Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя
(повне найменування вищого навчального закладу)
Факультет комп'ютерно-інформаційних систем і програмної інженерії
(назва факультету)
Кафедра комп'ютерних наук
(повна назва кафедри)

ПОЯСНЮВАЛЬНА ЗАПИСКА

до дипломного проекту (роботи)

магістр
(освітній ступінь (освітньо-кваліфікаційний рівень))

на тему: **Розробка веб-застосунку для візуалізації та аналізу мережі друзів за допомогою графів**

Виконав: студент (ка) 6 курсу, групи СНнм-61
спеціальності (напряму підготовки) 122
Комп'ютерні науки
(шифр і назва спеціальності (напряму підготовки))

(підпис) Радчук М. І.
(прізвище та ініціали)

Керівник (підпис) Ясній О. П.
(прізвище та ініціали)

Нормоконтроль (підпис) Мацюк О. В.
(прізвище та ініціали)

Рецензент (підпис) Цуприк Г. Б.
(прізвище та ініціали)

м. Тернопіль – 2020

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя
(повне найменування вищого навчального закладу)

Факультет комп'ютерно-інформаційних систем і програмної інженерії

Кафедра комп'ютерних наук

Освітній ступінь магістр

Напрям підготовки
(шифр і назва)

Спеціальність 122 "Комп'ютерні науки"
(шифр і назва)

ЗАТВЕРДЖУЮ

Завідувач кафедри к.т.н.,доц., Боднарчук І.О.

«__27_» травня 2020 р.

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ) СТУДЕНТУ

Радчуку Максиму Ігоровичу
(прізвище, ім'я, по батькові)

1. Тема проекту (роботи) Розробка веб-застосунку для візуалізації та аналізу мережі друзів за допомогою графів

Керівник проекту (роботи) Ясній Олег Петрович, доктор технічних наук, професор
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом по університету від «26» жовтня 2018 року № 4/7-802

2. Термін подання студентом проекту (роботи) 28.05.2020р.

3. Вихідні дані до проекту (роботи) наукові літературні джерела

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Огляд літературних джерел за тематикою магістерської роботи 2. Вибір технологій для розробки веб-застосунку для візуалізації та аналізу мережі друзів 3. Описання розробки веб-застосунку для візуалізації та аналізу мережі друзів 4. Спеціальна частина 5. Обгрунтування економічної ефективності 6.Охорона праці та безпека в надзвичайних ситуаціях. 7. Екологія. Список використаних джерел. Додатки.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
1. Тема дипломної роботи 2. Мета і завдання дослідження 3. Об'єкт та предмет дослідження 4. Актуальність теми 5. Роль соціальних мереж у сучасному житті 6. Що таке граф? 7. Соціальний граф 8. Отримання даних з мережі Facebook 9. Візуалізація графу бібліотекою plotly. 10,11. Розробка веб-додатку. 12. Порівняння можливих візуалізацій графу. 13. Висновок 14. Дякую за увагу.

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Спеціальна частина	Литвиненко Я.В., доцент	06.04.20р.	12.04.20р
Обґрунтування економічної			
ефективності	Матвійчук Л.П., доцент	13.04.20р.	19.04.20р.
Охорона праці			
Безпека в надзвичайних	Дмитроца Л.П., доцент	20.04.20р.	26.04.20р.
ситуаціях	Стадник І.Я., професор	20.04.20р.	26.04.20р.
Екологія	Лясота О.М., доцент	27.04.20р.	03.05.20р.

7. Дата видачі завдання 10 жовтня 2018р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Термін виконання етапів проекту (роботи)	Примітка
1	Затвердження теми дипломної роботи	29.10.2019р.	Виконано
2	Аналіз літературних джерел	30.10.19р.-10.11.19р.	Виконано
3	Обґрунтування актуальності дослідження	11.11.19р.-20.11.19р.	Виконано
4	Аналіз предмета дослідження та предметної області	21.11.19р.-29.11.19р.	Виконано
5	Розробка програми для отримання даних з соціальної	30.11.19р.-29.12.19р.	Виконано
	мережі Facebook		
6	Розроблення веб-додатку для візуалізації графу	13.01.20р.-17.02.20р.	Виконано
	друзів		
7	Оформлення розділу “Огляд літературних джерел за	18.02.20р.-27.02.20р.	Виконано
	тематикою магістерської роботи”		
8	Оформлення розділу “Вибір технологій для розробки	28.02.20р.-20.03.20р.	Виконано
	веб-застосунку для візуалізації та аналізу мережі друзів”		
9	Оформлення розділу “Описання розробки веб-	21.03.20р-05.04.20р.	Виконано
	застосунку для візуалізації та аналізу мережі друзів”		
10	Оформлення розділу “Спеціальна частина”	06.04.20р.-12.04.20р	Виконано
11	Оформлення розділу “Обгрунтування економічної	13.04.20р-.19.04.20р	Виконано
	ефективності”		
12	Оформлення розділу “Охорона праці та безпека в	20.04.20р.- 26.04.20р	Виконано
	надзвичайних ситуаціях”		
13	Оформлення розділу “Екологія”	27.04.20р.- 03.05.20р	Виконано
14	Перевірка дипломної роботи на плагіат	04.05.20р.-08.05.20р.	Виконано
15	Нормоконтроль	13.05.2020р.	Виконано
16	Попередній захист дипломної роботи	14.05.2020р.	Виконано
17	Захист дипломної роботи	28.05.2020р.	

Студент _____ Радчук М. І.
(підпис) (прізвище та ініціали)

Керівник проекту (роботи) _____ Ясній О. П.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Розробка веб-застосунку для візуалізації та аналізу мережі друзів за допомогою графів // Дипломна робота ОКР “Магістр” // Радчук Максим Ігорович // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп’ютерно-інформаційних систем і програмної інженерії, кафедра комп’ютерних наук, група СНнм-61 // Тернопіль, 2020 // 146 с., 32 рис., 5 табл., 2 додат., 49 джер..

Ключові слова: СОЦІАЛЬНИЙ ГРАФ, ВІЗУАЛІЗАЦІЯ, ВЕБ-ЗАСТОСУНОК

Метою даної магістерської роботи є дослідження та пошук рішень для отримання даних з найпопулярнішої соціальної мережі Facebook та розробка веб-додатку, який буде візуалізувати отримані дані. У процесі розробки потрібно обрати найбільш підходящі технології для реалізації веб-додатку. Користувач повинен мати змогу завантажити свої дані у веб-додаток та мати можливість взаємодіяти з цим графом у режимі реального часу. Також потрібно дослідити та описати можливість альтернативних представлень графа.

Об’єкт дослідження – процес візуалізації мережі друзів у соціальній мережі Facebook.

Предмет дослідження – технології візуалізації соціального графа та зчитування даних з веб-сторінки.

В першій частині проведено огляд літературних джерел за тематикою магістерської роботи. Основну увагу при цьому спрямовано на дві теми – графи та соціальні мережі.

У другій частині представлено основні технології, якими скористаємося для розробки веб-додатку. Усі технології поділено на категорії. У кожній категорії представлено найпопулярніші. Обґрунтовано вибір кожної технології.

У третій частині розглянуто отримання даних за допомогою веб-скрапінгу, імплементацію веб-додатку для візуалізації соціального графу та порівняно різні можливі представлення.

У спеціальній частині проведено огляд інтегрованих середовищ розробки та описано, для чого вони потрібні і які мають переваги. Здійснено обґрунтований вибір інтегрованого середовища.

Результатом дипломної роботи магістра є повністю функціонуюча програма для отримання даних про друзів з мережі Facebook та веб-додаток для представлення отриманих даних у вигляді інтерактивного графа з можливістю тонкої конфігурації.

ANNOTATION

Development of web-application for the network of friends visualization and analysis using graphs // Diploma work degree "Master" // Radchuk Maksym // Ternopil Ivan Pul'uj National Technical University, Department of Computer Information Systems and Software Engineering, Department of Computer Science // Ternopil, 2020 // P. 146, Fig. – 32, Tables – 5, Annexes. – 2, References – 49.

The purpose of this master's thesis is to research and find solutions to obtain data from the most popular social network Facebook and develop a web application that will visualize the obtained data. During the development process, one needs to choose the most appropriate technologies for developing a web application. The user must be able to upload its data to the web application and be able to interact with this graph in real time. Also, the possibility of alternative representations of the graph should be explored and described.

The object of the study is the process of visualizing a network of friends on the social network Facebook.

The subject of the research are technologies of visualization of the social graph and data reading from the web page.

In the first part, a review of literary sources about the master's thesis was preformed. The main attention is focused on two topics - graphs and social networks.

The second part presents the main technologies that will be used to develop a web application. All technologies are divided into categories. Each category presents the most popular ones. The choice of each technology is substantiated.

In the third part, there was covered the data obtaining using web scraping, implementing a web application to visualize a social graph, and comparing different possible representations.

In the special section, there was carried out the review of the integrated development environments and described what they are used for and what are their advantages. A reasonable choice of the integrated environment has been made.

The result of the master's thesis is a fully functional program for obtaining data about friends from the Facebook network and a web application for presenting the obtained data in the form of an interactive graph with the possibility of fine configuration.

Key words: SOCIAL GRAPH, VISUALIZATION, WEB APPLICATION

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

WebDriver – Інтерфейс програмної бібліотеки, яка дозволяє іншим програмам взаємодіяти з браузером.

JSON (англ. JavaScript Object Notation) – Текстовий формат обміну даними між комп'ютерами.

REST (англ. Representational State Transfer) – Передача репрезентативного стану.

DOM (англ. Document Object Model) – Об’єктна модель документа.

CSS (англ. Cascading Style Sheets) – Каскадні таблиці стилів.

HTML (англ. Hypertext Markup Language) – Мова розмітки гіпертексту.

SVG (англ. Scalable Vector Graphics) – Масштабована векторна графіка.

URL (англ. Uniform Resource Locator) – Уніфікований локатор ресурсів.

ЗМІСТ

ДОДАТКИ

ВСТУП

Графи широко застосовують у багатьох сферах науки і техніки, зокрема для представлення ієрархії файлів та тек, для зображення молекул, для представлення генеалогічних дерев, турнірних таблиць, для зберігання інформації у базах даних та інше. Графи є невід'ємною частиною сучасних технологій. Вони допомагають розв'язувати складні задачі, таких, як пошук оптимального шляху, так більш наглядно представляти дані.

З іншого боку, соціальні мережі тісно увійшли у наше життя. Мільйони користувачів кожен день відвідують одну з популярних соціальних мереж – Facebook, Instagram та Twitter. Зараз тяжко уявити життя без них, адже вони дозволяють швидко зв'язатися з друзями, дізнатись новини з їхнього життя, поширити необхідну інформацію чи навіть продати товар.

Оскільки граф підходить для розв'язування багатьохзадач, ним також можна зручно представляти і аналізувати соціальні мережі. Представивши мережі друзів у вигляді графа, можна побачити зв'язки між друзями і зрозуміти, наскільки тісно вони переплітаються. Витягуючи додаткові дані з профілів, можна глибше проаналізувати зв'язки та представити певну статистику.

Мета дипломної роботи магістра – дослідити та відшукати рішення для отримання даних з найпопулярнішої соціальної мережі Facebook та розробити веб-додаток, у якому візуалізуватимемо отримані дані. У процесі розробки потрібно обрати найбільш підходящі технології для розробки веб-додатку. Користувач повинен мати змогу завантажити свої дані у веб-додаток та мати можливість взаємодіяти з цим графом у режимі

реального часу. Також потрібно дослідити та описати альтернативні представлення графа.

Об’єкт дослідження – процес візуалізації мережі друзів у соціальній мережі Facebook.

Предметом дослідження – технології візуалізації соціального графа та зчитування даних з веб-сторінки.

Для досягнення мети потрібно виконати наступні завдання:

- Зібрати необхідну інформацію про веб-скрапінг мережі Facebook, види візуалізації графів у веб-застосунках та бібліотеки для розробки веб-застосунку.
- Вибрати мову програмування та бібліотеки для побудови веб-скрапера, візуалізатора графу та створення веб-застосунку.
- Розробити веб-скрапер для мережі Facebook, котрий повинен отримувати список спільних друзів у кожного з друзів користувача.
- Створити веб-застосунок для імпорту даних про граф друзів. Користувач має змогу завантажити дані у одному з популярних форматів.
- Розробити візуалізації для графа. Візуалізація має бути інтерактивною, щоб користувач міг вносити зміни і бачити їх у режимі реального часу.
- Вбудувати візуалізації для графа у веб-застосунок. Після імпорту даних користувач може миттєво побачити візуалізацію графу у веб-додатку.
- Надати різних можливості для візуалізації і визначити найбільш вдалі методи їх застосування.

Наукова новизна. Здійснено аналіз і досліджено різні види візуалізації соціального графа та порівняно їх сильні та слабкі сторони.

Зокрема, показано, для яких конкретних задач кожна з візуалізацій підходить найліпше.

Результатом дипломної роботи магістра повинна бути повністю функціонуюча програма для отримання даних про друзів з мережі Facebook та веб-додаток для представлення отриманих даних у вигляді інтерактивного графа з можливістю тонкої конфігурації.

1 ОГЛЯД ЛІТЕРАТУРНИХ ДЖЕРЕЛ ЗА ТЕМАТИКОЮ МАГІСТЕРСЬКОЇ РОБОТИ

1.1 Що таке граф?

Граф – це математичне зображення мережі та описує взаємозв'язок між лініями та точками (рис. 1.1). Граф складається з деяких точок і ліній між ними. Довжина ліній і положення точок не мають значення. Кожен об’єкт у графі називається вузлом.

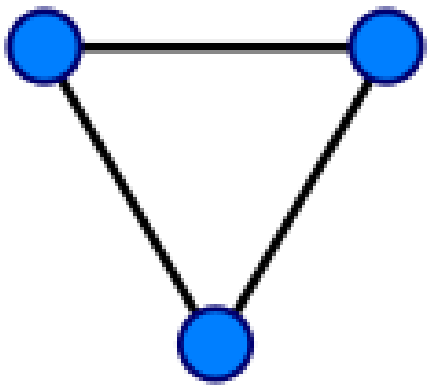


Рисунок 1.1 – Простий граф з трьома вершинами

Граф "G" – це набір вершин, які називаються вузлами "v", з'єднаними ребрами, які називаються посиланнями "e". Таким чином G = (v, e).

Вузол "v" – точка перетину графіка. Він позначає таке місце, як місто, перехрестя дороги або транспортний термінал (станції, гавані та аеропорти).

Край "e" – це зв'язок між двома вузлами. Посилання позначає переміщення між вузлами. Він має напрямок, який зазвичай представлений стрілкою. Якщо стрілка не використовується, це означає, що посилання є двонаправленою.

Географію транспорту можна визначити за допомогою графа. Більшість мереж, а саме дорожня, транзитна та залізнична мережі, визначаються більше за своїми зв’язками, ніж за вузлами. Але це не стосується всіх транспортних мереж. Наприклад, повітряні мережі визначаються більше їхніми вузлами, ніж їхніми зв’язками, оскільки зв'язки здебільшого чітко не визначені. Телекомунікаційна система також може бути представлена у вигляді мережі. Мобільні телефонні мережі або Інтернет – це найскладніший граф. Однак стільникові телефони та антени можуть бути представлені у вигляді вузлів, тоді як посилання можуть бути окремими телефонними дзвінками. Ядро Інтернету або серверів також може бути представлено у вигляді вузлів, тоді як фізична інфраструктура між ними, як волоконно-оптичні кабелі, може виконувати роль ланок. Це говорить про те, що всі транспортні мережі можуть бути певним чином представлені теорією графів.

У інформатиці графи використовуються для представлення потоку обчислень. Карти Google використовують графіки для побудови транспортних систем, де перетин двох (або більше) доріг вважається вершиною, а дорога, що з'єднує дві вершини, вважається краєм, тому їх навігаційна система базується на алгоритмі обчислення найкоротшого шлях між двома вершинами.

У Facebook користувачі вважаються вершинами, і якщо вони є друзями, то між ними є край. У алгоритмі пропозицій друзів Facebook використовується теорія графіків. Facebook – приклад ненаправленого графу.

У Всесвітній павутині веб-сторінки вважаються вершинами. Існує край від сторінки u до іншої сторінки v, якщо є посилання сторінки v на сторінці u. Це приклад графа спрямованого. Це була основна ідея алгоритму рейтингу сторінок Google.

В операційній системі ми зустрічаємо графи розподілу ресурсів, де кожен процес і ресурси вважаються вершинами. Краї відводяться від ресурсів до виділеного процесу або від процесу запиту до запитуваного ресурсу. Якщо це призведе до будь-якого формування циклу, то настане тупикова ситуація.

1.2 Соціальна мережа в інтернеті

Альтернативно називається віртуальною спільнотою або профільним сайтом, соціальна мережа – це веб-сайт, який об'єднує людей для спілкування, обміну ідеями та інтересами або створення нових друзів. Цей тип співпраці та обміну відомий як соціальні медіа. На відміну від традиційних медіа, створених не більше десяти людей, сайти соціальних медіа містять вміст, створений сотнями, а то й мільйонами різних людей. Нижче наведено невеликий перелік деяких найбільших соціальних мереж

.

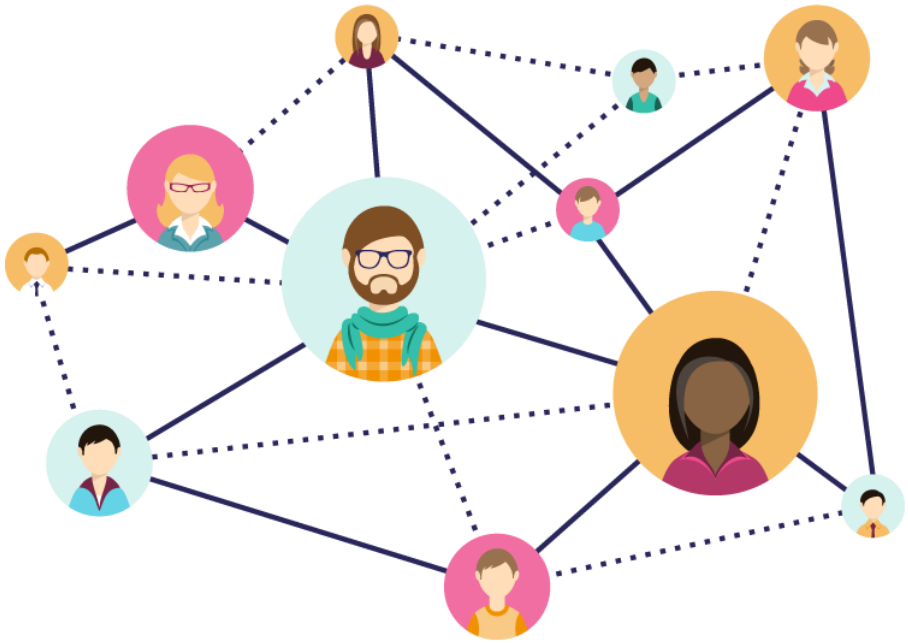


Рисунок 1.2 – Приклад зв’язків у соціальній мережі

Соціальні мережі надають змогу людям бути на зв’язку зі своїм колом друзів та родиною. Соціальні мережі – це легкий метод для того, щоб знайти те, що спільного кожен має у вашому соціальному колі. Соціальні мережі також широко використовуються для пошуку цікавих і незвичних речей в Інтернеті, адже час від часу ваші друзі та родина діляться своїми інтересами, які схожі до ваших.

Сайт соціальних мереж під назвою Bolt.com, що був створений у 1996 році Джейн Маунт та Ден Пелсон, хоч і не вважається першим реальним веб-сайтом соціальних мереж, але технічно був першим створеним. Офіційне його закриття відбулося у жовтні 2008 року.

1.3 Небезпеки соціальних мереж

Молодь особливо схильна до залежності від Інтернету: у фазі чи житті, коли соціальний контакт з однолітками відіграє головну роль у самооцінці та ідентифікації, лайки та прохання про дружбу спокушають людей проводити все більше часу перед екран.

Як і в залежності від азартних ігор, високе відчуття, коли організм вивільняє ендорфіни, може відчуватися лише за частину секунди - коли повідомлення показує, що у вас є повідомлення або друзі сподобалось ваше повідомлення. Як тільки смартфон зникає з поля зору, багато людей починають відчувати себе некомфортно і ніби їх не вистачає. Важко уявити час, коли смартфонів не існувало.

Хоча деякі люди отримують щоденну дозу щастя з Інтернету, багатьом доводиться готуватися до найгіршого, коли входять у свою соціальну мережу: вони стають жертвами кібер-знущань або переслідування. Студенти, яких вибирають на уроці, часто виявляють, що

ця необдумана поведінка переливається в цифровий світ. Це може включати погрози насильством, наклеп або навіть витоку особистих образів. Жертвам сталкерів часто стикаються із загрозливими повідомленнями. Завантажуючи фотографії, які кожен може побачити, користувачі значно полегшують потенційним сталкерам дістати патрони.

Тому батьки повинні поговорити зі своїми дітьми про небезпеку соціальних медіа, перш ніж дозволяти їм створити обліковий запис. Особливо важливо зосередитись на важливості налаштувань конфіденційності. Чим менше особистих даних, які є загальнодоступними, тим краще. TrendMicro проаналізував різні джерела, які показали, що винуватці в основному використовують інформацію про школу жертви (за даними 61% користувачів), рідне місто (48%) або плани відпустки (26%), щоб переслідувати або погрожувати.

Якщо ви орієнтуєтесь в Інтернеті, ви залишите сліди. Кожен, хто оприлюднить свої часові шкали в Facebook і годує гіганта Силіконової долини інформацією про свій вік, улюблену музику / ігри / бренди тощо, в кінцевому підсумку залишить цифровий слід таким же великим, як у Годзілла. Ви можете це прочитати чітко в загальних положеннях та умовах: Facebook не тільки має права на всі зображення, які ви завантажуєте на платформу, але також може продавати дані загальнодоступних профілів (тобто як цифрове досьє) своїм партнерам. Однак багато користувачів не сприймають це як проблему: зрештою, чверть опитаних були раді бачити особисту рекламу на основі оцінки даних. Це значно спрощує пошук споживчих товарів.

Однак кожен повинен знати, що це може закінчитися тим, що ваші дані потраплять до рук злочинців. Крім того, користувачі рідко знають про те, як далеко їхні дані подорожують в мережі. Навіть якщо ви завантажуєте додаток, у вас часто є вибір дозволити додатку доступ до певної

інформації. Ці особисті дані - це те, що робить користувачів соціальних мереж цікавими для компаній - іноді ви можете заробляти реальні гроші, продаючи цю інформацію або принаймні пристосовуючи рекламу для користувача.

1.4 Соціальний граф

Соціальний граф – це контекстна соціограма, яка описує всіх членів, організацій, груп та інших компонентів користувачів соціальної мережі та зв'язок між ними. Соціальний графік допомагає проілюструвати та відобразити загальну структуру та взаємозв'язок членів соціальної мережі.

Facebook був першим, хто реально розповів про "Соціальний графік" як спосіб описати їхню соціальну мережу, але Бред Фіцпатрік, творець LiveJournal, а зараз інженер в Google, справді перший, хто справді розповів про практичні програми та потенціал соціальних графів. Бред визначив соціальний граф як "глобальне відображення всіх і як вони пов'язані"

По суті, соціальний граф – це відображення соціальної мережі, яка є соціальною структурою, що складається з осіб (або організацій), званих "вузлами", які пов'язані (з'єднані) одним або декількома специфічними типами взаємозалежності, такими як дружба, споріднення, фінансовий обмін, неприязнь чи стосунки переконань, знань чи престижу.

Очевидно, що контекст, МЕСпро який ми говоримо тут, – це Інтернет та спільноти соціальних мереж, у яких, схоже, беруть участь усі. Проблема з Social Graphs, як визначає Бред – не існує єдиного соціального графа (або навіть множинного, який взаємодіє), який був би всебічним і децентралізованим. Швидше за все, існує сотні розсіяних соціальних графів, більшість яких сумнівної якості.

Бред вирішив знищити бар'єри між цими соціальними мережами та зробити соціальний граф надбанням громади. Google разом з Fitzpatrick навіть запустили інтерфейс програмування додатків (API), щоб веб-розробники мали змогу використовувати дані користувачів соціальних мереж.

Якщо ви зможете зламати бар'єри між мережами, можна побачити, що всі у вашій соціальній мережі досягають в одному зручному місці - або принаймні подолати розрив між мережами. Здогадайтесь, хто хоче це зробити найбільше: Google, Bing і те, що залишилося від Yahoo. Пошукові системи мають укладені пропозиції для індексації Twitter та Facebook, і вже експериментують із включенням результатів "Соціального пошуку" від людей чи підприємств у вашій соціальній графіці до звичайних результатів пошуку.

1.4.1 Небезпеки соціального графу

Висунуто кілька питань стосовно існуючої реалізації соціальної графіки, що належить Facebook. Наприклад, на даний момент служба соціальних мереж не знає про стосунки, налагоджені між окремими особами на іншій службі. Це створює Інтернет-досвід, який є непростим, а натомість забезпечує фрагментарний досвід через відсутність відкрито доступного графіка між службами. Крім того, існуючі служби по-різному визначають відносини.

Станом на 2010 рік, соціальний граф Facebook – це найбільший набір даних у соціальних мережах у світі, і він містить найбільшу кількість визначених зв’язків між найбільшою кількістю людей серед усіх веб-сайтів, оскільки це найпоширеніший сервіс соціальних мереж у світі. Концерн зосередив увагу на тому, що соціальний граф Facebook належить компанії та не ділиться іншими службами, що надає їй головну перевагу

перед іншими службами та заважає його користувачам брати свій граф із собою до інших служб, коли вони хочуть це зробити, наприклад, коли користувач незадоволений Facebook. Google намагається запропонувати вирішити цю проблему, створивши API Social Graph, випущений у січні 2008 року, який дозволяє веб-сайтам отримувати загальнодоступну інформацію про людину для формування портативної ідентичності особи, щоб представити ідентифікацію користувача в Інтернеті. Однак це не спричинило бажане використання Google, і тому було відкликано у 2012 році. Facebook представив власний API API на конференції 2010 року. Обидві компанії монетизують зібрані набори даних за допомогою прямого маркетингу та соціальної комерції. У грудні 2016 року Microsoft придбала LinkedIn за 26,2 мільярда доларів. Метрики взаємовідносини подають характер взаємовідносин одного соціального об'єкта з іншими соціальними об'єктами.

1.4.2 Моделі

У цьому пункті наведено загальновідомі моделі графів, які потенційно можуть замінити «реальні» соціальні графи. Цей список не є повним адже існує більше можливих моделей графів для вирішення цього завдання.

Функціонально-керовані моделі (англ. Feature-driven Models) створюються для відтворення статистичних властивостей діаграми, таких як поступовий розподіл та динамічні зміни щільності діаграми.

- Модель Барабаш-Альберт – є однією з декількох запропонованих моделей, що генерують безмасштабні мережі. Вона включає два важливих загальних поняття: ріст та привілейованість;
- Модель «Палаючий ліс» (Forest Fire) – це будь-яка з ряду динамічних систем, що демонструють самоорганізовану критичність.

Модель визначається як стільниковий автомат на сітці з осередками Ld. L – довжина бічної сітки, а d – її розмірність.

Навмисно-керовані моделі (англ. Intent-driven Models) емулюють процес створення оригінального графа.

- Випадковий обхід/випадкові блукання (Random Walk) – це випадковий процес при якому ходок неодноразово рухається по краю до випадково вибраного сусіда, поки k краї не будуть перекреслені. Краї та вершини можуть відвідуватися кілька разів. Випадкова прогулянка проста, якщо наступна вершина обрана з однаковою ймовірністю з боку сусідів поточного;
- Найближчий сусід (Nearest Neighbor) – це спрямований графік, причому P є його вершинним набором і з направленим ребром від p до q, коли q є найближчим сусідом p.

Рисунок 1.3 – Моделі соціальних графів [16]

Структурно-керовані моделі (англ. Structure-driven Models) містять статистичні дані зі структури діаграми, щоб відповідний генератор міг відтворювати випадкові діаграми з однаковими структурними обмеженнями.

- Графи Кронекера (Kronecker graphs) – є конструкцією для генерації послідовності графіків з невеликого базового графіка шляхом повторення продукту Кронекера;
- dK-графи (dK-graphs) – це графи із заданим розподілом ступеня. Показано, що розподіл вузлів ступеня є визначальним властивістю для багатьох реальних характеристик графіка.

1.4.3 Аналіз соціальної мережі

Аналіз соціальних мереж – це складний процес дослідження соціальних структур завдяки використанню мереж та теорії графів. Він описує мережеві структури з точки зору вузлів (виділених дійових осіб, людей або речей розташованих в мережі) та зв'язків, країв або зв'язків (зв’язків чи взаємодій), які їх з'єднують. Приклади соціальних структур, які в більшості описуються за допомогою аналізу соціальних мереж, містять мережі соціальних медіа, поширення мемів, циркуляцію інформації, мережі дружби та знайомств, бізнес-мережі, мережі знань, складні робочі стосунки, соціальні мережі, графіки співпраці, споріднення, передачу хвороби та сексуальні стосунки. Ці мережі часто візуалізуються за допомогою соціограм, в яких вузли представлені у вигляді точок, а зв’язки представлені у вигляді рядків. Ці візуалізації забезпечують засіб якісної оцінки мереж, змінюючи візуальне зображення їх вузлів та країв, щоб відобразити цікаві атрибути.

Аналіз соціальних мереж став ключовим методом сучасної соціології. Він також здобув велику пітдртимку в галузі антропології, біології, демографії, комунікаційних досліджень, економіки, географії, історії, інформатики, організаційних досліджень, політології, охорони здоров'я, соціальної психології, досліджень з розвитку, соціолінгвістики та інформатики, і це зараз зазвичай доступний як споживчий інструмент.

Візуалізація соціальних мереж є важливою складовою для аналізу даних мережі та передачі результату аналізу. Представлено безліч методів візуалізації даних, які одержано за допомогою аналізу соціальних мереж. Багато програмного забезпечення мають модулі для візуалізації мережі. Дослідження даних здійснюється з використанням представлення вузлів та зв'язків у різних макетах та присвоєння вузлам кольорів, розмірів та інших розширених властивостей. Візуальне уявлення мереж може служити сильним способом передачі комплексної інформації, але необхідно бути пильним при розтлумачуванні властивостей вузлів та графів лише з візуальних дисплеїв, адже вони можуть неправильн представити структурні властивості, які краще відображаються за допомогою кількісного аналізу.

Графи, що підписуються, можуть бути використані для ілюстрації як добрих так і поганих стосунків між людьми. Позитивний край між двома вузлами позначає позитивні відносини, такі як дружба, союз, побачення, а негативний край між двома вузлами позначає негативні відносини (ненависть, гнів). Підписані графи в соціальній мережі можна використовувати для прогнозування подальшої еволюції графу. У підписаних соціальних мережах існує поняття "врівноважений" та "незбалансований" цикли. Збалансований цикл визначається як цикл, де добуток усіх ознак позитивний. Згідно теорії балансу, врівноважені графіки представляють групу людей, які навряд чи змінять свою думку щодо інших людей у групі. Неврівноважені графіки представляють групу людей, які мають велику ймовірність змінити свою думку людей у своїй групі. Наприклад, група з 3 осіб (A, B і C), де A і B мають позитивні стосунки, B і C мають позитивні стосунки, але C і A мають негативні стосунки - це неврівноважений цикл. Ця група, ймовірно, перетворюється на збалансований цикл, такий як той, де B має лише хороші стосунки з A, і

обидва A і B мають негативний зв’язок із C. Використовуючи концепцію врівноваженого та незбалансованого циклів, еволюція підписані графіки в соціальній мережі можна передбачити.

1.5 Граф інтересів

Граф інтересів – це цифрове зображення ваших інтересів, так само, як ваш соціальний графік є цифровим зображенням ваших соціальних зв'язків.

