Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет комп’ютерної інженерії та управління
(повна назва)

Кафедра електронних обчислювальних машин
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

Рівень вищої освіти другий (магістерський)

Інтелектуальна система прийняття рішення для організації гуманітарної логістики
(тема)

Виконав:

студент II курсу, групи СПм-22-3

Яковлев А.В.
(прізвище, ініціали)

Спеціальність

123 «Комп’ютерна інженерія»
(код і повна назва спеціальності)

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма

Системне програмування
(повна назва освітньої програми)

Керівник: доц. Ільїна І.В.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри ЕОМ _____ Коваленко А.А.
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет комп’ютерної інженерії та управління

Кафедра електронних обчислювальних машин

Рівень вищої освіти другий (магістерський)

Спеціальність 123 «Комп’ютерна інженерія»
(код і повна назва)

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Системне програмування
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
“____” _____ 20___ р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

студенту Яковлєву Андрію Віталійовичу
(прізвище, ім’я, по батькові)

1. Тема роботи Інтелектуальна система прийняття рішення для організації гуманітарної логістики

затверджена наказом по університету від “ 01 ” квітня 2024 р. № 257 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 15червня 2024

3. Вхідні дані до роботи вхідні дані, браузер, доступ в Інтернет, Webstorm

4. Перелік питань, що потрібно опрацювати у роботі Аналіз інтелектуальних інформаційних систем та галузей їх застосування, розглянути інструментарій для створення системи підтримки прийняття рішень, розглянути вже існуючі інтелектуальні системи підтримки прийняття рішень та їх функціонал, проєктування та моделювання системи підтримки прийняття рішень, програмна реалізація

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) 11

6. Консультанти розділів роботи (заповнюється за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз інтелектуальних інформаційних систем	01.04.2024	
2	Розгляд інструментів для створення СППР	10.04.2024	
3	Розгляд існуючих СППР в логістиці	12.04.2024	
4	Проєктування та моделювання СППР	13.04.2024	
5	Програмна реалізація	28.04.2024	
6	Оформлення матеріалів кваліфікаційної роботи	20.05.2024	
7	Подання кваліфікаційної роботи керівникові	08.06.2024	
8	Подання кваліфікаційної роботи на рецензування	12.06.2024	

Дата видачі завдання 01 квітня 2024 р.

Студент _____
(підпис)

Керівник роботи _____ доц. Ільїна І.В.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 105 с., 12 рис., 3 табл., 2 дод., 40 джерел.

СИСТЕМИ ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ, ГУМАНІТАРНА ЛОГІСТИКА, TYPESCRIPT, VUE.

Метою кваліфікаційної роботи є розробка моделі системи підтримки прийняття рішень в гуманітарній логістиці. Для створення моделі та доказу концепції також були проаналізовані: інструменти для створення СППР, існуючі СППР в логістиці, технології для реалізації.

Програмна реалізація продукту розроблена на мові програмування TypeScript. В WebStorm був створений доказ концепції, який підтверджує що реалізація моделі на практиці можлива.

ABSTRACT

Master’s thesis: 105 pages, 12 figures, 3 tables, 2 appendices, 40 sources.

DECISION SUPPORT SYSTEMS, HUMANITARIAN LOGISTICS, TYPESCRIPT, VUE.

The objective of the qualification work is to develop a model of a decision support system in humanitarian logistics. To create the model and demonstrate the proof of concept, various tools for building decision support systems, existing decision support systems in logistics, and relevant technologies for implementation were analyzed.

The software implementation of the product was developed using the TypeScript programming language. A proof of concept was created in WebStorm, demonstrating that the practical implementation of the model is feasible.

ЗМІСТ

-

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

DSS – система підтримки прийняття рішень (англ., Descision Support System)

СППР – система підтримки прийняття рішень

IT – інформаційні техналогії (англ., Information technologies)

IDSS – інтелектуальна система підтримки прийняття рішень (англ., Intelligent Descision Support System)

ІІС –інформаційна система

C/S – клієнт/сервер (англ., Client/Server)

B/S – браузер/сервер (англ., Browser/Server)

AI – Штучний інтелект (англ., Artificial intelligence)

ШІ – Штучний інтелект

HR – людський ресурс (англ., Human resourse)

HRM – кадровик (англ., Human resourse manager)

ML – машинне навчання (англ., Machine learning)

ІСПР – інтелектуальна система підтримки рішень

ІСППР – інтелектуальна система підтримки прийняття рішень

AIAC – автоматизовані інформаційно-аналітичні системи

SWOT – сильні сторони, слабкі сторони, можливості, загрози (англ., Strengths, Weaknesses, Opportunities, and Threats)

СПР – система прийняття рішень

C/S – клієнт/сервер (англ., Client/Server)

B/S – браузер/сервер (англ., Browser/Server)

AI – Штучний інтелект (англ., Artificial intelligence)

ШІ – Штучний інтелект

HR – людський ресурс (англ., Human resourse)

HRM – кадровик (англ., Human resourse manager)

ML – машинне навчання (англ., Machine learning)

ВСТУП

В сучасному світі, де гуманітарні кризи та надзвичайні ситуації стають невід'ємною частиною нашого життя, роль волонтерських організацій у сфері логістики стає дедалі важливішою. Забезпечення швидкої та ефективної допомоги у потрібний момент вимагає високого рівня організації та точності при прийнятті рішень. У цьому контексті використання систем підтримки прийняття рішень набуває ключового значення для координації дій волонтерів у логістичних процесах [1].

Гетерогенний обмін інформацією не зміг задовольнити більше систем, диверсифікованих і численних структур пропозиції та попиту на людські ресурси з обох сторін попиту, таким чином реалізувати обмін інформацією, а інформація про ресурси є фундаментальною для вирішення проблеми. На основі «хмарних обчислень», що виникають, автор обговорює архітектуру та метод впровадження системи підтримки прийняття рішень щодо архівної інформації (DSS) людських ресурсів на основі «хмарних обчислень».

Хмарні обчислення базуються на зростанні, використанні та доставці взаємопов’язаних веб-сервісів, які зазвичай передбачали надання динамічних, легко розширюваних і часто віртуалізованих ресурсів через Інтернет. У вузькому сенсі хмарні обчислення стосуються режиму доставки та використання базової IT-інфраструктури та отримують необхідні ресурси через мережу за вимогою та легко розширеним способом. У широкому розумінні хмарні обчислення стосуються режиму надання та використання послуг, що означає отримання необхідних послуг через мережу за вимогою та легко розширюваний спосіб. Такими послугами можуть бути інформаційні технології та програмне забезпечення, пов’язані з Інтернетом або інші послуги. Він йде після централізованих обчислень, розподілених обчислень, настільних обчислень і мережевих обчислень. Порівняно з грід-обчисленнями хмарні обчислення мають більше переваг [2]. Хмарні

обчислення мають такі характеристики [3]:

- вони інтегрують та оптимізують обчислювальні ресурси для покращення загальної обчислювальної потужності;
- режим розподіленого управління центром обробки даних може бути реалізований для підвищення стійкості системи до стихійних лих;
- апаратне та програмне забезпечення можуть бути ізольовані одне від одного, щоб зменшити взаємозалежність обладнання;
- він реалізує модульний дизайн платформи розробки.

СППР – це свого роду інтелектуальна комп'ютерна інформаційна система, яка допомагає менеджерам приймати наукові напівструктуровані рішення за допомогою взаємодії людини з комп'ютером і моделювання даних. Система підтримки прийняття рішень, застосована в людських ресурсах, заснована на теорії сховищ даних і спираючись на технологію бізнес-аналітики, дозволяє користувачам створювати і керувати багатовимірними тривимірними моделями даних, аналізувати і відображати дані в диверсифікованих діаграмах [4].

В даний час багато вітчизняних підприємств і установ реалізували електронну інформатизацію внутрішніх кадрових ресурсів і розробили і застосували велику кількість систем управління для реалізації щоденного управління персоналом підприємств [5]. Каран та ін. [6] вважають, що людські ресурси – це сукупність людського капіталу, і дуже легко скопіювати систему управління людськими ресурсами на практиці, але дуже важко отримати висоту стратегічної позиції людських ресурсів, і ефект буде бути дуже поганим, і навіть буде негативний ефект. Джільяреллі та ін. [7] висунув, що практика людських ресурсів є фундаментальною та є джерелом, а практика є передумовою успіху стратегії людських ресурсів підприємства, яку не можуть скопіювати конкуренти. Водночас вони зазначають, що кадрова практика є органічною системою багаторівневої співпраці, взаємозалежності та взаємодоповнення.

Розенталь та ін. [10] запропонували, що планування та прогнозування

людських ресурсів є стратегічним заходом, який зазвичай використовується у великих і середніх установах і підприємствах, і є процесом попередньої підготовки до задоволення потреб клієнтів, що постійно змінюються. Планування людських ресурсів є важливою частиною та ядром стратегії розвитку компанії. Хантінгтон та ін. [11] зазначає, що планування трудових ресурсів є стратегією розвитку підприємства. Менеджери завершують виробництво та управління, використовуючи передову теорію та практику, разом із безперервним розвитком суспільства та різноманітними змінами всередині та за межами підприємств – це баланс між постачанням робочої сили. Lappalainen та ін. [12] вважають, що суттю планування людських ресурсів є напрямок розвитку компанії. Вище керівництво бере за ціль напрямок розвитку та попит на людські ресурси та організовує та планує виробництво відповідно до фінансового бюджету. Wall та ін. [13] використовують технологію Інтернет/Інтранет, щоб забезпечити комплексне рішення для прийняття рішень щодо людських ресурсів підприємства, таких як організаційна структура підприємства та схема планування структури персоналу, і генерувати результати.

Більшість дослідників погоджуються, що мета систем підтримки прийняття рішень, полягає в тому, щоб підтримати рішення неструктурованого управління та забезпечити обробку знань за допомогою комунікаційних можливостей [7]. IDSS може використовувати знання певної області та виконувати деякі типи інтелектуальної поведінки, наприклад навчання та прийняття рішень на основі міркування. Програми IDSS використовуються в різних областях, таких як: розробка та планування, управлінські рішення, підприємства та виробничі галузі, послуги, тощо.

Метою дослідження є аналіз підходів та методів для створення моделі підтримки прийняття рішень саме для організації гуманітарної логістики, проаналізувати існуючі методи розв’язання задачі, ознайомитись з основними підходами та вибрати найбільш підходящий в даній ситуації. На основі аналізу, проведеного в даній роботі можливо створити модель, яка

допоможе відібрати продуктивні шляхи врегулювання більшості процесів організації гуманітарної логістики.

1 ГАЛУЗІ ЗАСТОСУВАННЯ І ВИДИ СИСТЕМ ПРИЙНЯТТЯ РІШЕННЯ

ІІС – це комп'ютерна система, що складається з 5 основних взаємодіючих компонентів:

- мовної підсистеми (механізм забезпечення зв'язку між користувачем та іншими компонентами);
- інформацією підсистеми (сховище даних та засобів їх обробки);
- підсистеми управління знаннями (сховище знань про проблемну галузь, таких як процедури, евристики та правила, та засоби обробки знань);
- підсистеми керування моделями;
- підсистеми обробки та вирішення завдань (сполучна ланка між іншими підсистемами) [8].

Система може здійснювати оптимізацію та порівняльний аналіз, а також реалізувати інтеграцію та інтуїтивне вираження інформації про дані. Сяо та ін. [14] і Stone et al. [15] обидва обговорювали підтримку прийняття рішень щодо людських ресурсів на підприємствах. Наразі технічна платформа, яка використовується системою управління персоналом у країні та за кордоном, поступово переходить від автономної до мережевої інтегрованої системи, а режим розробки системи також поступово оновлюється від традиційної архітектури C/S до B/S архітектура. Технологія розробки заснована на архітектурі браузер-проміжне програмне забезпечення-сервер на основі трирівневої архітектури.

Додатки веб-сервісів на різних платформах розгортаються та реалізуються за допомогою таких технологій, як фонова база даних і проміжне програмне забезпечення [16]. Однак із постійним зростанням розвитку бізнесу багато систем управління людськими ресурсами більше не можуть задовольнити їхні потреби. По-друге, недостатньо досліджень щодо реалізації функції підтримки прийняття рішень у системі управління

персоналом. Bohlouli та ін. [17] представив системний компонент, заснований на підтримці прийняття рішень на основі знань, який може відображати статус прибутків і статус людських витрат підприємств у формі різноманітних інтуїтивно зрозумілих діаграм і надавати певну підтримку прийняття рішень підприємствам для формулювання ефективних стратегій розвитку. Деякі інші відомі постачальники програмного забезпечення також вивчають застосування підтримки прийняття рішень в управлінні людськими ресурсами [18].

В умовах переходу до економіки, що базується на знаннях, забезпечення ефективної роботи та конкурентоспроможності організації, особливо такої як університет, потребує підвищення уваги до персоналу та людського фактору. Працівники організації розглядаються як основний стратегічний ресурс, що забезпечує її діяльність та досягнення поставлених цілей. Відповідно до цієї концепції персонал є одним з основних ресурсів організації, яким необхідно грамотно керувати, створювати оптимальні умови для його розвитку, вкладати в це необхідні кошти. Основу концепції управління персоналом складають зростаюча роль особистості працівника, знання його мотиваційних установок, уміння формувати їх і спрямовувати відповідно до завдань, що стоять перед організацією.

Проблема відбору персоналу посаду належить до категорії слабоструктурованих завдань, традиційно зведених до прийняття рішень. У реалізації подібних завдань істотну роль відіграє думка особи, яка приймає рішення. Інтелектуальну підтримку політики вибору визначає конкретний керівник. При прийомі на роботу необхідно визначити наявність чи відсутність у кандидата необхідної для ефективної роботи компетенції – сукупності знань, навичок, здібностей, соціальних і особистісних показників і норм поведінки працівників, визначених цілями організації та завданням конкретної ситуації.

Слабоструктуроване завдання відбору персоналу характеризується такими особливостями:

- багатофакторність та багатокритеріальність;
- якісний та кількісний характер критеріїв та показників;
- необхідність врахування думки у процесі оцінки;
- залежність від вимог роботодавця, які визначають «портрет спеціаліста» на зайняття конкретної посади.

Перелічені особливості «занурюють» завдання прийому на роботу в нечітке середовище, зумовлюють прийняття рішень щодо відбору найбільш прийнятного кандидата на вакансію у нечіткій ситуації.

Загалом, інтелект – це здатність думати й розуміти, а не робити щось інстинктивно чи автоматично [13]. Основні ідеї інтелекту полягають у дослідженні процесів мислення людей і копіюванням цих процесів за допомогою машин. Вивчення штучного інтелекту (AI) полягає в тому, як змусити комп'ютери робити речі, у яких на даний момент люди є кращими, таких як:

- вміння вчитися на досвіді;
- сприйняття та міркування;
- робити висновки на основі міркування;
- швидко й успішно реагувати на незнайомі ситуації;
- визнавати відносну важливість різних елементів у ситуації;
- робіть висновки в ситуації, де існують нечіткість і невизначеність;
- використовувати знання та досвід для маніпулювання навколишнім середовищем.

