МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова

Навчально-науковий інститут комп'ютерних наук та управління проектами
Кафедра програмного забезпечення автоматизованих систем
Спеціальність 121 «Інженерія програмного забезпечення»
Освітня програма «Інженерія програмного забезпечення»

«Допущений до захисту»
Завідувач кафедри
_____ проф. Приходько С.Б.
«___» _____ 2024 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття ступеня вищої освіти «магістр»

на тему: Нелінійна регресійна модель для оцінювання розміру веб-застосунків, що створюються з використанням фреймворку NextJS, та розробка програми для її реалізації

Виконав: студент групи 6151м
_____ Грицань В.Ю.__
(підпис, ПІБ)
Керівник роботи:
_____ Приходько С.Б.__
(посада, науковий ступень вчене звання)
_____
(підпис, ПІБ)

Миколаїв – 2024 р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова

Навчально-науковий інститут комп'ютерних наук та управління проектами

Кафедра програмного забезпечення автоматизованих систем

Спеціальність 121 «Інженерія програмного забезпечення»

Освітня програма «Інженерія програмного забезпечення»

«ЗАТВЕРДЖУЮ»

Гарант освітньої програми

_____ проф. С.Б.Приходько

(підпис)

« 14 » жовтня 2024 р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття ступеня вищої освіти «магістр»

Студенту Грицаню Валерію Юрійовичу

(Прізвище, ім'я, по батькові)

1. Тема роботи: Нелінійна регресійна модель для оцінювання розміру веб-застосунків, що створюються з використанням фреймворку NextJS, та розробка програми для її реалізації

Керівник роботи Приходько С.Б.

Затверджені наказом ректора № 1070-УЧ від « 11 » _____10_____ 2024 р.

2. Термін подання роботи: 09.12.2024 р.

3. Вихідні дані по роботі:

4. Перелік питань, що належать до розробки (найменування розділів)

Вступ (Актуальність теми. Зв'язок роботи з науковими програмами, планами, темами. Мета і завдання дослідження. Об'єкт дослідження. Предмет дослідження. Методи дослідження. Наукова новизна одержаних результатів. Практичне значення одержаних результатів. Особистий внесок здобувача. Апробація результатів досліджень**.** Публікації.); Огляд літератури та обґрунтування необхідності проведення досліджень за обраною темою; Викладення результатів власних досліджень з висвітленням того нового, що пропонується; Розділ з розробки програмного забезпечення; Організаційно-економічний розділ; Розділи з охорони праці та охорони навколишнього середовища; Висновки; Список використаних джерел; Додатки (технічне завдання, текст програми, опис програми, інструкція користувача, програма і методика випробувань програмного забезпечення)

5. Перелік презентаційних матеріалів 1.Тема, 2. Актуальність теми, 3. Мета і завдання дослідження, 4. Об'єкт та предмет дослідження, 5. Методи дослідження, 6. Наукова новизна одержаних результатів, 7. Практичне значення одержаних результатів, 8. Особистий внесок здобувача 9. Аналіз існуючих методів та математичних моделей для оцінювання розміру веб-застосунків, 10. Апробація результатів досліджень та публікації 11. Аналіз вимог до програми 12.Розробка програми, 13. Екранна форма застосунку, 14 Висновки

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання 14.10.2024 р.

КАЛЕНДАРНИЙ ПЛАН

Номер	Назва етапів роботи	Терміни виконання	Примітка
1.	Підготовка розділу ВСТУП	16.10.2024	НС*
2.	Підготовка розділу з огляду літератури та обґрунтування необхідності проведення досліджень за обраною темою	18.10.2024	НС*
3.	Підготовка розділу (ів) за результатами власних досліджень	21.10.2024	НС*
4.	Підготовка розділу з розробки програмного забезпечення	27.11.2024	
5.	Підготовка організаційно-економічного розділу	29.11.2024	
6.	Підготовка розділу з охорони праці	02.12.2024	
7.	Підготовка розділу з охорони навколишнього середовища	04.12.2024	
8.	Підготовка розділу ВИСНОВКИ	05.12.2024	
9.	Оформлення списку використаних джерел та додатків	06.12.2024	
10.	Подання на кафедру ПЗАС тексту остаточного варіанту роботи, підписаного її керівником, у роздрукованому та електронному форматі разом із заявами щодо самостійності виконання роботи та ідентичності друкованої та електронної версії роботи (Додатки 1 і 2 «Порядку здійснення заходів з перевірки робіт на наявність текстових збігів/ідентичності/схожості із використанням програмно-технічних засобів», який введений в дію наказом ректора НУК за №20 від 20.01.2020 р.)	09.12.2024	не пізніше, ніж за 14 діб до захисту (згідно п.4.1 зазначеного Порядку)
11.	Підготовка презентації та доповіді	13.12.2024	
12.	Попередній захист роботи на засіданні кафедри ПЗАС	16.12.2024	
13.	Подання на кафедру ПЗАС електронних версії наступних документів у форматі pdf: кваліфікаційної роботи; файлу-опису кваліфікаційної роботи (згідно Додатку до наказу ректора НУК за №287-уч від 19.05.2020 р.); презентації доповіді	19.12.2024	