Граф інтересів має два основні типи вузлів (люди та інтереси) та два основні типи зв’язків між вузлами (зв’язки між людьми та інтереси). Коли у людини (Фреда) є інтерес до поняття (скажімо, С. Ф. Гіганти), тоді між Фредом і Гігантами існує зв'язок між особою та інтересом. Це може бути слабкою ланкою (Фред піклується про «Гігантів») або сильною ланкою (Фред - великий шанувальник). Крім того, якщо час є фактором, посилання Фреда на «Гігантів» може слабшати або зміцнюватися, коли його інтерес змінюється. Зв'язок між інтересами - це зв'язок між інтересами, який вказує на зв’язок між інтересами. Наприклад, «Гіганти» – команда з бейсболу вищої ліги (MLB), MLB – найпрестижніша професійна ліга бейсболу в США, а бейсбол – спорт. Відповідно, існують зв'язки за інтересами між Giants та MLB, MLB та бейсболом, та бейсболом та спортом. Інтерес-інтересні зв’язки важливі для додання контексту інтересу та для виявлення інтересу вищого рівня людини. Наприклад, якщо Фред також є в The Dodgers і The Mariners, то ви можете використати посилання за інтересами, щоб виявити, що Фред справді займається бейсболом.

Ряд компаній починає створювати та використовувати графи інтересів. Google, Facebook та Twitter мають багаті набори даних про інтереси. Google знає, які результати пошуку ви натискаєте та коли ви

з'являєтесь на сторінці, де розміщуються оголошення Google. Facebook знає, коли вам подобається/діліться веб-сторінкою, сайтом, сторінкою тощо. Twitter знає, про що ви твітте, та посилання, якими ви ділитесь у твітах. Ряд стартапів використовує наявні дані або створює власні дані для побудови графів інтересів. Наприклад, Гунч – це побудова "графа смаку" на основі того, як ви відповідаєте на питання уподобань. Тим часом Gravity викоаристовує природну обробку мови для вилучення публічних соціальних даних (тобто, що ви публічно говорите, ділитесь тощо у соціальних мережах).

Стосунки на графі інтересів асиметричні, тобто вони мають односторонній зв’язок (а не двосторонні дружні стосунки). Користувачі можуть наслідувати іншу людину або сподобатися, не вимагаючи, щоб ця людина наслідувала користувачів або сподобалась їм. Це означає, що вони організовані навколо інтересів, а не дружби. Насправді дружби описуються соціальним графом, картиною того, хто знає кого. Соціальний граф досліджений для різних цілей, включаючи рекламу, але його найкраще використання має тенденцію обмежуватись рекомендаціями щодо створення більше дружби. Можливо, ви бачили розділи "Люди, яких ви можете знати" на LinkedIn, які спонукають вас до спілкування з більшою кількістю людей. Важливо робити відмінність того, що знання реальних людей у реальному житті не вимагається графом інтересів. Ви можете сподобатися взуттям Тома, Converse та Puma, не відвідуючи їхніх роздрібних магазинів.

А оскільки взаємне слідування не потрібно, люди можуть слідувати виходячи з того, що хочуть, а не від тих, кого вони вже знають. Це означає, що наступне – це прагнення. Ви можете слідкувати за Біл Гейтсом у Twitter, тому що ви захоплюєтесь його благодійницькою роботою та хочете дізнатися більше про неї, без Білла Гейтса не потрібно вирішувати,

що вам варто відповідати натомість. Асиметричне слідування підкреслює інтереси, оскільки допомагає людям досягти своїх побажань та сподівань.

Зважаючи на характер соціальних платформ, інтереси (та граф інтересів) є загальнодоступними. Інформація про те, кого і за якими людьми слідкують, є стандартною публічною інформацією в профілях користувачів, а це означає, що вона не тільки розкривається, але і неінвазивна. Однак користувачі можуть зробити свої профілі приватними, а деякі рекламні платформи використовують це як функцію автоматичного відмови.

1.5.1 Отримання даних для графу інтересів

Існує два підходи до збору даних. Ви можете заявити, що потрібно, щоб користувач надавав інформацію або ви можете слідкувати за діяльністю користувача та з'ясовувати інтереси самостійно (тобто явні та неявні).

Явний підхід може створювати перешкоди для користувачів, і це може спричинити запитання на увазі користувачів, чому вони повинні надавати детальну інформацію про їх інтерес, тоді як у них є можливість безпосередньо шукати потрібний товар або послугу. Прикладом явного підходу може бути Pinterest.

Неявний підхід найкраще можна визначити за допомогою прикладу, тобто Amazon. Він використовує неявний підхід до збору даних. На сайті є все, історія вашої покупки, історія ваших пошукових запитів, і він навіть виконує різні відносини (тобто люди, які придбали це, теж купували це, або люди, які шукали цей предмет, також шукали його).

Незалежно від того, як ви збираєте інформацію, але пам'ятайте, що коли користувач повинен вводити свої інтереси, платформа, як правило, має тенденцію генерувати неточні профілі інтересів користувачів. Отже,

наша пропозиція - використовувати неявний підхід та збирати дані відповідно до реальних інтересів користувачів.

1.5.2 Очищення даних

У соціальних мережах кожна дія користувача розглядається як сигнал. Наприклад: якщо вам подобається зображення вашого друга, знятого на футбольному стадіоні, дивлячись гру між Chicago Bears & NY Giants, це обов'язково не означає, що вам подобається футбол. З цієї вашої дії може бути ряд результатів. Список може включати:

- Ви любите дивитися футбол;
- Вам подобаються Чиказькі ведмед;.
- Вам подобаються NY Giants;
- Вам просто сподобалася картина;
- Ви хотіли почати розмову.

Однак якщо ви завантажили зображення себе на футбольному стадіоні, спостерігаючи за грою, це, безумовно, означатиме, що вам подобається Футбол, вам подобається одна з команд, що грають у гру, або, можливо, обидві команди.

Алгоритми дуже типові, тому не хвилюйтесь, якщо ви помилилися в першій спробі. Пам'ятайте, що досконалі системи не створюються з першої спроби, вони змінюються або змінюються відповідно до вимог з часом.

1.5.3 Порівняння графу інтересів і соціального графу

Незважаючи на те, що природно дружити з спільними інтересами, соціальні кола, до яких ми належимо, набагато різноманітніші. Соціальний графік, рекомендуючи те, що нам подобається нашим друзям, і навпаки – це не єдиний спосіб дізнатися про те, що можна успішно продавати. Інший

спосіб маркетологи можуть використовувати інформаційний потік у соціальних мережах через графи інтересів.

Ми не повинні мислити графіки інтересів як заміну соціальних графіків – вони просто кращі для різних речей. Інтернет-роздрібні торговці повинні пам’ятати, що відносини між потенційним клієнтом та його друзями не завжди визначають тип товару чи пропозицію, яку він отримає. Іноді люди не вибирають нічого, що отримали їхні друзі просто тому, що вони не мають однакових інтересів. Тут відображається граф інтересів. Він показує можливі інтереси модельного клієнта незалежно від його соціального походження. Виходячи з цих інтересів, маркетологи можуть підготувати пропозицію, яка просто не могла залишитися непоміченою для такого клієнта.

Незважаючи на те, що модель соціальних графів стає сильною і її дані сьогодні користуються великим попитом, маркетологи майбутнього можуть шукати більш оптимізовані дані графіків відсотків. Незабаром ми можемо помітити менше «ваших друзів на кшталт цього» індикаторів і більше «ви можете зацікавитись» знаками, коли ми переглядаємо Інтернет, шукаючи новий фільм або нову книгу для читання.

Коротше кажучи, граф інтересів можна вважати альтернативою для соціального графа. Якщо у вашому бізнесі є або матимуть певні інструменти рекомендацій, може бути більш розумним використовувати модель графа інтересів, щоб отримати набагато більш точні результати, ніж ті, що спостерігаються під час використання соціальної моделі графіка.

1.6 Висновки до першого розділу

У розділі проведено огляд літературних джерел за тематикою дипломної роботи магістра. Основну увагу при цьому спрямовано на дві теми – графи та соціально мережі.

Спочатку розглянуто поняття графа і де його застосовують. Описано основні види графів та їх практичну цінність, а також соціальні мережі і їхній вплив на сучасне суспільство. Як відомо, соціальні мережі проникли у кожен куточок людського життя і тому дані, які зберігають у них мають надзвичайну цінність для досліджень.

Поняття соціальних мереж і графів збігаються у двох видах графів – соціальний граф та граф інтересів. Перший вид графу чудово підходить для соціального пошуку та генерації рекомендацій. Описано метрики, які при цьому застосовують та можливі проблеми. Граф інтересів підходить для вивчення захоплень користувача та створення націленої реклами. У роботі генеруватимемо соціальний граф для аналізу мережі друзів і виділення окремих груп друзів.

2 ВИБІР ТЕХНОЛОГІЙ ДЛЯ РОЗРОБКИ ВЕБ-ЗАСТОСУНКУ ДЛЯ ВІЗУАЛІЗАЦІЇ ТА АНАЛІЗУ МЕРЕЖІ ДРУЗІВ

2.1 Вибір даних для візуалізації

Для роботи даного застосунку потрібно представити певну вибірку даних, яку важко відтворити самостійно. У даному випадку мережу друзів найкраще будувати на базі даних з соціальних мереж.

Існує багато різних соціальних мереж Twitter, LinkedIn, Facebook, Instagram, Вконтакті і т.д. Усі перелічені соціальні мережі надають програмний інтерфейс для отримання даних про профіль користувача та його мережу друзів. Ці програмні інтерфейси базуються на технології REST і тільки Facebook і Instagram користуються власною розробкою – GraphQL. Інша перевага Facebook – велика кількість даних, які можна отримати. Порівняно з іншими соціальних мережами є гнучке налаштування дозволів і користувач може відкрити доступ до великої кількості особистих даних. На жаль, Instagram не надає цих переваг, оскільки з новим оновленням кількість персональних даних, яку можна отримати про користувача значно зменшилась.

2.1.1 Facebook API

Платформа Facebook – термін, яким позначають набір послуг, інструментів і продуктів, котрі надає служба соціальних мереж Facebook для сторонніх розробників для створення власних додатків і служб, які отримують доступ до даних у Facebook.

Поточна платформа Facebook з'явилася у 2010 році. Платформа пропонує набір програмних інтерфейсів та інструментів, які дозволяють розробникам інтегруватися з відкритим "соціальним графом" особистих відносин та іншими речами, такими як пісні, місця та сторінки Facebook. Програми на facebook.com, зовнішні веб-сайти та пристрої дозволяють отримати доступ до графа.

Graph API є ядром платформи Facebook, що дозволяє розробникам читати та записувати дані у Facebook [17]. Graph API надає простий, узгоджений вигляд соціального графу Facebook, котрий однозначно відображає об'єкти на графі (наприклад, людей, фотографії, події та сторінки) і зв'язки між ними (наприклад, стосунки з друзями, загальний вміст і теги фотографій).

Перевірка автентичності Facebook дозволяє програмам розробників взаємодіяти з API для користувачів Facebook, а також забезпечує механізм однозначного входу у веб-, мобільні і настільні програми.

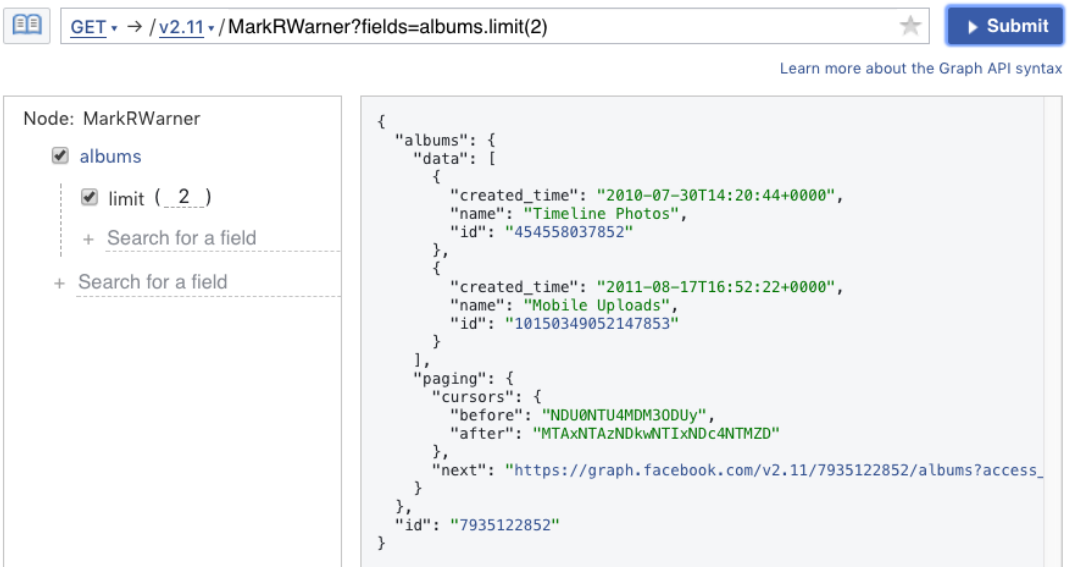


Рисунок 2.1 – Приклад запиту в Facebook Graph API

Соціальні плагіни, включно з кнопками "Подобається", "Рекомендації" та "Діяльність", дозволяють розробникам надавати своїм користувачам соціальний досвід лише за допомогою декількох рядків HTML. Всі соціальні плагіни –розширення Facebook, створені таким чином, що жодні дані користувача не надаються спільно із сайтами, на яких вони відображаються. З іншого боку, соціальні плагіни дозволяють Facebook відстежувати звички перегляду користувачами через будь-які сайти, які містять плагіни.

Facebook Connect, також званий «Вхід через Facebook», подібно до OpenID – єнабір API для аутентифікації через Facebook, яким розробники можуть користуватися для того, щоб допомогти своїм користувачам підключатися та обмінюватися з друзями Facebook таких користувачів (увімкненням та виключенням Facebook) та збільшенням участі у їхньому сайті або додатку. У такому випадку, учасники Facebook можуть входити на веб-сайти, програми, мобільні пристрої та ігрові системи третьої сторони за допомогою свого Facebook-ідентифікатора і, увійшовши в систему, можуть з'єднуватися з друзями через засоби масової інформації та публікувати інформацію та оновлювати свій профіль Facebook.

Входом через Facebook не можна користуватися у місцях, де нема доступу до Facebook, навіть якщо сайт третьої сторони звідти доступний . За даними Facebook, користувачі, які увійшли до The Huffington Post з Facebook, витратили більше часу на сайт, ніж середній користувач.

Графічний API названо на честь ідеї "соціального графа" – подання інформації на Facebook. Він складається з:

- вершини – в основному окремі об'єкти, такі як користувач, фото, сторінка або коментар;
- ребра – з'єднання між колекцією об'єктів і одним об'єктом, наприклад, фотографії на сторінці або коментарі на фото;

- поля – дані про об'єкт, наприклад, день народження користувача або ім'я сторінки.

Зазвичай, вершинии дозвовляють отримати дані про конкретний об'єкт, ребрами – ти колекції об'єктів на одному об'єкті, а поля – дані про окремий об'єкт або кожний об'єкт у колекції.

Багато розробників додатків Facebook намагаються створити вірусні програми. Стенфордський університет навіть запропонував заняття восени 2007 року під назвою Комп'ютерні науки (CS) 377W: "Створити привабливі веб-додатки, що користуються метриками та навчанням на Facebook". Численні додатки, створені класом, були дуже успішними, і вони потрапили до числа найліпших додатків Facebook.

У 2011 році The Guardian висловив стурбованість тим, що користувачі, які публікують вміст через постачальника третьої сторони, схильні втрачати свої веб-позиціонування, якщо їхня служба видалена; і відкритий графік може змусити приєднати присутність до соціальних служб Facebook навіть для людей, які використовують власні канали публікації. У червні 2018 року The New York Times розкритикувала партнерство Facebook з виробниками пристроїв, повідомляючи, що дані, доступні цим виробникам, "викликають занепокоєння щодо захисту приватного життя компанії та дотримання ухвали про дозвіл 2011 року з Федеральною комісією торгівлі".

2.1.2 GraphQL

GraphQL – відкритий вихідний код даних і мова маніпуляції для API, а також середовищем виконання у реальному часі для запитів з існуючими даними. GraphQL створено внутрішньо Facebook у 2012 році, перш ніж він з’явився публічно у 2015 році. 7 листопада 2018 року проект

GraphQL перенесено із Facebook на новостворений фонд GraphQL, організований некомерційною Linux Foundation.

GraphQL забезпечує ефективний, потужний і гнучкий підхід до розробки веб-API, порівняно із REST та іншими архітектурами веб-служб ідозволяє клієнтам визначати структуру необхідних даних, і точно таку ж структуру даних повертати з сервера, що перешкоджає надмірному поверненню великої кількості даних, однак це має наслідки для того, яким має бути ефективне кешування веб-результатів запитів. Гнучкість і багатство мови запитів також додають складності, яка може бути надлишковою для простих API.

Рисунок 2.2 – Приклад GraphQL запиту

GraphQL не прив'язано до конкретної бази даних або механізму зберігання даних, а натомість його підтримують існуючим кодом і даними.

Службу GraphQL створюють через визначення типів і полів на цих типах, а потім вона надає функції для кожного поля певного типу.

GraphQL підтримує читання, запис (мутацію) і підписку на зміни даних (оновлення у реальному часі).

Як тільки служба GraphQL виконується (як правило, на URL-адресі веб-служби), їй можна надіслати запити GraphQL для перевірки та виконання. Отриманий запит спочатку перевіряють, щоб переконатися, що він стосується тільки визначених типів і полів, а потім запускає надані функції для отримання результату.

Основними клієнтами GraphQL є клієнт Apollo і Relay. Сервери GraphQL доступні для декількох мов, включно із Haskell, JavaScript, Python, Ruby, Java, C#, Scala, Go, Elixir, Erlang, PHP, R і Clojure.

GraphQL проста у вивченні та сильно типізована мова запитів. Це дозволяє розробникам та бізнес-менеджерам візуалізувати свої схеми, користуючись відкритим редактором Visual GraphQL Editor, який дозволяє створювати асоціації між фрагментами тексту та відповідними тегами.

Методи класифікації текстів знаходяться на межі двох областей – інформаційного пошуку і машинного навчання. Їх схожість полягає в способах уявлення самих документів і способах оцінки якості алгоритмів. Заразі існує велика кількість методів і їх різних варіацій для класифікації текстів. Кожна група методів має свої переваги і недоліки, області застосування, особливості та обмеження.

Особливий інтерес має випадок, коли дані надходять у вигляді потоку, наприклад у телекомунікаційних мережах. Певні труднощі виникають через те, що навчання моделі завжди ґрунтується на сукупності властивостей набору документів. Ці сукупні властивості можуть змінюватися з плином часу, і при побудові потокового класифікатора необхідно враховувати можливі зміни вихідного розподілу даних. Бажано, щоб обраний метод міг підтримувати інкрементне навчання, тобто щоб класифікатор навчався на кожному окремо взятому зразку у режимі

реального часу. При інкрементному навчанні навчальні приклади надходять послідовно у процесі роботи алгоритму, так що класифікатор повинен постійно коригувати результати навчання і донавчатися. За неінкрементного навчання вся навчальна вибірка подається одразу повністю. Зрозуміло, що у разі такого навчання поведінка класифікатора у процесі роботи змінюється, що зменшує його передбачуваність і може ускладнити налаштування системи. У той же час інкрементне навчання му набагато гнучкіше, тобто, адаптується до умов, що змінюються.

Особливості процесу класифікації у потоці пов'язані ще з тим, що не завжди вдається контролювати швидкість надходження даних. Деякі класи документів можуть зустрічатися у потоці тільки час від часу. Виявити цей рідкісний клас буває непросто, і класифікувати тексти у таких випадках надзвичайно складно.

2.1.3 REST

REST або RESTful (Представницький стан передачі) призначений для використання переваг існуючих протоколів. Хоча REST може використовуватися майже на будь-якому протоколі, він зазвичай використовує переваги HTTP при використанні для веб-API. Це означає, що розробникам не потрібно встановлювати бібліотеки чи додаткове програмне забезпечення, щоб скористатися дизайном API REST. REST API Design був визначений доктором Роєм Філдінгом у своїй докторській дисертації 2000 року. Це примітно своїм неймовірним шаром гнучкості. Оскільки дані не прив’язані до методів та ресурсів, REST має можливість обробляти декілька типів дзвінків, повертати різні формати даних та навіть структурно змінюватись при правильній реалізації гіпермедіа, що призводить великої кількості задіяних окремих ресурсів.

Ця свобода та гнучкість, притаманна дизайну API REST, дозволяють створити API, який відповідає вашим потребам, а також задовольняє потреби дуже різноманітних клієнтів. На відміну від SOAP, REST не обмежується XML, але натомість може повертати XML, JSON, YAML або будь-який інший формат залежно від того, що запитує клієнт. І на відміну від RPC, користувачам не потрібно знати назви процедур або конкретні параметри в певному порядку.

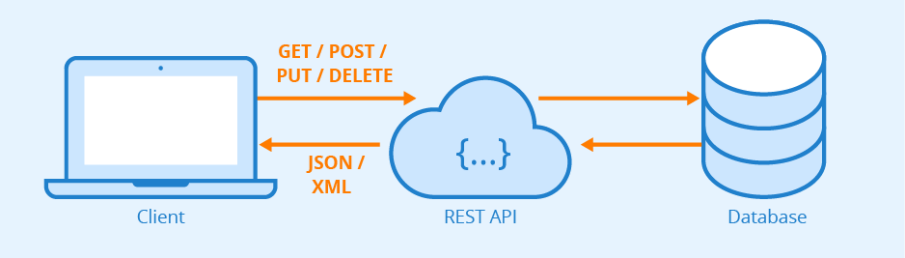


Рисунок 2.3 – Спрощене представлення REST запиту

Однак у дизайну API REST є недоліки. Ви можете втратити можливість підтримувати стан в REST, наприклад, в межах сеансів, а новішим розробникам це може бути складніше. Також важливо зрозуміти, що робить REST API RESTful, і чому ці обмеження існують перед тим, як створити ваш API. Зрештою, якщо ви не розумієте, чому щось спроектовано таким, яким він є, ви можете перешкоджати зусиллям, навіть не усвідомлюючи цього.

Хоча більшість API претендують на RESTful, вони не відповідають вимогам та обмеженням, які стверджує доктор Філдінг. Існує шість основних обмежень для дизайну API REST, про які слід пам'ятати, вирішуючи, чи це правильний тип API для вашого проекту.

Обмеження клієнт-сервер працює на концепції, що клієнт і сервер повинні бути відокремлені один від одного і дозволяти еволюціонувати

окремо і незалежно. Іншими словами, я повинен мати можливість вносити зміни до свого мобільного додатку, не впливаючи ні на структуру даних, ні на дизайн бази даних на сервері. У той же час, я повинен мати можливість змінювати базу даних або вносити зміни до свого серверного додатку, не впливаючи на мобільного клієнта. Це створює розділення проблем, дозволяючи кожній програмі зростати та масштабуватися незалежно від інших, а також дозволяти вашій організації швидко та ефективно рости.

API REST не має статусу, тобто виклики можна здійснювати незалежно один від одного, і кожен виклик містить усі дані, необхідні для успішного завершення. API REST не повинен покладатися на збереження даних на сервері або сеанси, щоб визначити, що робити з викликом, а скоріше покладатися лише на дані, що надаються в цьому виклику. Ідентифікаційна інформація не зберігається на сервері під час здійснення дзвінків. Натомість кожен виклик має в собі необхідні дані, такі як ключ API, маркер доступу, ідентифікатор користувача тощо. Це також допомагає підвищити надійність API, маючи всі дані, необхідні для здійснення дзвінка, замість того, щоб покладатися на серію викликів зі станом сервера для створення об’єкта, що може призвести до часткових збоїв. Натомість, щоб зменшити потреби в пам’яті та зберегти вашу програму максимально масштабованою, API RESTful вимагає, щоб будь-який стан зберігався на клієнті, а не на сервері.

Оскільки API без стану може збільшити накладні запити, обробляючи великі навантаження вхідних та вихідних дзвінків, API REST повинен бути розроблений для заохочення зберігання даних, що кешуються. Це означає, що коли дані кешуються, відповідь повинна вказувати на те, що дані можуть зберігатися до певного часу (закінчується), або у випадках, коли дані потребують у режимі реального часу, що відповідь не має кешуватися клієнт. Вмикаючи це критичне

обмеження, ви не тільки значно зменшите кількість взаємодій з вашим API, зменшивши використання внутрішнього сервера, але і надасте своїм користувачам API інструменти, необхідні для забезпечення найшвидших та найефективніших програм. Майте на увазі, що кешування виконується на стороні клієнта. Незважаючи на те, що ви можете мати можливість кешувати деякі дані в вашій архітектурі для виконання загальної продуктивності, ціль полягає в тому, щоб проінструктувати клієнта про те, як це робити і чи може клієнт тимчасово зберігати дані.

Ключовим фактором для роз'єднання клієнта від сервера є наявність єдиного інтерфейсу, який дозволяє незалежно розвивати додаток, не маючи сервісів, моделей або дій програми, щільно пов'язаних із самим шаром API. Уніфікований інтерфейс дозволяє клієнту розмовляти з сервером однією мовою, незалежно від архітектурного резервного середовища. Цей інтерфейс повинен забезпечувати незмінні, стандартизовані засоби комунікації між клієнтом і сервером, такі як використання HTTP з ресурсами URI, CRUD (створення, читання, оновлення, видалення) та JSON.

Багатошарова система дозволяє інкапсулювати застарілі системи та переміщувати менш поширені функціональні можливості до спільного посередника, а також захищати від них більш сучасні та поширені компоненти. Крім того, багатошарова система надає вам свободу переміщати системи в архітектуру та виходити з неї в міру розвитку технологій та послуг, збільшуючи гнучкість та довговічність, доки ви тримати різні модулі якомога вільніше. Існує велика перевага безпеки щодо використання шаруватої системи, оскільки вона дозволяє зупиняти атаки на проксі-шарі або в інших шарах, не дозволяючи їм потрапляти до фактичної архітектури сервера. Використовуючи багатошарову систему з проксі-сервісом або створивши єдину точку доступу, ви зможете тримати

критичні та вразливіші аспекти своєї архітектури за брандмауером, запобігаючи безпосередній взаємодії з ними клієнтом. Майте на увазі, що безпека не базується на єдиному рішенні "зупинити всіх", а на тому, щоб мати декілька шарів з розумінням того, що певні перевірки безпеки можуть не виконати або обійти. Таким чином, чим більше безпеки ви зможете впровадити у свою систему, тим більше шансів на те, щоб запобігти збитковій атаці.

2.2 Вибір фреймворку для розробки веб-застосунку

Вибір фреймворку для розробки веб-застосунку є одним з найважливіших рішень, тому варто детально розглянути усі можливі варіанти та порівняти їх переваги та недоліки. Зараз будь-який веб-додаток користується мовою програмування JavaScript, тому фреймворк вибрано саме на базі цієї мови.

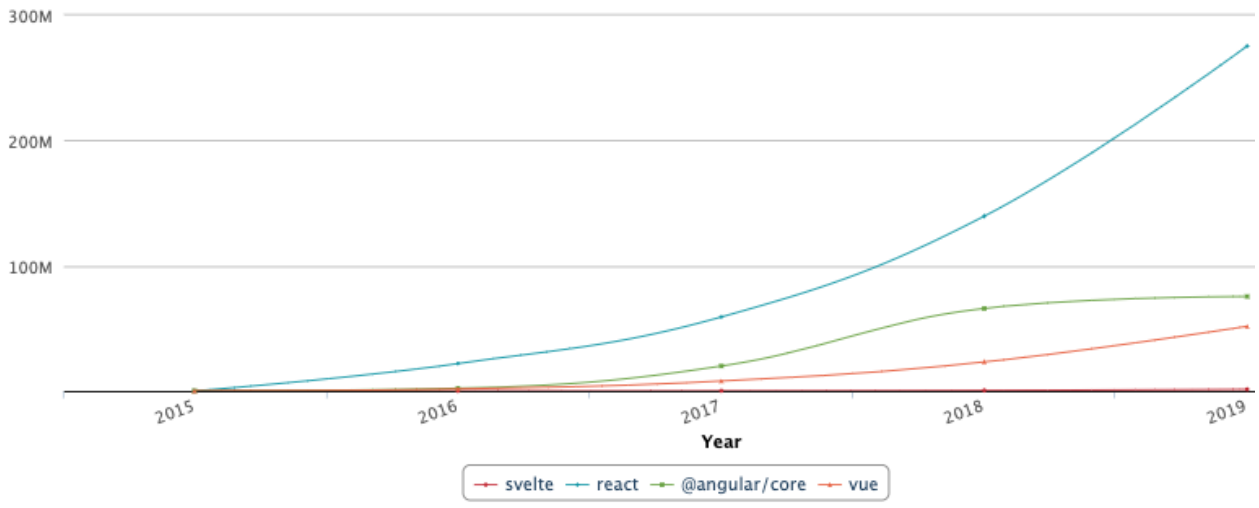


Рисунок 2.4 – Популярність JavaScript фреймворків [19]

Зараз найпопулярнішими фреймворками для веб-розробки є: React.JS, Angular, Vue.JS, що можна побачити з рисунку 2.4. Популярність фреймверку впливає на поширеність, кількість додаткових бібліотек та активну спільноту розробників. Саме тому виберемо один фреймворк із трьох найпопулярніших для розв'язування поставлених задач.

2.2.1 Angular

Angular (в основному так називають фреймворк Angular 2 або Angular 2+, а самі тільки версії після другої) — front-end фреймворк з відкритим вихідним кодом, написаний на TypeScript, який розробляють під чітким керівництвом Angular Team у компанії Google, а також широкою мережею приватних розробників та корпорацій. Angular – старий AngularJS, який переосмислили та повністю переписали незмінною командою розробників.

Як згадано вище, Angular – ретельно переписаний AngularJS.

Додано Angular CLI, що дає змогу розпочати створення нового додатка, просто написавши команду ng new.

Angular не користується концепцією "області видимості" або контролерів, натомість, головнаархітектурнаою концепціяею – ієрархію компонентів

Angular має інший синтаксис написання виразів, застосовуючи "[ ]" для біндінгу даних властивостей, і "( )" для біндінгу даних івентів.

Модульність – значну частина основного функціоналу перенесено у модулі.

Angular рекомендує та застосовує розроблену Microsoft мову – TypeScript. TypeScript — надмножина ECMAScript 6 (ES6), і є зворотно сумісною зі стандартом ECMAScript 5 (тобто JavaScript). Angular також має такі ES6-можливості, як:

- Анонімні функції
- Ітератори
- Цикли типу For/Of
- Python-подібні генератори
- Рефлексію

Ітеративні зворотні виклики реалізовано через RxJS. RxJS дещо обмежує видимість станів та можливості відлагодження, але, застосовуючи такі плагіни, як ngReact та ngrx, то легко виправити.

На рисунку 2.5 зображено схематичне представлення модуля в Angular. То базовий концепт, який є певною групою компонентів, сервісів та директив, котрі можуть функціонувати самостійно. Увесь додаток на Angular складається з таких модулів, які підвантажуються динамічно на вимогу користувача, щодозволяє розділити логіку і розподілити імплементацію функціональності між різними групами розробників.