Результатом поєднання інтелектуальні здібностей та комп'ютерної системи є розумна машина. Цей тип машини має допомагати людям приймати рішення, шукати інформацію, контролювати складні об'єкти та, нарешті, розуміти значення слів. Для того, щоб розробити інтелектуальну комп'ютерну систему, ми повинні утримувати, організовувати та використовувати людські експертні знання в деяких вузьких сферах знань, підвищити обчислювальну потужність ядра системи за допомогою складних алгоритмів, що використовують сенсорну обробку [14]. Крім того,

інтелектуальна система повинна бути здатна діяти належним чином у невизначеному середовищі, щоб збільшити ймовірність успіху – досягнення поведінкових підцілей, які підтримують кінцеву мету системи.

Інтелектуальна система підтримки прийняття рішень (IDSS) розроблена, щоб допомогти особам, які приймають рішення, на різних етапах прийняття рішень шляхом інтеграції інструментів моделювання та людських знань. IDSS – це інструмент, які допомагає у процесі прийняття рішень у випадках, коли існує невизначеність або неповна інформація або де рішення, пов’язані з ризиком, повинні прийматися на основі людського судження. IDSS, як випливає з назви, використовується для підтримки прийняття рішень і не призначений замінити завдання особи, яка приймає рішення. Скоріше це інтерактивна система, досить гнучка, адаптована та спеціально розроблена для підтримки вирішення неструктурованої проблеми управління для покращення якості рішень. IDSS є більше когнітивною, а не технологічною системою, фундаментальна відмінність полягає в тому, що навіть базові характеристики інтелекту не можуть бути визначені в механіці.

Останнім часом з’явилося досить багато комп’ютерних додатків, які застосовували інтелектуальні технології і використовували концепції та компоненти DSS. У цьому дослідженні я зосереджуюсь на додатках IDSS, у які вбудовано відповідні інтелектуальні методи. Насправді існують різні типи технологій штучного інтелекту, які використовуються для міркування, машинного навчання, автоматичного програмування, штучного життя, аналізу даних і візуалізації даних. Комп’ютер може «думати» за допомогою фреймів (наприклад, семантичної мережі), на основі правил, регістрів і розпізнавання шаблонів. Очікується, що IDSS включатиме знання про конкретну область і здійснюватиме певні типи інтелектуальної поведінки, наприклад навчання та міркування, для підтримки процесу прийняття рішень. Інтелектуальна поведінка, представлена інтелектуальною системою, пов’язана зі здатністю збирати та об’єднувати знання предметної області, вчитися на отриманих знаннях, міркувати про такі знання та мати можливість

видавати рекомендації та обґрунтовувати результати по запиту. Ці здібності використовуються для підтримки процесів прийняття рішень. Існують різні типи інтелектуальних методів, які застосовуються в програмах IDSS. З дослідження літератури, ми виявили, що більшість програм використовують «систему основану на знаннях».

Інтелектуальний аналіз даних – це підхід, якому зараз приділяється велика увага, і його визнають новим інструментом аналізу. Це пояснюється широкою доступністю величезних обсягів даних і важливою необхідністю перетворення таких даних на корисну інформацію та знання. Проблеми інтелектуального аналізу даних зазвичай класифікуються як кластеризація, асоціація, класифікація та прогнозування [9]. Протягом багатьох років інтелектуальний аналіз даних використовував різні методи, включаючи статистику, нейронну мережу, дерево рішень, генетичний алгоритм і методи візуалізації. Тим не менш, його застосування в УЛР зустрічається рідко [9]. Додатки прогнозування, які використовують інтелектуальний аналіз даних у HRM, є нечастими, наприклад: для прогнозування тривалості обслуговування, премій за продаж, індексів стійкості страхових агентів або аналізу неправильної поведінки операторів [5]. Інтелектуальний аналіз даних є одним із найкращих підходів до аналізу записів у баз даних. Проаналізовані результати можуть бути використані для майбутнього планування. Метод інтелектуального аналізу даних також застосовувався в проблемних сферах кадрової роботи, але зосереджувався на завданнях відбору персоналу, і застосовувався в інших видах діяльності, таких як планування, навчання, управління талантами, тощо. Останнім часом, відповідно до нових вимог у відділах кадрів, шукають нові методи аналізу даних [5]. Це можна реалізувати шляхом ідентифікації згенерованих шаблонів з існуючих даних у базах даних HR як корисних знань. Шаблони можна генерувати за допомогою деяких основних методів інтелектуального аналізу даних, наприклад кластеризації, об’єднання, прогнозування та класифікації. Існує багато завдань, пов’язаних із людськими ресурсами, які можна вирішити за

допомогою методів інтелектуального аналізу даних, таких як оцінка продуктивності співробітників, методи консультування та управління ефективністю рішень [5]. Для отримання результатів інтелектуального аналізу даних, які підходять для завдань з пошуку людських ресурсів, слід дотримуватися кількох кроків у процесі інтелектуального аналізу даних. Першим кроком процесу інтелектуального аналізу даних є отримання основних наборів даних для вибору набору даних. Вони можуть бути зібрані з оперативних баз даних кадрових ресурсів або з вибраного сховища даних людських ресурсів. Потім вибрані дані проходять очищення та попередню обробку для усунення розбіжностей і невідповідностей у наборі даних і водночас для покращення якості набору даних. Далі набір даних аналізується для виявлення шаблонів, які представляють зв’язок між даними, застосовуючи алгоритми, такі як нейронні мережі, наприклад дерево рішень, груба теорія множин тощо. Потім шаблони перевіряються за допомогою нових наборів даних. Крім того, повинна бути можливість трансформувати створені шаблони в дієві плани, які, ймовірно, допоможуть співробітникам людських ресурсів досягти своїх цілей. Етапи процесу видобутку повторюються, доки не буде отримано значущі знання. Шаблон, який задовольняє ці умови, стає організаційним знанням і може бути використаний у будь-яких пов’язаних програмах HR для завдань людських ресурсів.

Запропонована структура HR IDSS містить чотири типи система підтримки прийняття рішень.

1.1 Система керована даними

Основний підхід цієї системи використовується для розробки прогностичної моделі та для виявлення можливих шаблонів і правил з існуючої системи баз даних. В базах даних кадрів, пов’язаних з такими проблемами як прогнозування продуктивності співробітників, ми можемо використовувати

інформацію про персонал, дані оцінки продуктивності та інші пов’язані бази даних. Відповідні дані будуть перетворені на корисні знання як прогностичну модель за допомогою прогностичного моделювання.

1.2 Система на основі моделі

Система на основі моделі повинна зберігати сконструйовану модель, існуючу модель та пов’язані моделі, які можна використовувати у відповідному процесі прийняття рішень.

1.3 Система керована знаннями

Ця система містить набір фактів і правил. У запропонованій структурі KBS міститиме інформацію про шаблони, правила асоціації та будь-які пов’язані факти та правила.

1.4 Комунікаційні системи

Повинна бути механізмом висновків у програмі HR DSS, яка контролює взаємодію між різними частинами програми HR. По суті, цей компонент буде діяти як інтерфейс між користувачем і самою системою.

HR IDSS, як частина додатків інтелектуальної системи, відіграє ті самі ролі, щоб сприяти процесу прийняття рішень. Застосування та інтелектуальні методи HR IDSS потребують багато уваги та зусиль, як з боку академіків, так і практиків. Незважаючи на потенціал HR IDSS є кілька особливостей. По-перше, у HRM є багато проблемних областей, які необхідно досліджувати. Дослідники повинні визначити проблемні області, де потрібні інструменти для перетворення невизначених і неповних даних у корисні знання. З цієї причини для програми HR IDSS слід використовувати підхід машинного навчання. По-друге, необхідно використовувати гібридні інтелектуальні

методи, так як вони є найкращим підходом для підтримки прийняття рішень, особливо в міркуваннях і навчанні. По-третє, необхідно постійно вдосконалювати основні знання для ефективної HR IDSS. Цей процес можна покращити шляхом постійного розвитку веб-інструментів, бездротового протоколу та системи підтримки прийняття групових рішень. Однак, на даний момент для створення ефективної системи необхідно більше застосування HR IDSS та інтелектуальних методів до різних проблемних областей у сфері HRM.

2 АНАЛІЗ ІСНУЮЧИХ ІНТЕЛЕКТУАЛЬНИХ СИСТЕМ ПРИЙНЯТТЯ РІШЕНЬ

2.1 Огляд сучасних систем прийняття рішень, та її економічна ефективність

Прийняття рішень – невід’ємна частина нашого повсякденного життя. За деякими оцінками, ми щодня свідомо робимо вибір близько 35 000 разів. Із розвитком горизонтальних організаційних структур, де децентралізоване прийняття рішень стало нормою, кількість цих рішень лише зростає. Замість того, щоб чекати на схвалення вищого керівництва, співробітники всіх рівнів отримують право на самостійне прийняття рішень, часто колективно. Інформаційні системи підтримки рішень (ІСПР) відіграють важливу роль у сучасному світі, допомагаючи організаціям та окремим особам приймати обґрунтовані рішення на основі аналізу даних. Серед найефективніших методів, які використовуються в ІСПР для прийняття рішень, варто виділити штучний інтелект (ШІ), машинне навчання (ML) та аналітику даних, кожен з цих методів має свої унікальні переваги та може значно підвищити ефективність процесу прийняття рішень.

ІСПР допомагають користувачам у процесі прийняття рішень та забезпечують підтримку в різних контекстах завдань. Думки людини та комп'ютерна інформація працюють разом для прийняття рішень, при цьому ІСППР підсилює міркування та оцінки осіб, які приймають рішення (ОПР), але не замінює їх і не скасовує, контроль все ж залишається за людиною. Досягнення найбільш ефективних рішень в ІСПР забезпечується принципом інтерактивного вирішення завдань. Користувач має можливість підтримувати безперервний діалог із системою, крім того, при вирішенні складних масштабних проблем забезпечується колективне прийняття рішень, коли до процесу у багатокористувацькому режимі залучається необхідна кількість

експертів.

Потужність обчислень та робота в реальному часі дозволяє ІСППР інтегрувати моделі та аналітичні методи зі стандартним доступом до даних і вибірками з них. Для надання допомоги при прийнятті рішень активуються одна або кілька моделей. Бази даних включають історію поточних і попередніх операцій, а також зовнішню інформацію та дані про середовище, а візуалізація результатів обробки інформації на кожній ітерації у вигляді діаграм, графіків, просторової інформації на електронних картах сприяє глибокому аналізу ситуації користувачем.

Для ІСППР характерною є оперативна взаємодія із зовнішнім середовищем та актуальне оновлення інформації завдяки роботі з мережею. На всіх етапах процесу забезпечується захист конфіденційної інформації, оскільки зазвичай інформація, що стосується рішень, прийнятих керівництвом, може викликати інтерес конкурентів та зловмисників. ІСППР орієнтовані на гнучкість і адаптивність для пристосування їх до змін середовища або модифікації підходів до вирішення завдань, обраних користувачем. ОПР має адаптуватися до змінних умов самостійно і відповідно шляхом надання необхідних запитів готувати до них систему. Водночас ІСППР не нав'язує користувачеві визначеного процесу прийняття рішень, а слідує лише за запитами та вказівками користувача, ці системи прості у використанні (звісно, для осіб, які мають досвід роботи з комп'ютером), вони забезпечують легке пересування по системі, і користувач почувається комфортно та впевнено.

Останнім часом сформувалося четверте покоління автоматизованих систем – автоматизовані інформаційно-аналітичні системи (АІАС). У них широко використовуються аналітичні засоби обробки інформації та забезпечуються процеси прийняття рішень, сучасні АІАС включають риси (елементи) всіх попередніх поколінь. У таких системах є власне ІСППР та їх сучасна модифікація – інтелектуальні системи підтримки прийняття рішень.

Перш за все, ІСППР використовують знання експертів для підтримки

процесу прийняття рішень, що дозволяє системам надавати більш точні та релевантні рекомендації. По-друге, вони застосовують методи штучного інтелекту для моделювання свідомих розумових процесів людини, що підвищує точність і ефективність аналізу даних. Нарешті, ІСППР забезпечують максимальне використання можливостей як людських експертів, так і програмно-технічних засобів для вирішення управлінських завдань.

ІСППР набувають все більшого поширення, особливо в економічній сфері, де вони об'єднуються під загальною назвою "дорадчі економічні системи" (ДЕС). Метою створення ДЕС є надання управлінському персоналу знань, яких їм бракує у процесі виконання своїх професійних обов’язків, а також навчання конкретним діям, необхідним для реалізації рекомендацій системи з подальшим контролем їх виконання. Другий клас ДЕС включає два підкласи систем: нейромережеві обчислення та орієнтовані на природно-мовні запити. На даний момент практична реалізація таких систем є складним завданням і питанням майбутнього.

Для подальшої роботи важливо було розглянути інструменти розробки ІСПР.

Таблиця 2.1 – Інструментарій для розробки СППР

Етап	Інструментарій (методи та заходи)	Проміжні результати
1	2	3
Аналіз і декомпозиція процесу прийняття рішення	Протоколи нарад. Наказ про створення системи	Протокол збору даних опитувань і декомпозиції задачі
	Визначення задачі прийняття рішення і декомпозиція її на окремі операції	Зведена таблиця опису ситуацій

Продовження таблиці 2.1

1	2	3
	Протокол збору даних щодо обмежень на процес прийняття рішень	Аналіз і визначення обмежень при прийнятті рішень
	Протокол збору даних щодо припустимих функцій системи	Визначення функціональних можливостей СППР
	Протокол збору даних щодо припустимих вимог	Формулювання вимог ОПП та функціональних вимог
Функціональне проектування та розробка специфікацій	Перелік методів і правил для вибору рішення	Визначення необхідних методів і способів автоматизованої підтримки прийняття рішення
	Функціональна архітектура системи	Детальне проектування діалогових вікон, структури керування СППР, програмних специфікацій
Реалізація, верифікація та супровід системи	Створення програмної системи і її тестування, протокол тестування	

Існує кілька моделей інтелектуальних систем прийняття рішень, але жодна з них не гарантує ідеального результату у всіх ситуаціях. У різних контекстах можуть бути ефективні різні моделі, наприклад, для прийняття складних стратегічних рішень може підійти модель SWOT-аналізу, яка

допомагає оцінити сильні і слабкі сторони, можливості та загрози. Для оперативних рішень більш придатною може бути модель «плюсів і мінусів», що дозволяє швидко зважити переваги і недоліки кожного варіанту.

Крім ознайомлення з різними моделями ухвалення рішень, важливо також розуміти, які упередження можуть впливати на ваш вибір. Наприклад, підтверджувальне упередження (коли ми шукаємо інформацію, яка підтверджує наші попередні переконання) або упередження втрати (коли страх втрати переважає над бажанням здобути вигоду). В контексті гуманітарної допомоги, де кожне рішення може мати критичне значення для життя та благополуччя людей, СППР стають невід'ємним інструментом управління та прийняття стратегічних рішень.

Класифікація СППР може проводитися за різними критеріями. За областю застосування розрізняють системи для бізнесу і менеджменту, інжинірингу, фінансів, медицини та охорони навколишнього середовища. Це дозволяє адаптувати систему до конкретних потреб та особливостей галузі. Класифікація СППР за областю застосування стає ключовою для розуміння їхнього впливу на гуманітарну допомогу. У зазначених областях, таких як бізнес і менеджмент, фінанси, медицина та інжиніринг, СППР використовуються для планування та управління різними аспектами гуманітарної діяльності. В тому числі використання СППР у фінансовій сфері дозволяє ефективно управляти фінансовими ресурсами, щоб забезпечити оплату гуманітарних послуг та придбання необхідного обладнання. Другий підхід до класифікації - це розподіл за відношенням між даними та моделями, така методика включає у себе різноманітні системи доступу до даних, аналізу інформації, обчислень фінансових наслідків та інші. Третя класифікація СППР - за типом використаного інструментарію, тут вже охоплюються моделі, що базуються на класичних аналітичних методах, на історичних даних, на груповому прийнятті рішень та на знаннях, що машинно виводяться. Використання штучного інтелекту, бізнес-аналітики, оптимізаційних систем та інших інструментів дозволяє досягти

більш точних та ефективних результатів у вирішенні навіть найскладніших завдань [19].