* - за результатами наукового стажування (НС), яке було з 02.09.2024 по 13.10.2024 р.

Студент _____ Грицань В.Ю.
(підпис) (ПІБ)

Керівник роботи _____ Приходько С.Б.
(підпис) (ПІБ)

РЕФЕРАТ

Грицань Валерій Юрійович

«Нелінійна регресійна модель для оцінювання розміру веб-застосунків, розроблених за допомогою фреймворку Next.js, та створення програмного інструменту для її реалізації»

Кваліфікаційна робота на здобуття освітнього рівня магістра зі спеціальності 121 – «Інженерія програмного забезпечення». Національний університет кораблебудування імені адмірала Макарова, Миколаїв, 2024 р.
Обсяг роботи: 85 стор., 7 табл., 7 рис., 23 використаних джерел., 5 додатків:

Актуальність теми: Зумовлена необхідністю точнішого оцінювання розміру (в тисячах рядків коду) веб-застосунків на основі фреймворку Next.js, оскільки моделі для подібних оцінок у цьому середовищі досі не були сформовані.

Мета дослідження: Підвищення точності прогнозування розміру (в тисячах рядків коду) веб-застосунків, що розробляються з використанням Next.js, шляхом застосування нелінійної регресійної моделі.

Об’єкт дослідження: Процес оцінювання розміру (в тисячах рядків коду) веб-застосунків, створених із застосуванням Next.js.

Предмет дослідження: Нелінійні регресійні моделі для оцінювання розміру (в тисячах рядків коду) веб-застосунків на основі Next.js.

Методи дослідження: Використано методи математичної статистики, теорії ймовірностей, а також методи лінійного й нелінійного регресійного аналізу.

Наукова новизна: Удосконалено нелінійну регресійну модель для оцінки розміру (в тисячах рядків коду) веб-застосунків на фреймворку Next.js. Використання нормалізації на основі десяткового логарифму дозволяє підвищити точність оцінювання порівняно з існуючими моделями для інших технологій, таких як Java. Нова модель показала кращі результати за критеріями R^2, MMRE і PRED(0.25).

Практичне значення: Створено програмний інструмент для оцінки розміру (в тисячах рядків коду) веб-застосунків Next.js, написаний мовою Python. Ця програма може бути корисною на ранніх етапах проєктування програмного забезпечення.

Апробація результатів: Основні положення та результати роботидоповідалися та обговорювалися на VI Всеукраїнській науково-практичній конференції «Сучасні інформаційні технології в освіті і науці» («УМАНСЬКИЙ ДЕРЖАВНИЙ ПЕДАГОГІЧНИЙ УНІВЕРСИТЕТ ІМЕНІ ПАВЛА ТИЧИНИ», м. Умань, 14–15 листопада 2024 р.).

Публікації: Основні положення й результати кваліфікаційної роботи опубліковано у 1 науковій праці – тезах конференції.

Ключові слова: оцінка обсягу коду, веб застосунки, нелінійна регресійна модель, фреймворк Next.js.

ABSTRACT

Hrytsan Valeriy Yuriyovych

"A nonlinear regression model for estimating the size of server applications developed using the Next.js framework and creating a software tool for its implementation"

Qualification work for obtaining a master's degree in the specialty 121 – "Software Engineering". Admiral Makarov National University of Shipbuilding, Mykolaiv, 2024

Volume 85 p., 7 tables, 7 figures, 23 references, 5 appendices:

Topic Relevance: Due to the need for a more accurate estimation of the size of server applications based on the Next.js framework, since models for such estimates in this environment have not yet been formed.