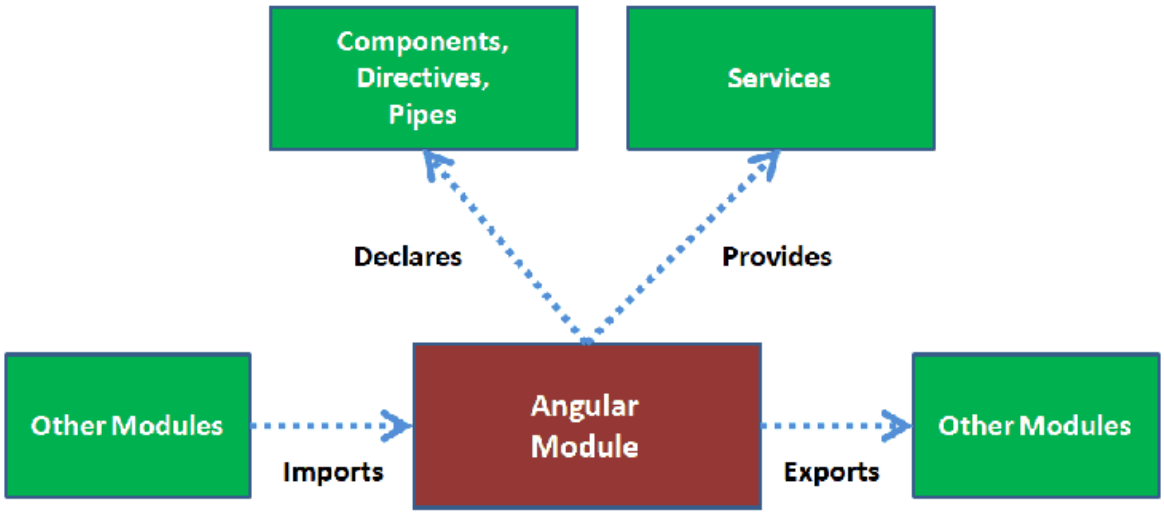


Рисунок 2.5 – Схематичне представлення модуля в Angular [48]

Версія Angular 9 з’явилася на початку 2020 року. У ній версії внесено значні зміни у роботу компілятора та рантайму для підвищення швидкодії та зменшення обсягу вихідного файлу.

2.2.2 Vue.js

Vue.js йде в аналогічному напрямку як Angular і React і стає все більш популярним. Мінімальна версія рамки має розмір всього 20 кілобайт і тому досить компактна. Це також дуже швидкою завдяки віртуальному DOM. Основна ідея полягає в тому, щоб досягти хороших результатів при найменших зусиллях, так що зазвичай лише кілька рядків коду ведуть до мети. Vue.js ідеально підходить для роботи з компонентами і вимагає відносно невеликих витрат, тим більше завдяки однофайловим компонентам, які вміщують увесь код від HTML до CSS до Javascript в одному файлі. Крім того, ви навіть можете інтегрувати Vue.js в інші рамки, такі як React, ви не зобов'язані фіксованим специфікаціям виробника.

Згідно з різними відгуками, Vue.js дуже просто в початку роботи з нею і не має особливо крутої кривої навчання. Це пов'язано з тим, що окрім Javascript, HTML та CSS, не потрібно жодної іншої мови для використання Vue.js, на відміну від Angular with Typescript або React with JSX. Крім того, інтеграція до загальних редакторів дуже хороша. Під Firefox та Chrome є навіть плагін для браузера, що робить роботу ще простішою. Спільнота також дуже активна, що не в останню чергу пов’язано з великою популярністю Vue.js.

Завдяки простому створенню компонентів, Vue.js також цікавий для більших команд і дозволяє складно структурувати комплексні завдання і одночасно залишатися достатньо єдиними, щоб користувач міг завжди добре орієнтуватися. Щоб численні компоненти не стали занадто великими, Vue.js пропонує однофайлові компоненти, в яких HTML, CSS та

Javascript поєднуються у файлі Vue, тому над цим потрібно працювати лише в одному місці. Вам не потрібно постійно відкривати кілька файлів. Потім компоненти одного файлу вже попередньо компілюються в поєднанні з webpack, що зберігає роботу браузера. Єдина проблема тут полягає в тому, що ваш редактор повинен також розпізнавати та підтримувати цю комбінацію трьох мов, щоб правильно виділити відповідний синтаксис для кожного розділу компонентів одного файлу. Але плагіни пропонують цей функціонал.

Вище згадана ефективність також дуже добре відповідає фреймворку, як і добре розроблений набір інструментів додатків та плагінів. За допомогою Vue-cli ви можете за короткий час запустити проект і навіть отримати шаблони, які можна одразу використовувати. Крім того, документація Vue.js дуже детальна та зрозуміла і надалі пропонує прямі рішення для типових завдань із документацією. І якщо ви не знайдете те, що шукаєте, ви можете здійснити пошук Awesome List, де перераховано численні ідеї, реалізовані за допомогою Vue.js.

2.2.3 React.js

React іноді помилково називається фреймворком JS, React дійсно нагадує фреймворк тим, що він глибоко інтегрований з бібліотеками Redux і Flux, які нав'язують свої власні архітектурні зразки. React – це бібліотека, яку часто порівнюють з AngularJS, популярною системою JS, і має як обмеження, так і переваги.

React є дуже гнучким і може використовуватися для проектів інтерфейсу будь-якого розміру та платформи, включаючи мобільні, веб-розробки та настільні комп’ютери. Розробники часто використовують React для малих та середніх проектів, де глобальні моделі не потрібні. Так само багато світових відомих компаній використовують React у своїх

продуктах, наприклад в Netflix, Dropbox, Khan Academy, Reddit, Imgur та багато інших.

В Infopulse ми бачимо Реагування як дуже корисну для наших потреб. Одне з наших рішень на основі React, розроблене для зовнішнього замовника, - це всеосяжна багатомодульна веб-програма. Близько шести людей зараз працюють над цим проектом. Випуск продукту планується на літо 2017 року з подальшою підтримкою та обслуговуванням.

Система оцифровує та автоматизує масивний робочий процес, який пропонує максимальну прозорість та безпеку. Застаріла система, заснована на JQuery, була нестабільною, незахищеною, погано працювала і не мала багатьох функцій. Нам не вдалося оновити або перенести застарілу систему, і довелося переробити все, використовуючи React для розробки інтерфейсу та C# для бекенда. Порівняно з звичайними структурами MVC та іншими JS, React пропонує багато переваг. Масштабованість та гнучкість програмного забезпечення. Як і Angular, React використовує компонентний метод розробки, який можна використовувати для створення масштабованих та гнучких додатків. Фреймворк інтерфейсу Material пропонує широку базу компонентів і модулів, які представляють ідеальну взаємодію з бібліотекою ReactJS.

Якщо ми детальніше подивимось на API React, то ми можемо перевіряти стан компонента щоразу. Верхні компоненти API React розроблені для функціонального підходу. Компоненти поділяються на класи, щоб їх можна було легко переносити, розширювати та масштабувати. Крім того, основні функціональні можливості React значно розширюються бібліотеками Redux або Flux. Наприклад, можна зберегти історію всієї програми в локальній пам'яті за допомогою бібліотеки Redux. API React дозволяє створювати інкапсульовані компоненти. Отже, якщо у нас є сторінка з меню та сіткою, стандартний підхід MVC рекомендує

створити окремі моделі для кожного з елементів сторінки. React описує це з моделлю для об'єднання та обробки даних через Redux.

2.3 Збереження і керування станом

Для будь-якого веб-додатку важливо мати єдине джерело правди – стан, до котрогомає бути доступ з будь-якого компоненту нашого веб-додатку. Найбільш поширеною технологією для React для керування станом є Redux. Також нам потрібно отримувати дані з GraphQL API, для цього зручно користуватися бібліотекою Apollo Client.

2.3.1 Apollo GraphQL

Платформа Apollo GraphQL – реалізація GraphQL, яка допомагає керувати даними від хмари до інтерфейсу. Ним можна користуватися поступово, і його можна накласти на існуючі служби, включаючи API REST і бази даних. Apollo містить дві бібліотеки з відкритим вихідним кодом для клієнта і сервера, на додаток до інструментарію розробника, який забезпечує все, що потрібно для найдійного запуску графічного API .

Apollo впроваджує найефективніші практики та моделі для побудови та експлуатації шару GraphQL, від інструментів та робочих процесів, які допомагають вашим командам співпрацювати у розробці, до інтеграції з уже існуючими системами розробки та інфраструктури. Це дозволяє плавно розширювати API GraphQL від однієї команди до всієї компанії. Apollo дозволяє складати окремі послуги GraphQL в єдину, уніфіковану схему, без жодних точок відмови або розгортання точок.

Apollo фіксує цінні метадані на рівні поля для кожної операції над GraphQL API і добре представляє схему GraphQL та спосіб її застосування.

Приклад застосування Apollo видно на рисунку 2.6. Інтерфейс користувача звертається до Apollo за даними. Далі Apollo вибирає, чи йому потрібно виконати REST чи GraphQL запит, або, можливо, такі дані вже закешовано і можна повернути миттєво без необхідності виконувати додатковий запит.

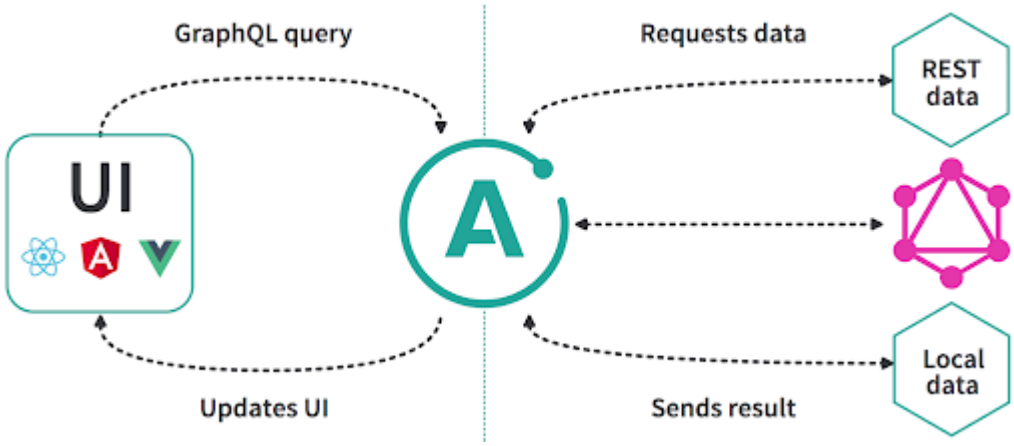


Рисунок 2.6 – Приклад застосування Apollo [47]

Інструменти Apollo застосовують для управління безпечними списками відомих клієнтів і запитів, перевірки зміни схеми щодо спостережуваного виробничого трафіка, а також налаштування гнучких робочих процесів розробки GraphQL, інтегрованих із системами керування джерелами та CI / CD.

Apollo – єдине джерело істини для API GraphQL. Центральний реєстр схеми в платформі Apollo GraphQL поширює всі зміни і деталі того, хто торкнувся ваших даних, тому багато команд можуть співпрацювати з повною видимістю і безпекою на одному графіку даних.

2.3.2 Redux

Redux – це інструмент управління станом. Хоча він використовується в основному з React, він може використовуватися з будь-якою іншим фреймворком JavaScript або бібліотекою. Він легкий у 2 КБ (включаючи залежності), тому вам не потрібно турбуватися про те, щоб збільшити розмір активів вашої програми.

За допомогою Redux стан вашої програми зберігається в магазині, і кожен компонент може отримати доступ до будь-якого стану, який йому потрібен з цього магазину. Давайте зануримось трохи глибше, щоб зрозуміти, чому може знадобитися інструмент управління станом.

Більшість бібліотек, таких як React, Angular тощо, побудовані так, щоб компоненти могли внутрішньо керувати своїм станом, не потребуючи зовнішньої бібліотеки чи інструменту. Це добре для додатків з малою кількістю компонентів, але в міру того, як програма збільшується, управління станами, що поділяються між компонентами, стає справою. Приклад роботи Redux представлено на рисунку 2.7.

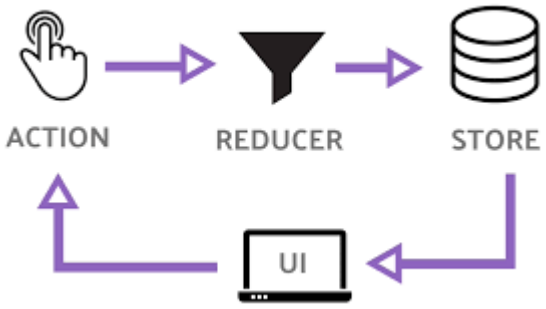


Рисунок 2.7 – Порядок дій у Redux [21]

У додатку, де дані діляться між компонентами, це може заплутати справжнє знання, де має жити стан. В ідеалі дані в компоненті повинні

жити лише в одному компоненті, тому обмін даними між компонентами братів стає важким.

Наприклад, у React, для обміну даними між братами та сестрами, держава повинна жити в батьківському компоненті. Метод оновлення цього стану забезпечується материнським компонентом і передається як реквізит цим компонентам братів.

2.4 Візуалізація графа

Після того, як вибрано Facebook API для отримання даних, Apollo для отримання даних з GraphQL API, Redux для управління станом та React бібліотеку для розробки додатку, залишився останній елемент додатку – візуалізація графа.

Для цієї задачі найбільш зручними і поширеними є JavaScript бібліотеки, які будуть граф у SVG форматі. Це дозволяє динамічно їх будувати та обирати з тисячі різних можливих імплементацій. Далі розглянемо три найпопулярніші бібліотеки для візуалізації – D3.js, Chart.js, Vis.js.

2.4.1 D3.js

D3.js (або просто D3 для документів, керованих даними) – бібліотека JavaScript для створення динамічних, інтерактивних візуалізацій даних у веб-браузерах, котра користується широко розповсюдженими стандартами SVG, HTML5 і CSS. Вона – наступник попередньої системи Protovis. На відміну від багатьох інших бібліотек, D3.js забезпечує великий контроль над кінцевим візуальним результатом. Її розробку відзначено у 2011 році, оскільки версія 2.0.0 з’явилася у серпні 2011 року.

D3.js користуються на сотнях тисяч веб-сайтів. Деякі популярні варіанти користування включають створення інтерактивної графіки для онлайн-новинних веб-сайтів, інформаційних панелей для перегляду даних і створення географічних карт. Крім того, експортна природа SVG дозволяє користуватися графічними зображеннями, створеними D3, у друкованих виданнях.

Здіснювали також попередні спроби візуалізації даних до веб-браузерів. Найбільш помітними прикладами були інструменти Prefuse, Flare і Protovis, які можна вважати прямими попередниками D3.js.

Prefuse – інструментарій для візуалізації, створений у 2005 році, який ґрунтувався на Java, а візуалізації візуалізувалися у браузерах з плагіном Java. Flare – схожий інструментарій з 2007 року, який користувався ActionScript, і потребував плагіна Flash для рендеринга.

У 2009 році, виходячи з досвіду розробки та використання Prefuse і Flare, Джефф Хеер, Майк Бостоk, і Вадим Огівецький зі Стенфордської групи візуалізації Стенфордського університету створили Protovis, бібліотеку JavaScript для створення SVG-графіки з даних. Бібліотека відома практикам та науковцям із візуалізації даних.

У 2011 році розвиток Protovis зупинено, щоб зосередитися на новому проекті D3.js. Ознайомившись з досвідом роботи з Protovis, Bostock, разом з Heer і Ogievetsky, створили D3.js, щоб забезпечити більш виразну структуру, яка в той же час зосереджується на веб-стандартах і забезпечує ліпшу продуктивність.

Рисунок 2.8 – Приклад візуалізації з D3.js [46]

Бібліотека JavaScript D3.js, вбудована у веб-сторінку HTML, користується попередньо створеними функціями JavaScript для вибору елементів, створення об'єктів SVG, їх стилювання або додавання до них переходів, динамічних ефектів або підказок. Такими об'єктами також можна користуватися за допомогою CSS. Великі набори даних можна легко прив'язати до об'єктів SVG за допомогою простих функцій D3.js для створення багатого тексту та графічних діаграм. Дані можуть бути у різних форматах, найчастіше JSON, значення, розділені комами (CSV) або geoJSON, але, якщо потрібно, функції JavaScript можна записати для читання інших форматів даних.

Центральний принцип дизайну D3.js полягає в тому, щоб дозволити програмісту спочатку скористатися селектором стилів CSS для вибору заданого набору вузлів об'єктної моделі документа (DOM), а потім за допомогою операторів маніпулювати ними, як от у jQuery. Наприклад, за допомогою D3.js можна вибрати всі елементи HTML ... , а потім змінити колір тексту.

2.4.2 Vis.js

Vis.js – динамічна бібліотека візуалізації на основі браузера. Бібліотеку розроблено так, щоб нею можна було легко скористатися,

опрацьовувати великі обсяги динамічних даних, а також маніпулювати та взаємодіяти з ними. Бібліотека складається з компонентів DataSet, Timeline, Network, Graph2d і Graph3d.

Vis.js дозволяє:

- Відображати динамічні, автоматично організовані перегляди мережі.
- Створювати повністю налаштовувану, інтерактивну шкалу часу з елементами та діапазонами.
- Будувати графіки та стовпчикові діаграми на інтерактивній шкалі та персоналізувати їх
- Створювати інтерактивні, анімовані 3D графіки. Поверхні, лінії, крапки і блокування стилю з коробки.
- Керувати неструктурованими даними за допомогою DataSet. Додавати, оновлювати та видаляти дані і прослуховувати зміни у даних.

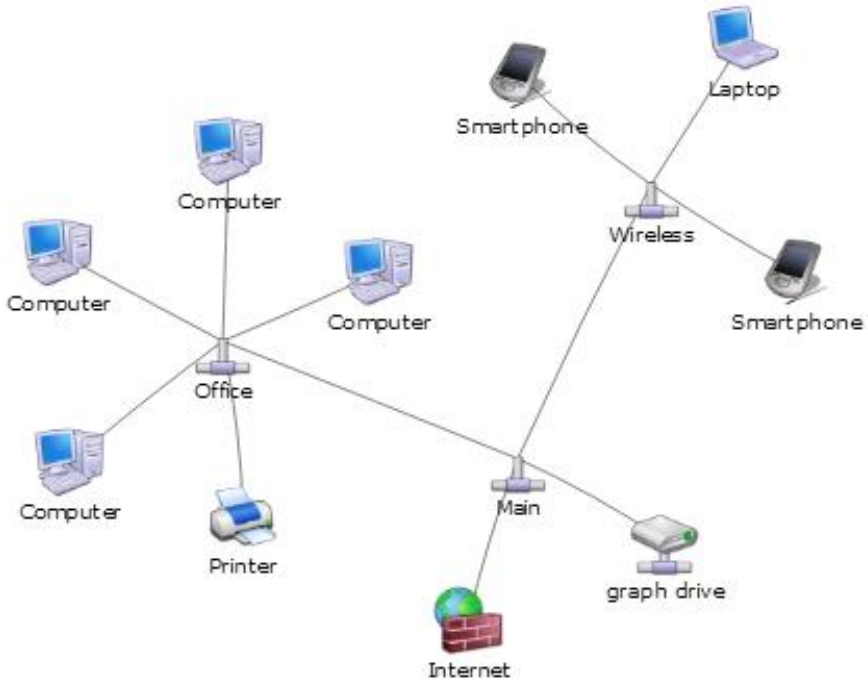


Рисунок 2.9 – Приклад візуалізації з Vis.js

Бібліотеку vis.js запропоновано компанією Almende B.V. Vis.js працює у Chrome, Firefox, Opera, Safari, IE9 + і більшості мобільних браузерів (з повною підтримкою дотику). Цей сайт містить документацію, завантаження та живі приклади vis.js. Вихідний код vis.js доступно у Github.

2.4.3 Chart.js

Chart.js – популярна бібліотека з відкритим кодом, яка допомагає нам створювати дані у веб-додатках. ЇЇ можна дуже точно налаштувати, але всіх його варіанти залишається проблемою для деяких людей. Розглянемо із простий приклад .

Налаштування досить просте для користувача React. Компонент Chart,js, як і будь-який інший компонент React, приймає реквізит, і саме через ці реквізити Chart.js знає, що робити потрібно виконати. Два основних реквізити, – дані, які підтримують дані, і опції, що пропонуються. Є ще багато інших, якими можна користуватися для подальшого налаштування діаграми.

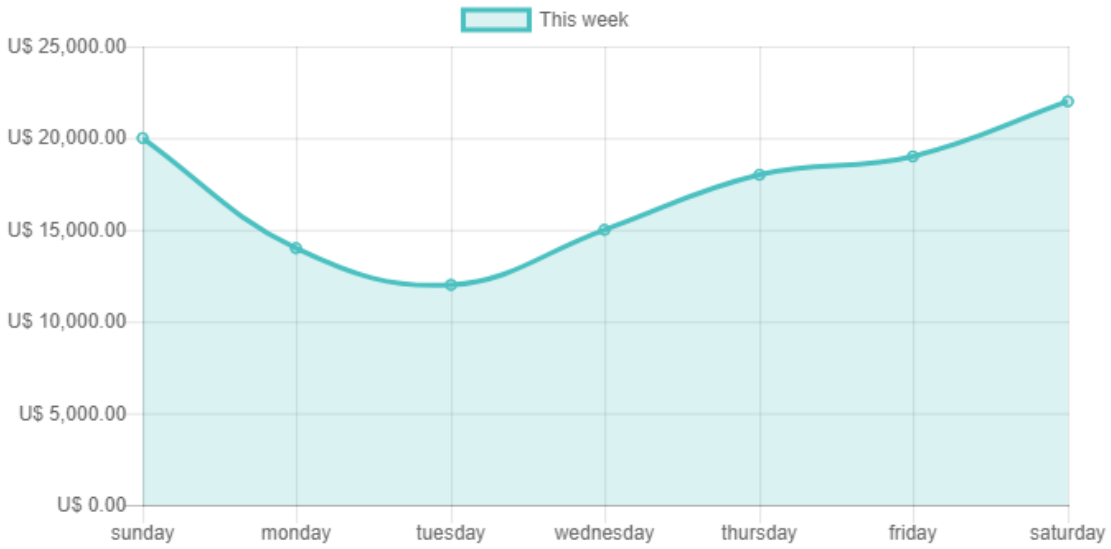


Рисунок 2.10 – Приклад візуалізації з Chart.js

Chart.js може побудувати осі динамічно на основі поданих даних. Якщо потрібно додатково налаштувати його, можна, пропустити додаткові опції у пропуску опцій.

2.5 Веб-скрапінг даних

Розглянуто основні інструменти та бібліотеки, якими користуватимуться для розробки веб-застосунку у даній роботі. Після певних змін у Facebook API, така соціальна мережа більше не надає доступу до списку друзів. Для того, щоб обійти це таке обмеження, необхідно власноруч витягнути усі необхідні дані з Facebook. Для цієї задачі найбільше підходить техніка веб-скрапінгу.

Веб-скрапінг (англ. web scraping) – це техніка, якою користуються для вилучення великої кількості даних з веб-сайтів, за допомогою якої дані витягують та зберігають у локальний файл на комп'ютері або в базі даних у таблиці (формат електронних таблиць).

Дані, що відображають на більшості веб-сайтів, можна переглядати лише за допомогою веб-браузера. Вони не пропонують можливості збереження копії цих даних для особистого користування. Єдиний варіант –скопіювати та вставити дані вручну – дуже кропітка робота, яка може зайняти багато годин, а іноді й днів. Веб-скрапінг –техніка автоматизації цього процесу, так що замість копіювання даних вручну з веб-сайтів, програмне забезпечення для веб-скрапінгу виконає те саме завдання за набагато менше часу.

Програмне забезпечення для веб-скрапінгу автоматично завантажує та витягує дані з декількох сторінок веб-сайтів на основі вашої потреби. То або створений на замовлення для конкретного веб-сайту, або такий, який

можна налаштувати для роботи з будь-яким веб-сайтом. Натисканням кнопки можна легко зберегти дані, доступні на веб-сайті, у файл на своєму комп’ютері.

Практичні сценарії використання:

1. Витягнути деталі продукту, включаючи ціну, зображення тощо з веб-сайтів електронної комерції для заповнення інших веб-сайтів, моніторингу конкуренції та ін.

2. Витягнути контактні дані про бізнес, включаючи ім’я, адресу, електронну пошту, телефон, веб-сайт тощо з Карт Яндекс, Google тощо для маркетингу та створення нових клієнтів.

3. Витягнути реквізити власності, а також контактні дані агента з веб-сайтів про нерухомість.

Методи веб-скрапінгу:

1. Застосування програмного забезпечення. Програмне забезпечення для веб-скрапінгу поділяють на дві категорії. Перше, котре можна встановити локально на вашому комп'ютері, і друге, котре працює в хмарному браузері. WebHarvy, OutWit Hub, Visual Web Ripper та ін. – приклади програмного забезпечення для веб-скрепінгу, яке можна встановити на комп'ютері, тоді як import.io, Mozenda – приклади хмарних платформ для вилучення даних.

2. Написання коду. Можна найняти розробника для створення спеціального програмного забезпечення для вилучення даних для вашої конкретної потреби. Розробник може, у свою чергу, скористатися веб-інтерфейсами для скрепінгу, що допомагає йому / їй легко розробляти програмне забезпечення. Наприклад, apify.com дозволяє легко отримати API, щоб отримати дані з будь-якого веб-сайту.

2.5.1 Веб-скрапінг за допомогою ПЗ

Існує багато програмних засобів, якими можна налаштовувати веб-скрапінг. Програмне забезпечення може здійснити автоматичний запит для розпізнавання структури даних сторінки. В інакшому випадку вонон надає інтерфейс запису, який усуває необхідність вручну писати код веб-скрепінгу, або деякі функції сценаріїв, якими вилучати та перетворювати вміст та інтерфейси бази даних, які можуть зберігати отримані дані у локальних базах даних. Деяким програмним забезпеченням для веб-скрепінгу також можна витягувати дані безпосередньо з API.

Octoparse – надійний гусеничний веб-переглядач веб-сайтів для отримання майже всіх видів даних, потрібних на веб-сайтах. Можна користуватися Octoparse для видобутку веб-сайту з його широкими функціональними можливостями. Він має 2 види режиму роботи – режим шаблону завдань та розширений режим. Користувальницький інтерфейс через вибір пунктів меню може допомогти вам пройти весь процес вилучення. Як результат, ви можете легко перетягувати вміст веб-сайту і зберігати його у структурованих форматах, таких як EXCEL, TXT, HTML або бази даних за короткий проміжок часу. На додачу, Octoparse повинен задовольняти найповніші потреби користувачів у сканування, як основні, так і розширені, без будь-яких навичок кодування.

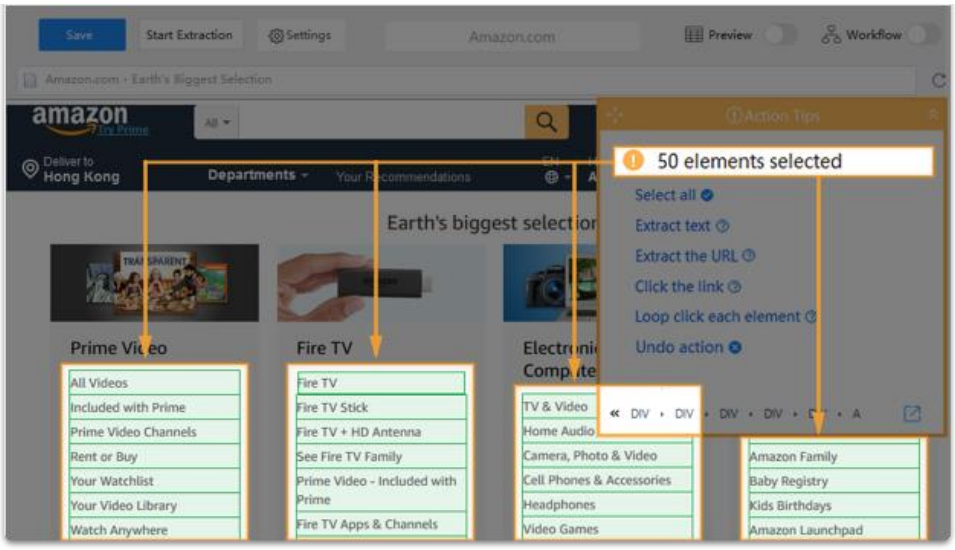


Рисунок 2.11 – Вигляд веб-сайту Octoparse [15]

WebCopy ілюстративний, як і його назва. Це безкоштовний сканер веб-сайтів, який дозволяє копіювати частково або повністю веб-сайти локально на жорсткий диск для офлайн-довідки. Можна змінити його налаштування, щоб повідомити боту, як потрібно сканувати. Крім того, можна також налаштувати псевдоніми домену, рядки агентів користувача, документи за замовчуванням тощо. Однак WebCopy не включає віртуальний DOM або будь-яку форму розбору JavaScript. Якщо веб-сайт широко застосовує JavaScript для роботи, швидше за все, WebCopy не зможе створити справжню копію. Швидше за все, він не буде правильно опрацьовувати динамічні макети веб-сайтів через JavaScript.

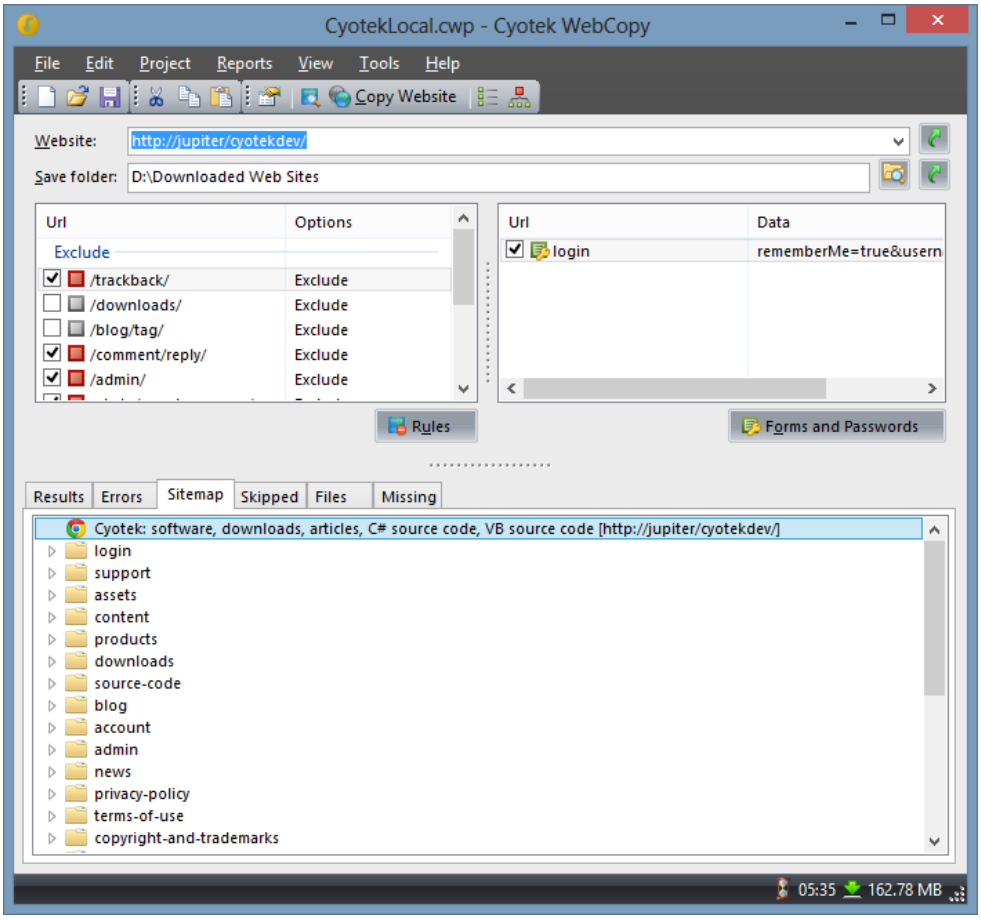
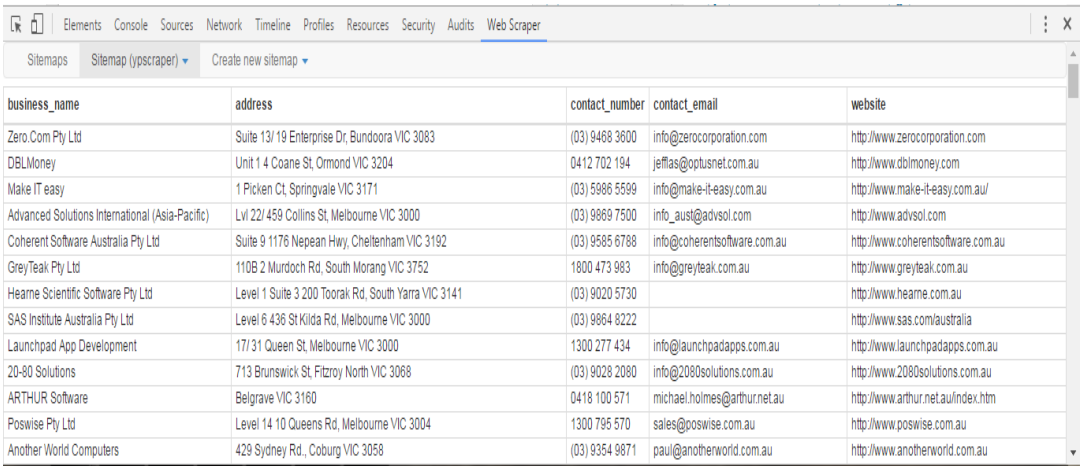


Рисунок 2.12 – Вигляд програми Cyotek WebCopy

Scraper – розширення для Chrome з обмеженими можливостями вилучення даних, однак корисними для онлайн-досліджень, щотакож дозволяє експортувати дані до електронних таблиць Google. Такий засіб призначено для початківців і експертів. Можна легко скопіювати дані у буфер обміну або зберегти у електронних таблицях за допомогою OAuth. Скрепер може автоматично генерувати XPaths для визначення URL-адрес для сканування. Він не пропонує послуги повного скрепінгу, але більшості людей не потрібно все повністю налаштовувати.



business_name	address	contact_number	contact_email	website
Zero.Com Pty Ltd	Suite 13/ 19 Enterprise Dr, Bundoora VIC 3083	(03) 9468 3600	info@zerocorporation.com	http://www.zerocorporation.com
DBLMoney	Unit 1 4 Coane St, Ormond VIC 3204	0412 702 194	jefflas@optusnet.com.au	http://www.dblmoney.com
Make IT easy	1 Picken Ct, Springvale VIC 3171	(03) 5986 5599	info@make-it-easy.com.au	http://www.make-it-easy.com.au/
Advanced Solutions International (Asia-Pacific)	Lvl 22/ 459 Collins St, Melbourne VIC 3000	(03) 9869 7500	info_aust@advsol.com	http://www.advsol.com
Coherent Software Australia Pty Ltd	Suite 9 1176 Nepean Hwy, Cheltenham VIC 3192	(03) 9585 6788	info@coherentsoftware.com.au	http://www.coherentsoftware.com.au
GreyTeak Pty Ltd	110B 2 Murdoch Rd, South Morang VIC 3752	1800 473 983	info@greyteak.com.au	http://www.greyteak.com.au
Hearne Scientific Software Pty Ltd	Level 1 Suite 3 200 Toorak Rd, South Yarra VIC 3141	(03) 9020 5730		http://www.hearne.com.au
SAS Institute Australia Pty Ltd	Level 6 436 St Kilda Rd, Melbourne VIC 3000	(03) 9864 8222		http://www.sas.com/australia
Launchpad App Development	17/ 31 Queen St, Melbourne VIC 3000	1300 277 434	info@launchpadapps.com.au	http://www.launchpadapps.com.au
20-80 Solutions	713 Brunswick St, Fitzroy North VIC 3068	(03) 9028 2080	info@2080solutions.com.au	http://www.2080solutions.com.au
ARTHUR Software	Belgrave VIC 3160	0418 100 571	michael.holmes@arthur.net.au	http://www.arthur.net.au/index.htm
Poswise Pty Ltd	Level 14 10 Queens Rd, Melbourne VIC 3004	1300 795 570	sales@poswise.com.au	http://www.poswise.com.au
Another World Computers	429 Sydney Rd., Coburg VIC 3058	(03) 9354 9871	paul@anotherworld.com.au	http://www.anotherworld.com.au

Рисунок 2.13 – Приклад використання Scrapper у браузері Chrome

З наведених вище прикладів можна помітити, що веб-скрапінг можна здійснювати скориставшись ПЗ у різній формі – плагін для браузера, десктопний додаток або веб-сайт. У всіх формах веб-скрапінг за допомогою ПЗ дозволяє швидко і просто витягувати основні дані зі сторінки та зберігати їх на комп'ютері користувача. Найбільша перевага даного методу – простота, адже ПЗ не вимагає знання програмування.