Таблиця 2.2 – Класифікація систем підтримки прийняття рішень

Критерій класифікації	Варіанти	Опис
1	2	3
За областю застосування	Бізнес і менеджмент	Системи, що використовуються для прийняття рішень у сфері бізнесу та менеджменту, наприклад, для планування, прогнозування, бюджетування, аналізу ринку тощо.
	Інжиніринг	Системи, що використовуються для прийняття рішень у сфері інженерії, наприклад, для проектування, тестування, моделювання, оптимізації тощо.
	Фінанси	Системи, що використовуються для прийняття рішень у сфері фінансів, наприклад, для інвестування, оцінки ризиків, управління портфелем тощо.
	Медицина	Системи, що використовуються для прийняття рішень у сфері медицини, наприклад, для діагностики, лікування, планування лікування тощо.

Продовження таблиці 2.2

1	2	3
	Охорона навколишнього середовища	Системи, що використовуються для прийняття рішень у сфері охорони навколишнього середовища, наприклад, для моніторингу довкілля, оцінки ризиків, планування природокористування тощо.
За співвідношенням дані/модель (методика Стивена Альтера)	FDS (File Drawer Systems)	Системи, що базуються на ручному аналізі даних та не мають формальних моделей.
	DAS (Data Analysis Systems)	Системи, що використовують прості статистичні методи для аналізу даних.
	AIS (Analysis Information Systems)	Системи, що використовують складні аналітичні методи та моделі для аналізу даних.
	AFM(s) (Accounting & Financial models (systems))	Системи, що використовуються для бухгалтерського обліку та фінансового моделювання.
	RM(s) (Representation models (systems))	Системи, що використовуються для створення моделей, які представляють реальні об'єкти або процеси.
	OM(s) (Optimization models (systems))	Системи, що використовуються для оптимізації рішень.

Продовження таблиці 2.2

1	2	3
	SM(s) (Suggestion models (systems))	Системи, що пропонують користувачам можливі рішення.
За типом використаного інструментарію	Model Driven	Системи, що базуються на формальних моделях для прийняття рішень.
	Data Driven	Системи, що базуються на даних для прийняття рішень.
	Communication Driven	Системи, що сприяють спілкуванню та співпраці між користувачами для прийняття рішень.
	Document Driven	Системи, що базуються на документах та звітах для прийняття рішень.
	Knowledge Driven	Системи, що використовують знання та досвід експертів для прийняття рішень.

Надалі розглянемо найпоширеніші методики ІСПР:

- машинне навчання (Machine Learning). Підхід зосереджений на створенні алгоритмів, які можуть навчатися на даних і робити прогнози або приймати рішення без явного програмування, він використовується для класифікації, кластеризації, регресії та інших завдань аналізу даних;

- нейронні мережі (Neural Networks). Спеціалізований тип алгоритмів машинного навчання, який моделює роботу людського мозку,

використовуються для розпізнавання образів, обробки природної мови, прогнозування та інших завдань;

- алгоритми класифікації (Classification Algorithms). Ці алгоритми призначені для визначення категорій, до яких можна віднести об'єкти на основі їх характеристик, використовуються для прийняття рішень у багатьох областях, таких як медицина, фінанси та бізнес;

- алгоритми кластеризації (Clustering Algorithms). Дані алгоритми групують схожі об'єкти разом на основі їх характеристик, допомагаючи відкривати приховану структуру даних та використовуються для сегментації ринків, аналізу вибірки споживачів та інших завдань;

- оптимізаційні алгоритми (Optimization Algorithms). Призначені для знаходження оптимальних рішень у складних умовах, використовуються для планування маршрутів, управління запасами, оптимізації виробничих процесів та інших завдань;

- статистичні методи (Statistical Methods). Використовуються для аналізу статистичних властивостей даних, включаючи розподіл ймовірностей, кореляцію та інші показники, вони допомагають робити висновки на основі наявних даних та прогнозувати майбутні події.

2.2 Оптимальна система для гуманітарної логістики

У сучасному світі гуманітарні кризи стають все більш поширеними і загостреними проблемами, що потребують ефективного та оперативного реагування, гуманітарна логістика, яка включає в себе планування, організацію та координацію розподілу гуманітарної допомоги, відіграє ключову роль у забезпеченні потреб населення в умовах кризи, однак, викликані складністю та непередбачуваністю ситуацій, управління гуманітарними операціями може стати великим викликом для організацій та агентств.

Системи прийняття рішень в гуманітарних кризах виконують кілька

ключових функцій, по-перше, вони забезпечують збір і аналіз даних про потреби постраждалих, охоплюючи інформацію про кількість людей, які потребують допомоги, а також типи та обсяги необхідних ресурсів. Наприклад, важливо знати, скільки людей потребує медичної допомоги, скільки - харчів, а скільки - притулку. По-друге, системи прийняття рішень допомагають визначити пріоритети розподілу ресурсів, тобто враховуючи обсяги потреб та обмеженість ресурсів, ці системи допомагають визначити, які потреби є найбільш критичними та потребують негайного вирішення. Наприклад, при великій кількості постраждалих може бути необхідно встановити пріоритет на надання медичної допомоги тим, хто потребує її найбільше. Ну і по-третє, системи прийняття рішень допомагають у плануванні логістичних операцій, тобто оптимізовувати маршрути доставки, управління запасами та координацію дій різних гуманітарних організацій. Такий підхід дозволяє ефективно використовувати обмежені ресурси та максимально швидко надавати допомогу постраждалим [20].

ІСППР варто використовувати для управління запасами гуманітарної допомоги, щоб гарантувати, що необхідні товари доступні в потрібний час і в потрібному місці, в майбутньому таке його використання допомогатиме запобігти дефіциту та забезпечити ефективне використання ресурсів. Щодо використання для аналізу історичних даних, даних про поточні потреби та інших факторів, щоб прогнозувати майбутній попит на гуманітарну допомогу, такий метод допомагатиме організаціям заздалегідь запастися необхідними товарами, щоб уникнути дефіциту. Окрім цього варто звернути увагу і на такі можливості ІСППР як управління ланцюгом постачання, оптимізація запасів, розподіл допомоги. Тобто інтелектуальні системи прийняття рішень варто використовувати для розробки оптимальних маршрутів, що дозволятиме мінімізувати час та витрати на доставку допомоги до постраждалих областей. Шляхом аналізу географічних даних, інформації про стан доріг та інших чинників, інтелектуальні системи прийняття рішень можуть знайти оптимальний шлях для доставки допомоги,

що забезпечить її швидку та ефективну розподіл. Також ці системи допомогають організаціям управляти запасами, щоб гарантувати, що необхідні товари доступні в потрібний час і в потрібному місці.

За останні два десятиліття розвиток систем підтримки прийняття рішень для гуманітарної логістики зріс в рази. Найбільш використовувані системи зосереджувалися головним чином на контролі запасів, зазвичай залишаючи осторонь логістику, пов'язану з транспортуванням і розподілом постачань. Саме з цієї причини системи, описані нижче, головним чином використовуються на етапах підготовки та реагування, через важливість, яку мають контроль запасів і управління автопарком на цих двох етапах; однак, залежно від конкретної ситуації, вони також можуть бути корисними на етапах відновлення та пом'якшення наслідків [21].

Нижче коротко описані деякі з найважливіших систем.

SUMA та LSS. У 1992 році Панамериканська організація охорони здоров'я разом з регіональним офісом Всесвітньої організації охорони здоров'я за підтримки різних країн (переважно Нідерландів) розробили Систему управління гуманітарними поставками (SUMA), яка є однією з найбільш повних систем, здатних керувати інформацією та ресурсами під час лиха. Ця система використовувалася як підтримка інвентаризації для інших завдань в організаціях роботодавців. Повна система складається з трьох модулів:

SUMA Central: це ядро системи, яке зазвичай розташоване в центрі операцій з надзвичайних ситуацій. Приймаючі рішення на цьому рівні збирають і визначають усі параметри проблеми, такі як стратегічні місця відновлення та доставки, місця для складів тощо.

Field Unit: його мета – керувати різними операціями в ключових місцях, таких як аеропорти, морські порти та місця, визначені на вищому рівні. Всі отримані постачання сортуються та класифікуються для кращого задоволення потреб постраждалого населення.

Управління запасами: реєструє всю інформацію, пов'язану з

отриманими та доставленими постачаннями. Цей модуль балансує наявні запаси на певному складі.

Після того, як стратегічні параметри були визначені центральною частиною SUMA, постачання повинні бути належним чином керовані. Для цієї мети використовуються польові підрозділи (Field Unit), які сортують і маркують зібрані постачання на різних пунктах прийому. Класифікація поділяється на три типи предметів: «Терміново! Негайний розподіл», «Не терміновий розподіл» та «Непріоритетні предмети». Додатково предмети розділяються на різні групи: продукти харчування, ліки тощо, а потім інформація передається до центральної частини SUMA. Нарешті, постачання інвентаризуються і доставляються до інших складів або розподільчих центрів, доходячи до постраждалого населення під координацією операційного центру.

Як показано вище, ця система вимагає наявності гарного інструменту для передачі даних між трьома модулями, що не завжди можливо в місці, яке постраждало від лиха. Ця система використовувалася до 2005 року, коли було впроваджено її наступника під назвою Система підтримки логістики (LSS), яка базувалася на програмному забезпеченні SUMA. Агентства, такі як РАНО і ОСНА, серед інших, зробили цей проєкт можливим. LSS полегшує обмін інформацією між НУО, донорами та постраждалими країнами.

Як і SUMA, LSS класифікує і сортує отримані постачання. Однак структура системи надає особливу увагу обробці постачань, які включені в базу даних:

- входи: різні отримані постачання, що прибули в певний локальний пункт, реєструються в системі, а також класифікуються і сортуються;

- поставки: система відстежує відправлення постачань, беручи до уваги початкові та кінцеві пункти, а також основні характеристики відправлення;

- трубопровід: очікувані постачання відображаються в цьому модулі системи;

- запит: ця частина системи керує інформацією про ті місця, які потребують конкретних постачань, і є корисною для визначення, як розподілити необхідні постачання;

- кошик запасів: після того, як постачання були класифіковані в групи, ця система відстежує доступні запаси для кращого управління майбутніми потребами;

- імпорт/експорт: у цій частині система може обмінюватися інформацією про постачання з іншими стратегічними місцями, зовнішніми системами або агентствами.

SUMA використовується з 1995 року, коли вона була застосована під час урагану Луїс на Карибських островах, а потім розгорнута під час інших природних катастроф, таких як ураган Мітч, що вразив Центральну Америку в 1998 році. LSS використовується з 2005 року, коли вона була застосована під час тропічного шторму Стан в Сальвадорі. Серед багатьох інших випадків, її також розгорнули під час землетрусів 2005 і 2007 років, які вразили Пакистан і Перу відповідно, і виверження вулканів 2008 року в Еквадорі. Для отримання детальнішої інформації про розгортання SUMA і LSS, див. [22-24]. Багато агентств використовували SUMA/LSS як підтримку гуманітарної логістики, таких як ООН, ВООЗ, ВПП, ООН, ЮНІСЕФ, УВКБ ООН, ПАОЗ, CTS, Червоний Хрест та багато інших організацій.

HLS та HELIOS. Гуманітарне логістичне програмне забезпечення (HLS) [34] є централізованою веб-системою управління ланцюгами постачання, яка була розроблена в 2003 році Інститутом Фріца у співпраці з Міжнародною федерацією Червоного Хреста (IFRC) і Червоного Півмісяця. Воно було створено для відстеження постачань і фінансування від пожертв до доставки, підвищення прозорості пожертв, прискорення ланцюга допомоги і надання детальнішої інформації особам, які приймають рішення, в штаб-квартирі і на місцях. Система надає онлайн-огляд ланцюга постачань допомоги і дозволяє швидко управляти ресурсами за допомогою веб-списків постачальників і каталогів, що дозволяє робити замовлення безпосередньо

онлайн. HLS вперше використовувалася IFRC у 2004 році під час кількох рятувальних операцій, таких як землетрус у Марокко, цунамі в Південній Азії або ураган на Гаїті. Пізніше вона також використовувалася під час таких операцій, як землетрус у Пакистані в 2005 році і землетрус і цунамі в Індонезії в 2006 році, серед інших.

Між 2007 і 2008 роками HLS було замінено більш просунутим програмним забезпеченням, також розробленим Інститутом Фріца, під назвою HELIOS [25]. Це спеціальний інструмент підтримки прийняття рішень для гуманітарної логістики, який надає організаціям дані в режимі реального часу для покращення управління ланцюгами постачання гуманітарної допомоги. Він складається з п'яти модулів:

- модуль управління проектами адмініструє відкриті проекти, оцінює основні потреби та керує запитами. Це головний модуль, який координує виконання інших модулів за потреби;

- модуль обробки запитів управляє інформацією щодо виданих запитів, таких як дані про відправника, необхідні товари, місця забору та доставки тощо;

- модуль управління складами займається управлінням запасами і може бути активований, коли надходить запит або здійснюється доставка;

- модуль мобілізації зосереджений на управлінні пожертвами, від грошових коштів до натуральних товарів та послуг. Він використовується організаціями для моніторингу та звітування про пожертви, значно покращуючи інформацію, що надається донорам щодо використання їхніх пожертв;

- модуль закупівель використовується для адміністрування та закупівлі товарів, управління відповідними запитами на котирування та виконання замовлень.

Операції автоматизуються через веб-систему для покращення координації між агентами, але вона також може працювати в автономному режимі, якщо немає інтернет-з'єднання. У 2008 році World Vision International

і Oxfam GB вперше впровадили HELIOS у двох локаціях кожен. Пізніше, у 2009 році, Oxfam розширив систему до 20 різних країн.

SAHANA. Це система управління катастрофами [26], яка була створена ІТ-спільнотою Шрі-Ланки після землетрусу та цунамі в Індійському океані в 2004 році. Вона була задумана як інтернет-система для управління зусиллями з надання допомоги під час фаз реагування та відновлення після катастрофи. Насправді, слово «Sahana» означає «допомога» сингальською мовою, однією з національних мов Шрі-Ланки.

Першим власником ІСППР Sahana був Фонд програмного забезпечення Ланка, що зробило Sahana глобальним проєктом відкритого програмного забезпечення, який підтримується волонтерами та кількома національними органами влади. У 2009 році була створена неприбуткова організація під назвою Фонд програмного забезпечення Sahana, щоб продовжити розвиток цього проєкту, залучаючи експертів з управління катастрофами.