Research goal: To improve the accuracy of predicting the size of server applications developed using Next.js by applying a nonlinear regression model.

Object of research: The process of estimating the size of server applications created using Next.js. Subject of research: Nonlinear regression models for estimating the size of server applications based on Next.js.

Research methods: Methods of mathematical statistics, probability theory, as well as methods of linear and nonlinear regression analysis are used.

Scientific contribution: Improved nonlinear regression model for estimating the size of applications on the Next.js framework. The use of logarithm-based normalization allows for improved estimation accuracy compared to existing models for other technologies, such as Java. The new model showed better results according to the criteria R^2, MMRE and PRED(0.25).

Practical significance: A software tool for estimating the size of Next.js server applications, written in Python, has been created. This program can be useful in the early stages of software design. Approbation of results:

Approbation of the thesis results: The main provisions and results of the work were reported and discussed at the VI All-Ukrainian Scientific and Practical Conference "Modern

Information Technologies in Education and Science" ("Uman State Pedagogical University named after Pavla Tychyna", Uman, November 14–15, 2024).

Publications: The main provisions and results of the qualification work were published in 1 scientific paper - conference abstracts.

Keywords: code volume estimation, web applications, nonlinear regression model, framework, Next.js.

Зміст

ВСТУП

Актуальність теми. на початкових етапах розробки будь-якого програмного забезпечення виникає потреба у прогнозуванні трудомісткості створення проєкту, що дозволяє ефективніше планувати ресурси та строки виконання. Одним із ключових показників для такої оцінки, зокрема в моделі COCOMO II [1, 2], є розмір вихідного коду. Точна оцінка цього показника особливо важлива для складних веб веб-застосунків, де навіть незначне відхилення може призвести до помилок у розрахунках часу та витрат.

Останнім часом значно зросла популярність веб веб-застосунків на основі JavaScript, особливо завдяки середовищу Node.js, що використовує Chrome V8. Ця технологія стала основою для багатьох вебплатформ, сервісів та корпоративних систем через її продуктивність, масштабованість і можливість використання однієї мови на фронтенді та бекенді. Фреймворк **Next.js** набув широкого застосування завдяки можливостям рендерингу, зручному налаштуванню маршрутизації та оптимізації продуктивності. Однак, попри популярність, наразі бракує ефективних інструментів для точного оцінювання розміру проєктів на основі Next.js.

Для розробників і менеджерів важливо мати можливість прогнозувати обсяг коду на ранніх стадіях, оскільки це впливає на загальну структуру проєкту, планування тестування та розгортання. Існуючі моделі оцінювання, створені для інших технологій, таких як PHP [3-7], не враховують специфіку Next.js, що може знижувати точність оцінок.

Таким чином, розробка нелінійної регресійної моделі для Next.js є важливим завданням, яке дозволить підвищити точність прогнозування розміру коду, забезпечить краще планування ресурсів та підвищить загальну ефективність розробки сучасних веб веб-застосунків.

Мета та завдання дослідження. Підвищення точності прогнозування розміру (в тисячах рядків коду) веб-застосунків на фреймворку Next.js за допомогою нелінійної регресійної моделі.

- Виконати аналіз існуючих моделей оцінювання розміру веб-застосунків для Node.js-фреймворків.
- Зібрати метрики з проєктів на основі Next.js для побудови моделі.
- Визначити оптимальні методи нормалізації даних (десятковий логарифм, Box-Cox або Johnson).
- Розробити та перевірити нелінійну регресійну модель.
- Створити програму для оцінки розміру (в тисячах рядків коду) веб-застосунків Next.js на Python.

Об’єкт дослідження: процес оцінювання розміру (в тисячах рядків коду) веб-застосунків, що створюються з використанням фреймворку NextJs.

Предмет дослідження: нелінійні регресійні моделі для оцінювання розміру (в тисячах рядків коду) веб-застосунків, що створюються з використанням фреймворку NextJS.

Методи дослідження: методи теорії ймовірностей та математичної статистики, лінійного та нелінійного регресійного аналізу.

Наукова новизна: Удосконалено нелінійну регресійну модель для оцінки розміру (в тисячах рядків коду) веб-застосунків на фреймворку Next.js. Використання нормалізації на основі десяткового логарифму дозволяє підвищити точність оцінювання порівняно з існуючими моделями для інших технологій, таких як Java. Нова модель показала кращі результати за критеріями R^2, MMRE і PRED(0.25).