2.5.2 Веб-скрапінг з використанням бібліотек

Веб-скрапінг із застосуванням бібліотек є не менш популярним порівняно з ПЗ. Це досягається завдяки наявності безлічі бібліотек, які значно спрощують написання коду. Для ще більшого спрощення цього завдання найчастіше застосовують мову програмування Python, яка дозволяє писати лаконічний код. У даному пункті розглянемо декілька бібліотек для Python [20].

Beautiful Soup – бібліотека Python для витягування даних з HTML та XML-файлів. Вона призначена для таких проектів, як сканування монітору. Така бібліотека пропонує прості методи та пайтонічні ідіоми для

навігації, пошуку та зміни дерева розбору. Такий інструмент автоматично перетворює вхідні документи в Unicode, а вихідні документи на UTF-8.

Lxml –інструмент Python для бібліотек C libxml2 та libxslt. Його визнано однією із багатофункціональних і простих для користування бібліотек для обробки XML та HTML мовою Python. Він унікальний у тому випадку, коли поєднує у собі швидкість та особливості XML цих бібліотек з простотою вбудованого API Python і є в основному сумісним, але перевершує добре відомий ElementTree_API.

MechanicalSoup – бібліотека Python для автоматизації взаємодії з веб-сайтами. Така бібліотека автоматично зберігає та надсилає файли cookie, переслідує переадресації та може переходити посилання та надсилати форми. MechanicalSoup надає аналогічний API, побудований на запитах гігантів Python (для сеансів HTTP) та BeautifulSoup (для навігації по документах). Однак цей інструмент не змінювався протягом декількох років, оскільки він не підтримував Python 3.

Python Requests – єдина бібліотека HTTP для мови Python. Вона дозволяє користувачеві надсилати запити HTTP/1.1, і нема необхідності вручну додавати рядки запитів до ваших URL-адрес або формувати-кодувати ваші POST-дані. Існує низка функціональної підтримки, такої, як перевірка SSL у стилі браузера, автоматична декомпресія, автоматичне декодування вмісту, підтримка проксі-серверів HTTP (S) та багато іншого. Запити офіційно підтримують Python 2.7 & 3.4–3.7 та працюють на PyPy.

Scrapy – відкрита бібліотека для отримання потрібних користувачеві даних із веб-сайтів. Написана мовою Python, Scrapy –швидке веб-сканування та скрепінг на високому рівні для Python. Ним можна користуватися для широкого спектру задач – від пошуку даних до моніторингу та автоматизованого тестування. В основному це програма для написання веб-павуків, які сканують веб-сайти та витягують з них дані.

Павуки –класи, які визначає користувач, а Scrapy користується павуками для скрепінгу інформації з веб-сайту (або групи веб-сайтів).

Selenium Python –веб-інструмент автоматизації з відкритим кодом, який пропонує простий API для написання функціональних або інтегрованих тестів за допомогою Selenium WebDriver. Selenium – сукупність різних програмних засобів, кожен з яких має різний підхід до підтримки автоматизації тестів. Весь набір інструментів призводить до багатого набору функцій тестування, спеціально орієнтованих на потреби тестування веб-додатків усіх типів. За допомогою API Selenium Python користувач може отримати доступ до всіх функцій Selenium WebDriver інтуїтивно зрозумілим чином. Наразі підтримуються версії Python 2.7, 3.5 та вище.

Urllib –пакет Python, яким можна користуватися для відкриття URL-адрес. Він збирає декілька модулів для роботи з URL-адресами, такими як urllib.request для відкриття та зчитування URL-адрес, які в основному є HTTP, urllib.error модуль визначає класи винятків для винятків, підняті urllib.request, urllib.parse модуль визначає стандартний інтерфейс для Уніфікованого локатора ресурсів (URL) містить рядки у компонентах, а urllib.robotparser забезпечує єдиний клас, RobotFileParser, який відповідає на запитання про те, чи може конкретний користувацький агент отримати URL-адресу на веб-сайті, який опублікував файл robots.txt.

Загалом з усіх бібліотек вибрано Selenium. Така бібліотека містить широкий набір інструментів, який дозволяє легко імітувати певні дії користувача, як, наприклад, введення імені і паролю користувача та вхід на веб-сайт. Також вона має широку підтримку серед спільноти та велику кількість прикладів застосування.

Головною ж перевагою бібліотек на ПЗ для задачі веб-скрапінгу є можливість автоматизації. У нашому випадку потрібно відкривати сотні

сторінок друзів та переглядати у них спільних друзів, що займає досить багато часу. При використанні ПЗ доводилось би заходити на сторінку кожного з друзів, переглядати список усіх друзів, і тоді вибирати потрібний елемент веб-сторінки та експортувати дані. При застосуванні мови програмування таку задачу можна автоматизувати і зберегти усі дані вже в обробленому вигляді в один файл. Єдиною небезпекою в даному випадку може бути заборона деякими веб-сайтами користуватис веб-скрапінг. Цей пункт розглянемо детальнішу у частині імплементації.

2.6 Висновки до другого розділу

У другому розділі представлено основні технології, якими скористаємося для розробки веб-додатку. Усі технології поділено на категорії. У кожній категорії представлено найпопулярніші. Обґрунтовано вибір кожної технології.

Спочатку представлено групу технологій, якими користуватимемося для отримання даних для візуалізації. Основною є Facebook API, оскільки саме з цієї соціальної мережі отримуватимемо список друзів. Принцип роботи та можливість застосування GraphQL і REST також описано в цьому пункті.

Далі представлено найпопулярніші фреймворки для розробки веб-застосунку. Застосувавши веб-фреймворк, заощадимо час, необхідний для розробки веб-застосунку. З усіх представлених фреймворків вибрано React через його прості налаштування, великий набір додаткових бібліотек та широку підтримку розробників.

У наступному пункті показано популярні JavaScript бібліотеки для візуалізації даних. З трьох претендентів обрано D3.js, оскільки вона дозволяє гнучко налаштовувати вигляд та має велику кількість

стандартних візуалізацій, які чудово підходять для представлення соціального графу.

Далі пункті описано можливості веб-скрапінгу даних. Оскільки Facebook API більше не надає доступу до списку спільних друзів користувача, доводиться користуватися даною технікою для отримання даних. Веб-скрапінг із застосуванням бібліотек у нашому випадку є набагато зручнішим варіантом, адже дозволяє автоматизувати процес отримання даних завдяки імітуванню дій реального користувача.

3 ОПИСАННЯ РОЗРОБКИ ВЕБ-ЗАСТОСУНКУ ДЛЯ ВІЗУАЛІЗАЦІЇ ТА АНАЛІЗУ МЕРЕЖІ ДРУЗІВ

3.1 Отримання даних з мережі Facebook

Для того, щоб створити соціальний граф, спочатку потрібно отримати дані з однієї із популярних соціальних мереж. У нашому випадку вибрано мережу Facebook, адже вона є найбільшою в світі та налічує більше 2 мільярдів користувачів. Завдяки її популярності можна легко побудувати доволі великий і детальний граф друзів. Звичайно нам також знадобиться аккаунт з мінімумом 100 друзями. Для даної роботи вибрано аккаунт Максима Радчука з кількістю друзів рівною 156.

3.1.1 Веб-скрапінг даних з мережі Facebook

Наступний крок – вибір правильного інструменту для отримання даних. Як описано у попередньому розділі, отримати дані за допомогою штатних інструментів Facebook уже неможливо і тому обрано інший шлях. У цьому нам як раз і допоможе техніку веб-скрапінгу. Вона дозволить отримати усі дані, які користувач бачить у браузері. Завдяки автоматизації веб-скрапінгу з використання мови програмування Python, розв’язати таку задачу можна, мінімізувавши кількості монотонної роботи.

Для запуску коду користувачу потрібно мати встановленим середовище Python, а також додаткові бібліотеки. Найважливішою з бібіліотек є Selenium, яка дозволяє імітувати дії реального користувача за

допомогою спеціальних функцій, які можна викликати у програмі. Також для правильної роботи потрібно установити Chrome WebDriver, який служить мостом між браузером Google Chrome та програмним інтерфейсом бібліотеки Selenium.

Після установки усіх залежностей користувач може запустити код, написавши команду “python3”, а далі назву файлу із кодом. Далі система запитає імя користувача та його пароль, а також унікальний ідентифікатор. Даний ідентифікатор може бути як автоматично згенерованим у вигляді цифр, так і заданий користувачем. Цей ідентифікатор можна вставити в адресну стрічку для пошуку користувача у мережі Facebook.

Рисунок 3.1 – Приклад запуску програми і введення даних

Коли програма почне працювати, вона відкриє браузер Google Chrome, введе логін та пароль користувача. Далі, скориставшись унікальним ідентифікатором користувача, перейде у список його друзів. Далі, витягнувши усі унікальні ідентифікатори друзів, почне переходити на сторінку кожного з них і прогортатиме її повністю (лістинг 3.1). Прогортування здійснюється за допомогою Chrome WebDriver. Для цього створено цикл, у якому постійно п прогрортуємо на величину видимої області. Далі порівнюємо видиму область у даний момент і ту, яка залишилась. Якщо прогорнуто сторінку до кінця, то цикл завершується і повертається вся HTML розмітка сторінки. Додатково, щоб прогортування імітувало дії користувача і не викликало підозр у веб-сайту, час від часу робимо паузи у виконанні завдяки функції “time.sleep”. З розмітки

сторінки користувача отримуємо лише список спільних друзів, який і дасть нам можливість згодом побудувати соціальний граф.

```
driver.get(url)
    last_height = driver.execute_script("return
document.body.scrollHeight")

    while True:
        driver.execute_script("window.scrollTo(0,
document.body.scrollHeight);")
        time.sleep(SCROLL_PAUSE_TIME)
        new_height = driver.execute_script("return
document.body.scrollHeight")
        if new_height == last_height:
            break
        last_height = new_height
    html_source = driver.page_source
    return html_source
```

Лістинг 3.1 – Автоматична прокрутка сторінки

Під час виконання програми усі дані зберігатимуться у файлі формату pickle. Це формат, яким користуються для потокового запису інформації. Він займає мінімум місця на диску і дозволяє швидко отримувати доступ до файлу. Іншою перевагою є також можливість зберігати файли у процесі роботи програми, тому, якщо програма перерветься, то при наступному запуску вона продовжить виконання з того ж місця. Це особливо важливо, оскільки навіть у випадку 156 друзів виконання програми зайняло майже 1 годину.

3.1.2 Обробка і очищення даних

Після виконання графу отримані дані все ще не придатні для візуалізації, тому нам потрібно їх додатково опрацювати. Для цієї задачі ідеально підходить Jupyter Notebook (рис. 3.2). Дана програма дозволяє

легко суміщати звичайний текст із кодом, а також виводити проміжні результати. Jupyter Notebook надає простий користувацький інтерфейс та автоматично зберігає код.

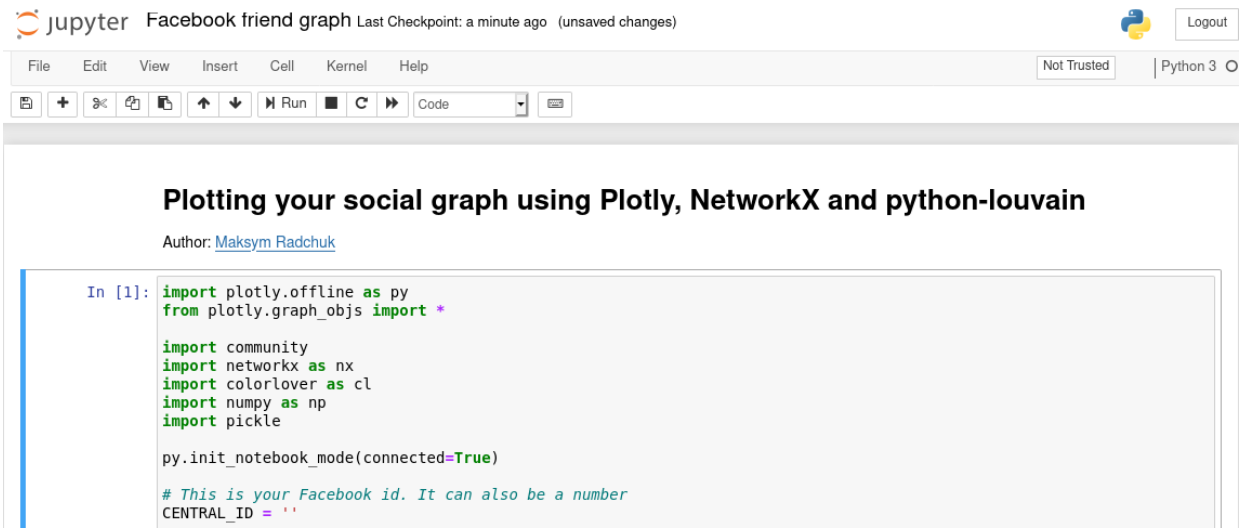


Рисунок 3.2 – Вигляд коду у програмі Jupyter Notebook

Нам знадобляться також дві додаткові бібліотеки – networkx та plotly. Перша дозволить перетворити дані у справжній граф, відфільтрувати непотрібні дані та зберегти у фоматі JSON, який можна скористатися для подальшої візуалізації. Друга дозволить швидко візуалізувати дані, перевірити правильність обробки даних та згрупувати друзів.

Спочатку завантажуємо дані, отримані під час веб-скрапінгу у змінну “friend_graph”. Далі перетворюємо список друзів і записуємо усі їх зв’язки (ребра графу), при цьому фільтруючи їх (лістинг 3.2). Фільтр у даному випадку – кількість спільних друзів. Якщо їх більше двох, то залишаємо дані зв’язки. Більш слабкі зв’язки відфільтрували, щоб не засмічувати граф великою кількістю надлишкової інформації.

```
edges = []
nodes = [CENTRAL_ID]
central_friends = {}
for k, v in friend_graph.items():
    intersection_size=len(np.intersect1d(list(friend_graph.keys()),
v))
    if intersection_size > 2:
        central_friends[k] = v
for k, v in central_friends.items():
    for item in v:
        if item in central_friends.keys() or item == CENTRAL_ID:
            edges.append((k, item))
```

Лістинг 3.2 – Перетворення графу

Далі створюємо пустий граф за допомогою бібліотеки networkx та додаємо створені вершини та ребра графу. На цьому етапі граф уже готовий і можна ним користуватися у нашому додатку. Для цього networkx надає можливість експорту в різні формати файлів, такі як GraphML, XML, YAML, Pickle, JSON. З них усіх ми оберемо формат JSON, оскільки він найпоширеніший і найкраще підтримує візуалізацію засобами D3.js.

3.1.3 Візуалізація графа бібліотекою plotly

Як зазначалось вище, граф візуалізуємо бібліотекою plotly. Дана бібліотека дозволяє швидко візуалізувати граф навіть за стандартної конфігурації. Даною бібліотека часто візуалізовують графіки в області Big Data. Вона продукує графи, які потребують мінімум конфігурації та з коробки підходять для використання у звітах. На рисунку 3.3 зображено візуалізацію графа друзів бібліотекою plotly. Групи друзів помічено різними кольорами.

Рисунок 3.3 – Візуалізація графу засобами plotly

Оскільки відфільтровано ребра для друзів, які мають менше двох спільних друзів із заданим користувачем, можна чітко побачити, що утворилась окрема група вершин зліва. В даному випадку, то колеги по роботі. Інші групи, які утворились – друзі з різних університетів, друзі зі школи та друзі з Америки. Як бачимо, візуалізація дозволяє нам чітко побачити як наші друзі зв’язані і розбити їх по групам.

3.2 Створення веб-додатку для візуалізації графа

Отримавши і перетворивши дані за допомогою Python, можна візуалізувати дані, застосувавши JavaScript. Дана мова програмування дозволяє побудувати веб-додаток, доступний для користувачів у браузері. Оскільки браузер можна запускати на абсолютно різних пристроях – від розумних годинників до комп’ютерів – це дозволить користуватися додатоком більшій кількості людей. Додаток має бути простим у застосуванні та представляти користувачу інтерактивну візуалізацію його даних графа друзів. В даній роботі скористалися даними із соціальної

мережі Facebook, однак користувач може скористся будь-якою соціальною мережею або ж навіть створювати граф власноруч.

3.2.1 Розробка інтерактивної візуалізації

Для розробки веб-додатка з інтерактивною візуалізації, скористаємося бібліотекою React, яка допоможе простіше налаштувати та ініціалізувати проект. Вона зразу ж пропонує створити базовий проект, модифікуючи який, можна швидко створити інтерактивну веб-сторінку. React створює веб-сторінки, комбінуючи різні компоненти. Компоненти – невеликі незалежні частини коду, які відповідають за візуалізацію частини веб-сторінки.

У нашому випадку скористалися компонентом “Graph” (лістинг 3.3), який доступний для React і надає широкі можливості зміни візуалізації, а також багато інтерактивних елементів. Він базується на бібліотеці D3.js та повторно користується значною кількістю її елементів, розширюючи їх.

```
<Graph
   id='graph-id'
   data={socialGraphData}
   config={this.myConfig}
   onClickNode={this.onClickNode}
   onDoubleClickNode={this.onDblClickNode}
   onRightClickNode={this.onRClickNode}
   onClickLink={this.onClickLink}
   onRightClickLink={this.onRClickLink}
   onMouseOverNode={this.onMouseOvrNode}
   onMouseOutNode={this.onMouseOtNode}
   onMouseOverLink={this.onMouseOvrLink}
   onMouseOutLink={this.onMouseOtLink}/>
```

Лістинг 3.3 – Використання компоненту Graph

Даний компонент пропонує нам значну кількість параметрів для конфігурації. Зокрема параметр “data” передає соціальний граф. Наведено велику кількість параметрів для виконання функцій при маніпулювання користувачем вершинами та ребрами. Зокрема, коли користувач натискає лівою кнопкою миші, правою кнопкою миші, натискає двічі або проводить мишкою.

Не менш важливим вхідним параметром є конфігурація графа, яка задається програмно (лістинг 3.4), але користувач може змінювати її впродовж роботи з веб-додатком. У ній можна задати параметри вершини такі як колір, розмір та колір підсвічування при наведенні. Для ребер графа доступні параметри кольору підсвічування та типу проведення лінії. Тип проведення лінії може бути прямим, частково або повністю заокругленим.

```
nodeHighlightBehavior: true,
highlightDegree: 2,
node: {
    color: 'green',
    size: 200,
    highlightStrokeColor: 'darkblue'
},
link: {
    highlightColor: 'blue',
    type: 'CURVE_SMOOTH'
},
d3: {
  gravity: -100,
  linkLength: 100,
  linkStrength: 0.5,
```

Лістинг 3.4 – Конфігурація графа

В окремому об’єкті розташовано параметри для бібліотеки D3.js. Зокрема, дуже цікавим є параметр “gravity”, оскільки він відповідає за те, наскільки вершини притягуються або відштовхуються одна від одної. Параметр “linkStrength” відповідає за силу, з якою ребра зіштовхуються. А параметр “linkLength” відповідає за довжину ребер. Отже, три вище описаних параметри відповідатимуть за довжину ребер між вершинами.

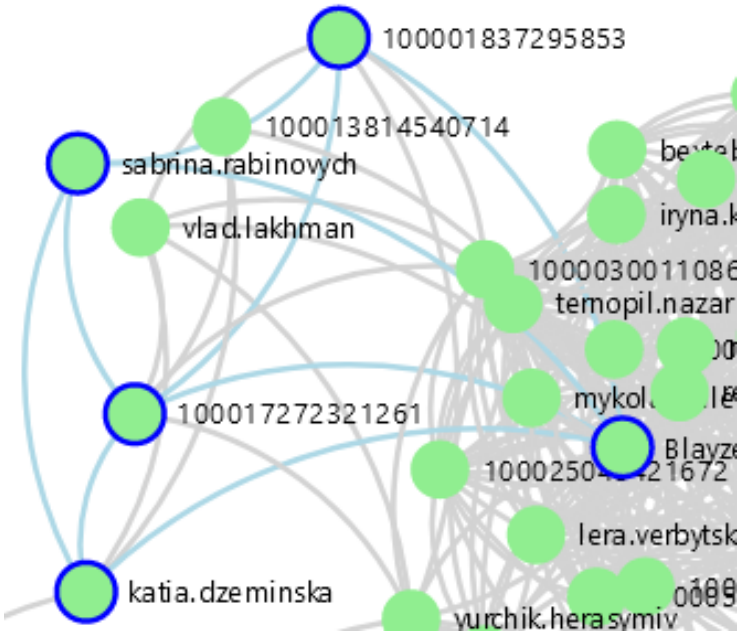


Рисунок 3.4 – Приклад підсвічування спільних друзів

На рисунку 3.4 зображено приклад застосування параметрів “nodeHighlightBehavior” та “highlightDegree”. Перший вмикає підсвітку сусідніх з обраною точкою вершин, а другий задає, наскільки близькі вершини потрібно підсвічувати. В даному випадку,підсвічуємо тільки сусідні вершини. Також можна побачити, що кольори підсвітки вершин і ребер є такими, які задано у конфігурації графу. Завдяки такій

інтерактивності можна легко побачити друзів, які мають зв’язки з нашими друзями.

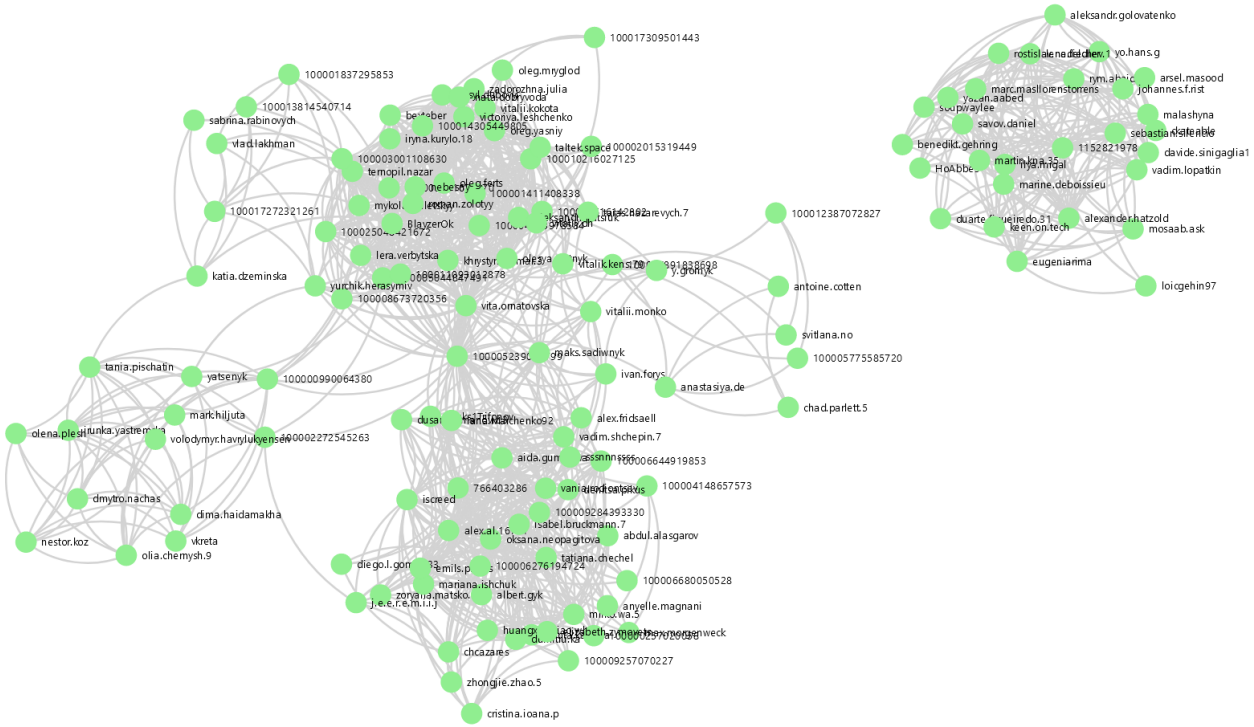


Рисунок 3.5 – Вигляд побудови графа у браузері

На рисунку 3.5 уже повноцінно візуалізуваний граф. Порівнюючи його з графом на рисунку 3.3, можна помітити набагато більшу кількість ребер, оскільки їх усіх видно. Це дозволяє нам візуально побачити, наскільки сильно наші друзі зв’язані одне з одним. Наприклад, у центрі графа видно найбільшу кількість зв’язків друзів одне з одним, тоді як ,у нижньому лівому куті кількість ребер набагато менша. Також даний граф набагато інформативніший, оскільки показує одразу усі назви вершин – унікальні ідентифікатори друзів. Він може бути й більш інтерактивним, що побачимо у наступному пункті.

3.2.2 Додаткові можливості візуалізації

У попередньому пункті, розглянутовізуалізацію за допомогою конфігурування компоненту “Graph”. Додатково можна збільшити кількості даних у графі. Приміром, можна для кожного друга додати інформацію про стать, наявність мотоцикла і/або автомобіля і т. ін. (рис. 3.6).

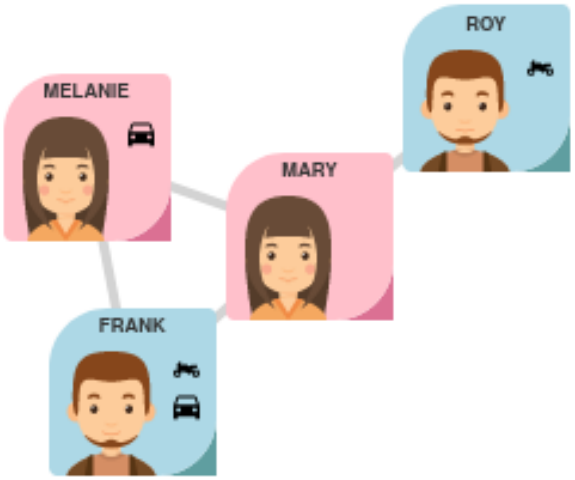


Рисунок 3.6 – Приклад графа з додатковими даними

Для того, щоб це виконати, достатньо для кожної вершини, окрім поля унікального ідентифікатора додати поля статі, наявності машини та наявності мотоцикла. При цьому сам компонент “Graph” дозволяє змінити вигляд кожної вершини залежно від переданих даних. Даний вигляд задають у форматі SVG. Це формат векторної графіки, який чудово виглядає на будь-якому моніторі користувача.

Матриця суміжності може бути іншим варіантом представити соціальний графна. Це квадратна матриця, розмір якої відповідає кількості вершин у графі. Кожен елемент матриці відповідає кількості ребер між двома вершинами. У випадку соціального графу у нас буде завжди

максимум тільки одне ребро. При розробці програмного забезпечення даний вид матриці часто використовується для збереження графу, оскільки займає мінімум дискового простору. На рисунку 3.7 зображено матрицю суміжності на прикладі спільних появ персонажів роману Віктора Гюго “Знедолені” [14].