Sahana - це автоматизована вебсистема, зосереджена головним чином на координації та плануванні гуманітарних операцій. З моменту першого впровадження у 2004 році програмне забезпечення було вдосконалено, і наразі воно містить такі інструменти [27]:

- Sahana Vesuvius: розроблена за участі Національної медичної бібліотеки США, зосереджена на потребах медичної спільноти у підготовці до катастроф та реагуванні на них. Використовується для допомоги лікарням, медичним установам і юрисдикціям у зв'язку записів постраждалих з повідомленнями про зниклих безвісти;

- Sahana Mayon: це інструмент для управління персоналом і ресурсами, який дозволяє керувати великою кількістю ресурсів. Він містить рішення для управління сценаріями катастроф, об'єктами та персоналом;

- Sahana Eden: гнучкий інструмент з кількома модулями, які можуть використовуватися на різних етапах управління катастрофами: зниження ризиків, підготовка, реагування та відновлення. Основні модулі такі: реєстр організацій (відстеження активних організацій для забезпечення

можливостей співпраці та координації), відстеження проєктів (організація проєктів та визначення найбільших потреб), управління людськими ресурсами (управління волонтерами та людьми, залученими до проєкту, ведення списку контактів і відстеження їх місцезнаходження та навичок), інвентаризація (управління запасами предметів і запитами складів, запис і автоматизація транзакцій для відправки та отримання вантажів), управління активами (управління активами, такими як транспортні засоби, радіообладнання або генератори, відстеження їх місцезнаходження та стану та забезпечення їх ефективного використання), оцінки (збір та аналіз інформації з оцінок різних організацій за допомогою користувацьких звітів, графіків і карт), сценарії та події (планування декількох сценаріїв відповідно до людських ресурсів, об'єктів тощо), картографія (інтегрована функціональність картографування, що забезпечує ситуаційну обізнаність і підтримку стандартних форматів з інших джерел і ГІС, таких як ризики природних катастроф, населення або погода) і управління притулками (відстеження інформації про притулки та людей, які прибувають і від'їжджають). Усі ці модулі, разом із додатковими необов’язковими модулями, можуть бути налаштовані та персоналізовані користувачами відповідно до їхніх конкретних потреб.

Sahana вперше була розгорнута Центром національних операцій (CNO) під час цунамі в Шрі-Ланці у 2005 році. З того часу вона використовувалася кількома організаціями для надання допомоги, наприклад, під час землетрусу в Кашмірі у 2005 році в Пакистані (Національною базою даних і реєстрації уряду Пакистану), землетрусу в Йогьякарті у 2006 році в Індонезії (Індонезійським джерелом допомоги) або під час повені у Венесуелі в 2010 році (урядом Венесуели). Інструмент Sahana Eden, нещодавно впроваджений, вперше був використаний для реагування на катастрофи під час землетрусу на Гаїті в 2010 році, коли був самостійно розгорнутий Фондом програмного забезпечення Sahana, також вперше. Система використовувалася в багатьох інших операціях з надання допомоги, деталі див. у [27].

HFOSS. Проект HFOSS [28] – це проект з відкритим вихідним кодом, який бере участь у кількох інших невеликих проектах різного спрямування. Зокрема, у сфері управління катастрофами HFOSS зробив внесок у проект Sahana, вже згаданий раніше, та розробив інші інструменти з відкритим вихідним кодом для гуманітарної допомоги. Один із таких інструментів – Collabbit [29], веб-програмне забезпечення для обміну інформацією між організаціями під час надзвичайних ситуацій. Наприклад, його використовувала Служба надзвичайних ситуацій Армії порятунку в Нью-Йорку для координації програми обіду на День подяки у 2008 році, щоб нагодувати понад 10 000 людей, або для моніторингу центрів охолодження по всьому Нью-Йорку для людей, які шукають порятунку від спеки влітку. Інший інструмент, розроблений HFOSS, це Posit [30], додаток для Android, який використовують рятувальники для складання карт району катастроф за допомогою мобільного телефону та зв’язку з центральним сервером і з іншими рятувальниками. Наприклад, його використовувала неприбуткова організація ACDI/VOCA під час операцій з надання продовольчої та медичної допомоги у віддалених сільських районах Гаїті після землетрусу 2010 року.

DMIS. Система інформації з управління катастрофами (DMIS) [31] – це вебінструмент, розроблений Міжнародною федерацією Червоного Хреста, що дозволяє отримувати доступ до інформації в реальному часі про тенденції катастроф, наявні ресурси та бази даних, для підтримки ефективної підготовки та реагування на катастрофи. Вона доступна лише для персоналу Червоного Хреста і Червоного Півмісяця, делегацій і штаб-квартири в Женеві.

LOGISTIX. У 2006 році організація "Лікарі без кордонів" розробила інструмент для інвентаризації медичних потреб під назвою LogistiX [32]. Він зосереджується на інвентаризації ліків і використовується у повсякденній роботі організації. Доступ до нього мають лише "Лікарі без кордонів", і організація проводить кілька курсів, щоб навчити своїх волонтерів

користуватися цим інструментом.

Варто зазначити, що, окрім раніше представлених систем, існують також інші системи, які не були спеціально розроблені для гуманітарної логістики, але також використовуються НУО для надання гуманітарної допомоги. У цій групі варто відзначити систему під назвою FleetWave [33], яка є веб-орієнтованою корпоративною інформаційною системою, зосередженою на управлінні автопарком, яку використовували, наприклад, МКЧХ, МФЧХ і ВПП.

Зазначимо, що ці системи головним чином зосереджені на управлінні інформацією, що полегшує доступ до даних щодо активних організацій, людських ресурсів і бенефіціарів, розміщених або запланованих замовлень, запасів і наявних матеріалів, карт тощо, і покращує зв’язок між агентствами та організаціями, які беруть участь. Ця інформація є важливою для прийняття рішень і дуже корисна на практиці, значно покращуючи ефективність операцій з надання допомоги. Однак, наскільки нам відомо, існуючі системи не впровадили жодної моделі прийняття рішень, такої як ті, що доступні в літературі і обговорювалися в попередньому розділі, для надання автоматизованих детальних планів інвентаризації або розподілу як результат.

В останні роки спостерігається вибух літератури щодо управління катастрофами та надзвичайними ситуаціями (рисунок 2.1).

Рисунок 2.1 – Фази циклу управління катастрофами

Математичні моделі стали важливим інструментом для вирішення питань гуманітарної логістики при катастрофах і надзвичайних ситуаціях, допомагаючи у процесах прийняття рішень, які виникають під час спроб реагування на наслідки. Незважаючи на те, що потрібні подальші дослідження для врахування додаткових невизначеностей, з якими стикаються у гуманітарній логістиці, і притаманних складнощів через проблеми комунікації та координації у моделей, у цьому напрямку було зроблено багато роботи.

2.3 Проблеми та виклики впровадження ІСПР в гуманітарну логістику

Впровадження інтелектуальних систем підтримки прийняття рішень (ІСПР) в гуманітарну логістику може принести значні переваги, такі як підвищення ефективності, результативності та прозорості операцій. Проте, існують також певні проблеми та виклики, які необхідно враховувати.

Розробка та впровадження ІСПР можуть бути надзвичайно дорогими, особливо для невеликих організацій. Високі початкові інвестиції у технології, програмне забезпечення та навчання персоналу можуть стати

бар'єром для багатьох гуманітарних організацій, навіть якщо вони усвідомлюють потенційні переваги використання таких систем. Доступ до фінансування та підтримки є критичним фактором для подолання цього виклику.

Однією з головних складностей є технічна складність самих ІСПР. Багато систем мають складні користувацькі інтерфейси, які можуть бути незрозумілими для користувачів без спеціальної підготовки, що ускладнює навчання та адаптацію працівників, а також може призвести до помилок в роботі системи. Крім того, функціональність таких систем часто є дуже широкою, що, з одного боку, є перевагою, а з іншого – недоліком, оскільки користувачам важко осягнути всі можливості системи та ефективно їх використовувати.

Інтеграція нових ІІС з існуючими системами також є серйозною проблемою, часто існуючі системи мають різні формати даних та протоколи зв'язку, що робить інтеграцію складною та затратною. Міграція даних з однієї системи в іншу вимагає ретельного планування і виконання, щоб уникнути втрати даних або їх пошкодження.

Складність збору та обробки даних в умовах гуманітарних криз теж є значущою проблемою. У кризових ситуаціях інфраструктура часто зруйнована, а доступ до ресурсів обмежений, що ускладнює збір інформації про кількість та розподіл людей, потреби в ресурсах, а також процеси розподілу та доставки допомоги. Більшість даних зазвичай є неповними або недостовірними через хаос та невизначеність ситуації. Ще однією проблемою є відсутність стандартизованих процедур та форматів для збору та обробки даних у гуманітарних кризах. Різні організації можуть використовувати різні системи та методики, що ускладнює обмін інформацією та координацію дій. Недостатня синхронізація даних може призвести до дублювання ресурсів, неефективного використання та збільшення ризику помилок.

У гуманітарній логістиці, де діємо в областях з кризовою ситуацією та потребою в надзвичайних заходах, етика має вирішальне значення. Збір,

зберігання та використання даних повинні відбуватися з повагою до конфіденційності осіб, їх прав та гідності. Один із великих викликів полягає в забезпеченні конфіденційності та приватності даних, зокрема особистих даних постраждалих, це означає, що гуманітарні організації мають бути особливо обережними з обробкою інформації та забезпечити її захищеність від несанкціонованого доступу.

В районах, порушених катастрофами або конфліктами, існує значна ймовірність, що інфраструктура буде пошкоджена або повністю зруйнована. Відновлення інфраструктури, такої як мережі зв'язку та електропостачання, може зайняти значний час і потребує значних фінансових ресурсів, що значно ускладнює встановлення та ефективне функціонування інформаційних систем, які залежать від доступу до цієї інфраструктури [16].

Низька доступність до Інтернету також може стати серйозним обмеженням, у багатьох регіонах, де діють гуманітарні організації, доступ до швидкого та надійного Інтернету може бути обмеженим або навіть відсутнім, це ускладнює використання веб-заснованих інформаційних систем, які потребують постійного зв'язку з Інтернетом для своєї роботи [16].

Звісно ж до проблем можна віднести і політичну волю, яка грає важливу роль у впровадженні інформаційних систем у гуманітарній логістиці. Без належної підтримки від керівництва організацій та урядових структур, важко забезпечити успішне впровадження та функціонування цих систем. Необхідно, щоб керівництво організацій та урядові структури розуміли важливість та переваги використання інформаційних систем у гуманітарній логістиці. Вони повинні бути готові виділяти необхідні ресурси та створювати сприятливе середовище для їх застосування.

Хоча інформаційні системи можуть забезпечити швидкі та ефективні рішення, важливо, щоб вони не заміняли людський фактор в процесі прийняття рішень. Людський досвід, судження та співчуття не можуть бути повністю замінені технологіями. Системи повинні бути спрямовані на підтримку та полегшення прийняття рішень людьми, а не на їх заміну.

Важливо, щоб людський фактор залишався ключовим у процесі прийняття рішень, особливо в умовах кризових ситуацій, коли потрібне гнучке та відповідальне вирішення проблем.

2.4 Тенденції розвитку ІІС у логістиці

Першою ключовою тенденцією є використання штучного інтелекту (ШІ) у розвитку ІСПР. Методи машинного навчання, нейронні мережі та алгоритми глибокого навчання використовуються для аналізу даних, прогнозування попиту, оптимізації маршрутів та управління запасами. Це дозволяє автоматизувати та оптимізувати логістичні процеси, зменшуючи витрати та підвищуючи ефективність операцій [16].

Другою важливою тенденцією є інтеграція з Інтернетом речей (IoT). Збір даних з датчиків, пристроїв та обладнання дозволяє отримувати реальні дані про логістичні процеси. ІСПР можуть використовувати ці дані для моніторингу, аналізу та управління логістичними операціями в реальному часі, що підвищує їхню точність та реактивність до змін.

Третьою тенденцією є зростання автоматизації та роботизації логістичних процесів. Сучасні ІСПР можуть автоматично виконувати багато рутинних операцій, що полегшує роботу персоналу та знижує його втомленість, підвищуючи продуктивність та якість роботи.

Четвертою тенденцією є розширена аналітика, яка дозволяє компаніям отримувати більш детальні та контекстуалізовані дані для прийняття рішень. ІСПР надають можливість аналізувати великі обсяги даних та прогнозувати майбутні події, що дозволяє компаніям зробити більш обґрунтовані та ефективні рішення.

Останньою, але не менш важливою, тенденцією є підвищення гнучкості та адаптивності ІСПР до змін у бізнес-середовищі. Сучасні ІСПР стають більш гнучкими та адаптивними, що дозволяє компаніям швидко

реагувати на змінні умови та вимоги клієнтів, забезпечуючи їм конкурентні переваги на ринку.

Таблиця 2.3 – Аналіз тенденцій ІСПР

Тенденція	Опис	Приклади
1	2	3
Штучний інтелект (ШІ)	Використання алгоритмів ШІ для автоматизації завдань, таких як прогнозування попиту, оптимізація маршрутів та розподіл ресурсів.	- Система прогнозування попиту на гуманітарну допомогу на основі даних про попередні стихійні лиха та поточні умови. - Система оптимізації маршрутів для доставки гуманітарної допомоги до постраждалих районів. - Система розподілу ресурсів для забезпечення того, щоб допомога дісталася людям, які найбільше її потребують.
Машинне навчання (ML)	Навчання ІСПР на основі історичних даних та даних в режимі реального часу для покращення їхньої точності та ефективності.	- Система ML, яка навчається на даних про минулі гуманітарні операції для покращення прогнозування потреб та планування логістики. - Система ML, яка використовує дані датчиків та соціальних мереж для відстеження переміщення людей та потреб в гуманітарній допомозі в режимі реального часу.

Продовження таблиці 2.3

1	2	3
Інтернет речей (IoT)	Підключення датчиків та інших пристроїв до ІСПР для збору даних в режимі реального часу про умови зберігання та транспортування гуманітарної допомоги.	- Датчики температури та вологості для моніторингу умов зберігання продуктів харчування та медикаментів. - Датчики GPS для відстеження руху вантажів та запобігання крадіжкам.
Хмарні обчислення	Розгортання ІСПР в хмарі для покращення їхньої масштабованості, доступності та безпеки.	- Хмарна платформа для управління запасами гуманітарної допомоги та координації логістичних операцій. - Мобільний додаток для гуманітарних працівників, який дає їм доступ до ІСПР в режимі реального часу.
Великі дані	Аналіз великих обсягів даних про гуманітарні кризи для виявлення нових тенденцій та можливостей.	- Аналіз даних соціальних мереж для виявлення районів, які потребують допомоги. - Використання даних про мобільні телефони для відстеження переміщення людей та потреб в гуманітарній допомозі.

Загалом, інтелектуальні системи підтримки прийняття рішень у логістиці відіграють важливу роль у покращенні ефективності та результативності логістичних операцій [18]. Їхній розвиток спрямований на вдосконалення аналітичних та прогностичних можливостей, автоматизацію рутинних процесів, інтеграцію з сучасними технологіями та підвищення гнучкості та адаптивності до змін у середовищі. Інтелектуальні системи є

ключовим інструментом для досягнення конкурентних переваг у сучасній логістичній індустрії та дозволяють компаніям ефективно відповідати на виклики та можливості глобального ринку.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ПРОЕКТУ

У цьому розділі розглянуто процес створення дизайну та розробки функціональності програмного забезпечення. Детально проаналізовано та вибрано відповідні мови програмування і середовища розробки. Описано процес налаштування середовища розробки, включаючи встановлення необхідних інструментів і компонентів. Також проведено глибокий аналіз JavaScript-бібліотек, які можуть бути корисними для реалізації проекту.

