialchars($_REQUEST['REFRESH_ID']);
                $return['application_token'] = htmlspecialchars($_REQUEST['APP_SID']);
            }
        }
        return $return;
    }
    public static function setDataExt($data)
    {
        static::$dataExt = $data;
    }
}
```

settings.php:

```
<?php
define('C_REST_CLIENT_ID','local.618fb673429c87.46665445');//Application ID
define('C_REST_CLIENT_SECRET','i6wwtCN7fsGLw36MvF53o3sO6tEclNINUdanXUSKqFT1PR9qdp');//Applicati
on key
// or
//define('C_REST_WEB_HOOK_URL','https://rest-api.bitrix24.com/rest/1/doutwqkjxgc3mgc1/');//url on creat
Webhook

//define('C_REST_CURRENT_ENCODING','windows-1251');
//define('C_REST_IGNORE_SSL',true);//turn off validate ssl by curl
//define('C_REST_LOG_TYPE_DUMP',true); //logs save var_export for viewing convenience
define('C_REST_BLOCK_LOG',true);//turn off default logs
//define('C_REST_LOGS_DIR', __DIR__ .'/logs/'); //directory path to save the log
```

ДОДАТОК В

(обов’язковий)

Матеріали презентації для захисту ДРМ

Багатокритеріальна модель формування оптимального штату виконавців-розробників IT-компанії

Студентка групи ПМм-20-1 Шевчук Ольга

Керівник роботи к.т.н., доцент Драч Ілона Володимирівна

Метою дипломної роботи є побудова математичної моделі формування оптимального штату виконавців-розробників IT-компанії та розробка на її основі програмного застосунка для системи Бітрікс24.

Завдання:

- Зібрати та проаналізувати існуючі підходи та системи в області розподілу праці
- Детально ознайомитися з теорією прийняття рішень при нечіткому відношенні переваги на множині альтернатив
- Побудувати математичну модель формування оптимального штату виконавців-розробників IT-компанії
- Розробка застосунка для системи Бітрікс24 для формування оптимального штату виконавців-розробників IT-компанії на основі багатокритеріального моделювання

Об'єкт дослідження: структура виконуваних завдань та якісні характеристики виконавців-розробників IT-компанії ПП «Авіві» для формування оптимального призначення для виконання завдання.

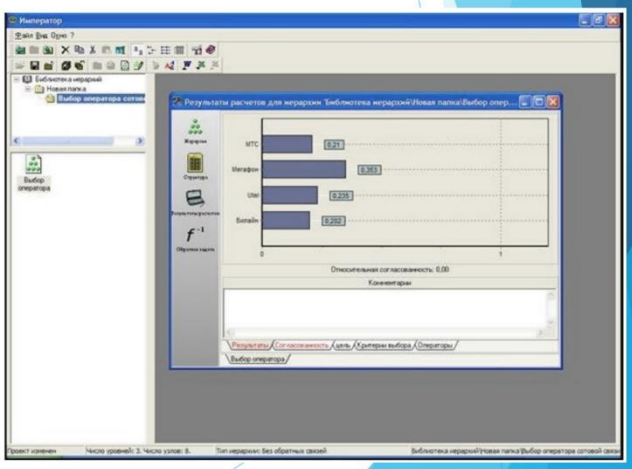
Предмет дослідження: моделювання прийняття управлінських рішень на основі нечіткого відношення переваги.

Науково-практична новизна результатів дипломної роботи

Побудовану багатокритеріальну модель взято за основу розробки застосунка для системи Бітрікс24 для формування оптимального штату виконавців-розробників IT-компанії. Розроблений застосунок допоможе приймати оптимальні рішення про призначення завдань виконавцям-розробникам, тим самим прискорити цей процес та мінімізувати перевантаженість програмістів.