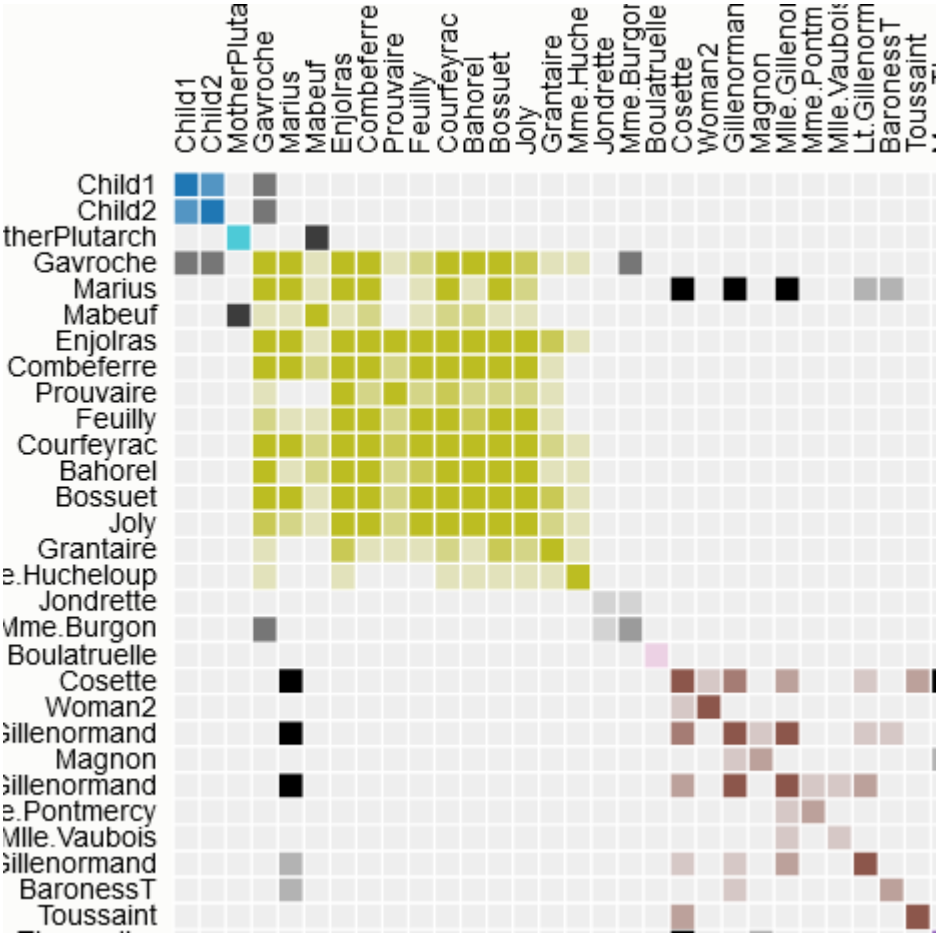


Рисунок 3.7 – Приклад візуалізації матриці суміжності

З наведених вище рисунків можна побачити, що є різні варіанти представити соціальний граф, якими не б скористалися у даній роботі, але можна реалізувати в майбутньому.

Зокрема представлення додаткових даних для вершин є складнішою задачею для використання з мережею Facebook, оскільки вона

не надає простого доступ до даних. У нашому випадку довелось би шукати кожне з потрібних полів на сторінці користувача. Соціальні мережі, такі як Вконтакті або Twitter, більш лояльні до отримання даних про користувачів і дозволяють одразу отримувати всі необхідні публічні дані.

Візуалізація у вигляді матриці суміжності потребує від користувача передачу дещо іншого формату даних із зазначенням, до якої групи кожна з вершин входить та побудови окремої компоненти. Безумовно, її імплементація розширить можливості веб-додатку та дасть більше інструментів для дослідження соціального графу.

3.3 Висновки до третього розділу

У третьому розділі розглянуто отримання даних за допомогою веб-скрапінгу, імплементацію веб-додатку для візуалізації соціального графа та порівняно різні можливі представлення.

У першому пункті описано процес отримання даних за допомогою веб-скрапінгу інтернет сторінок друзів у Facebook. Наведено інструменти, якими для цього в скористалися та процес зчитування і запису результатів. Далі показано процес трансформації та очищення даних, включаючи інструменти та бібліотеки, якими скористалися для досягнення бажаного результату. Показано можливість візуалізації графа у Python бібліотекоюе plotly.

У другому пункті описано розробку веб-додатка. та основні бібліотеки, якими скористалися – React та D3.js. Проілюстровано роботу з компонентом “Graph”, а саме, його конфігурацію та додаткові параметри, які він приймає. Показано вигляд готового графа у веб-додатку. Також описано додаткові можливості візуалізації – завдяки представленню

додаткової інформації та альтернативний варіант представлення графу у вигляді матриці.

4 СПЕЦІАЛЬНА ЧАСТИНА

4.1 Огляд інтегрованих середовищ розробки

Сьогодні JavaScript користуються у багатьох застосунках. Найчастіше JavaScript застосовують разом із HTML5 та CSS,створючи передні веб-сторінки. Але JavaScript також допомагає створювати мобільні додатки, і він займає важливе місце на задньому кінці у вигляді серверів Node.js. На щастя, інструменти розробки JavaScript – як редактори, так і інтегроване середовище розробки (IDE) –-дозволяють розв’язувати все нові задачі.

Навіщо користуватися IDE замість редактора? Основна причина полягає в тому, що IDE може відлагоджувати і іноді профілювати код. IDE також підтримують системи ALM, інтегруючись з Git, GitHub, Mercurial, Subversion та Perforce для контролю версій. Однак, оскільки більше редакторів додають гачки до цих систем, підтримка ALM стає менш диференційованою.

4.2 Вибір середовища розробки для мови програмування Python

Коротко описано популярні інструменти загального призначення, IDE та редактори коду, якими можна користуватися для розробки на JavaScript.

4.2.1 Eclipse

Спочатку, під впливом IBM VisualAge, Eclipse є одним з трьох великих IDE Java. Він доступний разом із розширюваною системою

плагінів. Щоб користуватися Eclipse для розробки JavaScript, а також з іншими мовами програмування, потрібно встановити спеціальні плагіни.

Eclipse – одне із перших IDE, що запускається під GNU Classpath. Інтегроване середовище розробки чудово поєднує продуктивність, надійність та стабільність. Налаштування проекту Oomph дозволяє автоматизувати та відтворювати однакові робочі простори.

Eclipse з інструментами розробки JavaScript має потужний редактор JavaScript і налагоджувач на основі Chrome, однак він не знає про TypeScript, яким користоується Angular, або про файли ES6 та JSX, якими користується React.

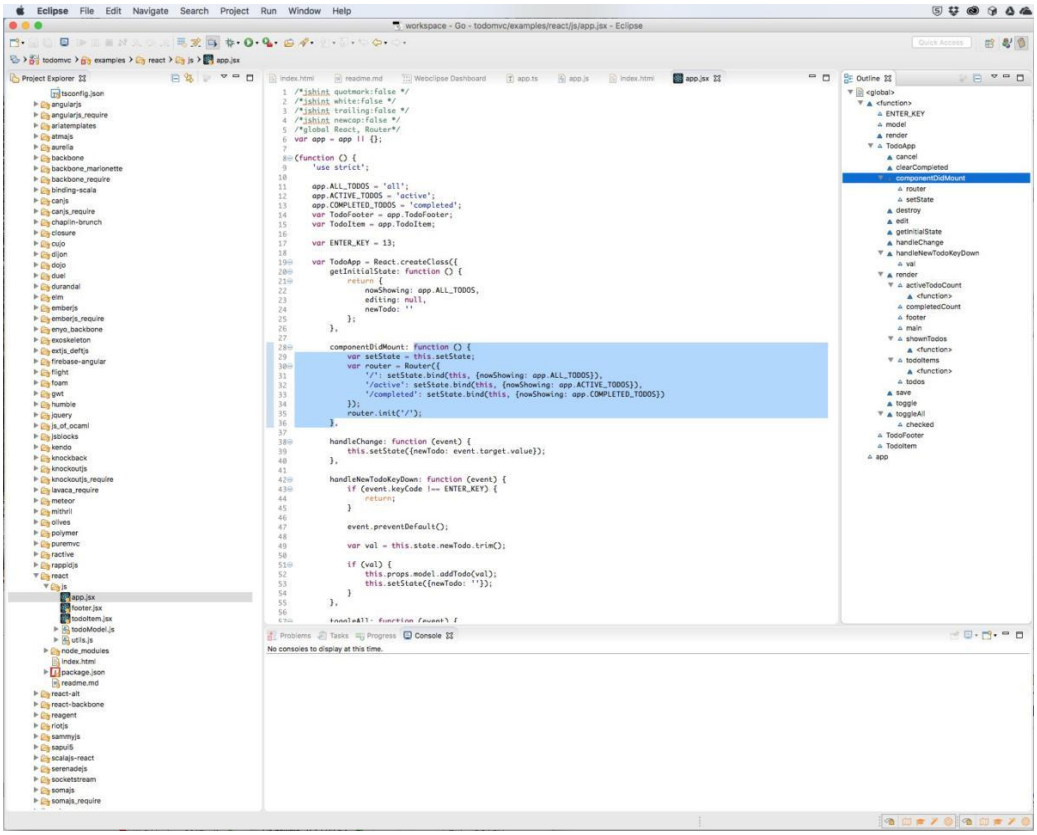


Рисунок 4.1 – Вигляд Eclipse

Eclipse завжди мав величезну кількість плагінів. Зокрема, TypeScript підтримує безкоштовний плагін TypeScript 1.0.0. Підтримку Angular, TypeScript та ES6 реалізовано у некомерційному Angular IDE (за CodeMix, раніше Webclipse), а для проектів React з файлами JSX рекомендовано скористатися TypeScript IDE. Якщо додати більше одного плагіна,, потрібно розв'язати суперечку щодо того, який слід редагувати файли TypeScript.

4.2.2 Komodo IDE

Komodo IDE забезпечує вдосконалене редагування JavaScript, підсвічування синтаксису, навігацію та налагодження, але воно не містить перевірки коду JavaScript. Для цього завжди можна запустити JSHint в оболонці [22].

Komodo підтримує десятки мов програмування та розмітки. Завдяки широкому спектру підтримки мови програмування та розмітки, включаючи рефакторинг, налагодження та профілювання, Komodo IDE є дуже добрим вибором для цілісного розвитку мов з відкритим кодом.

Komodo має модуль рефакторингу коду для всіх мов, для яких він забезпечує розумні підказки для коду: PHP, Perl, Python, Ruby, Tcl, JavaScript та Node.js. На жаль, характер "найменшого спільного знаменника" такого підходу обмежує можливості перейменування змінних та членів класу та вилучення коду у метод. Однак, це декілька найбільш корисних випадків.

Komodo IDE дозволяє як редагувати стовпці, так і декілька варіантів. Це забезпечує майже співвідношення з Sublime Text і TextMate, що стосується масових редагувань. Поки ми порівнюємо, Komodo – це більше IDE, тоді як Sublime Text – набагато швидше. І поки ми кажемо про

ефективність, швидкість Komodo помітно зросла порівняно зі старими версіями, у малюванні екрану, пошуку та синтаксисі.

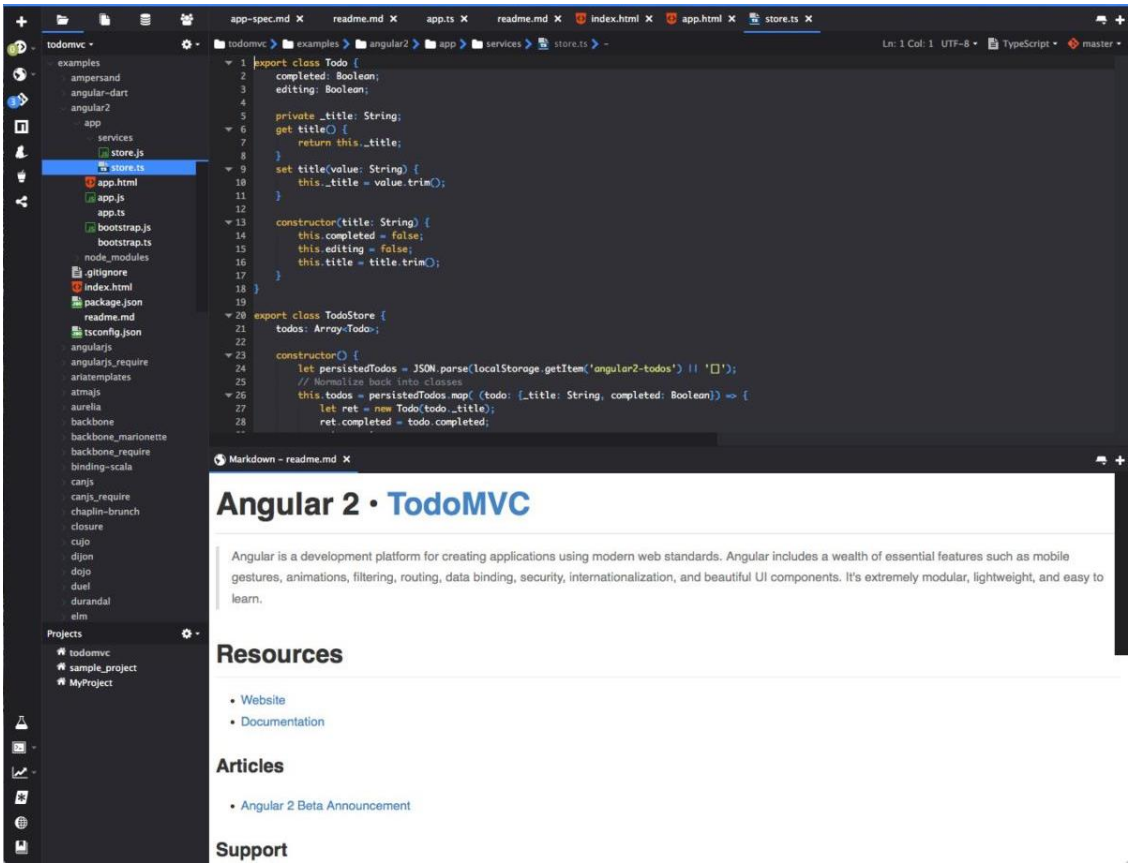


Рисунок 4.2 – Вигляд Komodo IDE

Komodo IDE має декілька функцій, яких не вистачає більшості конкуруючих продуктів. Один – його HTTP Inspector, який відмінно підходить для налагодження зворотних викликів Ajax. Іншим є його інструментарій Rx (регулярний вираз або регулярний вираз), який є чудовим способом побудови та тестування регулярних виразів для JavaScript, Perl, PHP, Python та Ruby.

Співпраця – ще один диференціатор ID-коду Komodo, щось на кшталт Документи Google для коду. Можна створювати сеанси для груп

файлів, додавати контакти до сеансів як співавторів, а потім працювати разом над тими ж файлами одночасно з синхронізацією в реальному часі.

Співпраця не є заміною для контролю вихідного коду, але це корисна можливість. Komodo IDE інтегрує контроль вихідного коду за допомогою CVS, Subversion, Perforce, Git, Mercurial та Bazaar. Підтримуються лише основні операції з управління версіями. Розширені операції, такі як розгалуження, повинні здійснюватися за допомогою окремого клієнта управління вихідним кодом.

Хоча у Komodo відсутній власний формат документа JavaScript, він користується найліпшим безкоштовним відкритим кодом для цієї мети. З цього вікна формат за замовчуванням для файлів JavaScript – JS Beautifier, але ще дев'ять варіантів доступно через спадне меню.

Komodo IDE підтримує налагодження JavaScript на стороні клієнта у Chrome, і він може налагоджувати Node.js як локально, так і віддалено. Він також налагоджує Perl, Python, PHP, Ruby, Tcl та XSLT.

Komodo IDE містить переглядач DOM, який дозволяє переглядати документи XML і HTML у вигляді дерев, атакож дозволяє здійснювати пошук XPath для фільтрації дерева.

Модулі профілювання коду та модульні тестування коду Komodo не підтримують JavaScript. Однак JavaScript і Node.js підтримує модуль розумних функцій Komodo, який реалізує перегляд коду, автоматичне завершення та підказки.

Komodo IDE може публікувати групи файлів через FTP, SFTP, FTPS або SCP. Komodo також може синхронізувати файли та виявляти потенційні конфліктні публікації, які можуть призвести до того, щопереписуються зміни інших людей.

Загалом, Komodo – добре, але не чудове JavaScript IDE, а також добриц, але не чудовий редактор JavaScript. Однак він, загалом, може

задовольнити потреби користувача, особливо якщо потрібнопрацюєте вати також з Perl, Python, PHP, Ruby, Tcl або XSLT.

4.2.3 Apache NetBeans

NetBeans дуже добре підтримує JavaScript, HTML5 та CSS3 у веб-проектах, в також Cordova / PhoneGap для створення мобільних додатків на основі JavaScript. NetBeans –не найшвидше IDE у блоці, але воно є одним із найповніших.

Редактор JavaScript NetBeans забезпечує підсвітку синтаксису, автоматичне доповнення та складання коду, майже настільки, наскільки очікувано. Функції редагування JavaScript також працюють для коду JavaScript, вбудованого у файли PHP, JSP та HTML. Підтримка jQuery передається в редактор. NetBeans 8.2 містить нову або вдосконалену підтримку для фреймворків Node.js та Express, Gulp, Grunt, AngularJS, Knockout.js, Jade, Mocha та Selenium.

Аналіз коду працює у фоновому режимі під час редагування, надаючи попередження та підказки. Налагодження працює у вбудованому веб-переглядачі WebKit та в Chrome із встановленим роз'ємом NetBeans. Налагоджувач може встановити точки перемикання DOM, лінії, події та XMLHttpRequest, і відображатиме змінні, годинник та стек викликів. Інтегроване вікно журналу браузера відображає винятки, помилки та попередження браузера.

NetBeans може налаштувати та провести тестування одиниць за допомогою JsTestDriver, файла JAR (архіву Java), який можна завантажити безкоштовно. Налагодження тестів одиниць вмикається автоматично, якщо вказано Chrome із NetBeans Connector як один із браузерів JsTestDriver, коли налаштовуємо JsTestDriver у вікні Послуги.

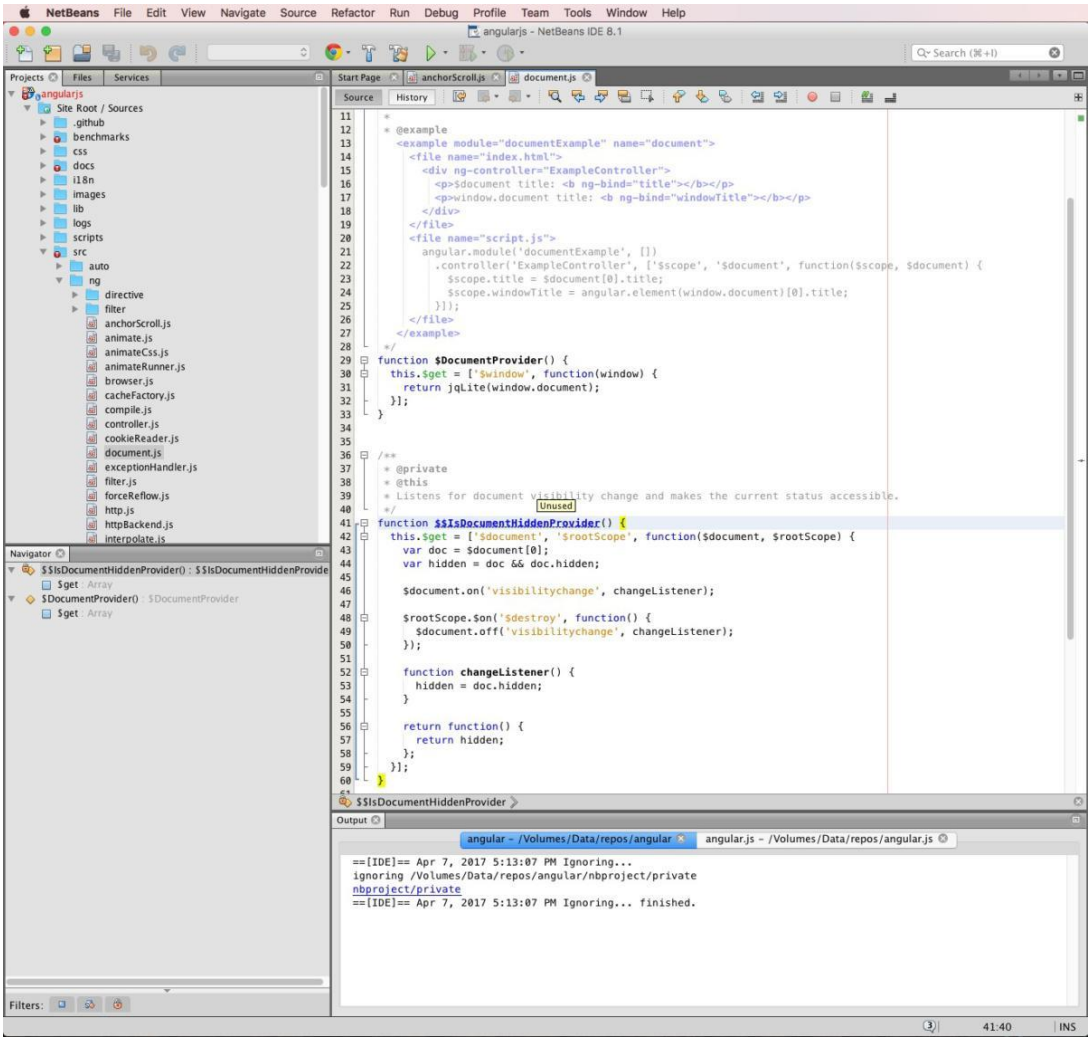


Рисунок 4.3 – Вигляд Apache NetBeans

Коли налагоджуємо веб-додаток у Chrome за допомогою з'єднувача NetBeans та редагуємо CSS з Інструментів для розробників Chrome, зміни відстежуються NetBeans та зберегатимуться у файлах CSS. Однак, якщо файли CSS створено на аркушах стилів LESS або SASS, доведеться вручну оновити вихідний аркуш, оскільки файли CSS – лише компільований вихід.

У вбудованому веб-переглядачі WebKit та в Chrome із встановленим роз'ємом NetBeans можна користуватися мережевим монітором NetBeans для перегляду заголовків запитів, відповідей та стеків викликів для зв'язку

REST. Для зв'язку WebSocket відображаються як заголовки, так і текстові кадри. Загалом, NetBeans забезпечує дещо ліпший досвід налагодження в Chrome, ніж реалізовано у Firefox з Firebug.

Загалом, NetBeans є гідним претендентом на розробку JavaScript, HTML5 та CSS3 на стороні клієнта, особливо якщо також займаєтеся розробкою Java, PHP або C ++ на сервері. Якщо у вас немає бюджету на WebStorm і не подобається Microsoft, виявите, що NetBeans виконує всі поставлені задачі.

4.2.4 Visual Studio

Visual Studio дуже добре служить як JavaScript IDE, хоча це і є кращим .Net IDE, і це не так добре, як WebStorm для JavaScript. Хоча воно також дуже добре працює як редактор JavaScript, він є кращим редактором C#, однак не такий добрий і не такий швидкий, як Sublime Text для JavaScript [23].

Як видно із знімка екрана, наведеного нижче, Visual Studio 2017 добре справляється з кольором синтаксису JavaScript та складанням коду, а також з навігацією у коді JavaScript: Клацнувши правою кнопкою миші на функцію або ім’я члена, і можна легко перейти до визначення або знайти всі посилання. Завершивши перегляд визначення, можна натиснути стрілку назад у верхній частині інтерфейсу, щоб повернутися назад.

Можна легко вставити фрагменти та обрамити свій вибір відповідним кодом, таким як HTML або URL-кодування рядкових змінних. Окрім JavaScript, HTML та CSS, можна редагувати файли Markdown і бачити відредагований Markdown, а також працювати з TypeScript.

Крім того, можна програмувати будь-якою мову .Net, на C ++ та Python. І як це підтримувалося раніше із Visual Studio, можна працювати із базами даних безпосередньо з IDE. Visual Studio особливо потужний для

роботи працює із базами даних SQL Server. Можна користуватися Visual Studio замість студії управління SQL Server для більшості операцій з базою даних, які потрібно виконати як програміст.

Visual Studio підтримує відлагодження майже у будь-якому веб-переглядачі, у якому ви хочете запустити його, включаючи браузери на мобільних пристроях та в емуляторах. Він також має два власних веб-браузери: звичайний внутрішній веб-браузер, який є підтримує версією Internet Explorer, та інспектор сторінки, який показує вам відтворену сторінку разом із усіма джерелами та стилями. Хоча Page Inspector робить багато потенційно трудомістких завдань, щоб налаштувати себе на сторінку. Однак після того, як ви перебуваєте на ній, ви можете залишитися там, без необзідності жонгляванням Visual Studio, браузером та інструментом для розробників браузера.

Продуктивність Visual Studio зазвичай досить висока, якщо наявно достатньо пам’яті та потужності процесора, але це, як правило, вимагає значних ресурсів. Visual Studio чудово діагностує продуктивність додатків, але то не дуже корисно для звичайного коду JavaScript, який, як правило, запускається всередині браузера. Visual Studio має специфічні терміни функціонування JavaScript, реагування на HTML інтерфейс користувача та засоби пам’яті JavaScript, але вони застосовуються лише до проектів універсальної платформи Windows на основі JavaScript, а не до веб-проектів, які зазвичай користуються JavaScript.

Visual Studio побудована в архітектурі, що підтримує можливість використання доповнень (Add-Ins), — плагінів від сторонніх розробників, що дозволяє розширювати можливості середовища розробки. Також підтримує велику кількість мов включаючи англійську, російську та німецьку.

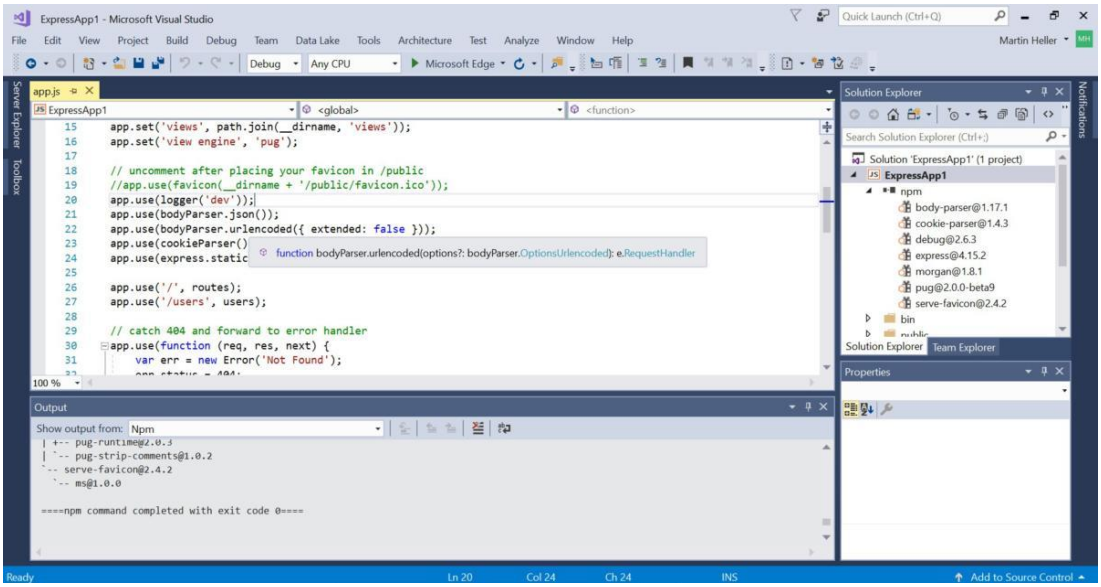


Рисунок 4.4 – Вигляд Visual Studio

Visual Studio включає в себе відмінне редагування додатків Node.js, IntelliSense, профілювання, інтеграцію NPM, підтримку TypeScript, налагодження локально та віддалено (Windows, MacOS, Linux) та налагодження у веб-додатках Azure та хмарних службах Azure. Він також підтримує CSS, HTML, JavaScript, TypeScript, CoffeeScript та менше. Сюди входить запуск JSHint під час введення тексту, що дозволяє мінімізувати файли JavaScript з контекстного меню, а також автоматично збирати файли CoffeeScript при збереженні, показуючи побічний попередній перегляд створеного JavaScript.

4.2.5 Visual Studio Code

Visual Studio Code – безкоштовний легкий редактор та IDE від Microsoft. У ньому є компоненти Visual Studio, поєднані з оболонкою Atom Electron із відкритим кодом, що відмінно підтримує для розробки ASP.Net

Core з C # та для Node.js з TypeScript і JavaScript., Код Visual Studio також працює на MacOS та Linux. Скріншот нижче виконано на MacOS.

Код Visual Studio має надзвичайно гарне завершення коду JavaScript завдяки включенню компілятора TypeScript та двигуна Salsa. Visual Studio Code надсилає код JavaScript компілятору TypeScript у фоновому режимі, щоб проаналізувати типи та створити таблицю символів. Результати можна побачити у вікні внизу екранного зображення, де відображається інформація про метод hasOwnProperty.

Така ж таблиця символів дозволяє IntelliSense надавати великі спливаючі списки параметрів для заповнення коду під час введення виразу. Отримуємо автоматичне закриття дужок, параметри автоматичного заповнення слів, автоматичні списки методів після введення "." та автоматичні списки параметрів у методі. Можна вдосконалити IntelliSense, додавши посилання на файли d.ts з DefinitelyTyped, а Visual Studio Code запропонує виконати це для користувача , коли він визначає поширені проблеми, наприклад, наявність __dirname, що є вбудованою змінною Node.js.

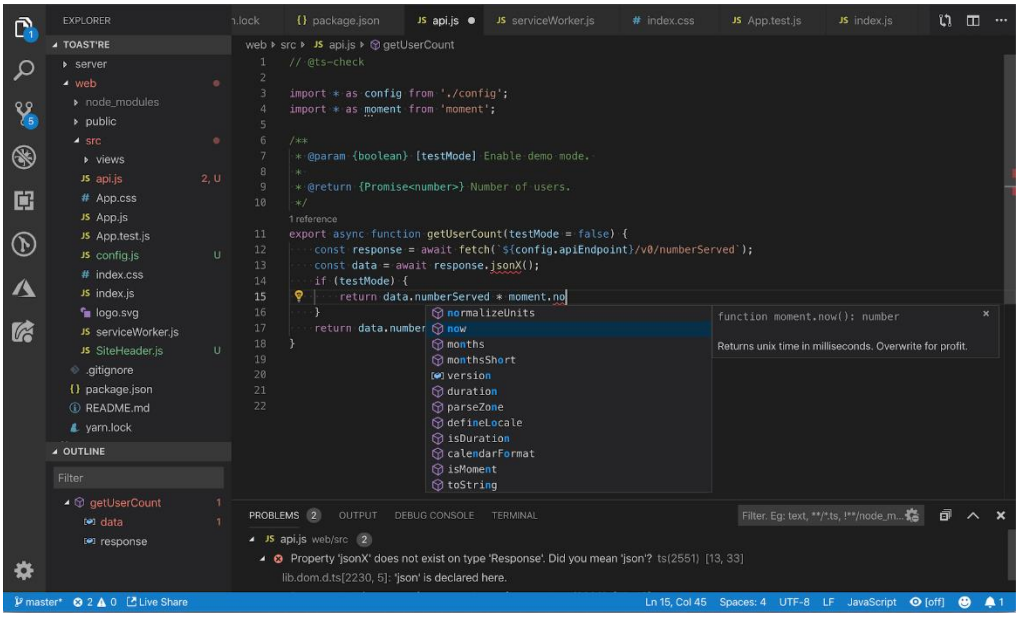


Рисунок 4.5 – Вигляд Visual Studio Code

Підтримка Git дуже хороша і досить проста на практиці . Відладчик Visual Studio Code забезпечує відмінний досвід налагодження для розробки Node.js (та розробки ASP.Net). Visual Studio Code має дуже хороші інструменти для HTML, CSS, Less, Sass та JSON, інструментів, заснованих на тій самій технології, що користується інструментами для розробників Internet Explorer F12. Окрім того, вона має настроювану інтеграцію із зовнішні бігуни завдань, такі як gulp та jake [25].

Visual Studio Code містить надійну екосистему плагінів – наприклад, для підтримки Angular та React.

4.2.6 JetBrains WebStorm

WebStorm JetBrains, незважаючи на свою скромну ціну, є високотехнологічним IDE для веб-розробників, зосереджуючись на передній частині HTML, CSS та JavaScript. JetBrains також продає IDE для Java, PHP, Ruby та Python, які мають спільний двигун.

Як редактор проектів веб-розробок, WebStorm такий же хороший, як і все, що там реалізовано. У ньому є все, чого очікує користувач, а також багато приємних сюрпризів. Зазвичай очікують забарвлення синтаксису та обмеження коду. Не можна очікувати точного заповнення коду для складних випадків із змішаною мовою, наприклад, JavaScript, який створює HTML.Тоді як деякі редактори коду впускають і розглядають HTML як звичайний рядок, WebStorm розпізнає вбудований HTML і аналізує наступний рівень. Доповнення коду JavaScript для ключових слів, міток, змінних, параметрів та функцій WebStorm ґрунтується на DOM, і яке підтримує популярні для браузера методи.