Практичне значення: Створений інструмент дозволяє оцінювати розміру (в тисячах рядків коду) коду на ранніх етапах розробки, що сприяє ефективнішому плануванню ресурсів.

Особистий внесок здобувача. Кваліфікаційна робота є самостійно виконаною науковою працею. Усі наукові результати отримано автором особисто. У праці [8], яка була опублікована у співавторстві, автору належить лінійна регресійна модель для оцінювання розміру (в тисячах рядків коду) веб-застосунків, що створюються з

використанням фреймворку NextJs, на основі нормалізуючого перетворення десяткового логарифму.

Апробація результатів: Основні положення та результати роботидоповідалися та обговорювалися на VI Всеукраїнській науково-практичній конференції «Сучасні інформаційні технології в освіті і науці» («УМАНСЬКИЙ ДЕРЖАВНИЙ ПЕДАГОГІЧНИЙ УНІВЕРСИТЕТ ІМЕНІ ПАВЛА ТИЧИНИ», м. Умань, 14–15 листопада 2024 р.) [8].

Публікації: Основні положення й результати кваліфікаційної роботи опубліковано у 1 науковій праці – тезах конференції.

1 **АНАЛІЗ ІСНУЮЧИХ МЕТОДІВ ТА МАТЕМАТИЧНИХ МОДЕЛЕЙ ДЛЯ ОЦІНЮВАННЯ РОЗМІРУ ВЕБ-ЗАСТОСУНКІВ ТА ОБҐРУНТУВАННЯ НЕОБХІДНОСТІ ПРОВЕДЕННЯ ДОСЛІДЖЕНЬ ЗА ОБРАНОЮ ТЕМОЮ**

1.1 Аналіз існуючих методів та математичних моделей для оцінювання розміру веб-застосунків

Оцінювання трудомісткості розробки програмного забезпечення (ПЗ) є критичним завданням для управління проєктами. Воно дозволяє точніше планувати ресурси, строки та бюджет. Особливо актуальним це питання стає для сучасних веб-застосунків, які використовують фреймворки на кшталт Next.js [9]. Цей фреймворк, побудований на React, є популярним для створення веб-додатків з підтримкою рендерингу (SSR) та статичного генерування сторінок (SSG). Оцінювання трудомісткості Next.js-веб-застосунків потребує особливого підходу, враховуючи специфічні метрики та фактори, що відрізняють цей фреймворк від інших.

Сучасні методи оцінювання базуються на застосуванні регресійних моделей — лінійних і нелінійних. Лінійні моделі є простими у використанні, але мають обмеження, пов’язані з необхідністю нормального розподілу даних. Нелінійні регресійні моделі дозволяють моделювати більш складні залежності між факторами та результатами. Однією з найвідоміших моделей є COCOMO II [1,2], яка враховує такі фактори, як розмір коду (SLOC), складність проєкту та команда розробників.

Нелінійні моделі надають можливість точніше прогнозувати розмір і трудомісткість розробки ПЗ, враховуючи складні взаємозв'язки між метриками. Для створення такої моделі необхідно обрати відповідні предиктори. Для оцінювання розміру веб-застосунків у цьому дослідженні обрано метрику «кількість рядків коду» (LOC), оскільки вона є більш відповідною для сучасних
інструментів і технологій, таких як Next.js та NestJS. На відміну від метрик, що базуються на функціональних точках (Function Points), LOC безпосередньо відображає обсяг реалізованого функціоналу в термінах фізичного коду. Це дає змогу точно оцінити трудомісткість розробки та впровадження програмного забезпечення,

особливо у випадку високотехнологічних фреймворків із чітко визначеними структурами.

Метрика функціональних точок, хоч і є широко застосовуваною в інженерії програмного забезпечення, має суттєві недоліки щодо оцінювання сучасних веб-застосунків. По-перше, вона орієнтована переважно на традиційні системи з монолітною архітектурою, а сучасні веб-застосунки часто мають розподілену архітектуру з динамічним рендерингом і складною інтеграцією клієнт-сервер. По-друге, використання функціональних точок потребує деталізованого аналізу вимог, що може бути складним і неточним на ранніх етапах розробки.

Натомість метрика кількості рядків коду дозволяє проводити автоматизовані вимірювання, що забезпечує її зручність та об’єктивність. Крім того, дослідження Качмарека та Кухарського [10] виявило, що використання LOC для веб-застосунків із сучасними фреймворками дозволяє знизити похибку оцінювання розміру завдяки врахуванню специфіки кодової бази цих систем.