1 ІСНУЮЧІ ПІДХОДИ ТА СИСТЕМИ В ОБЛАСТІ РОЗПОДІЛУ ПРАЦІ

Типи інформації	Метод прийняття
Відсутність інформації про вподобання; кількісна і/або інтервальна інформація про наслідки. Якісна інформація про вподобання та кількісна про наслідки Якісна (порядкова) інформація про вподобання та наслідки Кількісна інформація про вподобання та наслідки	Метод з дискретизацією невизначеності Стохастичне домінування Методи прийняття рішень в умовах ризику та невизначеності на основі глобальних критеріїв Метод аналізу ієрархій Метод теорії нечітких множин Метод практичного прийняття рішень Метод вибору статистично ненадійних рішень Метод кривих байдужості для прийняття рішень в умовах ризику та невизначеності Метод дерев рішень Декомпозиційні методи теорії очікуваної корисності



2 БАГАТОКРИТЕРІАЛЬНИЙ ВИБІР АЛЬТЕРНАТИВ НА ОСНОВІ НЕЧІТКОГО ВІДНОШЕННЯ ПЕРЕВАГИ

- Нечітке відношення переваги
- Лінійність нечітких відношень
- Нечітка підмножина недомінованих альтернатив
- Чітко недоміновані альтернативи та їхні властивості
- Багатокритеріальний вибір альтернатив на основі нечіткого відношення переваги

3 МАТЕМАТИЧНА МОДЕЛЬ ФОРМУВАННЯ ОПТИМАЛЬНОГО ШТАТУ ВИКОНАВЦІВ-РОЗРОБНИКІВ ІТ-КОМПАНІЇ

Критерії та їх вага:

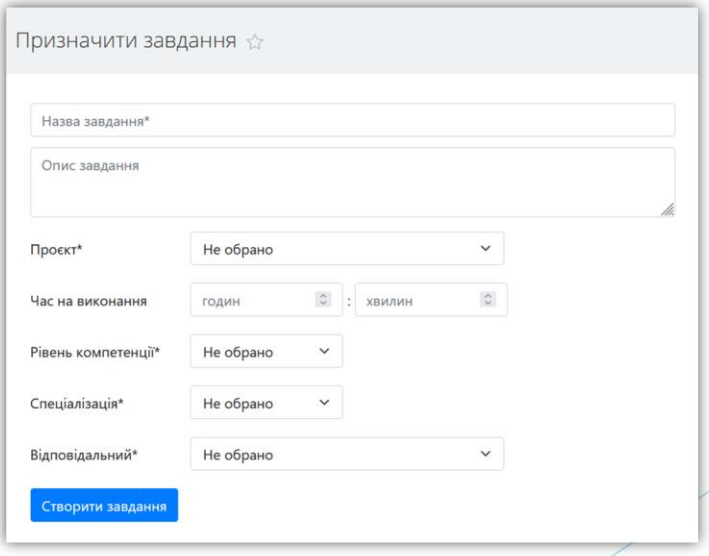
- R_1 - рівень компетенції - вага 0.5,
- R_2 - спеціалізація - вага 0.2,
- R_3 - обізнаність з проєктом - вага 0.2,
- R_4 - завантаженість програміста - вага 0.1.

- $R_1: x_1 \approx x_2, x_2 \approx x_3, x_3 \prec x_4, x_4 \approx x_5, x_5 \approx x_6, x_6 \succcurlyeq x_7, x_7 \approx x_8,\ x_8 \approx x_9, x_1 \prec x_9,$
- $R_2: x_1 \succ x_2, x_1 \approx x_3, x_3 \approx x_4, x_2 \approx x_5, x_4 \approx x_6, x_6 \approx x_7, x_2 \approx x_8,\ x_7 \approx x_9, x_1 \approx x_9,$
- $R_3: x_1 \approx x_2, x_2 \approx x_3, x_3 \approx x_4, x_4 \approx x_5, x_5 \approx x_6, x_6 \approx x_7, x_7 \prec x_8,\ x_8 \succ x_9, x_1 \approx x_9,$
- $R_4: x_1 \succ x_2, x_4 \succ x_7, x_2 \approx x_4, x_3 \prec x_5, x_4 \succcurlyeq x_7, x_5 \prec x_7, x_6 \prec x_3,\ x_8 \approx x_6, x_8 \succ x_9.$

Функція належності результуючої множини:

$$\mu_Q^{\text{нд}}(x_i) = [0.6; 0; 0; 1; 0; 0; 0; 0.4; 0];$$

4 РОЗРОБКА ЗАСТОСУНКА ДЛЯ ФОРМУВАННЯ ОПТИМАЛЬНОГО ШТАТУ ВИКОНАВЦІВ-РОЗРОБНИКІВ ІТ-КОМПАНІЇ



Висновки

При написанні дипломної роботи було виконано наступне:

- зібрано та проаналізовано вже існуючі підходи та системи в області розподілу праці;
- ознайомлено з теорією прийняття рішень при нечіткому відношенні переваги на множині альтернатив
- побудовано математичну модель за принципами прийняття рішень при нечіткому відношенні переваги на множині альтернатив та проаналізовано результат обрахунків;
- на основі отриманої моделі розроблено програмний застосунок для системи Бітрікс24, проведено апробацію та перевірку коректності отриманих результатів.