Перегляд і навігація надзвичайно важливі при перегляді коду. WebStorm легко переходить до декларацій та символів, а також знаходить та виділяє символи, міткі та файли.

JavaScript не тільки все ще розвивається, але він все ще містить різні реалізації в різних браузерах та інших середовищах. WebStorm дозволяє встановити версію мови JavaScript, як показує сумісність браузера з поточним вибором.

JavaScript, звичайно, є динамічно типізованою інтерпретованою мовою. Щоб додати ліпшу перевірку типу, дехто вважає за доцільніше писати на TypeScript. WebStorm підтримує TypeScript і включає компілятор TypeScript. Для досягнення компактнішого коду, деякі програмісти вибирають писати на CoffeeScript. WebStorm також підтримує такий функціонал і навіть додає підтримку налагодження вихідних карт для відстеження коду TypeScript, CoffeeScript або ECMAScript 6, який перекладається на JavaScript.

Перевірки коду, вбудовані в WebStorm, охоплюють багато поширених проблем JavaScript, а також такі, як у Dart, EJS, HTML, Інтернаціоналізація, Less, Sass, XML, XPath та XSLT. WebStorm включає JSHint (який рекомендує команда jQuery), JSLint та інші основні перевірки статичного коду JavaScript.

Налаштування Node.js для інших продуктів часто вимагає болісного сеансу в командній оболонці. WebStorm також дозволяє налагоджувати та профілювати програми Node.js та автоматично заповнювати членів класу CommonJS.

Окрім налагодження програм Node.js, WebStorm може відлагоджувати код JavaScript, що працює у Mozilla Firefox або Google Chrome. Він надає точки останову в HTML-файлах, а також файли JavaScript, а також дозволяє налаштувати властивості точки останову. Він

показує кадри, змінні та слідкує за змінними в інтерфейсі налагоджувача, а також забезпечує оцінку часу виконання виразів JavaScript (та вкладку елементів у Google Chrome).

Під час налагодження функція під назвою LiveEdit дозволяє змінювати код і змусити зміни негайно поширитись у браузер, де виконується сесія відлагодження. щоекономить час і допоможе уникнути поширеної проблеми, намагаючись з'ясувати, чому зміна нічого не містить, лише виявивши, що забули оновити веб-переглядач [24].

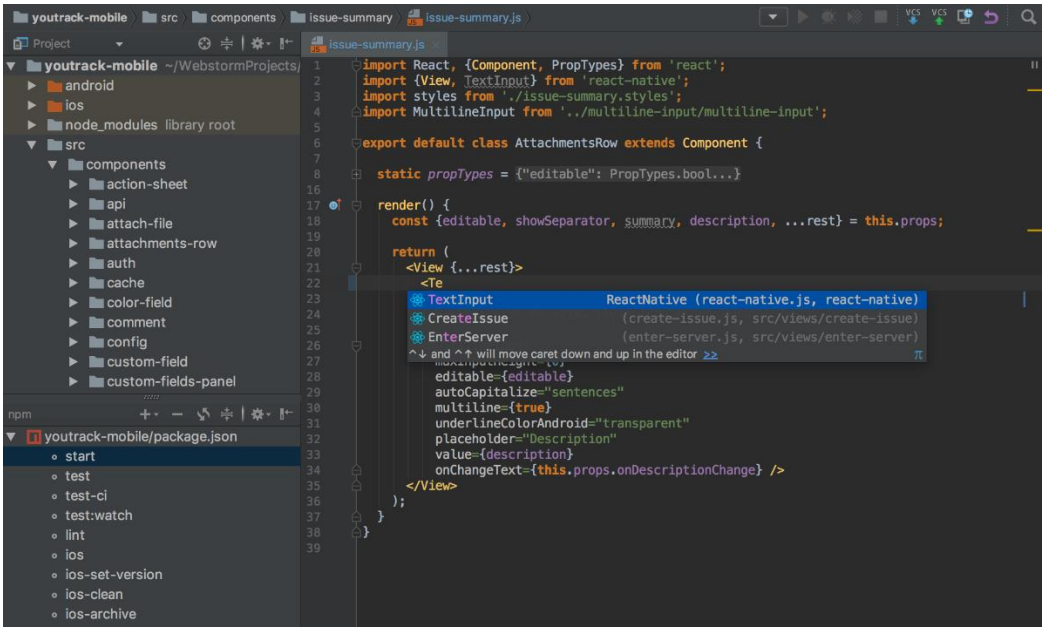


Рисунок 4.6 – Вигляд JetBrains WebStorm

На високому рівні WebStorm має достатньо документації для початку роботи та відповіді на основні питання. Однак, під час детальної роботи, можна натрапити на застарілі публікації блогу. У деяких випадках простіше експериментувати з програмою та, якщо потрібно, відновити свої файли, ніж шукати деталі того, як все мало б працювати.

Загалом, WebStorm –добрий вибір для серйозних розробників JavaScript / HTML5 / CSS, які хочуть повнофункціональне IDE ,мають

право на безкоштовну копію або невеликий бюджет на ліцензію. Однак, якщо також пишете багато коду на стороні сервера, який не є JavaScript, можливо, захочете дослідити IDE, яке підтримує серверні мови та в бази даних, а також JavaScript. Якщо вам дійсно не потрібно IDE, розгляньте Sublime Text або інший редактор з хорошим підсвічуванням синтаксису JavaScript.

4.2.7 Обґрунтування вибору інтегрованого середовища розробки

Хоча Eclipse не може впоратися із сучасними проектами Angular and React власними силами, він, як завжди, містить велику кількість плагінів. Для Angular і TypeScript можна додати Angular IDE (CodeMix), а для проектів React з файлами JSX можна додати TypeScript IDE. За допомогою одного або обох цих плагінів, Eclipse стає корисним для сучасних проектів JavaScript, хоча все ще не дуже швидкий.

Komodo IDE можна рекомендувати, якщо користуєтеся JavaScript у поєднанні з PHP, Perl, Python або Ruby, або займаєтесь великою роботою в Ajax, або пишете багато регулярних виразів. Інакше це не зовсім відповідає стандартам, встановленим WebStorm та Visual Studio Code.

NetBeans, хоч і безкоштовний, але не надто швидкий. Більше підходить, якщо користуватися JavaScript спільно з технологіями сервера Java.

Visual Studio дуже-дуже хороший як JavaScript IDE, але він важкий і ресурсомісткий. Зручний для розробки додатків для підприємств, що включають багато JavaScript. Якщо, окрім написання JavaScript або TypeScript користуються .Net або .Net Core для серверної частини, комунікації з базами даних, розгортанням контейнерів, службами Azure або розгортається Azure, то Visual Studio може бути правильним вибором IDE.

Visual Studio Code безкоштовний, легкий продукт, котрий поєднує в собі багато переваг Visual Studio з оболонкою Atom Electron з відкритим кодом, працює на MacOS і Linux, а також Windows, і має надійну екосистему плагінів.

WebStorm – JavaScript IDE, рекомендоване серйозним розробникам JavaScript / HTML5 / CSS, які хочуть повнофункціональне IDE і мають право на безкоштовну копію або мають певний бюджет на ліцензію. Завдяки чудовим можливостям редагування, аналізу коду, сильній інтеграції ALM та підтримці провідних фреймворків JavaScript, WebStorm перевіряє всі вікна для професійних розробників JavaScript.

Враховуючи всі вищеописані чинники, у процесі написання магістерської роботи скористалися середовищем Visual Studio Code, котренадає можливість розширення через велику екосистему плагінів, витрачає мало ресурсів комп'ютера та працює на усіх популярних операційних системах. Найліпша альтернативою – WebStorm, який набагато більше навантажує систему і займає більше часу при запуску проекту. Враховуючи, що магістерська розроблялась однією людиною, а не групою людей, більшість професійного функціоналу WebStorm є надлишковою.

4.3 Висновки до розділу

У спеціальній частині описано, що таке інтегроване середовище розробки, наведено основні переваги IDE, а також основні принципи, за якими відрізняють IDE від редакторів коду чи консолі. Проаналізовано переваги та недоліки інтегрованого середовища.

Здійснено огляд найпопулярніших IDE та редакторів коду для JavaScript, які застосовують сьогодні. Описано такі IDE, як: Eclipse, Visual

Studio, Komodo IDE, WebStorm, Apache NetBeans та редактор коду Visual Studio Code, який зараз дуже популярниий . Описано коротко характеристики наведених продуктів та коротко розглянуто основні переваги та недоліки для кожного із них.

Обґрунтовано вибір інтегрованого середовища розробки Visual Studio Code. Описано основні особливості та переваги даного IDE. Оскільки дане середовище розробки є легким і швидко запускається, і легко розширюється, воно чудово підходить для розробки даного проекту.

5 ОБҐРУНТУВАННЯ ЕКОНОМІЧНОЇ ЕФЕКТИВНОСТІ

Метою дипломної роботи магістра є дослідження та розробка веб-застосунку для візуалізації та аналізу мережі друзів за допомогою графів, яка складається з отримання даних про друзів з мережі Facebook та візуалізації у веб-застосунку засобами D3.js.

5.1 Розрахунок норм часу на виконання науково-дослідної роботи

Ефективне використання часу має велике значення тому, що коефіцієнт корисної дії залежить від оптимального використання часу.

Розробку дипломної роботи магістра поділено на декілька етапів, що дозволяє полегшити і структурувати побудови веб-додатку та супровідних програм.

Основні етапи при виконанні побудові веб-додатку для аналізу і візуалізації мережі друзів, наступні:

1. Підготовка опису задачі.

2. Збір необхідної інформації про веб-скрейпинг мережі Facebook, види візуалізації графів у веб-застосунках та бібліотеки для розробки веб-застосунку.

3. Розробка структурної схеми веб-застосунку для візуалізації та аналізу графу друзів у Facebook.

4. Вибір мови програмування та бібліотек для побудови веб-скрапера, візуалізатора графу та створення веб-застосунку.

5. Розроблення веб-скрапера для мережі Facebook.

6. Створення веб-застосунку для імпорту даних про граф друзів.

7. Розроблення візуалізації для графу.

8. Вбудовування візуалізації для графу у веб-застосунок.

9. Представлення різних можливостей візуалізації і визначення найбільш вдалих методів їх застосування.

Для оцінки тривалості виконання окремих етапів робіти використовують нормативи часу.

Виконавцем усіх операцій по побудові веб-скрапера та веб-застосунку являється програміст.

Витрати часу по окремих операціях технологічного процесу відображені в таблиці 5.1.

Таблиця 5.1 – Операції технологічного процесу та їх час виконання

№ п/п	Назва операції (стадії)	Виконавець	Середній час виконання операції, год.
1.	Підготовка опису задачі.	Програміст	12
2.	Збір необхідної інформації про веб-скрейпинг мережі Facebook, види візуалізації графів у веб-застосунках та бібліотеки для розробки веб-застосунку.	Програміст	30
3.	Розробка структурної схеми веб-застосунку для візуалізації та аналізу графу друзів у Facebook.	Програміст	8
4.	Вибір мови програмування та бібліотек для побудови веб-скрейпера, візуалізатора графу та створення веб-застосунку.	Програміст	10
5.	Розроблення веб-скрейпера для мережі Facebook.	Програміст	10

6.	Створення веб-застосунку для імпорту даних про граф друзів.	Програміст	48
7.	Розроблення візуалізації для графу.	Програміст	56
8.	Вбудовування візуалізації для графу у веб-застосунок.	Програміст	10
9.	Представлення різних можливостей візуалізації і визначення найбільш вдалих методів їх застосування.	Програміст	10
Разом			194

Загальні затрати часу на реалізацію даної роботи становить 194 години, найбільше часу витрачено на розроблення візуалізації для графу: 56 годин.

5.2 Визначення витрат на оплату праці та відрахувань на соціальні заходи

Відповідно до Закону України “Про оплату праці” заробітна плата – це “винагорода, обчислена, як правило, у грошовому виразі, яку власник або уповноважений ним орган виплачує працівникові за виконану ним роботу”.

Розмір заробітної плати залежить від складності та умов виконуваної роботи, професійно-ділових якостей працівника, результатів його. Заробітна плата складається з основної та додаткової оплати праці.

Основна заробітна плата нараховується за виконану роботу за тарифними ставками, відрядними розцінками чи посадовими окладами.

Додаткова заробітна плата – це складова заробітної плати працівників, до якої включають витрати на оплату праці, не пов’язані з

виплатами за фактично відпрацьований час. Нараховують додаткову заробітну плату залежно від досягнутих і запланованих показників, кваліфікації виконавців. Джерелом додаткової оплати праці є фонд матеріального стимулювання, який створюється за рахунок прибутку.

При розрахунку заробітної плати кількість робочих днів у місяці слід в середньому приймати – 24,5 дні/міс., або ж 196 год./міс. (тривалість робочого дня – 8 год.).

Місячний оклад кожного працівника слід враховувати згідно існуючих на даний час тарифних окладів. Згідно закону України «Про Державний бюджет України на 2020 рік», зокрема Статтею восьмою мінімальна заробітна плата у погодинному розмірі встановлена у розмірі 25,13 грн. Рекомендовані тарифні ставки: керівник дипломної роботи – 30,00...50,00 грн./год., інженер – 25,13...30,00 грн./год., програміст – 25,13...30,00 грн./год., консультант – 25,13...30,00 грн./год., технік – 25,13...30,00 грн./год., лаборант – 25,13...25,00 грн./год.

Основна заробітна плата розраховується за формулою:

$$З_{осн.} = Т_с \cdot К_г \,, \tag{5.1}$$

де $Т_с$ – тарифна ставка, грн.; $К_г$ – кількість відпрацьованих годин.

Оскільки всі види робіт в виконує інженер, то основна заробітна плата буде розраховуватись тільки за однією формулою

$$З_{осн.} = 25{,}13 \cdot 194 = 4875{,}22 \text{ грн.}$$

Додаткова заробітна плата становить 10–15 % від суми основної заробітної плати.

$$З_{дод.} = З_{осн.} \cdot К_{допл.}, \quad (5.2)$$

де $К_{допл}$ – коефіцієнт додаткових виплат працівникам, 0,1–0,15 (візьмемо його рівним 0,15).

$$З_{дод} = 4875{,}22 \cdot 0{,}15 = 731{,}28 \text{ грн.}$$

Звідси загальні витрати на оплату праці ($В_{о.п.}$) визначаються за формулою:

$$В_{о.п.} = З_{осн.} + З_{дод.} \quad (5.3)$$

$$В_{о.п.} = 4875{,}22 + 731{,}28 = 5606{,}50 \text{ грн.}$$

Крім того, слід визначити відрахування на соціальні заходи:

- єдиний соціальний внесок ЄСВ (прибутковий податок) – 22%;
- військовий збір – 1,5%.

У сумі зазначені відрахування становлять 23,5 %.

Отже, сума відрахувань на соціальні заходи буде становити:

$$В_{с.з.} = Ф_{оп} \cdot 0{,}235 \quad (5.4)$$

де $Ф_{оп}$ – фонд оплати праці, грн.

$$В_{с.з.} = 5606{,}50 \cdot 0{,}235 = 1317{,}52 \text{ грн.}$$

Проведені розрахунки витрат на оплату праці наведено у таблицю 5.2.

Таблиця 5.2 – Розрахунки витрат на оплату праці

з/п	Категорія працівників	Основна заробітна плата, грн.			Додаткова заробітна плата, грн.	Нарахув. на ФОП, грн.	Всього витрати на плату праці, грн. (6=3+4+5)
		Тарифна ставка, грн.	Кількість відпрацьова них год.	Фактично нарах. з/пл., грн			
А	Б	1	2	3	4	5	6
1.	програміст	25,13	194	4875,22	731,28	1317,52	6924,02

З таблиці розрахунки витрат на оплату праці видно що всього витрати на плату праці становить 6924,02 грн.

5.3 Розрахунок матеріальних витрат

Матеріальні витрати визначаються як добуток кількості витрачених матеріалів та їх ціни:

$$M_{ві} = q_i \cdot p_i \,, \tag{5.5}$$

де: q_i – кількість витраченого матеріалу і–го виду; p_i – ціна матеріалу і–го виду.

Звідси, загальні матеріальні витрати можна визначити:

$$З_{м.в.} = \sum M_{ві} \,, \tag{5.6}$$

Розрахунки занесемо у таблицю 5.3.

Таблиця 5.3 – Розрахунки матеріальних витрат

Найменування матеріальних ресурсів	Один. виміру	Норма витрат	Ціна за один., грн.	Затрати матер., грн.	Транс-портно-заготівель-ні витрати, грн.	Загальна сума витрат на матер., грн.
1. Основні матеріали..						
Використання мережі Internet	години	162	–	160	–	160
2. Допоміжні витрати						
Папір формату А4	шт.	200	0,18	45	–	45
Разом:						205

Загальні матеріальні витрати на Internet і Папір формату А4 становить 205 грн.

5.4 Розрахунок витрат на електроенергію

Затрати на електроенергію 1–ці обладнання визначаються за формулою:

$$З_в = W \cdot T \cdot S, \qquad (5.7)$$

де W – необхідна потужність, кВт; Т – кількість годин на реалізацію розробки; S – вартість кіловат-години електроенергії.

Вартість кіловат-години електроенергії слід приймати згідно існуючих на даний час тарифів. Отже, 1 кВт з ПДВ коштує 2,42 грн.

Потужність комп’ютера для створення дипломної роботи – 400 Вт, кількість годин роботи обладнання згідно таблиці 6.1 –194 години.

Тоді,

$$З_в = 0{,}4 \cdot 194 \cdot 2{,}42 = 187{,}79 \text{ грн.}$$

Згідно формули затрати на електроенергію де необхідна потужність множиться на кількість годин на реалізацію розробки і множиться на вартість кіловат-години електроенергії що в висновку дорівнює 187,79 грн.

5.5 Розрахунок суми амортизаційних відрахувань

Характерною особливістю застосування основних фондів у процесі виробництва є їх відновлення. Для відновлення засобів праці у натуральному виразі необхідне їх відшкодування у вартісній формі, яке здійснюється шляхом амортизації.

Амортизація – це процес перенесення вартості основних фондів на вартість новоствореної продукції з метою їхнього повного відновлення.

Для визначення амортизаційних використовується формула:

$$A = \frac{Б_В \cdot Н_А}{100\%}, \quad (5.8)$$

де A – амортизаційні відрахування за звітний період, грн.; $Б_В$ – балансова вартість групи основних фондів на початок звітного періоду, грн.; $Н_А$ – норма амортизації.

Комп’ютери та оргтехніка належать до четвертої групи основних фондів. Для цієї групи річна норма амортизації дорівнює 60 % (квартальна – 15 %).

Для даної дипломної роботи засобом розробки є комп’ютер. Його сума становить 17460 грн. Отже, амортизаційні відрахування будуть рівні:

$$A = 17460 \cdot 5\% / 100\% = 873 \text{ грн.}$$

Оскільки робота виконувалась 194 години, то амортизаційні відрахування будуть становити:

$$A = 873 \cdot 194 / 194 = 873 \text{ грн.}$$

Згідно формули для визначення амортизаційних де $Б_В$ множиться $Н_А$ і ділиться на 100% амортизація розробки становить 873 грн.

5.6 Обчислення накладних витрат

Накладні витрати пов’язані з обслуговуванням виробництва, утриманням апарату управління спілкою та створення необхідних умов праці.

В залежності від організаційно-правової форми діяльності господарюючого суб’єкта, накладні витрати можуть становити 20–60 % від суми основної та додаткової заробітної плати працівників.

$$Н_в = В_{о.п.} \cdot 0{,}2 \ldots 0{,}6 \,, \qquad (5.9)$$

де $Н_в$ – накладні витрати.

Отже, накладні витрати:

$$H_в = 5606{,}50 \cdot 0{,}2 = 1121{,}3 \text{ грн.}$$

Накладні витрати згідно розрахунку формули, становить 1121,3 грн.

5.7 Складання кошторису витрат та визначення собівартості науково-дослідницької роботи

Результати проведених вище розрахунків зведемо у таблицю 5.4.

Таблиця 5.4 – Кошторис витрат на НДР

Зміст витрат	Сума, грн.	В % до загальної суми
Витрати на оплату праці	5606,50	60,21
Відрахування на соціальні заходи	1317,52	14,15
Матеріальні витрати	205	2,21
Витрати на електроенергію	187,79	2,03
Амортизаційні відрахування	873	9,36
Накладні витрати	1121,3	12,04
Собівартість	9311,11	100,00

Собівартість ($С_в$) програмного продукту розрахуємо за формулою:

$$С_в = В_{о.п.} + В_{с.з.} + З_{м.в.} + З_в + А + Н_в \text{ .} \qquad (5.10)$$

Отже, собівартість програмного продукту дорівнює:

$$С_в = 5606{,}50 + 1317{,}52 + 205 + 187{,}79 + 873 + 1121{,}3 = 9311{,}11$$ грн.

Загальний кошторис витрат та визначення собівартості науково-дослідницької роботи становить 9311,11 грн.

5.8 Розрахунок ціни програмного продукту

Ціну науково-дослідної роботи можна визначити можна визначити за формулою:

$$Ц = \frac{C_В \cdot (1 + P_{рен}) + К \cdot B_{н.і.}}{К} \cdot (1 + ПДВ), \quad (5.11)$$

де $P_{рен.}$ – рівень рентабельності, 30 %; $К$ – кількість замовлень, од. (встановлюється лише при розробці програмного продукту та мікропроцесорних систем); $B_{н.і.}$ – вартість носія інформації, грн. (встановлюється лише при розробці програмного продукту); $ПДВ$ – ставка податку на додану вартість, (20 %).

Оскільки розробка є прикладною, і використовуватиметься тільки для одного підприємства, то для розрахунку ціни не потрібно вказувати коефіцієнти К та Ві.н , оскільки їх в даному випадку не потрібно.

Тоді, формула для обчислення ціни розробки буде мати вигляд:

$$Ц = C_В \cdot (1 + P_{рен}) \cdot (1 + ПДВ) \quad (5.12)$$

Звідси ціна на роботу складе:

$$Ц = 9311,11 \cdot (1 + 0,3) \cdot (1 + 0,2) = 14525,33 \text{ грн.}$$

Загальний розрахунок ціни програмного продукту становить 14525,33 грн

5.9 Визначення економічної ефективності і терміну окупності капітальних вкладень

Ефективність виробництва – це узагальнене і повне відображення кінцевих результатів використання робочої сили, засобів та предметів праці на підприємстві за певний проміжок часу.

Економічна ефективність (E_p) полягає у відношенні результату виробництва до затрачених ресурсів:

$$E_p = \frac{П}{C_B}, \tag{5.13}$$

де $П$ – прибуток; C_B – собівартість.

Плановий прибуток ($П_{пл}$) знаходимо за формулою:

$$П_{пл} = Ц - C_в . \tag{5.14}$$

Розраховуємо плановий прибуток:

$$П_{пл} = 14525,33 - 9311,11 = 5214,22 \text{ грн.}$$

Отже, формула для визначення економічної ефективності набуде вигляду:

$$E_p = \frac{П_{пл}}{C_в}. \quad (5.15)$$

Тоді,

$$E_p = 5214{,}22 / 9311{,}11 = 0{,}56.$$

Поряд із економічною ефективністю розраховують термін окупності капітальних вкладень (T_p):

$$T_p = \frac{1}{E_p}, \quad (5.16)$$

Термін окупності дорівнює:

$$T_p = 1 / 0{,}56 = 1{,}8 \text{ р.}$$

Згідно формул плановий прибуток від розробки становить 5214,22 грн., економічна ефективність дорівнює 0,56 а термін окупності становить 1,8 роки що вважається доцільним та економічно вигідним.

5.10 Висновки до обґрунтування економічної ефективності

В організаційно-економічній частині дипломної роботи освітнього рівня «магістр» було розраховано основні техніко-економічні показники для побудови веб-застосунку для візуалізації та аналізу мережі друзів (див. таблиця 5.5).

Орієнтоване значення економічної ефективності становить 0,56 що є достатньо високим значенням.

Період окупності повинен варіюватися від 1 до 3 років, тоді розвиток вважається доцільним та економічно вигідним. Термін окупності даної роботи становить 1,8 років.

Таблиця 5.5 – Техніко-економічні показники науково-дослідної роботи

№ п/п	Показник	Значення
1.	Собівартість, грн.	9311,11
2.	Плановий прибуток, грн.	5214,22
3.	Ціна, грн.	14525,33
4.	Економічна ефективність	0,56
5.	Термін окупності, рік	1,8

Отже, ця робота може бути реалізована та розвинена, оскільки вона є економічно вигідною для всіх основних технічних та економічних показників.

6 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

6.1 Інформаційна культура як важлива складова в управлінні охороною праці

Наслідком інформатизації та збільшення цінності інформації стала поява нового типу відносин між одиницями суспільства, які були опосередковані загальним інформаційним полем. Віртуалізація соціальних процесів під впливом науково-технічного прогресу призвела до появи нового типу стосунків, який має за основу глобальні телекомунікаційні мережі.

У теперішньому суспільстві виникає необхідність розвитку певного рівня інформаційної культури особистості, яка проявляється у різних сферах соціальної життєдіяльності та формує основу особистісної успішності суб'єкта соціальних відносин у соціальному середовищі. До одного із основних завдань соціалізації особистості відносять формування здатності керувати інформацією, орієнтуватися в ній та створювати на основі результатів власної індивідуальної активності знання, які мають певний смисл та суспільну цінність. Інформаційна культура включає в себе ряд соціально-професійних інваріантів, які використовуються людиною у процесі розв'язання як професійних так і побутових завдань. На даний момент, за результатами дослідження психографічних описів більшості професій соціономічного та технономічного циклу, можна зробити висновок, що інформаційні компетентності стають інваріантною складовою частиною професії. Як наслідок без якісного володіння ними суб'єкт соціальних відносин не зможе ефективно виконувати трудові

обов’язки та здійснювати інші види соціальної активності. Інформаційна культура, яка акумулює в собі ряд провідних інформаційних компетентностей, визначає не лише професійну успішність особистості, але й її соціальну життєздатність.

Рисунок 6.1 – Модель рівнів інформаційної культури особистості

Нижче наведена структура інформаційної культури особистості, яка складається з двох основних рівнів – технологічного та соціально-психологічного [26]. На кожному рівні розташовані по два етапи взаємодії особистості з інформаційною мережею у процесі професійної, дослідницької та інших видів соціальної активності.

Для підвищення ефективності системи управління охорони праці (СУОП) надважливою є роль формування і розвитку інформаційної культури фахівців ІТ-технологій, яка впливає на удосконалення інформаційного контуру сучасних підприємств, дозволяє створювати надійні прогнози щодо стану умов праці, показників здоров’я та працездатності, виробничого травматизму і професійної захворюваності, визначати політику розвитку підприємств, установ та організацій на основі різноманітних стратегій охорони праці. Поряд з інформаційною культурою

важливо використовувати в рамках СУОП «трикутник» її складових: правову (8,1 за 10-бальною шкалою), організаційну (8,0), управлінську (7,5). В управлінні охороною праці потрібно реалізувати основні положення, окремі теоретико-методологічні підходи інформаційного менеджменту. Головну роль та відповідальність за стан СУОП мають нести фахівці служби охорони праці сучасного підприємства. Сучасне суспільство називають інформаційним, оскільки йдеться про багатосторонні і кардинальні зміни у розвитку цивілізації. Інформаційне суспільство передбачає докорінну зміну, яка полягає у перетворенні інформації і знань у головний професійно-виробничий потенціал особистості, соціуму і держави.

Під інформаційною культурою розуміють сукупність, складову НІТ (новітні інформаційні технології), технологічну, правову, психологічну, соціологічну та ергономічну підсистеми, що сприяють спрямованому впливу на протікання соціальних процесів у суспільстві, колективі і вихованню свідомого відношення людини до праці, виконання прав та обов’язків [27]. Саме поняття інформаційної культури виникло в процесі активізації дослідницької уваги до механізмів інформаційного обміну у зв’язку зі значним підвищенням ролі інформації в соціальних процесах суспільства, яке розглядають як інформаційне суспільство знань, де в центрі знаходяться інформаційні технології. Є три принципові причини, в силу яких сьогодні необхідно дбати про інформаційну культуру компанії. По-перше, вона з часом стає найважливішою частиною загальної організаційної (корпоративної) культури компанії. Більшість компаній розуміють необхідність перетворень, орієнтованих на задоволення очікувань споживача. По-друге, інформаційні технології роблять можливим створення в компаніях комп'ютерних мереж, завдяки яким йде спілкування між менеджерами. Саме по собі створення такої мережі з

усіма її робочими станціями і мультимедійними можливостями не гарантує того, що інформація буде використовуватися розумніше та ефективніше. По-третє, для різних функціональних служб, підрозділів та робочих груп сучасних підприємств в сфері охорони праці інформаційна культура різна, а це означає відмінність методологічних підходів до процесів усвідомлення, збору, організації, обробки, поширення і використання інформації. Тому багато менеджерів погодяться з тим, що корпоративна інформаційна культура важлива для вироблення різних стратегій охорони праці та запровадження відповідних заходів з її вдосконалення. Поряд із задачами і здобутками окреслилися негативи використання інформаційних технологій:

1) надмірне інформаційне навантаження, суть якого полягає у тому, що кількість корисної інформації, яка надходить до мережі, перевищує психофізіологічні можливості її сприйняття людиною;

2) велика кількість інформації, яка сприймається, але не є корисною для фахівців в даний момент;

3) інформаційний голод, причиною якого є саме надлишок інформації, викликаний інформаційним перенавантаженням;

4) «інформоманія» як хвороба людини, яка робить останню знеособленою, залежною від перебування в інформаційному просторі і роботи з комп'ютером і чому вона віддає перевагу, уникаючи «живого» спілкування з людьми;

5) поява «кіберспільнот», що за своїми соціокультурними характеристиками набагато ближчі до представників інших культур у глобальному інформаційному просторі, ніж до своєї етнонаціональної спільноти чи решти населення, не охопленого Інтернетом;

6) індивідуалізм і дегуманізація способу життя «мешканців» Інтернету –відсутність готовності ділитися своїми знаннями.

6.2 Охорона праці в системі «людина-комп’ютер-середовище»

В сучасних умовах взаємодія людини з технікою значно ускладнилась, що вимагає комплексного підходу, який передбачає розгляд людини, технічних засобів праці та виробничого середовища, як взаємозв'язаних елементів єдиної системи. Все вищесказане в повній мірі відноситься й до системи «людина – комп'ютер – середовище». Спрощена модель, що ілюструє основні моменти взаємодії в цій системі наведені на рисунку 6.2.

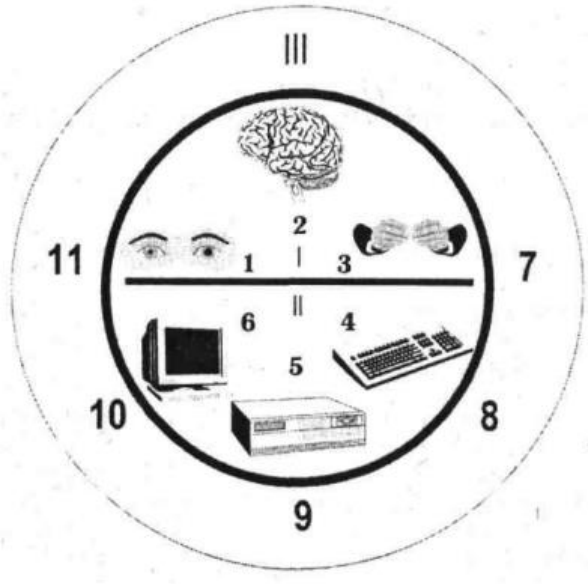


Рисунок 6.2 – Спрощена модель системи «людина – комп'ютер – середовище»

Як видно із рисунка, I – людина; II – комп'ютер; III – виробниче середовище; 1 – органи зору; 2 – опрацювання інформації (аналізування та прийняття рішень; 3 – керування; 4 – органи керування; 5 – комп'ютерні

операції; 6 – дисплей; 7 – мікроклімат; 8 – шум і вібрація; 9 – шкідливі речовини у повітрі; 10 – випромінювання; 11 – освітлення.

Таким чином, без необхідних знань про людину неможливе створення ефективних комп'ютеризованих систем та забезпечення оптимальних умінь праці [12].