3.1 Створення дизайну та розробка функціональності

Модель системи підтримки прийняття рішень детально описана в статті [15]. Формула для розрахунку пріоритету наступна:

$$K_i = \frac{0{,}4 \times I_i + 0{,}3 \times T_i + 0{,}2 \times A_i + 0{,}1 \times E_i}{S_i + D_i} \quad (3.1)$$

де:

I – Важливість (Importance);

T - Терміни (Terms);

E - Ефективність (Efficiency);

A - Доступність (Availability);

S - Специфікації (Specifications);

D - Відстань (Distance).

Проектування додатку було вирішено робити в Figma. Для початку потрібно було визначитися які саме потреби повинен покривати додаток. Було визначено, що можливості додатка мають бути підходящими для демонстрації роботи системи підтримки рішень іншими словами це має бути Proof of concept, а отже повинен включати наступні особливості:

- можливість реєстрації, логіну, створення акаунта;

- акаунти повинні бути розділені за ролями;
- можливість створення запиту на допомогу;
- можливість оцінки запиту за критеріями наведеними вище;
- можливість зміни статусу користувача або волонтера адміністрацією додатка;
- можливість додавати волонтерів.

Згідно до цих умов ролі мають бути наступними: super admin, admin, volunteer, volunteer responsible for delivery, user. Система ієрархічна тож той хто знаходиться найвище може робити все що роблять ті хто нижче. Розглянемо можливості кожного з них більш детально (рисунок 3.1).

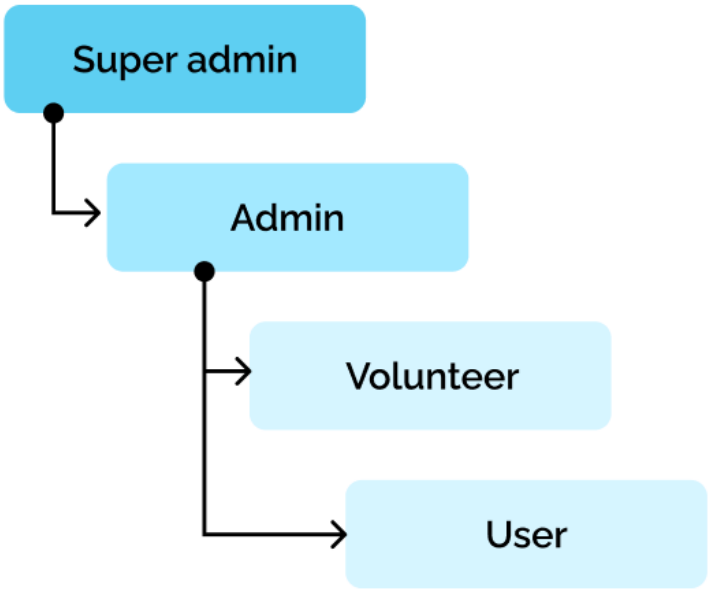

Рисунок 3.1 – Ієрархія ролей

Super admin:

- може створювати/видаляти волонтерів;
- може контролювати все, що відбувається в додатку.

Admin:

- проводить верифікацію волонтерів;

- менеджмент акаунтів волонтерів;
- фільтрація та оцінка вхідних запитів на допомогу;
- делегування запитів на допомогу відповідальному волонтеру (видання замовлення на допомогу).

Volunteer responsible for delivery:

- може додавати помічників-волонтерів;
- управляє статусом замовлення.

Volunteer – переглядає інформацію про замовлення.

User:

- може створювати запити на допомогу;
- може подавати заявку на становлення волонтером;
- бачить історію своїх запитів.

Далі розглянемо прототипи сторінок додатку. Першими будуть екрани логіну та реєстрації (рисунок 3.2-3.3).

Рисунок 3.2 – Екран логіну

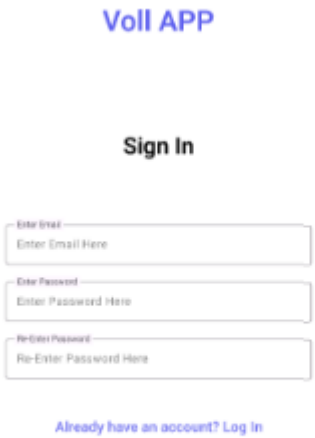

Рисунок 3.3 – Екран реєстрації

Далі створено екрани кожної ролі та ті графічні елементи до яких у них повинен бути доступ (рисунок 3.4-3.7).

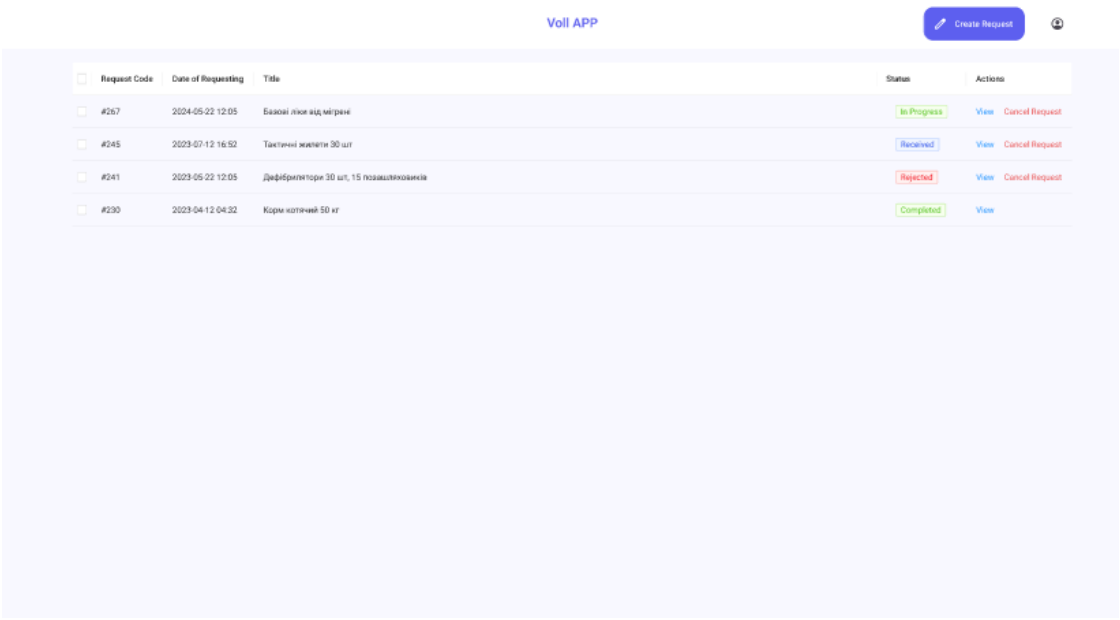

Рисунок 3.4 – Екран користувача

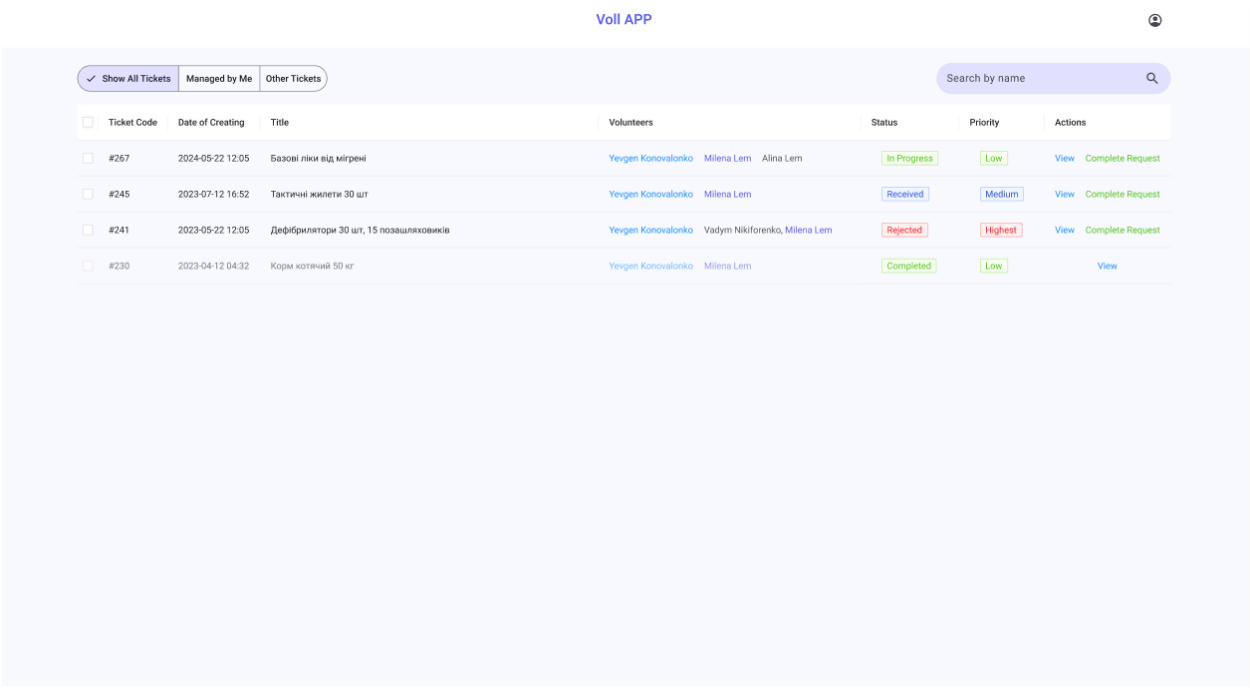

Рисунок 3.5 – Екран волонтера

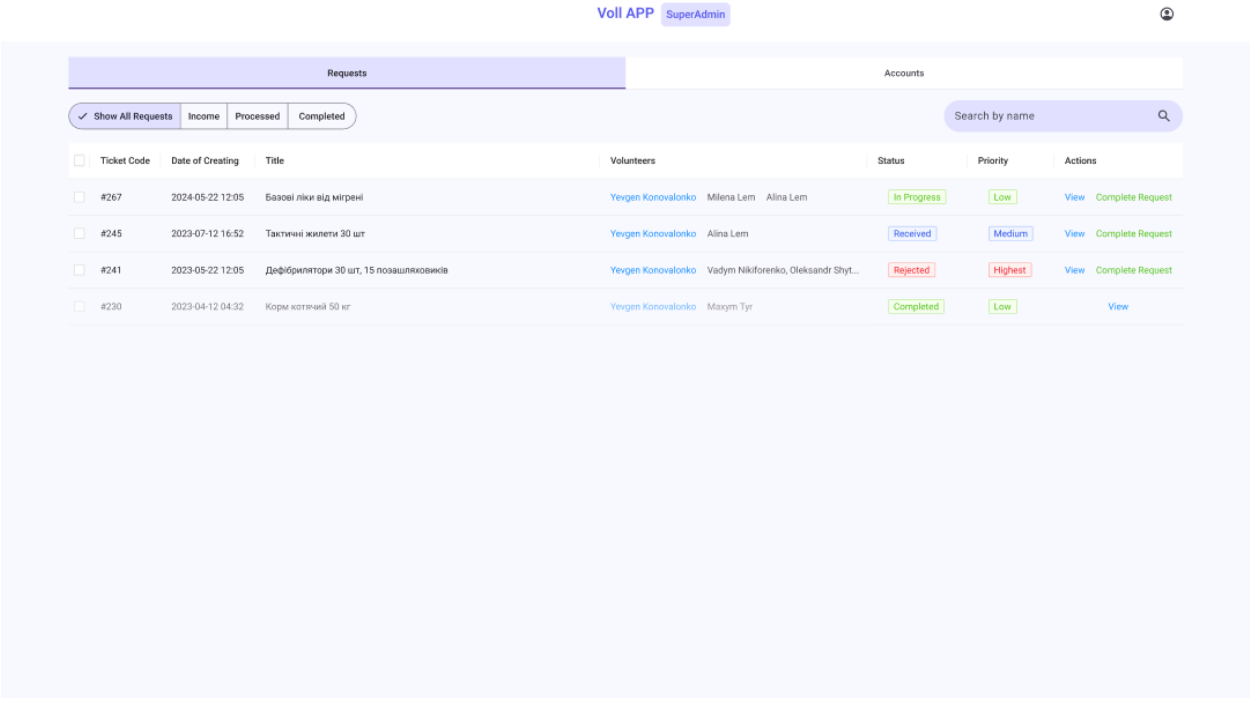

Рисунок 3.6 – Екран адміністратора з запитами

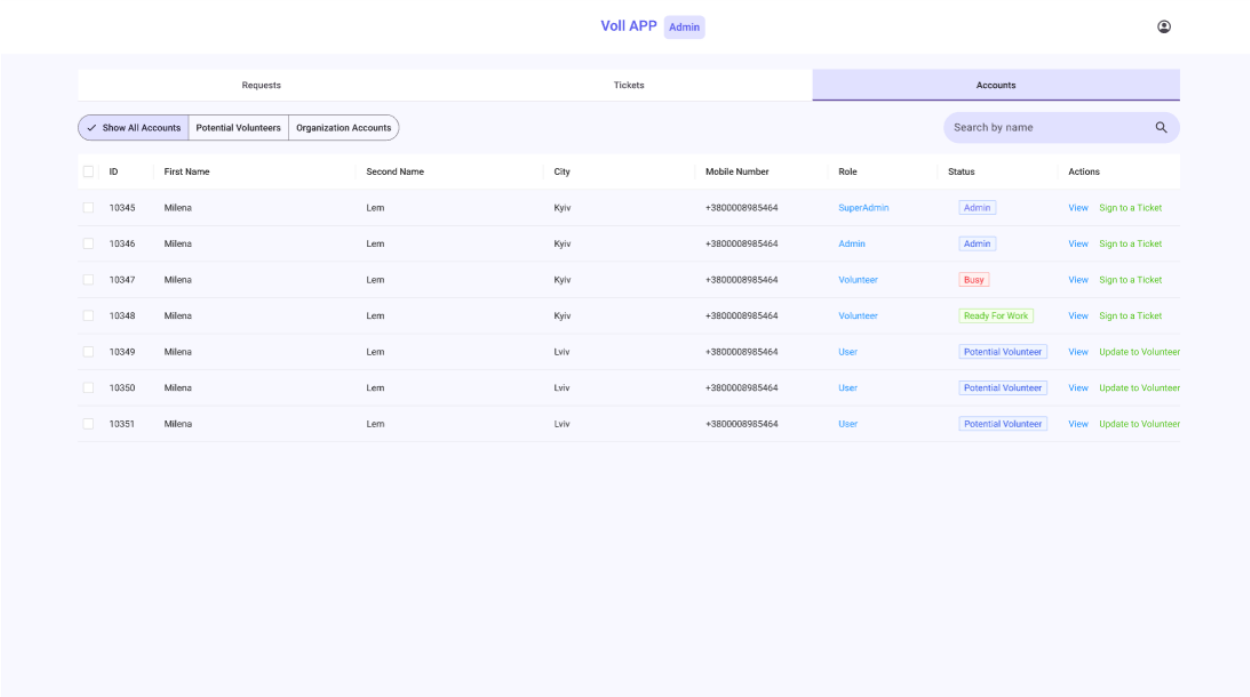

Рисунок 3.7 – Екран адміністратора з користувачами

3.2 Аналіз та вибір мов програмування

Волонтерська платформа повинна охоплювати максимальне число пристроїв, так як потенційні користувачі можуть мати дуже різне програмне забезпечення. За останніми даними Android використовується на 40,66% всіх гаджетів. Далі йдуть Windows (32,3% пристроїв) та iOS (15,97%). Значною популярністю користуються OS X (6,82% пристроїв, ОС використовується для MacBook). На інші операційні системи припадає лише 4,25% [1]. Для вибору мови програмування необхідно визначитися з програмним забезпеченням, на якому буде запускатися волонтерська платформа (рисунок 3.8). Оскільки ми прагнемо досягти максимально широкого охоплення пристроїв, оптимальним рішенням буде розробка платформи для веб-браузера. Це дозволить запускати платформу як локально на пристроях з будь-якою популярною операційною системою, так і розміщувати її на сервері для віддаленого доступу. Такий підхід забезпечить гнучкість і доступність платформи для користувачів, незалежно від їхніх технічних умов.