Розроблений програмний застосунок буде корисний менеджерам проєктів ІТ-компаній тим, що дозволяє швидко та обґрунтовано визначити найкращих претендентів для виконання завдання.

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

РЕЦЕНЗІЯ НА ДИПЛОМНУ РОБОТУ

Дипломник Шевчук Ольга Олександрівна

Тема Багатокритеріальна модель формування оптимального штату виконавців-розробників ІТ-компанії

Спеціальність 113 – Прикладна математика

Обсяг дипломної роботи:

Кількість листів креслень 0; кількість сторінок записки 78

1.Короткий зміст ДР та прийнятих рішень В роботі проаналізовано існуючі підходи та системи в області розподілу праці, побудовано математичну модель формування оптимального штату виконавців-розробників ІТ-компанії та на її основі розроблено програмний застосунок для системи Бітрікс24

2. Висновок про відповідність ДР поставленому завданню Дипломна робота відповідає поставленому завданню

3. Характеристика виконання кожного розділу роботи, ступінь використання останніх досягнень науки і техніки і передових методів роботи: Перший розділ роботи присвячений аналізу вже існуючих підходів та систем в області розподілу праці. У другому розділі розглянуті питання прийняття рішень при нечіткому відношенні переваги на множині альтернатив. У третьому розділі описано побудову математичної моделі формування оптимального штату виконавців-розробників ІТ-компанії. Для побудови моделі використовуються принципи прийняття рішень при нечіткому відношенні переваги на множині альтернатив. У четвертому розділі представлено основні моменти розробки програмного застосунка для системи Бітрікс24. Також проведено практичну апробацію та перевірку коректності отриманих результатів.

4. Позитивні сторони роботи Грунтовний аналіз існуючих підходів та системи в області розподілу праці, побудова математичної модель формування оптимального штату виконавців-розробників ІТ-компанії та розроблений на її основі програмний застосунок для системи Бітрікс24, можливість використання застосунка як на хмарній, так і на коробковій версії порталу.

5. Негативні сторони роботи присутня незначна кількість описок пов'язаних із автозаміною тексту.

6. Оцінка графічного оформлення та пояснювальної записки роботи пояснювальна записка оформлена згідно чинних вимог

7. Відгук про роботу в цілому Робота виконана відповідно до поставлених завдань

8. Інші зауваження

9. Оцінка дипломної роботи добре

РЕЦЕНЗЕНТ (прізвище, ім'я, по-батькові, посада, місце роботи) Кисіль Тетяна Миколаївна, кандидат фізико-математичних наук, доцент за кафедрою прикладної математики

"7" 12 2021 р. _____
(підпис)

Tue Dec 07 10:08:54 EET 2021, Стецюк Віктор Іванович, Хмельницький національний університет, ХНУ

Anti-Plagiarism v-15.257

Максимальное совпадение с одним документом 0.0%

Словари проверки: en_US, ru_RU, ua_UA. **Ошибок в документах: 8%**

ID: 98291 Название: Багатокритеріальна модель формування оптимального штату виконавців-розробників ІТ-компанії Добавлено в БД: 2021-12-07 Авторы: Шевчук Ольга Олександрівна Руководители: Драч Ілона Володимирівна Консультанты: Опоненты:	Документ		Суммарное совпадение по Базе Данных	
	Символы	Лексемы	Символы	Лексемы
	44929	714	677 (2%)	13 (2%)

Источник плагиата

ID	Описание	Наличие плагиата в документе	
		Символы	Лексемы

UNICHECK

Имя пользователя:
Kafedra TMIT KhNU

ID проверки:
1009571265

Дата проверки:
07.12.2021 12:09:52 EET

Тип проверки:
Doc vs Internet

Дата отчета:
07.12.2021 12:13:44 EET

ID пользователя:
100005657

Название файла: Шевчук ПМм-20

Количество страниц: 75 Количество слов: 10199 Количество символов: 70766 Размер файла: 1.85 MB ID файла: 1009577865

Обнаружены модификации текста (могут влиять на процент совпадений)

5.37%
Совпадения

Наибольшее совпадение: 3% с Интернет-источником (https://dokumen.pub/fundamentos-de-diseo-logico-y-de-comput...