Людина взаємодіє з технічними засобами праці, прагне уникнути несприятливого впливу з боку останніх. Тому необхідно створити такі умови праці та техніку, які б водночас із високою працездатністю людини, забезпечували ще й необхідну зручність у роботі, зберігали здоров'я, сили, професійне довголіття. При цьому необхідно враховувати реальні можливості людини, її антропометричні, фізіологічні та психологічні особливості.

Таким чином, без необхідних знань про людину неможливе створення ефективних комп'ютеризованих систем та забезпечення оптимальних умінь праці. Людина взаємодіє з технічними засобами праці, прагне уникнути несприятливого впливу з боку останніх. Тому необхідно створити такі умови праці та техніку, які б водночас із високою працездатністю людини, забезпечували ще й необхідну зручність у роботі, зберігали здоров'я, сили, професійне довголіття. При цьому необхідно враховувати реальні можливості людини, її антропометричні, фізіологічні та психологічні особливості.

Ефективна робота системи «людина-комп'ютер–середовище» значною мірою залежить від врахування фізіологічних можливостей та особливостей опорно-рухової системи людини. Зокрема, максимальна кількість рухів пальців руки за хвилину становить 380, кисті – 180, руки – 90, тулуба – 30. Рухи в горизонтальній площині менше втомлюють ніж у вертикальній, а рухи від тулуба точніші, ніж до тулуба [13].

Час сприйняття сигналів органами зору складає – 0,15-0,25 с, слуху – 0,10-0,20 с, відчуття – 0,10-0,25 с, болю – 0,15-0,90 с, температури – 0,25-1,60 с. Однак в умовах психологічного напруження цей час може бути значно збільшений.

Найбільшу небезпеку несе підвищене значення напруги, що подається з електромережі на блок живлення системного блоку. Несприятливий вплив на користувача може здійснювати шум, що створюється при роботі вентиляторів та накопичувачів системного блоку. Електромагнітні випромінювання, які виникають при роботі електронних компонентів блоку мають незначні рівні, тому можуть створювати хіба що радіочастотні перешкоди.

Дисплей. Дисплеї, сконструйовані на основі ЕПТ, є джерелом кількох видів електромагнітного випромінювання та полів, а саме:

- іонізуючого (рентгенівського) випромінювання;
- оптичного випромінювання;
- електромагнітних випромінювань та полів.

Необхідні концентрації позитивних та негативних іонів в повітрі робочих зон можна забезпечити застосуванням:

- генераторів негативних іонів;
- установок штучного зволоження;
- кондиціонерів;
- примусової вентиляції (провітрювання, системи загально обмінної припливно-витяжної вентиляції, пристрої місцевої вентиляції);
- захисних екранів, що заземлені.

Оптичне випромінювання виникає в результаті взаємодії електронів з шаром люмінофору. Дослідження показали, що в процесі роботи дисплея,

окрім видимого випромінювання, мають місце і інші види оптичного випромінювання (ультрафіолетове та інфрачервоне) [28].

Електромагнітні випромінювання та поля різних діапазонів частот (високих, низьких та вкрай низьких) виникають в системах горизонтальної та вертикальної розгортки та в результаті дії електронного променя.

Електростатичний заряд зосереджується переважно на ЕПТ дисплея. Він виникає у зв'язку з високим потенціалом розгону електронів та провідністю поверхні екрана. Основним джерелом статичних електричних полів є висока напруга (6-15 кВ), яка подається на блок анодів та внутрішню поверхню екрана ЕПТ.

Клавіатура. Практично всі клавіатури введення даних в світі побудовані на тому, що натиснення клавіші визначається шляхом сканування матриці клавіатури. При такому принципі роботи клавіатура випромінює електромагнітні поля низьких рівнів. Проте велика тривалість використання клавіатури протягом робочої зміни та невелика відстань нервових закінчень пальців користувача від джерел електромагнітного випромінювання – контактів клавіш, можуть бути причиною несприятливого впливу на його здоров'я.

Розглянемо шкідливу дію для кожного виду принтерів:

- Матричні принтери. Оскільки матричні принтери засновані на ударному принципі дії, то основним несприятливим фактором, що впливає на користувача при їх роботі є шум. Окрім того, їх використання підвищує запиленість повітряного середовища на робочому місці частинками паперу та барвника.

- Лазерні принтери. Можна виділити такі шкідливі та небезпечні фактори за порядком їх проявлення в процесі друкування: електростатичні заряди, лазерний промінь, висока температура. Самі електростатичні заряди, на використанні яких заснована робота лазерних принтерів,

особливої небезпеки для користувача не представляють, оскільки, заряджені ними елементи принтера захищені корпусом і контакт користувача з ними – неможливий. Найбільша небезпека, пов'язана з електростатичними зарядами, полягає у тому, що вони призводять до утворення озону. Для виключення виділення лазерними принтерами озону в сучасних моделях встановлюються озонові фільтри. Однак, слід зазначити, що через певний проміжок часу, визначений фірмою – виробником, фільтр втрачає свої захисні властивості і підлягає заміні. Варто також нагадати, що зменшенню концентрації озону сприяє регулярне провітрювання приміщення. Лазерний промінь, потрапивши на тіло чи орган зору користувача, може призвести до небажаних наслідків. Щоб уникнути цього, в принтері встановлено лазерний блок закритого типу. У випадку поломки принтера його ремонт здійснюють лише спеціалісти, які мають відповідну кваліфікацію та повноваження. Необхідність високої температури для фіксування тонера на папері призводить до підвищеного тепловиділення. Тому у лазерних принтерах вмонтовані вентилятори. Звичайно, робота лазерного принтера не може помітно вплинути на підвищення температури в приміщенні, однак, вплинути на підвищення запиленості повітря – може. Зокрема, при друкуванні на папері з низькою поверхневою міцністю відбувається відрив целюлозних волокон та частинок наповнювача, що входять до складу паперу. Окрім того, на таких сортах паперу погано закріплюється тонер. Таким чином, частинки паперу і тонера, підхоплені потоком повітря, потрапляють у виробниче середовище, підвищуючи його запиленість. Тепер стає зрозумілим, чому для роботи лазерного принтера без шкідливого ефекту пиління важливо використовувати сорти паперу, що зазначені в технічному паспорті.

- Струменеві принтери. Шкідливий вплив струменевих принтерів визначається, в першу чергу, шкідливістю компонентів, що входять до складу чорнила. Так, чорнило для струменевих принтерів компанії Epson на 90% складається з води, а решта – нетоксичні спирт та барвник (за даними виробника).

Пристрої безперебійного живлення (ПБЖ). Найбільш небезпечними та шкідливими факторами, пов'язаними з роботою ПБЖ є:

- можливість ураження електричним струмом;
- вплив електромагнітного випромінювання.

Щодо першого фактора, то в ПБЖ застосовано низку засобів, спрямованих на підвищення рівня електробезпеки. Зокрема, ПБЖ буде працювати лише у випадку, коли він підключений до розетки з заземленням.

Як відомо, інтенсивність електромагнітного випромінювання обернено пропорційна квадрату відстані до джерела. Тому, чим далі від користувача знаходиться ПБЖ, тим менша імовірність впливу електромагнітного випромінювання, що ним генерується.

З метою профілактики несприятливого впливу електромагнітного випромінювання від ВДТ на користувача необхідно:

- встановити на робочому місці відеотермінал, що відповідає сучасним вимогам стосовно захисту від випромінювань (MPR-II або TCO-95);
- встановити на ВДТ старої конструкції заземлений приекранний фільтр (незаземлений захисний екран відіграє лише декоративну роль щодо захисту від електромагнітного випромінювання);
- не переобтяжувати приміщення значною кількістю робочих місць з ВДТ;

- не концентрувати на робочому місці великої кількості радіоелектронних пристроїв;
- вимикати ВДТ, якщо на ньому не працюють, однак знаходяться неподалік від нього.

Електробезпека. Під час монтажу та експлуатації ліній електромережі необхідно повністю унеможливити виникнення електричного джерела загорання внаслідок короткого замикання та перевантаження проводів.

6.3 Забезпечення безпеки життєдіяльності при роботі з ПК

Заходи щодо усунення небезпеки ураження електричним струмом зводяться до правильного розміщення устаткування та електричних кабелів. Інші заходи щодо забезпечення електробезпеки, збігаються з загальними заходами пожежо- та електробезпеки.

В якості профілактичних заходів для забезпечення пожежної безпеки слід використовувати скриту електромережу, надійні розетки з пожежобезпечних матеріалів, силові мережі живлення устаткування виконувати кабелями, розрахованими на підключення в 3-5 разів більшого навантаження, включати й виключати живлення обладнання за допомогою штатних вимикачів. Треба регулярно робити очистку внутрішніх частин комп'ютерів, іншого устаткування від пилу, розташовувати комп'ютери на окремих неспалюваних столах. Для запобігання іскріння необхідно рідше встромляти і виймати штепсельні вилки з розеток [30].

Екран дисплея повинен бути розташованим перпендикулярно до напрямку погляду. Якщо він розташований під кутом, то стає причиною сутулості. Відстань від дисплея до очей повинна трохи перевищувати

звичну відстань між книгою та очима. Перед екраном монітора, особливо старих типів, повинен бути спеціальний захисний екран. При його відсутності треба сидіти на відстані витягнутої руки від монітора. Ще одним моментом, який стосується зору, є необхідність створення неоднорідного поля зору. Для цього можна розвісити на поверхнях (стінах) плакати та картини, виконані у спокійних тонах. Наприклад, пейзажі.

Важливою є форма спинки крісла, яка повинна повторювати форму спини. Висота крісла повинна бути такою, щоб користувач не почував тиску на куприк або стегна. Крісло бажано обладнати бильцями. Його потрібно встановити так, щоб не треба було тягтися до клавіатури. Періодично користувачу необхідно рухатися, вчасно змінювати положення тіла і робити перерви у роботі.

При напруженій роботі за комп'ютером щогодини необхідно робити перерву на 15 хвилин через кожну годину і треба займатися іншою справою. Декілька разів на годину бажано виконувати серію легких вправ для розслаблення.

Наслідками регулярної роботи з комп'ютером без застосування захисних засобів можуть бути: захворювання органів зору (60% користувачів); хвороби серцево-судинної системи (20%); захворювання шлунково-кишкового тракту (10%); шкірні захворювання (5%); різноманітні пухлини.

Режим праці та відпочинку при роботі з персональною електронно-обчислювальною машиною (ПЕОМ) залежить від категорії трудової діяльності. Всі роботи з ПЕОМ ділять на три категорії. Перша - епізодичне зчитування і робота з інформацією не більше 2-х годин за 8-годинну робочу зміну. Друга - зчитування інформації або творча робота не більше 4-х годин за восьми годинну зміну. Третя – зчитування інформації або творча робота тривалістю більше 4-х годин за зміну.

Якщо у приміщенні експлуатується більше одного комп'ютера, то треба врахувати, що на користувача одного комп'ютера можуть впливати випромінювання від інших, в першу чергу бокових, а також і задньої стінки сусіднього дисплея. Тому необхідний захист спеціальними фільтрами і щоб користувач розміщався від бічних і задніх стінок інших дисплеїв на відстані не ближче одного метра.

Отже, щоб запобігти негативним впливам необхідно знати й небезпечні сторони самого комп'ютера і правила безпечної роботи, знати засоби запобігання небезпек. Вони пов'язані перед усім із загально відомими небезпечними факторами – поразками електричним струмом, пожежонебезпечністю.

Негативні фактори впливу на здоров'я. Всесвітня організація охорони здоров'я (ВООЗ) роботу з персональним комп'ютером віднесла до небезпечних, бо їй притаманний фактор постійно діючого стреса. З-за цього небезпеці піддаються всі життєво важливі органи людини, з'являється ризик виникнення серйозних хвороб.

Електромагнітні поля біля комп'ютера (особливо низькочастотні) негативно впливають на людину і в першу чергу на її центральну нервову систему, викликаючи головний біль, запаморочення, нудоту, депресію, безсоння, відсутність апетиту, виникнення синдрому стресу. Причому нервова система реагує навіть на короткі за тривалістю впливи слабких полів: змінюється гормональний стан організму, порушуються біоструми мозку. Це призводить до погіршення зору, ускладненню серцево-судинних захворювань, зниженню імунітету, виникають негативні впливи на плин вагітності.

Характерною рисою професії оператора ПК є статичний режим роботи: великий обсяг праці треба виконувати в сидячому положенні. При цьому більшість груп м'язів постійно напружені, що призводить до

швидкої стомлюваності, сприяє розвитку фахових патологічних вигинів хребта: грудному гіперкифозу, сплощенню шийного лордозу і формуванню сколіозів. Неправильне розташування дисплеїв по висоті - занадто низьке або високе, під неправильним кутом - є головною причиною появи сутулості. Занадто високе розташування дисплея призводить до тривалої напруги шийного відділу хребта, що, зрештою, може призвести до розвитку остеохондрозу. Ненормальний стан хребта може стати причиною захворювання всього організму.

Основні "комп'ютерні" хвороби. Нерухома напружена поза оператора призводить до втоми і виникнення болю в хребті, шиї, плечових суглобах. Інтенсивна робота з клавіатурою викликає болючі відчуття в ліктьових суглобах, передпліччях, зап'ястях і пальцях рук.

У деяких операторів персонального комп'ютера (ПК) розвивається м'язова слабкість, відбувається зміна форми хребта (синдром тривалого статичного навантаження - СТСН), що може призвести до непрацездатності. Постійні користувачі ПК найчастіше піддаються психічним стресам, хворобам серцево-судинної системи і верхніх дихальних шляхів. Значному навантаженню піддається зоровий апарат.

При тривалій та інтенсивній роботі за комп'ютером з'являється синдром комп'ютерного стресу (СКС), який проявляється головною біллю, запаленням очей, алергією, дратівливістю, млявістю і депресією, погіршенням зосередженості і працездатності.

Причинами різноманітних симптомів СКС є 5 основних чинників: неправильна робота очей і поза тіла, носіння невідповідних окулярів або контактних лінз, неправильна організація робочого місця, розподілення фізичних, розумових, візуальних навантажень, низький рівень візуальної підготовленості для роботи з комп'ютером. Особливо це характерно для дітей, молодших школярів.

Існує медико-соціальна проблема - комп'ютер і здоров'я дітей. Тепер в світі існує потужна індустрія виробництва комп'ютерних ігор. Діти з великим задоволенням віддають цим гарним і захоплюючим, а часто і агресивним іграшкам свій вільний час. Діти в значно меншому ступені, чим дорослі, спроможні контролювати себе. їхня психіка дуже нестійка, тому надмірне захоплення комп'ютерними іграми може стати причиною дуже важких наслідків - розвивається підвищена збудженість, у школярів знижується успішність, діти стають примхливими, некерованими, перестають будь-чим цікавитися крім комп'ютера.

За своїм впливом на дитячий організм комп'ютерна гра подібна наркотику, тому необхідно дуже строго дозувати час комп'ютерних занять. Для організму дитини важливим є правильний розвиток опірно-рухового апарату, відхилення від норм розвитку якого є хвороби.

Інтенсивна і тривала робота з клавіатурою комп'ютера може стати джерелом важких професійних захворювань рук. Робота з клавіатурою є причиною 12% профзахворювань, викликаних повторюваними рухами.

Захворювання, пов'язані з "травмами повторюваних навантажень", пов'язані з хворобами нервів, м'язів і сухожилів. Наприклад тендит – запалення і набрякання сухожилів. Захворювання поширюється на кисті рук, зап'ястя, плечі. Інше – травматичний епікон-диліт – подразнення сухожилів, що з'єднують м'язи передпліч та ліктьових суглобів.

Хвороба Де-Кервена – різновид тендита, при якій страждають сухожилля, пов'язані з великим пальцем руки.

Тунельний синдром зап'ясного каналу – запалення медіального нерва руки з-за набряку сухожилів, синовиальної оболонки.

Для зменшення шкідливого впливу клавіатури фірма Microsoft розробила ергономічну клавіатуру, що своєрідною формою, конструкцією

знижує навантаження на руки. Творці клавіатури сподіваються, що вона застрахує оператора від тунельного синдрому зап'ястного каналу.

Інший пристрій, який привертає до себе увагу фахівців в області ергономіки - маніпулятор типу "миша". На жаль, навіть найерго-номічніша клавіатура не може цілком вирішити проблему, оскільки причини захворювання "травми повторюваних навантажень" дотепер цілком не виявлені.

Найбільш актуальним є захворювання, яке одержало назву "інтернетоманія", а психологи назвали її маніакальною залежністю від віртуального світу глобальних мереж. Збуджені, почервонілі очі, високий ступінь нервового і фізичного виснаження, сльозоточиве зівання - лише деякі симптоми цієї хвороби.

Сіткоголіки, або інтернетоголіки відчувають жагуче бажання знову і знову занурюватися у світ віртуальної реальності і довго не виходити з нього. На думку психологів інтернетоманія також руйнівна, як і алкоголізм або наркоманія. Вона веде до глибоких змін особистості, самоізоляції, втраті внутрішніх орієнтирів, неурівнова-женості психіки, що зовні часто проявляється розсіяністю і неохайністю, навіть байдужим відношенням до близьких.

6.4 Організація робочого місця користувача відеодисплейним терміналом

Діяльність більшості працівників сучасних професій у виробничій сфері пов’язана з використанням комп’ютерної техніки. Комп’ютер для сучасної людини є такою ж технічною необхідністю, як телевізор або холодильник. Побутові прилади ми використовуємо не задумуючись про їх

шкідливість або нешкідливість, усвідомлюючи лише переваги їх наявності. Щодо комп’ютерів, то існує багато інформації як про їх безпечність, так і шкідливість.

Наказом Держгірпромнагляду від 26.03.2010 р. № 65 затверджено Правила охорони праці під час експлуатації електронно-обчислювальних машин (далі — Правила № 65). Вони поширюються на всіх суб’єктів господарювання незалежно від форм власності, які у своїй діяльності здійснюють роботу, пов’язану з електронно-обчислювальними машинами (ЕОМ) з відеодисплейними терміналами (ВДТ), у т. ч. на тих, які мають робочі місця, обладнані ЕОМ з ВДТ і периферійними пристроями (ПП) [31].

Персональні комп’ютери (ПК) — це теж ЕОМ, тож у статті термін «комп’ютер» і абревіатура «ЕОМ» є синонімами.

Робочий стіл, крісло і інші елементи обладнання робочого місця повинні бути зручними для вас. Так, наприклад, незручне крісло в якому ви сидите багато годин на день, може призвести до розвитку самих різних захворювань.

Рисунок 6.3 – Правильна організація робочого місця

Вимоги до організації робочого місця наведено у розділі 4 ДСанПін 3.3.2.007-98, зокрема:

- робоче місце має бути організоване так, щоб світло падало зліва (для «правшів» і навпаки — для «лівшів»);
- при розміщенні декількох робочих місць в одному приміщенні мінімальна площа для одного робочого місця складає 6 м2;
- екран монітора має бути розміщений на оптимальній відстані від очей користувача (60-70 см), але не ближче 50 см;
- екран має бути розташований для забезпечення комфортного зорового спостереження у вертикальній площині під кутом + 30° до нормальної лінії погляду працівника;
- поверхня клавіатури має бути з антистатичними властивостями;
- під час роботи необхідно робити перерви для розвантаження очей;
- робочі місця з ВДТ і ПЕОМ під час виконання творчої роботи, яка потребує значної розумової напруги чи великої концентрації уваги, слід ізолювати одне від одного перегородкою висотою 1,5-2,0м
- для оздоблення приміщень з ВДТ повинні використовуватися дифузно-відбиваючі матеріали з коефіцієнтами відбиття: стелі – 0,7-0,8; стін – 0,4-0,5; підлоги – 0,2-0,3.

6.5 Висновок до розділу охорони праці та безпеки в надзвичайних ситуаціях

У першому пункті розглянуто інформаційну культуру в процесі управління охороною праці. Представлено теоретичне обґрунтування проблем, які виникають при недотриманні даної культури. Розглянуто історичні причини розвитку та поширення інформаційних технологій та їх вплив на охорону праці. Подано основні етапи взаємодії особистості з інформаційною мережею у процесі професійної діяльності.

У другому пункті здійснений аналіз системи «людина-комп'ютер-середовище». Наведено спрощену модель системи «людина-комп'ютер-середовище». Розглянуті основні чинники, які шкодять здоров'ю людини. Описано психологічні можливості та особливості людини. Розглянуто як береться психологічний тиск, його наслідки для організму та як протидіяти йому. Описано небезпеку яку несе комп'ютер та його апаратне забезпечення. Описано шкідливість дисплеїв та види електромагнітного випромінювання та полів, яке виділяють дисплеї. Розглянуто засоби концентрації іонів в повітрі в робочій зоні. Розглянуті електромагнітні випромінювання та поля різних діапазонів. Наведено шкідливої дії клавіатури. Описана шкідливу дії від таких видів принтерів як: матричні принтери, лазерні принтери та струменеві. Розглянуто пристрої безперебійного живлення і фактори, які потрібно застосовувати для підсилення рівня безпеки. Коротко розглянуто електробезпеку та як необхідно повністю унеможливити виникнення електричного джерела загорання.

У третьому пункті розглянуто правила безпеки при роботі з комп'ютером. Представлений режим праці та відпочинку, який залежить від категорії трудової діяльності. Наведено базові профілактичні заходи для забезпечення пожежної безпеки при роботі за комп'ютером. Надано вимоги до екрану за яким проводиться робота та крісла. Наведено основні хвороби, які можуть розвинутись при роботі за ПК, представлено їх

симптоми та причини виникнення. Додатково наведені приклади психологічних проблем зв’язаних з використанням комп’ютера і є не менш небазпечними ніж фізичні.

У четвертому пункті описана організація робочого місця користувача відеодисплейним терміналом на прикладі ПК. Описано основний закон який нормує правильну організацію робочого місця. Представлено основні пункти при організації робочого місця для забезпечення оптимальної продуктивності роботи. Приведено основні заміри для кожного пункту.

7 ЕКОЛОГІЯ

7.1 Зниження енергоємності та енергозбереження

Комп'ютери незамінні для нашої економіки та нашого повсякденного життя. Приблизно 300 мільйонів комп’ютерів використовуються у Сполучених Штатах — майже по одному на кожного чоловіка, жінку та дитину по всій країні і їх споживання електроенергії складає до 10 мільярдів доларів щорічно, що відносить їх до числа найбільших споживачів енергії в категорії електроніки. Для їх роботи потрібно 30 еквівалентних великих (500-мегаватних) електростанцій, які виробляють електроенергію цілодобово кожен день року, при цьому виділяється 65 мільйонів тонн забруднення двоокисом вуглецю (CO2). Немає стандартів ефективності на державному або на національному рівні, що вимагають, щоб ці всюдисущі машини були розроблені для енергоефективного використання. Це особливо важливо для настільних комп'ютерів, які споживають найбільше енергії та часто тримаються цілодобово, навіть коли ніхто не сидить перед ними [32].

Використання сучасних технологій для того, щоб зробити всі комп’ютери та монітори лише на 30 відсотків енергоефективнішими, може заощадити США 3 мільярди доларів щорічно та уникнути необхідності генерувати 29 мільярдів кіловат-годин електроенергії щороку — більше, ніж споживають усі домогосподарства в Індіані, або Лос-Анджелес та Фініксі разом [33]. Він також дозволить зберегти 20 мільйонів тонн забруднення вуглецю атмосфери, все це матиме нульовий вплив на продуктивність комп'ютера або зручність користувачів. Компанія NRDC та

її партнери розробили демонстраційний прототип, який показує, що використання енергії можна скоротити навпіл, використовуючи існуючі компонентні частини, і з мінімальними додатковими витратами.

Комп'ютери та підключені до них монітори відносяться до числа найбільших енергозатратних пристроїв у домах та бізнесах України. Якщо не зменшити використання енергії, це призведе до негативних наслідків для країни. У той час як портативні пристрої, такі як ноутбуки та планшети, як правило, розроблені для досягнення максимальної ефективності та тривалості роботи, перш ніж вичерпати енергію акумулятора, використання енергії настільних комп’ютерів зазнало лише обмеженого покращення, оскільки вони мають доступ до практично необмеженої кількості електроенергії з розеткових мереж. Виробники мають мало ринкових стимулів для впровадження кращих практик з енергоефективності. Адже користувач, а не виробник, оплачує рахунок за електрику.

Незалежно від дому чи бізнесу, більшість комп'ютерів проводять більшу частину часу в режимі очікування або виконуючи завдання низької інтенсивності. Користувачі або знаходяться вдалині від своїх комп’ютерів, за столом, але роблять щось інше, або навіть працюють за своїм комп’ютером, виконуючи легкі завдання, такі як перегляд Інтернету чи читання електронної пошти або більш енергозатратні завдання, такі як гра в комп'ютерні ігри чи редагування відео.

Проблема полягає в тому, що більшість сучасних комп’ютерів продовжують витрачати значну кількість енергії в режимі очікування або роботи низької інтенсивності. Їх компоненти — процесори, материнські плати та диски DVD / Blu-ray, графічні карти, джерела живлення, вентилятори та десятки інших — все ще використовують значну кількість електроенергії під час роботи на холостому ходу.

Законодавчо-нормативні документи України в енергетичній сфері регулюють питання енергозбереження, енергоефективності та встановлюють відповідну компетенцію органів державної влади, наділяючи їх необхідними правами. Окремі документи у сфері енергозбереження безпосередньо стосуються питань зменшення використання енергії, механізмів реалізації заходів із енергозбереження та їх фінансування.

Хоча нині у сфері енергоефективності діють близько 50 національних стандартів групи «Енергозбереження», проте в Україні не існує чіткого механізму стимулювання впровадження енергоощадних заходів, немає правил і механізмів їх регулювання, але є економічне стимулювання енергоефективності. Відповідно законодавству України поняття енергозбереження – це діяльність (організаційна, наукова, практична, інформаційна), яка спрямована на раціональне використання та економне витрачання первинної та перетвореної енергії і природних енергетичних ресурсів у національному господарстві і яка реалізується з використанням технічних, економічних та правових методів. Енергоефективність та енергозбереження взаємопов’язані, оскільки енергозбереження є головним фактором підвищення рівня ефективності використання паливно-енергетичних ресурсів. Поняття енергоефективності є дещо ширшим і містить не лише напрями безпосереднього енергозбереження, а й непрямі, які призводять до зниження споживання паливно-енергетичних ресурсів. Енергоефективність характеризує ступінь використання енергії на одиницю кінцевого продукту.

Питанням врегулювання енергозбереження та енергоефективності в Україні зайнялися Міністерство енергетики та вугільної промисловості та Державне агентство з енергоефективності та енергозбереження, в результаті чого було прийнято низку державних документів. В них

визначено оптимальні шляхи вирішення проблеми підвищення енергоефективності будь-якої галузі промисловості України через інвестиційно-інноваційний розвиток. Завдяки такому вибору забезпечується комплексний розвиток галузі за рахунок реалізації заходів, спрямованих на технічне оновлення виробництва, використовування науково-технічного потенціалу країни і формування високотехнологічного виробництва.

Об’єми поживання енергоресурсів в Україні приблизно в чотири рази більші, ніж у країнах ЄС, а тому потенціал для розвитку ефективного використання енергії в українських компаніях є величезним. Зокрема це стосується житлово-комунального господарства, яке є одним із головних «пожирачів» енергоресурсів в Україні. Хоча в структурі ВВП країни воно займає лише 6%, але майже 40% річного споживання електроенергії та газу припадає саме на даний сектор. Тому експерти ДП «Держзовнішінформ» відмічають, що потенціал енергозбереження в житлово-комунальному господарстві один із самих високих – більше 30% (близько 9,24 млн т н.е.).

Вирішення проблеми стимулювання енергозбереження вимагає комплексного підходу до її розв’язання. Необхідними є економічні, політичні та соціальні механізми стимулювання енергозбереження на регіональному рівні. Органи державного управління повинні залучити до здійснення основних заходів щодо розв’язання (мінімізації негативного впливу) проблеми енергозбереження фінансово-кредитні інституції (у тому числі комерційні банки) та державні Фонди, які працюють у сфері надання фінансово-кредитної підтримки суб’єктам підприємницької діяльності та населенню; державні та громадські асоціативні структури, які займаються питаннями енергозбереження; підприємницькі структури, які працюють у сфері енергозбереження та наукові інституції; засоби масової інформації.

7.2 Утилізація комп’ютерної техніки

Утилізація комп'ютерної техніки — це повна деконструкція електронних пристроїв для того, щоб скоротити витрати сировини та зберегти якомога більше матеріалів зі старої та зламаної техніки. У 2009 році, 38 % комп'ютерів і 25 % від загального обсягу електронних відходів було перероблено в Сполучених Штатах Америки, в порівнянні з 2006 роком відповідно 5 % і 3 % [34]. З моменту свого створення на початку 1990-х років, все більше і більше пристроїв переробляються в усьому світі за рахунок збільшення рівня інформованості та інвестицій. В основному електронна обробка відбувається для того, щоб відновити цінні рідкісноземельні і дорогоцінні метали, які знаходяться в дефіциті, а також пластмаси. Вони будуть перепродані або використані в нових пристроях після очищення, в результаті створюючи економіку замкненого циклу.

Переробка є екологічно чистою, оскільки вона запобігає потраплянню небезпечних відходів, у тому числі важких металів і канцерогенів, в атмосферу або водойми, а також утворенню звалищ. Хоча електроніка складає невелику частку від загального обсягу утворених відходів, вона набагато небезпечніша. Є суворі закони, спрямовані на дотримання і заохочення утилізації побутової техніки, найбільш впливовими з яких є Директива Електронних Відходів та Електронного Обладнання Європейського Союзу і Акт Національної Утилізації Комп'ютерів Сполучених Штатів.

Застарілі комп'ютери та стара електроніка є цінним джерелом для вторинної сировини при переробці, з іншого боку вони є джерелом токсинів та канцерогенів. Швидкий розвиток технології, низька початкова вартість та передбачене старіння призвели до швидко

зростаючого профіциту комп'ютерів та інших електронних компонентів по всьому світу. Технічні рішення доступні, але в більшості випадків перед застосуванням технічного рішення необхідно здійснити нормативно-правові основи, системи зборів, логістики, а також інші послуги. За оцінками Управління з охорони навколишнього середовища США, від 30 до 40 мільйонів залишків ПК класифікуються як «небезпечні побутові відходи». Рада національної безпеки вважає, що 75% всіх персональних комп'ютерів, проданих колись, тепер є електронним сміттям [35].

У 2007 Управління з охорони навколишнього середовища США заявило, що понад 63 мільйони комп'ютерів в США задля заміни були продані або викинуті. Сьогодні 15 % електронних пристроїв та устаткувань перероблюються в Сполучених Штатах. Більшість електронних відходів направляється на звалище або сміттєспалювальний завод, який випускає шкідливі елементи, такі як свинець, ртуть та кадмій в ґрунт, тим самим негативно впливаючи на навколишнє середовище.

Багато матеріалів, що використовуються в комп'ютерних пристроях, можуть бути відновлені для використання в майбутньому виробництві. Повторне використання олова, кремнію, заліза, які в достатній кількості присутні в комп'ютерах або інших електронних пристроях, може зменшити витрати на будівництво нових систем. Компоненти часто містять свинець, мідь, золото та інші цінні матеріали, придатні для утилізації.

Комп'ютерні компоненти містять багато токсичних сполук, таких як діоксини, поліхлоровані біфеніли (ПХБ), кадмій, хром, радіоактивні ізотопи і ртуть. А звичайний монітор комп'ютера може містити понад 6 % свинцю, велика частина якого в свинцевому склі з електронно-променевою трубкою (ЕПТ). А стандартні 15-дюймові (38 см) монітори комп'ютера можуть містити 1 кілограм (1.5 фунти) свинцю, але інші монітори можуть

мати до 4 кілограм (8 фунтів) свинцю. Друковані плати містять значні кількості свинцю та олова, припої, яких, швидше за все, потрапляють в ґрунтові води. Переробка (наприклад, спалювання та кислотні обробки) повинні зберегти ці дорогоцінні сполуки, але можуть створити або синтезувати отруйні побічні речовини.