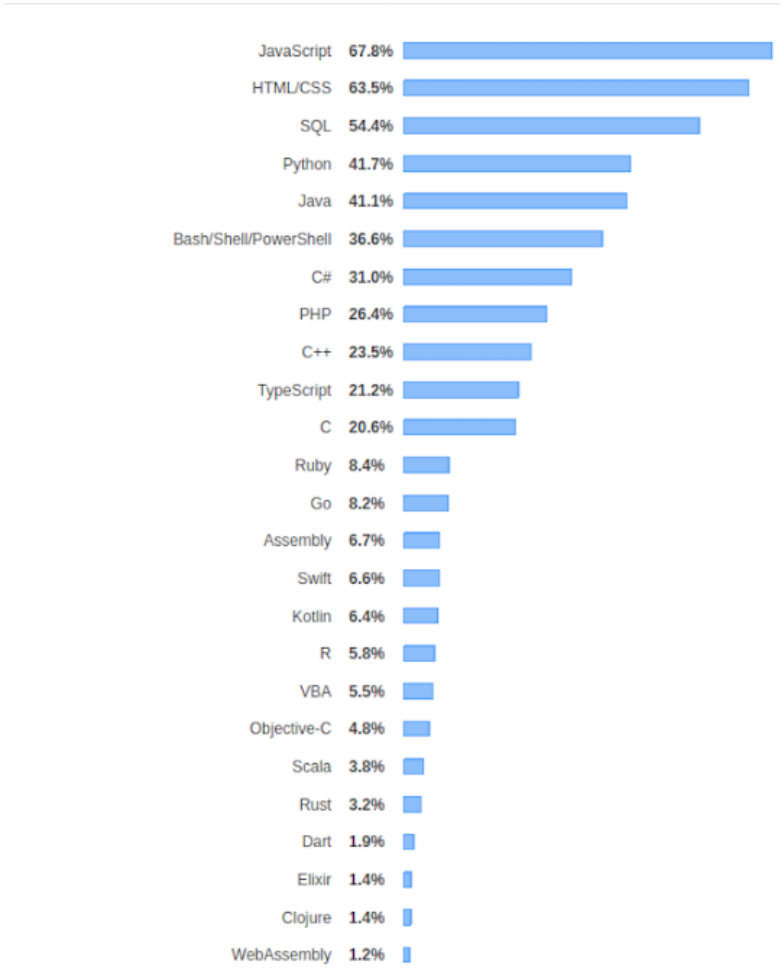

Рисунок 3.8 – Популярність мов програмування

Розглянемо більш детально декілька мов програмування:

HTML – це мова гіпертекстової розмітки утворена від англійської абревіатури «Hyper Text Markup Language».HTML – мова розмітки тексту, але браузерами та багатьма іншими програмами він інтерпретується як декларативна мова програмування, що містить інструкції, як обробляти розмічений текст. Такі інтерпретатори використовують семантику та синтаксис розмітки для перетворення на свій власний набір виконуваних інструкцій зіставляючи семантики HTML семантику своєї предметної області. HTML використовується для створення веб-сторінок, але має обмеження, коли йдеться про повністю адаптивні компоненти. Тому HTML слід використовувати лише для додавання текстових елементів і

структурування їх на сторінці. Для більш складних функцій HTML можна поєднувати з каскадними таблицями стилів (CSS) і JavaScript (JS).

JavaScript – це сценарна мова або мова програмування, яка дозволяє впроваджувати складні функції на веб-сторінках. Коли JavaScript був розроблений у 1995 році, він мав бути легкою мовою, яка використовувалася для того, щоб зробити веб-сторінки більш динамічними. Він додав прості ефекти, як показ і приховування елементів одним натисканням кнопки, але незабаром розробники зрозуміли, що він може зробити набагато більше.

Згодом JavaScript став однією з небагатьох мов, які можна було використовувати майже в кожному веб-браузері. Сьогодні більшість цих інших мов відійшли на другий план, залишивши JavaScript єдиною мовою програмування, яку можна використовувати в усіх популярних браузерах.

Наступний великий прорив для JavaScript стався, коли розробники почали експериментувати з ним поза межами веб-браузерів. Вони застосували його на бекенді веб-серверів, де він обробляв дані, які використовувалися на фронтенді. Після зростання популярності смартфонів JavaScript знайшов застосування у багатьох інструментах для мобільної розробки, які створюють додатки для платформ Android та iOS.

CSS – можна назвати умовною мовою програмування, що відповідає за зовнішній вигляд веб-сторінок, написаного за допомогою мови розмітки (HTML, XHTML, SVG, XUL). Стандарт CSS3 підтримуючий усіма сучасними браузерами здатний втілити найсміливіші фантазії дизайнера, створювати приголомшливі анімаційні ефекти, при цьому на відміну від JavaScript набагато продуктивніше.[16]

TypeScript – це статично типізована мова JavaScript, яка будується на основі існуючого синтаксису та функцій JavaScript. Це означає, що ви можете використовувати JavaScript у своєму коді TypeScript, але ви не можете використовувати TypeScript у своєму коді JavaScript, оскільки код, написаний на TypeScript, використовує функції та синтаксис, яких немає в JavaScript. TypeScript має бути скомпільований у JavaScript, щоб працювати у веб-

браузерах і середовищах, таких як Node.js. Після компіляції коду TypeScript у JavaScript отриманий код JavaScript не можна редагувати безпосередньо. Робочий процес для створення програм TypeScript полягає в тому, щоб написати їх у TypeScript, компілювати їх у JavaScript, а потім розгорнути. Хоча цей додатковий крок може здатися непотрібним додатковим ускладненням, він існує з дуже вагомої причини: TypeScript не прижився б, якби він був абсолютно новою мовою, але оскільки він компілюється в JavaScript і може працювати в існуючих системах, він був широко прийнятий.

Typescript – це мова програмування з відкритим кодом, яка додає до Javascript шар анотацій типів. Typescript також надає такі функції, як статичний тип, інтерфейси, класи, модулі та багато іншого. Це також допомагає (у більшості випадків) спростити життя розробників, забезпечуючи кращу перевірку помилок порівняно з Javascript, полегшуючи налагодження та підтримку коду.

Для розробки волонтерської платформи було прийнято рішення використати технології, засновані на мові програмування TypeScript, гіпертекстовій мові розмітки HTML та каскадних таблицях стилів CSS, оскільки ці технології є універсальними та широко розповсюдженими. TypeScript надає кілька вбудованих типів, відомих як примітивні типи, для роботи з різними типами даних. Використовуючи їх, ми можемо визначати змінні, параметри функцій, значення, що повертаються, та інші елементи коду.

3.3 Аналіз та налаштування середовища розробки

Для створення навчальної платформи також потрібно обрати середовище розробки. На сьогоднішньому ринку представлено багато середовищ розробки, як платних, так і безкоштовних

Visual Studio Code – редактор вихідного коду, розроблений Microsoft

для Windows, Linux та macOS. Позиціонується як «легкий» редактор коду для кросплатформної розробки веб- та хмарних програм. Включає в себе відладчик, інструменти для роботи з Git, підсвічення синтаксису, IntelliSense та засоби для рефакторингу. Має широкі можливості для кастомізації: теми користувача, поєднання клавіш і файли конфігурації. Розповсюджується безкоштовно, розробляється як програмне забезпечення з відкритим вихідним кодом, але готові зборки розповсюджуються ліцензією. Багато можливостей Visual Studio Code недоступні через графічний інтерфейс, часто вони використовуються через палітру команд або JSON-файли (наприклад, налаштування користувача). VS Code також дозволяє замінювати кодову сторінку за збереженням документа, символи перекладу рядка та мову програмування поточного документа. За допомогою вбудованого в продукт інтерфейсу користувача можна завантажити і встановити кілька тисяч розширень тільки в категорії «Мови програмування».[17]

WebStorm – один із безлічі редакторів коду для професійної розробки. Використовується переважно фронтенд-розробниками, яким потрібно більше працювати з JavaScript або Python, ніж зі стандартними CSS та HTML. Хоча з ними редактор коду теж непогано справляється. Відмінною особливістю є формат "все в коробці". Це означає, що розробнику не потрібно встановлювати жодних доповнень – редактор коду відмінно працюватиме з усіма технологіями після встановлення.

Перша і найпотужніша конкурентна перевага цього редактора коду – це відсутність необхідності встановлення якихось доповнень, щоб повноцінно працювати з проектами. За замовчуванням сюди вбудовані всі основні плагіни, фреймворки, аналізатор, зручне середовище для тестування коду, відладчик. Докладніше з функціоналом можна ознайомитись на офіційному сайті розробників [18].

Для розробки платформи було обрано середовище розробки WebStorm, оскільки воно має всі необхідні функції для роботи з Vue 3 та TypeScript. Щоб встановити WebStorm, слід перейти на офіційний сайт програми,

завантажити та інсталювати потрібну версію. Після цього середовище буде готове до використання, а також можна додатково встановити бажані плагіни.

3.4 Бібліотеки та інструменти для розробки платформи

Для оптимізації процесу розробки важливо також ознайомитися з різними бібліотеками та інструментами JavaScript. Давайте детальніше розглянемо деякі з них.

Vue 3 – це остання основна версія Vue.js, прогрессивного фреймворку JavaScript, який використовується для створення інтерфейсів користувача. Його розроблено для поступового впровадження, зосереджуючись на декларативному рендерингу та складі компонентів. Vue 3 приносить значні покращення продуктивності, нові функції та більш гнучку архітектуру порівняно зі своїм попередником, Vue 2.

Огляд функцій:

- composition API: Пропонує більш гнучкий і потужний спосіб компонувати логіку компонентів, полегшуючи повторне використання коду в різних компонентах;

- телепорт: нова функція, яка дозволяє розробникам відтворювати шаблон компонента в іншій частині DOM;

- ірагменти: підтримує кілька кореневих вузлів у компоненті, що забезпечує більш гнучкі структури шаблонів;

- реагування на основі проксі: замінює стару систему реагування на більш ефективний механізм на основі проксі, що забезпечує кращу продуктивність;

- покращена підтримка TypeScript: покращена інтеграція TypeScript із кращим визначенням типу та надійною підтримкою інструментів;

- паралельний рендеринг: оптимізований процес рендеринга для підвищення продуктивності, особливо в складних програмах;

- tree-Shaking: менші розміри пакетів завдяки автоматичному видаленню невикористаного коду під час процесу збирання;

- custom Renderer API: дозволяє розробникам створювати спеціальний рендерер для середовищ, що не є DOM, наприклад рідних мобільних або ігрових платформ;

- покращені прив’язки CSS: покращено обробку рішень CSS із областю видимості та CSS-in-JS;

- портал та Suspense: надає розширені функції для обробки асинхронних компонентів і відтворення вмісту в різних частинах DOM.

Ці функції роблять Vue 3 потужним і гнучким фреймворком для сучасної веб-розробки, що дозволяє розробникам створювати високопродуктивні додатки, які зручно підтримувати.

Vue-router – потрібен для створення односторінкових додатків (SPA) з Vue.js. Він дозволяє реалізувати маршрутизацію в додатку, що дозволяє переходити між різними компонентами або сторінками без перезавантаження сторінки. Це забезпечує більш швидку та плавну навігацію, покращуючи користувацький досвід. Vue Router також підтримує динамічні маршрути, вкладені маршрути та інші складні сценарії навігації.

Sass – це метамова, заснована на CSS, яка призначена для підвищення рівня абстракції CSS-коду та спрощення написання каскадних таблиць стилів. Деякі з її переваг включають:

- повна сумісність з усіма версіями CSS, що дозволяє легко використовувати будь-які доступні бібліотеки CSS;

- багатий набір функцій, який перевершує будь-яку іншу мову розширення CSS;

- більше восьми років активної розробки;

- висока популярність серед розробників, які часто вибирають Sass як основний інструмент для написання CSS. Підтримується та розвивається консорціумом високотехнологічних компаній і сотнями розробників;

- наявність численних фреймворків, побудованих на основі Sass. У

тому числі Quasar;

Quasar – це фреймворк з відкритим вихідним кодом Vue.js, ліцензований Массачусетським технологічним інститутом, який дозволяє вам, як веб-розробнику, швидко створювати адаптивні веб-сайти/додатки в багатьох варіантах:

- SPA (односторінковий додаток);
- SSR (додаток, що відтворюється на стороні сервера) (+ необов'язкове керування клієнтом PWA);
- PWA (прогресивна веб-програма);
- BEX (розширення браузера);
- мобільні програми (Android, iOS) через Cordova або Capacitor;
- багатоплатформні настільні програми (з використанням Electron).

Девіз Quasar: напишіть код один раз і одночасно розгорніть його як веб-сайт, мобільний додаток і/або додаток Electron. Так, єдина кодова база для всіх, яка допоможе вам розробити програму в рекордно короткі терміни за допомогою найсучаснішого інтерфейсу командного рядка та підкріплені найкращими практиками, неймовірно швидкі веб-компоненти Quasar. Використовуючи Quasar, вам не знадобляться додаткові важкі бібліотеки, такі як Hammer.js, Moment.js або Bootstrap. Завдяки простоті та потужності, запропонованій вам із коробки, Quasar зі своїм інтерфейсом команди наповнений повним набором функцій, які створені, щоб полегшити життя розробника. Quasar дивиться в майбутнє і встановлює власні високі та сучасні стандарти. Однак те, до чого більшість розробників звикли в певний момент часу, насправді не означає, що це найкраще рішення.

Quasar підходить для розробки додатку для волонтерів тому що:

- він заснований на Vue.js;
- ви отримуєте найсучасніший інтерфейс користувача (який відповідає рекомендаціям щодо матеріалів) для ваших веб-сайтів і програм із коробки;
- найкраща підтримка для настільних і мобільних браузерів (включно з iOS Safari) із коробки;

- він легко налаштовується (CSS) і розширюється (JS);
- це найбільш орієнтований на продуктивність фреймворк;
- він автоматично струсується деревом;
- неймовірна спільнота на нашому форумі та в чаті Discord;
- має регулярний цикл випуску, що включає нові функції;
- охоплює весь досвід розробки (включаючи навіть створення піктограм і заставок вашої програми) [19].

Vite – це локальний сервер розробки, створений Еваном Ю, творцем Vue.js, і використовується за замовчуванням у Vue та для шаблонів проектів React. Він підтримує TypeScript і JSX. Він використовує Rollup і esbuild внутрішньо для об'єднання. Він відстежує файли, коли вони редагуються, і після збереження файлу веб-браузер перезавантажує код, який редагується, за допомогою процесу під назвою «Гаряча заміна модулів» (HMR), який працює шляхом простого перезавантаження певного файлу, який змінюється за допомогою модулів ES6 (ESM). замість того, щоб перекомпілювати всю програму [20].

i18n – це фреймворк інтернаціоналізації, написаний на JavaScript і для нього. Розшифровуэться як нумеронім (i+ nternationalizatio + n), який використовується для слова інтернаціоналізація. Нумероніми також використовуються для визначення інших аспектів процесу виходу на глобальний рівень, наприклад l10n (локалізація), глобалізація (g11n), переклад (t10n).