Застосування метрики «кількість рядків коду» дозволяє уникнути додаткових труднощів, пов’язаних із переведенням функціональних характеристик у числові значення, що часто є суб’єктивним процесом. Використання LOC також узгоджується з попередніми дослідженнями у сфері веб-розробки, де ця метрика показала високі результати точності при оцінюванні розміру застосунків. Таким чином, вона є оптимальним вибором для даного дослідження.Для підвищення точності прогнозування часто використовують нормалізуючі перетворення. Найпоширенішими є перетворення Бокса-Кокса та Джонсона. Перетворення Джонсона типу SB є особливо ефективним для чотиривимірних моделей, що дозволяє адаптувати дані до нелінійної регресії. Однак ці перетворення вимагають додаткових обчислень і параметризації. Натомість десятковий логарифм є простішим у застосуванні й часто використовується у моделях COCOMO II та ISBSG. Для Next.js-веб-застосунків десятковий логарифм може бути обраний як базове перетворення завдяки його простоті та доступності.

У дослідженнях [3-6] трьохфакторні нелінійні регресійні моделі використовувались для оцінки PHP-веб-застосунків. Вони показали, що ключовими предикторами є кількість класів, глибина ієрархії та кількість методів на клас. Для Next.js-веб-застосунків необхідно адаптувати ці моделі, враховуючи особливості React-компонентів та структуру проєкту:

- **Компонентна архітектура**: на відміну від PHP-фреймворків, Next.js використовує компоненти, які можуть бути як простими, так і складними. Оцінка кількості компонентів і їхньої глибини вкладення може бути ключовим фактором.
- **SSR і SSG**: серверний рендеринг додає складності, тому кількість SSR-сторінок може слугувати додатковим предиктором.
- **API-інтеграції**: Next.js часто використовує API-ендпоінти для обробки запитів. Кількість та складність цих ендпоінтів впливає на загальну трудомісткість.

Використання нелінійних моделей дозволяє краще враховувати різноманіття факторів і їхній вплив на загальну складність. Такі моделі можуть передбачати трудомісткість навіть для великих проєктів, де лінійні підходи не дають точних результатів. Нормалізуюче перетворення у вигляді десяткового логарифму дозволяє спростити моделювання, зберігаючи при цьому достатню точність.

У випадку нормальності початкових даних в якості регресійної моделі можна взяти лінійну регресійну модель з наступним загальним виглядом:

$$Y = \hat{Y} + \varepsilon = \bar{Y} + (X^{+})\hat{b} + \varepsilon,$$

де $\hat{Y}$ – результат передбачення рівняння лінійної регресії для значень компонент вектору $X = \{X_1, X_2, \dots, X_k\}$, X^{+} – матриця центрованих змінних, яка містить наступні значення: $X_{1_i} - \bar{X}_1, X_{2_i} - \bar{X}_2, \dots, X_{k_i} - \bar{X}_{k1}$; $\hat{b}$ – оцінка вектору параметрів рівняння

лінійної регресії, $b = \{b_1, b_2, \dots, b_k\}^T$, ε – гаусівська випадкова величина, k – кількість факторів у моделі.

Застосування нормалізуючих перетворень є поширеним методологічним підходом при побудові регресійних моделей для даних, що відхиляються від гаусівського розподілу. У джерелі [11] запропоновано алгоритм формування нелінійних регресійних моделей, який передбачає послідовне виконання наступних етапів: спочатку виконують нормалізацію вхідних даних та вилучення викидів, потім будують лінійну регресійну модель для нормалізованих даних без викидів, і на завершальному кроці з використанням зворотного нормалізуючого перетворення отримують підсумкову нелінійну регресійну модель. Такий підхід дозволяє підвищити точність математичного моделювання та забезпечити коректність інтерпретації результатів у складних статистичних дослідженнях.