5.37% Источники из Интернета | 354 ..... Страница 77

Поиск совпадений с Библиотекой не производился

0% Цитат

Не найдено ни одной цитаты

Не найдено ни одной ссылки

0% Исключений

Нет исключенных источников

Модификации

Обнаружены модификации текста. Подробная информация доступна в онлайн-отчете.

Замененные символы | 31

Подозрительное форматирование | 21 страница

Завідувачу кафедри
телекомунікацій, медійних та
інтелектуальних технологій (ТМІТ)
Підченко С.К.
здобувача вищої студентки
2 курсу, гр. ПМм-20-1
Шевчук Ольги Олександрівни

Заява

З правилами чинного Положення „Про дотримання академічної доброчесності в Хмельницькому національному університеті" від 26.09.2020 (зі змінами від 26.11.2020), згідно з якими виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту та застосування заходів дисциплінарної та академічної відповідальності, ознайомлена. Про використання програмно-технічних засобів для перевірки кваліфікаційних робіт здобувачів вищої освіти на плагіат повідомлена та надаю свою згоду на обробку та збереження університетом моєї роботи в інституційному репозиторії університету.

Також надаю університету право на передачу моєї роботи для обробки та зберігання в базах даних програмно-технічних засобів (Unicheck та Anti-Plagiarism) та використання роботи для виявлення плагіату в інших роботах, які перевіряються програмно-технічними засобами та користувачами, що мають доступ до цих програмно-технічних засобів, виключно в обмежених цілях для виявлення плагіату в текстах робіт.

Робота для перевірки університетом надається в друкованому та електронному варіанті. Електронна версія моєї роботи збігається (ідентична) з друкованою.

6.12.21 р. Шевчук О.О.

РІШЕННЯ КАФЕДРИ

ТЕЛЕКОМУНІКАЦІЙ, МЕДІЙНИХ ТА ІНТЕЛЕКТУАЛЬНИХ ТЕХНОЛОГІЙ

ПРО ДОПУСК КВАЛІФІКАЦІЙНОЇ РОБОТИ ДО ЗАХИСТУ

Підтверджуємо ознайомлення з результатом звіту подібності щодо роботи, генерованого системою виявлення текстових збігів/ідентичності/схожості:

Назва: Багатокритеріальна модель формування оптимального штату виконавців-розробників ІТ-компанії

Автор**: Шевчук Ольга Олександрівна**

Спеціальність: **113 Прикладна математика**

Освітня програма: Прикладна математика

Науковий керівник: **к.т.н., доц. Драч Ілона Володимирівна**

Після аналізу звіту подібності зроблено такий висновок:

№	Висновок	Позначка про відповідність
1	Запозичення, виявлені в роботі, є законними і не є плагіатом (далі – зазначаються підстави віднесення запозичень до правомірних). Робота приймається до захисту.	**Відповідає**
2	Виявлені запозичення не є плагіатом, розміщені в розділах, які не описують безпосередньо авторське дослідження, але кількість цитат перевищує обсяг, виправданий поставленою метою роботи (далі – зазначаються детальні та аргументовані підстави віднесення запозичень до правомірних). Робота приймається до захисту, але має бути відкоригована. Відкоригований варіант має бути поданий на кафедру за 2 дні до захисту, разом із заявою щодо самостійності виконання письмової роботи та ідентичності друкованої та електронної версії роботи	
3	Виявлені запозичення не є плагіатом, але частково розміщені в розділах, які описують безпосередньо авторське дослідження, а кількість цитат перевищує обсяг, виправданий поставленою метою роботи. В зв'язку з цим мета роботи та поставлені завдання не були досягнені. Робота може бути допущена до захисту (наступного року) після того як буде відкоригована та допрацьована і успішно пройде повторну перевірку на академічний плагіат.	
4	Робота містить навмисні текстові спотворення, передбачувані спроби укриття запозичень або інші прояви академічного плагіату. Робота містить фабрикацію або фальсифікацію даних. Робота не допускається до захисту.	
5	Інше:	

Підтвердження: Запозичення у розмірі 5,37% є випадковими збігами та на рівень подібності не впливає. Модифікації тексту пов'язано із вилученням інформації типових бланків оформлення роботи.

7.12.2021р.

Відповідальний за контроль

плагіату за системою Unicheck: Олег ПИВОВАР

Зав. каф. ТМІТ Сергій ПІДЧЕНКО