Pinia – це бібліотека для керування станом у Vue, яка дозволяє спільно використовувати стан між компонентами та сторінками. Якщо ви знайомі з Composition API, можливо, ви думаєте, що можна вже поділитися глобальним станом за допомогою простого експорту const state = reactive({}). Це дійсно працює для односторінкових додатків, але відкриває вашу програму для вразливостей безпеки при серверному рендерингу. Але навіть у невеликих односторінкових додатках використання Pinia приносить багато переваг:

- інструменти для тестування;
- плагіни: розширюйте можливості Pinia за допомогою плагінів;
- повноцінна підтримка TypeScript або автозаповнення для користувачів JS;
- підтримка серверного рендерингу;
- підтримка Devtools;
- лінія часу для відстеження дій та мутацій;
- сховища з’являються в компонентах, де вони використовуються;
- подорож у часі та легше налагодження;
- гаряча заміна модулів;
- змінюйте свої сховища без перезавантаження сторінки;
- збереження будь-якого існуючого стану під час розробки.

3.5 Розробка платформи в WebStorm

Спочатку потрібно відкрити WebStorm і натиснути на кнопку New project, в лівому меню вибрати Vue.js та натиснути кнопку Create. В результаті буде створено базовий Vue.js проект та завантажені наступні директорії:

- node_modules – в якій зберігаються файли бібліотек;
- src – де знаходиться вихідний код, з якого потім буде компілюватися додаток;
- public – директорія в якій знаходяться статичні файли такі як логотип и тому подібне. Поки що вона нам не потрібна і ми можемо від неї позбавитись.

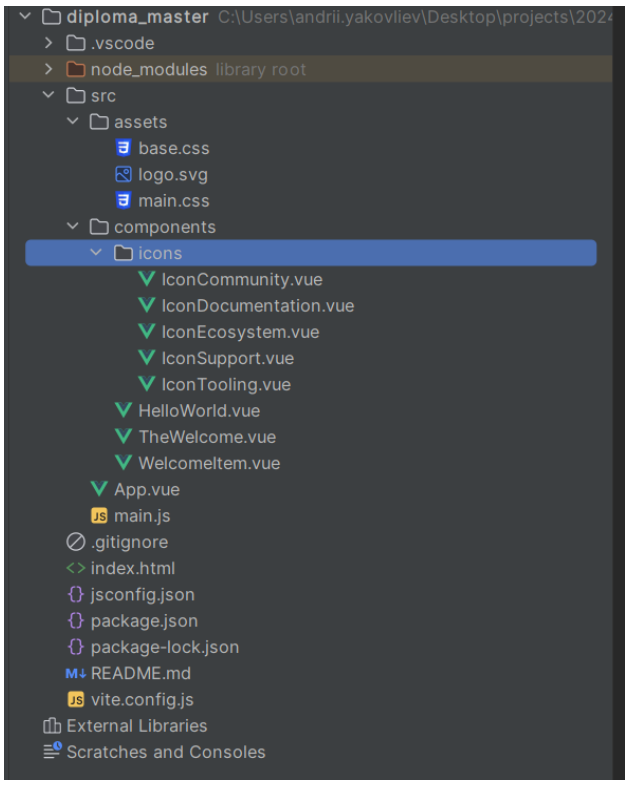

Рисунок 3.9 – Стартовий Vue додаток

Директорію public необхідно видалити. З директорії src видалити assets. З директорії components видалити все. Також необхідно стерти місця використання видалених файлів та імпорти в App.js.

Далі було встановлено допоміжні бібліотеки, а саме Qasar. Для цього необхідно відвідати офіційний сайт бібліотеки та згідно документації встановити пакети quasar та @quasar/extras за допомогою команди npm install --save. Після цього таким же чином встановити @quasar/vite-plugin та sass@^1.33.0.

Далі Qasar необхідно підключити. Для цього треба замінити код в файлі main.ts (ДОДАТОК Б.1) на:

Літинг 3.1 – Заміна коду в файлі main.ts

```
import { createApp } from 'vue'
import { Quasar } from 'quasar'
// Import icon libraries
```

```
import '@quasar/extras/material-icons/material-icons.css'
// Import Quasar css
import 'quasar/src/css/index.sass'
// Assumes your root component is App.vue
// and placed in same folder as main.js
import App from './App.vue'
const myApp = createApp(App)
myApp.use(Quasar, {
  plugins: {}, // import Quasar plugins and add here
})
// Assumes you have a <div id="app"></div> in your index.html
myApp.mount('#app')
```

Також зареєструємо Quasar в Vite для цього необхідно внести наступні зміни в файлі vite.config.js (ДОДАТОК Б.16):

Лістинг 3.2 – Реєстрація Quasar в Vite

```
import { fileURLToPath, URL } from 'node:url'
import { defineConfig } from 'vite'
import vue from '@vitejs/plugin-vue'
import { quasar, transformAssetUrls } from '@quasar/vite-plugin'
// https://vitejs.dev/config/
export default defineConfig({
  plugins: [
    vue({      template: { transformAssetUrls }    }),
    quasar({    sassVariables: 'src/quasar-variables.sass'    })
],
  resolve: {    alias: {      '@': fileURLToPath(new
URL('./src', import.meta.url))    }  }})
```

Далі потрібно створити файл за адресою src/quasar-variables.sass в якому задати кольори - це необхідно для стилізації додатку, без них Quasar не буде працювати. Виглядає це наступним чином:

Лістинг 3.3 – Файл за адресою src/quasar-variables.sass

```
$primary   : #1976D2
$secondary : #26A69A
$accent    : #9C27B0
$dark      : #1D1D1D
$positive  : #21BA45
$negative  : #C10015
$info      : #31CCEC
```

```
$warning   : #F2C037
```

Далі було встановлено Pinia за допомогою команди npm install pinia. В main.ts додано строчку const pinia = createPinia() та app.use(pinia). Після цього Vue-router за допомогою команди npm install vue-router, створено директорію router та в ній основний файл index.ts з базовою навігацією цей файл імпортовано в main.ts та додано за допомогою app.use(router).

Після цих дій ми маємо повністю готовий для розробки проект, всі необхідні бібліотеки встановлені та готові до використання.

Під час розробки до початкового дизайну були внесені зміни для покращення функціоналу. Для початку розроблені інтерфейси для даних згідно плану наведеного для оцінки замовлень. Перші інтерфейси – інтерфейси ролей та користувача, роль відповідального за доставку волонтера було вирішено виділити як окреме поле у інтерфейсі запиту:

Лістинг 3.4 – Інтерфейси ролей та користувача

```
export enum ROLES {
SUPER_ADMIN = "Super admin",    ADMIN = "Admin",    VOLUNTEER =
"Volunteer",
USER = "User"}
export interface IUser {
    id: string    name: string    email: string    phone: string
city: string   password: string
    role: ROLES}
```

Відповідно до інтерфейсів створено файл для керування користувачем:(ДОДАТОК Б.5). Опис функцій:

- login(email: string, password: string) - Знаходить користувача за його електронною адресою та паролем і входить в систему від його імені;

- logout() - Виходить із системи, видаляючи поточного користувача. Дозволяє користувачам вийти з системи, щоб захистити свої дані;

- changeRole(id: string, role: ROLES) - Змінює роль користувача на вказану. Дозволяє адміністраторам змінювати рівень доступу користувачів у

системі;

- isAdmin - Визначає, чи є поточний користувач адміністратором або супер-адміністратором;

- getUser(id: string) – Знаходить і повертає користувача за його ідентифікатором.

Цей модуль управляє даними користувачів у системі. Він дозволяє входити в систему, виходити з неї, змінювати ролі користувачів, а також переглядати інформацію про конкретних користувачів. Основні функції включають вхід користувача, вихід, зміну ролей та перевірку, чи є користувач адміністратором.

Далі інтерфейс замовлення та статуси які можуть бути надані замовленню відповідно до процесу обробки волонтерами.

Лістинг 3.5 – Інтерфейс замовлення та статуси

```
export enum STATUS {
 IN_PROGRESS = "In progress",    DONE = 'Done',    REJECTED =
"Rejected",
 PENDING = "Pending",    WAITING_FOR_ASSIGNMENT = "Waiting for
volunteer assignment"}
export interface IRequest {
 code: string    userId: string    title: string,
description: string,    date: string    status: STATUS
 priority: number    responsibleVolunteer?: IUser
assignedVolunteers: IUser[];}
```

На основі інтерфейсу створено файл який відповідає за запити(ДОДАТОК Б.4). Опис функцій:

- getRequest(code) Знаходить та повертає замовлення за його кодом. Дозволяє знайти конкретне замовлення для перегляду або редагування його деталей;

- assignResponsibleVolunteer(code, user) - Призначає відповідального волонтера для замовлення за його кодом. Якщо волонтер вже був у списку призначених, він буде видалений звідти.

- assignVolunteer(code, user) - Додає волонтера до списку призначених для замовлення. Якщо волонтер вже є відповідальним, він буде знятий з цієї позиції.

- unassign(code, userId) - Видаляє волонтера з замовлення за його кодом. Якщо цей волонтер був відповідальним, його статус буде знятий.

- setPriority(code, priority) - Встановлює пріоритет замовлення та переводить його у статус "Очікує призначення волонтера".

- setStatus(code, status) - Дозволяє оновити стан замовлення, наприклад, позначити його як "Виконується" або "Завершено".

- createOrder(title, description, userId) - Створює нове замовлення з вказаними заголовком, описом та ID користувача.

- getUserReq(userId) - Повертає всі замовлення, створені конкретним користувачем за його ID. Допомагає користувачу переглянути всі його запити на допомогу.

- getVolReq(userId) - Повертає всі замовлення, де конкретний волонтер призначений відповідальним або є в списку призначених.

Цей модуль управляє запитами на допомогу в системі. Він дозволяє створювати нові запити, призначати та знімати волонтерів, змінювати статус та пріоритет запитів, а також переглядати запити для конкретних користувачів або волонтерів. Це забезпечує ефективну організацію роботи та розподіл завдань між волонтерами.

Створено перший компонент додатку Header.vue, цей компонент відповідає за малювання верхньої частини, яка буде статична і доступна для всіх ролей і користувачів, проте з різним функціоналом(ДОДАТОК Б.7). Короткий опис компонента:

- create request: Це кнопка для створення нового запиту. Вона відображається тільки якщо користувач увійшов у систему;

- logout: Це кнопка для виходу із облікового запису. Вона також відображається тільки якщо користувач увійшов у систему;

- login: Це кнопка для входу в систему. Вона відображається, якщо

користувач не увійшов у систему.

Далі створені форми для реєстрації та логіну Auth.vue (ДОДАТОК Б.11). Компонент представляє собою декілька інпутів що роблять кнопки описано вище.

Наступний крок розробка компоненту що описує основну сторінку Home.vue (ДОДАТОК Б.8). Цей шаблон відображає різні компоненти в залежності від ролі користувача, якщо користувач увійшов у систему. Якщо користувач не увійшов у систему, відображається повідомлення "Login to continue". Ось як він працює:

- : Ця частина шаблону відображається тільки тоді, коли користувач увійшов у систему;

- <div v-if="user.role===ROLES.SUPER_ADMIN;

- user.role===ROLES.ADMIN">: Якщо роль користувача - SUPER_ADMIN або ADMIN, відображаються дві таблиці: RequestsTable та UsersTable для керування заявками та користувачами відповідно;

- : Якщо роль користувача - USER, відображається тільки RequestsTable для перегляду заявок, які створив користувач.;

- : Якщо роль користувача - VOLUNTEER, відображається RequestsTable для перегляду заявок, для яких потрібні волонтери, або заявок, які створив користувач, якщо вони є;

- <div v-else>Login to continue : Якщо користувач не увійшов у систему, відображається повідомлення "Login to continue".

Далі створена таблиця для керування користувачами UsersTable.vue (ДОДАТОК Б.12) Цей фрагмент коду визначає, які дії відображатимуться для кожної комірки таблиці в залежності від умов. Ось як це працює:

- <template #body-cell-actions="scope">: Цей фрагмент починається з оголошення шаблону для дій, які можна виконати з коміркою таблиці. #body-cell-actions вказує, що цей шаблон буде застосований до комірок з діями.

- <q-td :props="scope">: Це тег <q-td> з використанням пропсу :props, який передається об'єкту scope.

- : Ця частина відображається, якщо request не існує.

- <q-btn v-if="scope.row.role===ROLES.USER&&isAdmin">: Ця кнопка відображається, якщо роль користувача є USER і він є адміністратором (isAdmin).

- <q-btn v-else-if="scope.row.role===ROLES.VOLUNTEER&&isAdmin">: Ця кнопка відображається, якщо роль користувача є VOLUNTEER і він є адміністратором (isAdmin).

- <q-btn v-if="scope.row.role===ROLES.USER ||scope.row.role===ROLES.VOLUNTEER">: Ця кнопка відображається, якщо роль користувача є USER або VOLUNTEER.

- <div v-else>: Ця частина відображається, якщо request існує.

- : Ця частина відображається, якщо статус запиту не є IN_PROGRESS або DONE.

- <div v-else class="text-grey-6">: Ця частина відображається, якщо статус запиту є IN_PROGRESS або DONE.

Також після цього створено таблицю з запитами (ДОДАТОК Б.13).