Нормалізуюче перетворення негаусівського випадкового вектору P = $\{Y, X_1, X_2, \dots, X_k\}^T$ у гаусівський випадковий вектор T = $\{Z_Y, Z_1, Z_2, \dots, Z_k\}^T$, задається як

$$T = \psi(P), \tag{1.1}$$

а зворотне перетворення до (1.1) як

$$P = \psi^{-1}(T),$$

Тут $\psi -$ вектор, $\psi = \{\psi_Y, \psi_1, \psi_2, \dots, \psi_k\}^T$

Таким чином нелінійна регресійна модель, що побудована з використанням нормалізуючого перетворення [12], має вигляд

$$Y = \psi_Y^{-1}\left[\hat{X}_Y + \varepsilon\right] = \psi_Y^{-1}\left[\bar{Z}_Y + (Z_X^+)\hat{b} + \varepsilon\right],$$

де ψ_Y – перша компонента вектору ψ перетворення (1.1).

1.2 Обґрунтування необхідності проведення досліджень

З моменту появи Node.js його популярність стабільно зростає, привертаючи увагу як невеликих стартапів, так і великих корпорацій. Серед відомих компаній, що впровадили Node.js у свої проєкти, можна відзначити таких гігантів, як Netflix, NASA, eBay, LinkedIn та Uber [13].

Однак розробка на чистому JavaScript може бути досить трудомісткою і повільною. Щоб спростити процес створення веб-застосунків, з'явилися спеціальні фреймворки, які забезпечують стандартизовану структуру та зручні інструменти. Першими серед них стали ExpressJS та KoaJS. Ці фреймворки надають базовий функціонал для роботи з контролерами, маршрутизацією та обробкою запитів, але не забезпечують повну структуру проєкту.

Для вирішення цих викликів були створені більш комплексні фреймворки, які дозволяють швидко розгортати проєкти та легко масштабувати їх. Одним з таких фреймворків для фронтенд-розробки є Next.js. Ця технологія, вперше представлена у 2016 році, швидко здобула популярність завдяки зручності рендерингу (SSR) і статичному генеруванню сторінок (SSG).

У перші роки існування Next.js активно набирав популярність на GitHub, і вже у 2018 році кількість його зірок зросла на понад 300%. Спочатку Next.js не потрапив до основних рейтингів фреймворків на платформі State of JS [14], але згодом обійшов конкурентів за рівнем задоволеності користувачів. Наприклад, у 2020 році статистика показала, що близько 90% розробників високо оцінюють роботу з Next.js.

Процес зростання популярності цього фреймворку детально описано в матеріалах від statisticsanddata.org [15]. На сьогодні Next.js підтримується великим і активним ком'юніті, а також використовується такими відомими компаніями, як Netflix, Twitch і Hulu. Повний список організацій, які впроваджують Next.js у свої проєкти, можна знайти на офіційному вебсайті технології.

Попри зростання популярності Next.js, досліджень щодо оцінювання трудомісткості веб-застосунків на його основі досі недостатньо. Наявні моделі переважно стосуються веб-застосунків, створених за допомогою PHP або Java-фреймворків. Тому розробка моделей прогнозування трудомісткості для Next.js є актуальним завданням.

Під час аналізу останніх публікацій було виявлено, що більшість нелінійних регресійних моделей орієнтовані на PHP-фреймворки або застосунки на Java. Вони використовують предиктори на зразок загальної кількості класів, глибини ієрархії чи кількості методів на клас. Для Next.js-веб-застосунків необхідно адаптувати ці моделі, враховуючи специфіку компонентної архітектури React і особливості SSR.

Щоб підвищити точність прогнозування розміру веб-застосунків, доцільно використовувати нелінійні регресійні моделі з нормалізуючими перетвореннями. Найчастіше застосовуються перетворення Бокса-Кокса або Джонсона. Вони забезпечують високу точність, але потребують додаткових обчислень для підбору параметрів. Альтернативою є десятковий логарифм, який хоч і простіший у застосуванні, але у певних випадках показує задовільні результати.

З огляду на це, для розробки моделей оцінювання розміру Next.js-веб-застосунків пропонується використовувати десятковий логарифм як базове нормалізуюче перетворення. Це дозволить спростити обчислення та забезпечити достатню точність прогнозування трудомісткості.

1.3 Висновки до розділу 1

У цьому розділі представлено аналіз підходів до оцінювання розміру програмного забезпечення та визначено ключові прогалини в існуючих методологіях. Дослідження фокусується на особливостях оцінювання веб-застосунків, розроблених з використанням сучасних веб-фреймворків, зокрема NextJS. Проведений огляд дозволив встановити наступні принципові положення:

- Нелінійні регресійні моделі демонструють вищу точність порівняно з лінійними, що обумовлює необхідність розроблення спеціалізованої нелінійної моделі для веб-застосунків на NextJS.
- Використання нормалізуючих перетворень, зокрема десяткового логарифму, дозволяє підвищити якісні показники математичних моделей.
- В існуючих дослідженнях для оцінки програмного забезпечення найчастіше використовуються метрики концептуальної моделі даних:
    1. Загальна кількість класів
    2. Кількість міжкласових зв'язків
    3. Середня кількість атрибутів класу
- На момент дослідження відсутні спеціалізовані регресійні моделі для оцінювання розміру веб-застосунків на базі NextJS, що підтверджує актуальність запропонованої наукової роботи.