Експорт відходів у країни з більш низькими екологічними стандартами є основною проблемою. Базельська конвенція включає небезпечні відходи, але не регулює обмеження по кількості, такі як ЕПТ екрани, які не можуть бути експортовані трансконтинентально без попередньої згоди обох країн експорту на отримання відходів. Компанії можуть знайти її економічно ефективною в короткостроковій перспективі, щоб продати застарілі комп'ютери для менш розвинених країн з не суворими правилами. Вважається, що більшість надлишків ноутбуків направляються до країн, що розвиваються, під виглядом «звалища електронних відходів». Висока вартість роботи та багаторазове використання ноутбуків, комп'ютерів та комплектуючих (наприклад, оперативної пам'яті) може допомогти оплатити вартість транспортування багатьох непотрібних «товарів».

У Швейцарії перша система утилізації електронних відходів була створена в 1991 році, розпочиналася зі збору старих холодильників; протягом багатьох років у систему було додано всі інші електричні та електронні пристрої. Створено організацію відповідальності виробника — *SWICO*, в основному для обробки інформації, комунікації і організації технологій. Європейський Союз запровадив подібну систему в лютому 2003 року, під Директивою щодо відпрацьованого електричного й електронного обладнання.

Загальноєвропейське прийняття закону було повільним, Італія і Велика Британія були останніми державами-членами, які ухвалили

директиву як закон. Успіх директиви WEEE значно відрізняється у кожній державі, при цьому збір коливається від 13 до 1 кілограма на рік на душу населення. Комп'ютери та електронні відходи, зібрані з помешкань у Європі, розглядаються директивою WEEE за допомогою Проекту Відповідності Виробника (у межах якої виробники електроніки вкладають кошти до проекту центру утилізації побутових відходів (ЦУПВ).

Однак, обробка колишнього корпоративного комп'ютерного обладнання і супутнього електронного обладнання знаходиться поза *Проектом Відповідності Виробника* (більш відомий як не зобов'язані). У Великій Британії, відходи або застаріле комп'ютерне обладнання обробляють за допомогою третьої особи – уповноважених очисних споруд, які зазвичай стягують плату за її збір і обробку [36]. Підприємства шукають економічно ефективні способи для переробки великої кількості комп'ютерної техніки, але стикаються з більш складними технологічними процесами [37].

Підприємства також розглядають варіанти продажу або встановлення зв'язку з Виробниками Оригінального Устаткування (ВОУ) і організаціями з утилізації. Деякі компанії забирають непотрібне обладнання інших підприємств, стирають дані з систем і дають оцінку залишкової вартості продукту. Для пристроїв, які мають цінність, фірми купують запчастини, ремонтують і продають відновлені продукти тим, хто шукає більш дешеві варіанти, ніж покупка нових.

7.3 Висновок до екологічного розділу

В екологічній частині дипломної роботи освітнього рівня «магістр» було розглянуто два питання, перше – зниження енергоємності та енергозбереження, та друге – утилізація комп’ютерної техніки.

У першому пункті описано використання електроенергії сучасними комп’ютерами і порівняно з іншими енергозатратними активностями. Особливо виділені частини комп’ютера та процеси, які найбільше використовують електроенергію. Приведено декілька можливих швидких заходів для обмеження використання електроенергії комп’ютерами.

У наступному пункті описано, що таке утилізація комп’ютера, як вона проводиться та важливість її проведення як з екологічної так і з економічної точки зору. Описані основні державні заходи в різних країнах для забезпечення виконання утилізації комп’ютерів. Розглянуто основні небезпеки при проведенні утилізації неналежним шляхом. Описано проблеми переправлення відходів у країни, що розвиваються. Також розглянуто найбільш економічно вигідний варіант зменшення відходів – відновлення робочих пристроїв.

ВИСНОВОК

В даній дипломній роботі магістра створено програму для отримання даних з мережі Facebook шляхом веб-скрапінгу. Програма розроблена на мові програмування Python та використовує бібліотеку Selenium. Вона імітує дії реального користувача і записує спільних друзів користувача у файл. Після цього вона обробляє дані, фільтруючи дані і перетворюючи їх у граф. Вони далі експортуються у файл формату JSON, оскільки даний формат є широко поширеним і підтримується більшістю бібліотек для візуалізації даних у веб-додатку.

Представлено базові концепції графів. Описана їх важливість у сучасному житті та приведені приклади їх застосування. Представлено соціальний граф, який знаходиться на стику графу і соціальної мережі. Подано практичні сценарії його використання та корисні властивості для користувача.

Обґрунтовано вибір технологій та бібліотек для розроблення веб-додатку для візуалізації графу друзів. Зокрема, для кожної з категорій обрано найкращі технології на основі наявності потрібних функція, популярності та актуальності. Для кожної з технологій подана коротка довідка. На основі обраних технологій створено веб-додаток.

Процес імплементації веб-додатку розписаний крок за кроком. Ключові моменти додатку представлено у вигляді лістингу для демонстрації процесу розробку. Також описано компонент “Graph”, який використовувався для створення інтерактивного графу з можливістю легкої та гнучкої конфігурації.

Результатом роботи веб-додатку є візуалізації даних у вигляді графу. Приведено можливості зміни цієї візуалізації завдяки додатковим даним. Також описано можливості візуалізації даних використовуючи матрицю суміжності. Даний вид візуалізації пропонує чітку видимість зв’язків між друзями і дозволяє проводити більш точні дослідження.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Спекторський І. Я. Дискретна математика. — К.: «Політехніка», 2014. – 220 с.

2. Ловас Л., Пламмер М. Прикладные задачи теории графов. — М. : Мир, 1998. — 656 с.

3. Фляйшнер Г. Эйлеровы графы и смежные вопросы. — М. : Мир, 2002. — 176 с.

4. Оре О. Теория графов. — М. : Наука, 1980. — 336 с.

5. Н. В. Богатир. Вкоріненість і за її межами: вплив мереж. — Міжнародна конференція «Вкоріненість і за її межами: пояснюють чи соціологічні теорії економічну реальність ?» Жовтень 25-28, 2012, Москва, Росія, 2012. — 168-69 с..

6. В.М. Сазанів. Соціальні мережі як нова суспільна сфера. Системний аналіз та прогноз. — М. : Лабораторія СВМ, 2010. — 180 с.

7. Сутурин Г.С. Формування спільнот на основі граф-інтересів. — Красноярськ : Сучасні дослідження соціальних проблем, 2013. — № 1(13). — 215 с.

8. Тлумачний словник з інформатики / Г. Г. Півняк, Б. С. Бусигін, М. М. Дівізінюк та ін. — Д., Нац. гірнич. ун-т, 2010. — 600 с.

9. Загальний план дослідження функції та побудова її графіків // Вища математика в прикладах і задачах: Навчальний посібник. 2-ге видання. – К.: Центр учбової літератури / Клепко В.Ю., Голець В.Л.. — 2009. — С. 324. — 594 с.

10. Top 20 Web Crawling Tools to Scrape the Websites Quickly [Електронний ресурс] – Режим доступу до ресурсу:

https://www.octoparse.com/blog/top-20-web-crawling-tools-for-extracting-web-data.

11. How to Scrape Yellow Pages with Web Scraper chrome extension [Електронний ресурс] – Режим доступу до ресурсу: http://prowebscraping.com/how-to-scrape-yellow-pages-with-web-scraper-chrome-extension/.

12. Крушельницька Я. В. Охорона праці / Я. В. Крушельницька., 2013. – 367 с.

13. Яскілка В. Я. Охорона праці в галузі / В. Я. Яскілка, М. З. Олійник. – Тернопіль, 2015. – 56 с.

14. Les Misérables Co-occurrence [Електронний ресурс] – Режим доступу до ресурсу: https://bost.ocks.org/mike/miserables/.

15. Web Scrapping Tool and Free Web Crawlers [Електронний ресурс] – Режим доступу до ресурсу: https://www.octoparse.com/.

16. Соціальний граф [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Соціальний_граф.

17. Graph API [Електронний ресурс] – Режим доступу до ресурсу: https://developers.facebook.com/docs/graph-api/.

18. GraphQL | A query language for your API [Електронний ресурс] – Режим доступу до ресурсу: https://graphql.org/.

19. Top JavaScript Frameworks and Topics to Learn in 2020 and the New Decade [Електронний ресурс] – Режим доступу до ресурсу: https://medium.com/javascript-scene/top-javascript-frameworks-and-topics-to-learn-in-2020-and-the-new-decade-ced6e9d812f9/.

20. Top 7 Python Web Scraping Tools For Data Scientists [Електронний ресурс] – Режим доступу до ресурсу: https://analyticsindiamag.com/top-7-python-web-scraping-tools-for-data-scientists/.

21. Angular NgRx Starter Guide [Електронний ресурс] – Режим доступу до ресурсу: https://fireship.io/lessons/angular-ngrx-redux-starter-guide/.

22. Review: The 6 best JavaScript IDEs [Електронний ресурс] – Режим доступу до ресурсу: https://www.infoworld.com/article/3192844/review-the-6-best-javascript-ides.html?page=2.

23. Top 5 JavaScript IDEs [Електронний ресурс] – Режим доступу до ресурсу: https://jaxenter.com/top-5-javascript-ide-146609.html.

24. JavaScript in Visual Studio Code [Електронний ресурс] – Режим доступу до ресурсу: https://code.visualstudio.com/docs/languages/javascript/.

25. WebStorm: The Smartest JavaScript IDE by Jetbrains [Електронний ресурс] – Режим доступу до ресурсу: https://www.jetbrains.com/webstorm.

26. Кириченко В. В. Інформаційна культура у структурі професійної компетентності державних службовців – Херсон 2019. – 7с.

27. Шконда В.В., Кальянов А.В. Культорологічні засади професійно-особистісного становлення майбутніх фахівців : Монографія. – Донецьк, 2012. –262 с.

28. Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин, затверджені постановою Головного державного санітарного лікаря України від 10.12.1998 № 7 [Електронний ресурс] – Режим доступу до ресурсу: http://mozdocs.kiev.ua/view.php?id=2445

29. Охорона праці в галузі [Електронний ресурс] – Режим доступу до ресурсу: https://tiphaman.top/book_ohorona-praci-v-galuzi_876/.

30. Інструкція з охорони праці під час робіт на персональному комп'ютері і відеодисплейних терміналах [Електронний ресурс] – Режим

доступу до ресурсу: https://dnaop.com/html/31947/doc-instrukcijaz-ohoroni-pracipid-chas-robit-na-personalynomu-kompjuterii-videodisplejnih-terminalah.

31. Інструкція з охорони праці при роботі з персональним комп'ютером № 3 від 11.01.2016 [Електронний ресурс] – Режим доступу до ресурсу: http://dsum.edu.ua/wp-content/uploads/2015/11/ИОТ-№3-При-работе-на-компьютерах.pdf.

32. Аналіз та представлення різних електричних навантажень в НЕМС [Електронний ресурс] – Режим доступу до ресурсу: www.eia.gov/analysis/studies/demand/miscelectric/.

33. Пікслі Дж., Рос С. "Моніторинг використання режимів живлення комп'ютера в університеті" / Дж. Пікслі, С. Рос., 2014. - 102 с.

34. Утилизация компьютеров, списание компьютеров [Електронний ресурс] – Режим доступу до ресурсу: http://www.nemusor.ru/utilizacija-kompjuterov-spisanie-kompjuterov.

35. Утилизация оргтехники, утилизация компьютеров и оборудования [Електронний ресурс] – Режим доступу до ресурсу: http://www.pererabotka.org/util_org.htm.

36. Данилко В. К. Екологічна статистика / В. К. Данилко., 2003. – 368 с.

37. Тарасова В. В. Екологічна статистика / В. В. Тарасова., 2008. – 392 с.

38. Afra J. Mashhadi, Sonia Ben Mokhtar, Licia Capra. Habit: Leveraging Human Mobility and Social Network for Efficient Content Dissemination in MANETs / WoWMoM09, 2009. — 230 с.

39. How the Interest Graph will shape the future of the web [Електронний ресурс] – Режим доступу до ресурсу: https://miter.mit.edu/articlehow-interest-graph-will-shape-future-web/.

40. William Marcellino, Meagan Smith, Christopher Paul, Lauren Skrabala. Monitoring Social Media. Lessons for Future Department of Defense Social Media Analysis in Support of Information Operations/ Research Reports. RAND Corporation, 2013. – 92 с.

41. Elijah Meeks. D3.js in Action Data visualization with JavaScript 2nd Edition/Manning Publications Co., 2017. – 375 с.

42. Three Little Circles [Електронний ресурс] – Режим доступу до ресурсу: https://bost.ocks.org/mike/circles/.

43. Радчук М. І. Види небезпек у соціальних мережах / Радчук М. І. Тези доповіді на ІІІ Міжнародна студентська науково-технічна конференція «Природничі та гуманітарні науки. Актуальні питання». – Тернопіль, ТНТУ, 2020. – 261 с.

44. Радчук М. І. Практичне використання графу інтересів та соціального графу / Радчук М. І. Тези доповіді на ІІІ Міжнародна студентська науково-технічна конференція «Природничі та гуманітарні науки. Актуальні питання». – Тернопіль, ТНТУ, 2020. – 261 с.

45. Радчук М. І. Порівняння популярних бібліотек для візуалізації графу / Радчук М. І. Тези доповіді на ІІІ Міжнародна студентська науково-технічна конференція «Природничі та гуманітарні науки. Актуальні питання». – Тернопіль, ТНТУ, 2020. – 261 с.

46. Animated Basic Charts in D3 and React [Електронний ресурс] – Режим доступу до ресурсу: https://manavsehgal.com/animated-basic-charts-in-d3-and-react-e131635229c.

47. Add Apollo Link Functionality [Електронний ресурс] – Режим доступу до ресурсу: https://github.com/apollographql/apollo-android/issues/931.

48. Introduction to Angular Modules [Електронний ресурс] – Режим доступу до ресурсу: https://www.tektutorialshub.com/angular/angular-modules/.

49. Introduction to D3 / MIT Visualization Group / Observable [Електронний ресурс] – Режим доступу до ресурсу: https://observablehq.com/@mitvis/introduction-to-d3

ДОДАТКИ

Міністерство освіти і науки України,
Тернопільський національний технічний університет
імені Івана Пулюя
Маріборський університет (Словенія)
Технічний університет в Кошице (Словаччина)
Каунаський технологічний університет (Литва)
Львівський національний університет
імені Івана Франка,
Гірничо-металургійна академія ім. Станіслава Сташиця
(Польща)
Луцький національний технічний університет,
Чернівецький національний університет
імені Юрія Федьковича,
Вроцлавський економічний університет (Польща)
Донбаська державна машинобудівна академія

Студентське наукове товариство

III МІЖНАРОДНА

студентська науково - технічна конференція

"ПРИРОДНИЧІ ТА ГУМАНІТАРНІ НАУКИ.

АКТУАЛЬНІ ПИТАННЯ"

23-24 квітня 2020 р.

(збірник тез конференції)

Тернопіль 2020

ББК 72+34 (Укр)
М34

Матеріали ІІІ Міжнародної студентської науково - технічної конференції / Тернопіль: Тернопільський національний технічний університет ім. І.Пулюя (м. Тернопіль, 23-24 квітня 2020 р.), 2020.- 261 с.

В збірнику друкуються матеріали ІІІ Міжнародної студентської науково-технічної конференції. Тернопіль. – ТНТУ ім. І. Пулюя (23-24 квітня 2020р.) за наступними науковими напрямками:

математичне моделювання і механіка, машинобудування, машини та обладнання сільськогосподарського виробництва; приладобудування; матеріалознавство, міцність матеріалів і конструкцій; електротехніка, електроніка та світлотехніка; математика; фізика; хімія, хімічна, біологічна та харчова технології; обладнання харчових виробництв; інформаційні технології, гуманітарні науки, економіка, менеджмент, фінанси, біомедична інженерія; зварювання та споріднені процеси і технології, інженерія продукції.

.

Комп'ютерний набір, верстка та редагування:
науковий секретар Ігор Окіпний

Адреса конференції:
46001, м. Тернопіль, вул. Руська, 56
Тернопільський національний технічний університет ім. Івана Пулюя
e-mail: snt@tu.edu.te.ua
Тернопільський національний технічний університет ім. Івана Пулюя

УДК 621.326
Радчук М. –ст. гр. СНнм-61
Тернопільський національний технічний університет імені Івана Пулюя

ВИДИ НЕБЕЗПЕК У СОЦІАЛЬНИХ МЕРЕЖАХ

Науковий керівник: д.т.н., професор Ясній О. П.

Radchuk M.
Ternopil Ivan Pul'uj National Technical University

TYPES OF HAZARDS IN SOCIAL NETWORKS

Supervisor: professor Yasnij O. P.

Ключові слова: Соціальна мережа, Дезінформація
Keywords: Social Network, Disinformation

У доповіді будуть розглянуті потенційні небезпеки соціальних мереж та приведено конкретні приклади їхнього впливу на суспільне життя. Соціальні мережі можуть бути використані для поширення дезінформації. вдалим прикладом можна назвати злам облікового запису Associated Press в соціальній мережі мікроблогів Twitter в 2013 році та розміщення повідомлення про поранення Президента США Барака Обами внаслідок замаху на його життя. Новина, серед іншого, спричинила миттєвий обвал на фондових ринках. У відкритій доповіді американського аналітичного центру RAND зазначено, що аналіз соціальних мереж має великий потенціал використання в інформаційних операціях американськими військовими, оскільки дозволяє дослідити ставлення, світогляд та спілкування широкого кола осіб.

Велику увагу громадськості привернули випадки використання соціальних мереж російськими спецслужбами для впливу на вибори Президента США 2016 року. Так, в 2017 році були виявлені та висвітлені облікові записи в соціальних мережах, що належали російським операторам та які вдавали із себе американців. Наприклад, обліковий запис TEN_USA в Twitter вдавав із себе неофіційний обліковий запис осередку
Республіканської партії в штаті Теннессі. Йому вдалось здобути широку аудиторію, вступити в контакт з іншими консервативними та правими активістами в США.

Серед інших небезпек, можлива інфільтрація кола друзів зловмисниками з метою добування приватної інформації. Відомий приклад такої дії — створення Томасом Райаном з Provide Security підставного профілю молодої вродливої дівчини Робін Сейдж віком у 25 років, яка за легендою була фахівцем з 10-річним стажем роботи у сфері безпеки й мала вчений ступінь. Від імені свого віртуала Томас через популярні соціальні сервіси Facebook, LinkedIn і Twitter відправив запити щодо включення в коло друзів чоловікам і жінкам із середовища військових, співробітників сек'юриті компаній і державних чиновників.

Література

1. William Marcellino, Meagan Smith, Christopher Paul, Lauren Skrabala. Monitoring Social Media. Lessons for Future Department of Defense Social Media Analysis in Support of Information Operations/ Research Reports. RAND Corporation, 2013. – 92 с.

2. How A Russian Troll Fooled America [Електронний ресурс] – Режим доступу до ресурсу: https://medium.com/dfrlab/how-a-russian-troll-fooled-america-80452a4806d1

УДК 621.326
Радчук М. –ст. гр. СНнм-61
Тернопільський національний технічний університет імені Івана Пулюя

ПРАКТИЧНЕ ВИКОРИСТАННЯ ГРАФУ ІНТЕРЕСІВ ТА СОЦІАЛЬНОГО ГРАФУ

Науковий керівник: д.т.н., професор Ясній О. П.

Radchuk M.
Ternopil Ivan Pul'uj National Technical University

PRACTICAL USAGE OF THE GRAPH OF INTERESTS AND SOCIAL GRAPH

Supervisor: professor Yasnij O. P.

Ключові слова: Мережа, Граф
Keywords: Network, Graph

У доповіді буде порівняно граф інтересів із соціальним графом та подані приклади їх практичного використання. Граф інтересів — це онлайн представлення інтересів конкретної людини, отримане на основі її активності у соціальних мережах. Вершинами графа є захоплення особистості або її профіль в соціальній мережі. Ребра графа відображають взаємини між його вершинами. За допомогою графа інтересів можна зрозуміти, що людина хоче зробити, купити, куди хоче піти, з ким може зустрітися, за чиїми повідомленнями їй цікаво стежити або за кого вона готова проголосувати.

Граф інтересів і соціальний граф тісно взаємопов'язані, але це не одне і те ж саме. Граф інтересів використовується для створення мережі інтересів людей. У той час, як Facebook та інші соціальні мережі організовані навколо друзів людини, тобто навколо соціального графа, мережі захоплень створені навколо інтересів особистостей, їх графа інтересів. Подібно до того, як соціальний граф — це карта взаємозв'язків особистості з тими, хто «слідує» за нею в мережі, граф інтересів — це так само взаємозв'язок з інтересами особи в мережі.

Існує кілька способів використання графа інтересів, як з точки зору споживача, так і з точки зору бізнесмена. У поєднанні з соціальним графом, граф інтересів може застосовуватися для встановлення зв'язків між користувачами в соціальних мережах або в реальному світі. У таких мережах користувачі можуть вказувати і ділитися своїми захопленнями, але при цьому їм не обов'язково знати один одного. Граф інтересів так само може бути застосований в маркетингу.

Література

1. Afra J. Mashhadi, Sonia Ben Mokhtar, Licia Capra. Habit: Leveraging Human Mobility and Social Network for Efficient Content Dissemination in MANETs / WoWMoM09, 2009. — 230 с.

2. How the Interest Graph will shape the future of the web [Електронний ресурс] – Режим доступу до ресурсу: https://miter.mit.edu/articlehow-interest-graph-will-shape-future-web/.

УДК 621.326
Радчук М. –ст. гр. СНнм-61
Тернопільський національний технічний університет імені Івана Пулюя

ПОРІВНЯННЯ ПОПУЛЯРНИХ БІБЛІОТЕК ДЛЯ ВІЗУАЛІЗАЦІЇ ГРАФІВ

Науковий керівник: д.т.н., професор Ясній О. П.

Radchuk M.
Ternopil Ivan Pul'uj National Technical University

COMPARISON OF POPULAR LIBRARIES FOR GRAPH VISUALIZATION

Supervisor: professor Yasnij O. P.

Ключові слова: Візуалізація, Бібліотека
Keywords: Visualization, Library

У доповіді буде розглянуто найбільш популярні бібліотеки для візуалізації графів D3.js, Chart.js, Vis.js. Всі вони є JavaScript бібліотеками, які будують граф у SVG форматі, оскільки даний формат є легким і відображається однаково чітко незалежно від розширення монітору.

D3.js (або просто D3 для документів, керованих даними) – це бібліотека JavaScript для створення динамічних, інтерактивних візуалізацій даних у веб-браузерах. Він використовує широко впроваджені стандарти SVG, HTML5 і CSS. Вона є наступником попередньої системи Protovis. На відміну від багатьох інших бібліотек, D3.js забезпечує великий контроль над кінцевим візуальним результатом. Були різні попередні спроби візуалізації даних до веб-браузерів. Бібліотека JavaScript D3.js, вбудована у веб-сторінку HTML, використовує попередньо побудовані функції JavaScript для вибору елементів, створення об'єктів SVG, їх стилізування або додавання до них переходів, динамічних ефектів або підказок. Ці об'єкти також можна широко використовувати за допомогою CSS.

Vis.js – динамічна бібліотека візуалізації на основі браузера. Бібліотека розроблена так, щоб її можна було легко використовувати, обробляти великі обсяги динамічних даних, а також маніпулювати та взаємодіяти з даними. Бібліотека складається з компонентів DataSet, Timeline, Network, Graph2d і Graph3d. Вона дозволяє відображати динамічні, автоматично організовані мережі, створювати повністю налаштовувану, інтерактивну шкалу часу з елементами та діапазонами.

Chart.js - це популярна бібліотека з відкритим кодом, яка допомагає нам створювати дані в веб-додатках. Це дуже настроюється, але налаштування всіх його варіантів залишається проблемою для деяких людей. Chart.js достатньо розумний для побудови осі динамічно на основі даних, які йому передаються.

Література

1. Elijah Meeks. D3.js in Action Data visualization with JavaScript 2nd Edition/ Manning Publications Co., 2017. – 375 с.

2. Three Little Circles [Електронний ресурс] – Режим доступу до ресурсу: https://bost.ocks.org/mike/circles/.

З М І С Т

Секція: **Інформаційні технології**

Секція: **Математика**

Файл FacebookScrapper.py

```
from html.parser import HTMLParser
import re
import time
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
import os
from tqdm import tqdm
import pickle
import getpass

unique_id = input("Your unique id:")
username = input("Facebook username:")
password = getpass.getpass('Password:')

chrome_options = webdriver.ChromeOptions()
prefs =
{"profile.default_content_setting_values.notifications" : 2}
chrome_options.add_experimental_option("prefs",prefs)
driver = webdriver.Chrome(chrome_options = chrome_options)

driver.get('http://www.facebook.com/')

elem = driver.find_element_by_id("email")
elem.send_keys(username)
elem = driver.find_element_by_id("pass")
elem.send_keys(password)
elem.send_keys(Keys.RETURN)
time.sleep(5)

SCROLL_PAUSE_TIME = 2

def get_fb_page(url):
    time.sleep(2)
    driver.get(url)

    last_height = driver.execute_script("return
document.body.scrollHeight")

    while True:
        driver.execute_script("window.scrollTo(0,
document.body.scrollHeight);")
        time.sleep(SCROLL_PAUSE_TIME)
        new_height = driver.execute_script("return
document.body.scrollHeight")
        if new_height == last_height:
```

```
            break
        last_height = new_height
    html_source = driver.page_source
    return html_source

def find_friend_from_url(url):
    if re.search('com\/profile.php\?id=\d+\&', url) is not
None:
        m = re.search('com\/profile.php\?id=(\d+)\&', url)
        friend = m.group(1)
    else:
        m = re.search('com\/(.*)\?', url)
        friend = m.group(1)
    return friend

class MyHTMLParser(HTMLParser):
    urls = []

    def error(self, message):
        pass

    def handle_starttag(self, tag, attrs):
        if tag == "a":
            for name, value in attrs:
                if name == "href":
                    if
re.search('\?href|&href|hc_loca|\?fref', value) is not None:
                        if re.search('.com/pages', value) is
None:
                            self.urls.append(value)


my_url = 'http://www.facebook.com/' + unique_id + '/friends'

UNIQ_FILENAME = 'uniq_urls.pickle'
if os.path.isfile(UNIQ_FILENAME):
    with open(UNIQ_FILENAME, 'rb') as f:
        uniq_urls = pickle.load(f)
    print('We loaded {} uniq friends'.format(len(uniq_urls)))
else:
    friends_page = get_fb_page(my_url)
    parser = MyHTMLParser()
    parser.feed(friends_page)
    uniq_urls = set(parser.urls)

    print('We found {} friends, saving
it'.format(len(uniq_urls)))

    with open(UNIQ_FILENAME, 'wb') as f:
```

```
        pickle.dump(uniq_urls, f)

friend_graph = {}
GRAPH_FILENAME = 'friend_graph.pickle'

if os.path.isfile(GRAPH_FILENAME):
    with open(GRAPH_FILENAME, 'rb') as f:
        friend_graph = pickle.load(f)
    print('Loaded existing graph, found {}
keys'.format(len(friend_graph.keys())))


for url in tqdm(uniq_urls):
    friend_username = find_friend_from_url(url)
    if friend_username in friend_graph.keys():
        continue

    friend_graph[friend_username] = [username]
    mutual_url =
'https://www.facebook.com/{}/friends_mutual'.format(friend_use
rname)
    mutual_page = get_fb_page(mutual_url)

    parser = MyHTMLParser()
    parser.urls = []
    parser.feed(mutual_page)
    mutual_friends_urls = set(parser.urls)
    print('Found {} urls'.format(len(mutual_friends_urls)))

    for mutual_url in mutual_friends_urls:
        mutual_friend = find_friend_from_url(mutual_url)
        friend_graph[friend_username].append(mutual_friend)

    with open(GRAPH_FILENAME, 'wb') as f:
        pickle.dump(friend_graph, f)

driver.quit()
```

Файл facebook-friend-graph.ipynb

```
import plotly.offline as py
from plotly.graph_objs import *

import community
import networkx as nx
import colorlover as cl
import numpy as np
import pickle
```

```
py.init_notebook_mode(connected=True)

CENTRAL_ID = 'max.radchuk'

GRAPH_FILENAME = 'friend_graph.pickle'
with open(GRAPH_FILENAME, 'rb') as f:
    friend_graph = pickle.load(f)

edges = []
nodes = [CENTRAL_ID]
central_friends = {}

for k, v in friend_graph.items():
    intersection_size =
len(np.intersect1d(list(friend_graph.keys()), v))
    if intersection_size > 2:
        central_friends[k] = v

print('Firtered out {} items'.format(len(friend_graph.keys())
- len(central_friends.keys())))

# Extract edges from graph
for k, v in central_friends.items():
    for item in v:
        if item in central_friends.keys() or item ==
CENTRAL_ID:
            edges.append((k, item))
G = nx.Graph()
G.add_nodes_from([CENTRAL_ID])
G.add_nodes_from(central_friends.keys())

G.add_edges_from(edges)
print('Added {} edges'.format(len(edges) ))
from networkx.readwrite import json_graph
import json
json_node_link = json_graph.node_link_data(G)
with open('friend_graph.json', 'w', encoding='utf-8') as f:
    json.dump(json_node_link, f, ensure_ascii=False, indent=4)
```

Файл GraphViz.tsx

```
import React from 'react';
import './App.css';
import { Graph, GraphConfiguration } from 'react-d3-graph';
import socialGraphData from './friend_graph.json';
import "./App.css";

export default class App extends React.Component {
```

```
  constructor(props: Readonly<{}>){
    super(props);
  }
readonly myConfig: Partial<GraphConfiguration<{ "id": string;
}, { "source": string; "target": string; }>> = {
  nodeHighlightBehavior: true,
  highlightDegree: 2,
  node: {
      color: 'lightgreen',
      size: 200,
      highlightStrokeColor: 'blue'
  },
  link: {
      highlightColor: 'lightblue',
      type: 'CURVE_SMOOTH'
  },
  collapsible: true,
  d3: {
    alphaTarget: 0.05,
    gravity: -100,
    linkLength: 100,
    linkStrength: 0.5,
    disableLinkForce: false
  },
  height: 1440,
  width: 2500
};

onClickGraph = function(event: any) {
   console.log('Clicked the graph background');
};

onClickNode = function(nodeId: any) {
   console.log('Clicked node ${nodeId}');
};

onDoubleClickNode = function(nodeId: any) {
   console.log('Double clicked node ${nodeId}');
};

onRightClickNode = function(event: any, nodeId: any) {
   console.log('Right clicked node ${nodeId}');
};

onMouseOverNode = function(nodeId: any) {
   console.log(`Mouse over node ${nodeId}`);
};

onMouseOutNode = function(nodeId: any) {
```

```
   console.log(`Mouse out node ${nodeId}`);
};

onClickLink = function(source: any, target: any) {
   console.log(`Clicked link between ${source} and
${target}`);
};

onRightClickLink = function(event: any, source: any, target:
any) {
   console.log('Right clicked link between ${source} and
${target}');
};

onMouseOverLink = function(source: any, target: any) {
   console.log(`Mouse over in link between ${source} and
${target}`);
};

onMouseOutLink = function(source: any, target: any) {
   console.log(`Mouse out link between ${source} and
${target}`);
};

onNodePositionChange = function(nodeId: any, x: any, y: any) {
   console.log(`Node ${nodeId} moved to new position x= ${x}
y= ${y}`);
};

render() {
return (<div className="graph">
  <Graph
     id='graph-id'
     data={socialGraphData}
     config={this.myConfig}
     onClickNode={this.onClickNode}
     onDoubleClickNode={this.onDoubleClickNode}
     onRightClickNode={this.onRightClickNode}
     onClickLink={this.onClickLink}
     onRightClickLink={this.onRightClickLink}
     onMouseOverNode={this.onMouseOverNode}
     onMouseOutNode={this.onMouseOutNode}
     onMouseOverLink={this.onMouseOverLink}
     onMouseOutLink={this.onMouseOutLink}/>
</div>
);}
```