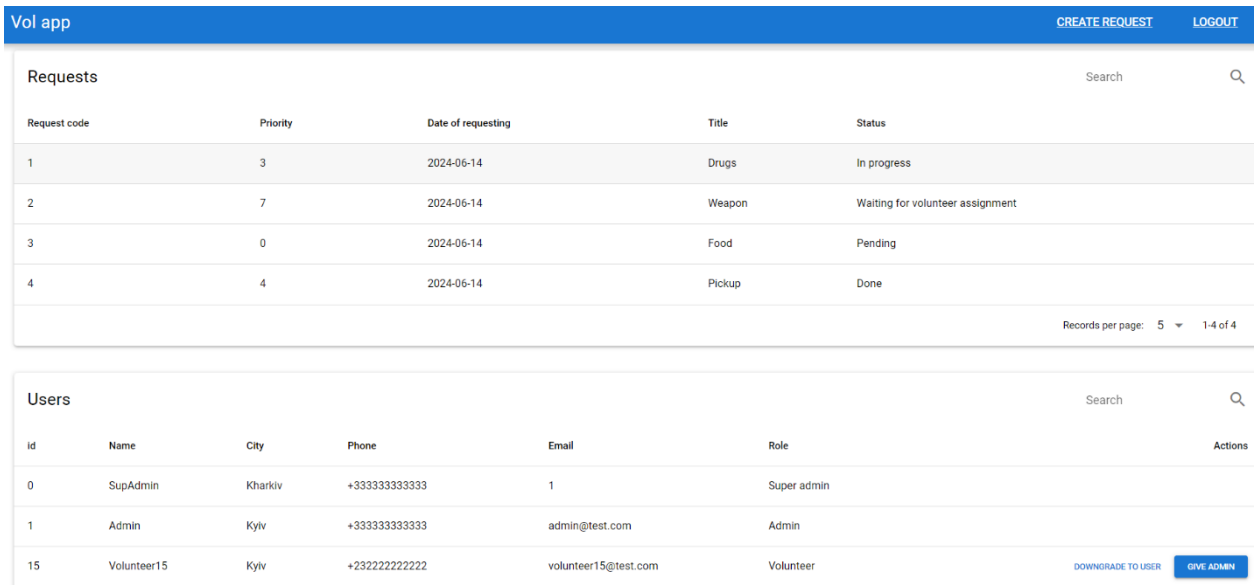

Рисунок 3.10 – Таблиці користувачів та замовлень

Та компонент що відповідає за відмальовування замовлення (ДОДАТОК Б.9). Цей шаблон відображає детальну інформацію про конкретний запит, включаючи інформацію про користувача, який створив запит, а також надає можливість виконання певних дій в залежності від ролі користувача та статусу запиту. Ось детальний опис того, що відображається в залежності від умов:

- : Цей контейнер відображається тільки тоді, коли існують дані про користувача, який створив запит (requestUser), і сам запит (request);

- <q-btn class="q-my-sm" flat @click="()=>router.back()">< Back</q-btn>: Кнопка для повернення на попередню сторінку;

- <h6 class="q-my-sm">{{ ${request.title} / #${request.code} }}</h6>: Заголовок, який відображає назву запиту та його код;

- : Відображає пріоритет запиту, якщо він встановлений (request.priority існує);

- Priority: та {{ request.priority }} : Мітка "Priority" та значення пріоритету запиту;

- : Відображає статус запиту, якщо він

встановлений (request.status існує);

- Status: та {{ request.status }} : Мітка "Status" та значення статусу запиту;

- Description: та {{ request.description }} : Мітка "Description" та опис запиту;

- <h6 class="q-my-sm">User info</h6>: Заголовок "User info";

- Інформація про користувача: Відображаються ім'я, роль, email, телефон та місто користувача, який створив запит;

- : Відображає секцію "Actions" для адміністраторів, якщо статус запиту не IN_PROGRESS, DONE або REJECTED;

- <q-btn color="primary" flat @click="openEstimate">Estimate request</q-btn>: Кнопка для оцінки запиту;

- <RateOrderDialog ="request" @close-estimate="closeEstimate" v-if="isEstimateOpened"/>: Компонент діалогового вікна для оцінки запиту, відображається, якщо isEstimateOpened має значення true;

- : Відображає можливість додавання волонтера, якщо статус запиту - WAITING_FOR_ASSIGNMENT;

- <UsersTable class="q-mt-sm" :users="availableVolunteers" :request="request"/>: Таблиця з доступними волонтерами;

- : Відображає секцію з волонтером, відповідальним за запит, якщо такий існує;

- <UsersTable :users="assignedVolunteers" :request="request"/>: Таблиця з призначеними волонтерами;

- : Кнопка для початку роботи над запитом, якщо відповідальний волонтер існує і статус

запиту не DONE, IN_PROGRESS або REJECTED;

- <q-btn v-if="(user.role===ROLES.ADMIN ||user.role===ROLES.SUPER_ADMIN)&&(request.status !== STATUS.DONE&&request.status!==STATUS.REJECTED)": Кнопка для відхилення запиту, якщо користувач є адміністратором або супер-адміністратором, і статус запиту не DONE або REJECTED;

- <q-btn v-if="(request?.responsibleVolunteer?.id===user?.id) || (user.role===ROLES.ADMIN || user.role===ROLES.SUPER_ADMIN && request.status===STATUS.IN_PROGRESS)": Кнопка для завершення запиту, якщо користувач є відповідальним волонтером або адміністратором, і статус запиту - IN_PROGRESS.

Таким чином, цей шаблон динамічно відображає інформацію про запит та надає можливість виконання різних дій в залежності від ролі користувача та статусу запиту.

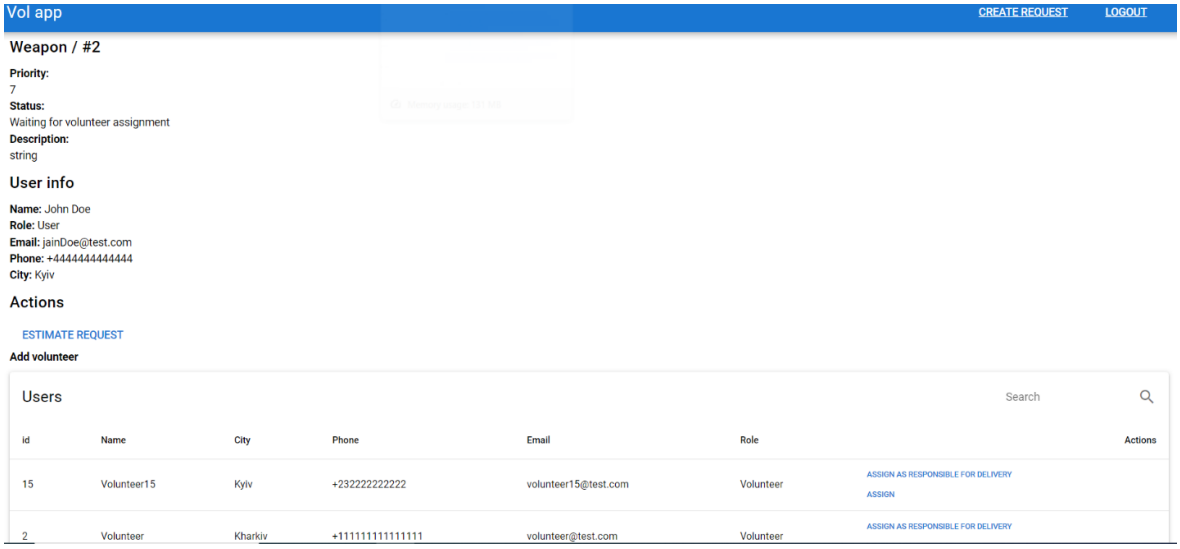

Рисунок 3.11 – Сторінка замовлення

Далі додано шаблон для додавання запиту на допомогу (ДОДАТОК Б.10). І компонент для оцінки (ДОДАТОК Б.4). На цьому розробка функціоналу завершена для MVP додатка.

ВИСНОВКИ

На підставі проведених досліджень були опрацьовані питання, що пов’язані з темою кваліфікаційної роботи.

Проаналізовано галузі застосування та види інтелектуальних систем прийняття рішень, розглянуті переваги та недоліки. Створено висновки щодо розробки інтелектуальної системи, яка допоможе обрати ефективні шляхи врегулювання більшості процесів організації гуманітарної логістики.

Описано процес створення інтелектуальної системи прийняття рішень для організації гуманітарної логістики. Розроблено додаток з імплементацією основної формули моделі.

Слід зазначити, що використання таких систем допомагає координувати дії волонтерів та оптимізувати логістичні процеси, що є ключовим у вирішенні невідкладних потреб. Алгоритм доставки волонтерською організацією, який було розглянуто, підкреслив важливість кожного кроку в організації та забезпеченні логістичної підтримки. Використання інноваційних підходів, таких як алгоритми маршрутизації та системи моніторингу, сприяє оптимізації роботи волонтерів та покращує комунікацію волонтерів, організації та споживачів. Розглянутий алгоритмів подальшому буде використаний для створення прототипу системи «Волонтерська організація», який відповідає визначеним критеріям. Загальний висновок полягає в тому, що використання систем підтримки прийняття рішень у волонтерській логістиці є необхідним елементом для забезпечення ефективності та успішності допомоги в умовах складних ситуацій. Досягнення найкращих результатів вимагає поєднання інновацій, організаційної точності та взаємодії всіх учасників процесу.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Використання системи підтримки прийняття рішень для організації гуманітарної логістики / І. В.Ільїна, В. В. Токарєв, А. В. Яковлєв, І. І. Шевченко. // 1. – 2024. – №75. – С. 88–91.

2. Гаврилова, Т. А. Бази знань інтелектуальних систем [Текст]: навчальний посібник/Т. А. Гаврилова, В. Ф. Хорошевський. - СПб. : Пітер, 2001. - 384 с. - ISBN 5-272-00071-4.

3. Галузинський, Г. П. Сучасні технологічні засоби обробки інформації [Текст]: навч. посіб. / Г. П. Галузінський, І. В. Гордієнко. - К.: КНЕУ, 1998. – 224 с.

4. Глібовець, Микола Миколайович. Штучний інтелект [Текст] : підручник/ М. М. Глібовець, О. В. Олецький. - К.: КМ Академія, 2002. - 366 с. – ISBN 966-518-153-X.

5. Дюк, В. А. Data Mining - інтелектуальний аналіз даних [Електронний ресурс]/В. А. Дюк. – Режим доступу: http://www.olap.ru/basic/dm2.asp.

6. Еддоус, М. Методи прийняття рішень [Текст]: навчальний посібник/М.Еддоус, Р. Стенфілд, І. І. Єлісєєва. - М.: Аудит: ЮНІТІ, 1997. - 590 с. - ISBN 5-85177-027-9.

7. Корнєєв, В. В. Бази даних. Інтелектуальна обробка інформації [Текст]/В. В. Корнєєв [та ін.]. - М.: Нолідж, 2000. - 352 с.

8. Пушкар, О. І. Системи підтримки прийняття рішень [Текст] : навч.посібник/О. І. Пушкар, В. М. Гіковатій, О. С. Євсєєв, Л. В. Потрашкова; ред. О. І. Пушкарь. - Харків: Інжек, 2006. - 304 с. – ISBN 966-392-066-1.

9. Черноруцький, І. Г. Методи оптимізації та прийняття рішень [Текст]: навчальний посібник / Ігор Георгійович Чорноруцький; СПбГ технічний ун-т. - СПб. : Лань, 2001. - 384 с. – (Підручники для вузів. Спеціальна література). -32 ISBN 5-8114-0387-9

10. Чіен К. Ф. і Чен Л. Ф. (2007). Використання грубої теорії множин для залучення та утримання високопотенційних талантів для виробництва напівпровідників.

11. IEEE Transactions on Semiconductor Manufacturing, 20 (4), 528-541.2. DeCenZo, D. A., & Robbins, S. P. (2005).

12. Основи управління персоналом (8-ме вид.). Нью-Йорк: John Wiley & Son.Inc. .3. Делен Д. і Пратт Д. Б. (2006). Інтегрована та інтелектуальна DSS для виробничих систем.

13. Експертна система з додатками, 30 (2), 325-336.4. Фей, Р. М., Мора-Каміно, Ф., Савадого, С., і Ніанг, А. (1998).

14. Інтелектуальна система підтримки прийняття рішень для управління системою зрошення. Доповідь, представлена на міжнародній конференції IEEE.

15. Хупер, Р. С., Гелвін, Т. П., Кілмер, Р. А., і Лібовіц, Дж. (1998). Використання експертної системи в процесі підбору персоналу. Експертні системи та програми, 14 (4), 425-432.

16. Малхотра, П., Берштейн, Ф., Фішер, Дж., МакКемміш, С., Андерсон, Дж., і Манасевич, Р. (2003). Он-лайн портал знань про рак Бреста: інтелектуальна система підтримки прийняття рішень, перспектива. Доповідь, представлена на 14-й Австралазійській конференції з інформаційних систем, Перт. 7. Негневицький, М. (2005).

17. Штучний інтелект: Посібник з інтелектуальних систем: Аддісон Веслі, Англія. 8. Цянь, З., Хуанг, Г. Х., і Чан, К. В. (2004). Розробка інтелектуальної системи підтримки прийняття рішень для контролю забруднення повітря на вугільних електростанціях.

18. Остропольська Є. В. ВПРОВАДЖЕННЯ ЗМІН ТА ПРИЙНЯТТЯ ЕФЕКТИВНИХ РІШЕНЬ [Електронний ресурс] / Євгенія Василівна Остропольська – Режим доступу до ресурсу: https://pdp.nacs.gov.ua/courses/vprovadzhennia-zmin-ta-pryiniattia-efektyvnykh-rishen_?course_enrollment_id=7007.

19. Запорожець Т. В. Застосування інтелектуальних технологій та систем штучного інтелекту для підтримки прийняття управлінських рішень. Механізми публічного управління. / Т. В. Запорожець., 2020.

20. Смоляк Ю. Ю. Штучний інтелект в управлінні підприємством: трансформація ролі менеджера в індустрії 4.0. / Ю. Ю. Смоляк, А. В. Холодницька.

21. Humanitarian aid [Електронний ресурс] – Режим доступу до ресурсу: https://civil-protection-humanitarian-aid.ec.europa.eu/what/humanitarian-aid_en.

22. The Humanitarian Supply Management System (SUMA) [Електронний ресурс] // The Pan American Health Organization. – 1992. – Режим доступу до ресурсу: http://www.disaster-info.net/SUMA/english/index.html.

23. Logistics Support System (LSS). [Електронний ресурс] // The Pan American Health Organization. – 2005. – Режим доступу до ресурсу: http://www.lssweb.net/.

24. The Fritz Institute. Humanitarian Logistics Software (HLS), 2003 [Електронний ресурс] – Режим доступу до ресурсу: http://www.fritzinstitute.org/prgTech-HLS.htm

25. The Fritz Institute. [Електронний ресурс]. – 2007. – Режим доступу до ресурсу: http://www.fritzinstitute.org/prgTech-HELIOS_Overview.html.

26. SAHANA system. [Електронний ресурс] // Lanka Software Foundation. – 2004. – Режим доступу до ресурсу: http://sahanafoundation.org/.

27. Lanka Software Foundation a. SAHANA deployments. 2004. [Електронний ресурс] – Режим доступу - [Електронний ресурс] – Режим доступу до ресурсу:http://sahanafoundation.org/ http://wiki.sahanafoundation.org/doku.php/deployments:start

28. HFOSS Institute. HFOSS. 2006. [Електронний ресурс] – Режим доступу до ресурсу:http://dev.hfoss.org/

29. HFOSS Institute. 2010. [Електронний ресурс] – Режим доступу до ресурсу:http://collabbit.org/about

30. HFOSS Institute . Posit. 2008 [Електронний ресурс] – Режим доступу до ресурсу: http://posit.hfoss.org/

31. The International Federation of Red Cross. The Disaster Management Information System (DMIS). 2001. [Електронний ресурс] – Режим доступу до ресурсу: https://www-secure.ifrc.org/DMISII/Pages/00_Home/login.aspx

32. Médecins Sans Frontières. LogistiX. 2006. [Електронний ресурс] – Режим доступу до ресурсу: ftp://support.geneva.msf.org/permanent/LOG/SupplyChainManagement

33. Chevin Fleet Solutions . FleetWave. 2001. [Електронний ресурс] – Режим доступу до ресурсу: http://www.chevinfleet.com/us/fleet_software_fleetwave_web_based_enterprise_fleet_management_solution.asp

34. The International Federation of Red Cross. The Disaster Management Information System (DMIS). 2001. [Електронний ресурс] – Режим доступу до ресурсу: https://www-secure.ifrc.org/DMISII/Pages/00_Home/login.aspx

35. Operating System Market Share Worldwide [Електронний ресурс] – Режим доступу до ресурсу: https://gs.statcounter.com/os-market-share/all/.

36. What is CSS? [Електронний ресурс]. – 2010. – Режим доступу до ресурсу: https://www.w3.org/standards/webdesign/htmlcss#whatcss.

37. Visual Studio Code [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Visual_Studio_Code.

38. WebStorm: обзор редактора кода и полезных фишек для веб-разработчиков и не только [Електронний ресурс] – Режим доступу до ресурсу: https://liquidhub.ru/blogs/blog/webstorm.

39. Introduction to quasar [Електронний ресурс] – Режим доступу до ресурсу: https://quasar.dev/introduction-to-quasar%5

40. Vite (software) [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Vite_(software).