Результати огляду підтверджують потребу в розробці інноваційного підходу до кількісної оцінки обсягу веб-застосунків у сучасному веб-розробленні.

2 ПОБУДОВА НЕЛІНІЙНОЇ РЕГРЕСІЙНОЇ МОДЕЛІ ДЛЯ ОЦІНЮВАННЯ РОЗМІРУ ВЕБ-ЗАСТОСУНКІВ, ЩО СТВОРЮЮТЬСЯ З ВИКОРИСТАННЯМ ФРЕЙМВОРКУ NEXTJS

2.1 Перевірка даних для побудови нелінійної регресійної моделі для оцінювання розміру веб-застосунків, що створюються з використанням фреймворку NextJS, на наявність мультиколінеарності та викидів

Для створення нелінійної регресійної моделі з метою оцінки розміру (в тисячах рядків коду) веб-застосунків, розроблених із використанням фреймворку Next.js, було відібрано 50 проєктів веб-застосунків із відкритим вихідним кодом, розміщені на платформі GitHub [16]. Для збору необхідних даних застосовувалася платформа SonarQube [17], яка дозволила отримати такі метрики: загальна кількість рядків у тисячах (lines), кількість рядків коду (KLOC), кількість файлів (files), кількість компонентів (components), кількість класів (classes) та кількість функцій (functions).

Для побудови регресійної моделі були відібрані три ключові метрики: кількість рядків коду (KLOC), кількість компонентів (components), та кількість функцій (functions). Зібрані дані з метриками представлені у таблиці 2.1.

Таблиця 2.1 – Початковий набір даних разом з розрахованою відстанню Махаланобіса (SMD)

№	LOC	Functions	Components	Smd
1	2	3	4	5
1	6,132	160	173	2,044875983
2	3,172	144	111	0,850926137
3	0,74	22	40	2,064498601
4	1,823	89	115	0,76765769
5	5,591	383	148	1,305483933
6	26,151	2354	176	1,420850238
7	36,797	3081	85	1,446426236
8	9,359	429	66	1,433106558
9	68,076	4283	139	0,322935633

Продовження табл. 2.1.

1	2	3	4	5
10	71,805	4231	133	0,433368769
11	14,439	1337	114	0,45286318
12	2,194	150	85	0,963686925
13	77,197	4563	135	0,505693543
14	2,851	203	101	0,727775506
15	2,285	203	98	0,755991272
16	1,599	846	109	0,567179408
17	4,422	236	123	0,819870257
18	75,984	4637	110	1,189622187
19	132,832	4400	133	0,493744173
20	40,447	2205	114	0,429357898
21	26,682	1378	81	1,178567569
22	12,353	1005	89	0,920285209
23	15,082	1037	60	1,695995494
24	1,997	79	67	1,365298336
25	1,078	78	61	1,51600263
26	7,086	341	46	1,955680238
27	5,893	350	107	0,669336168
28	6,936	337	105	0,679486169
29	5,024	213	54	1,718003261
30	23,205	1142	145	0,961662722
31	52,248	2531	168	1,136174891
32	6,241	417	82	1,037611145
33	57,081	2327	172	1,313596428
34	126,857	5645	198	1,287801145
35	11,766	6046	195	1,16633937
36	381,119	20421	245	4,307364858
37	2,601	1386	148	0,953201206
38	235,783	12239	225	2,124707435
39	39,456	2056	159	1,027504366
40	90,841	4358	185	1,156485709
41	8,231	4134	182	1,119228062
42	236,642	14548	225	2,704745463
43	122,913	6624	196	1,153699493
44	280,853	14384	232	2,633102271
45	65,234	3245	175	1,135314244
46	164,328	8562	210	1,443662602
47	35,642	1876	156	1,002557081

Продовження табл. 2.1.

48	98765	5234	188	1,083898277
49	54,321	2892	170	1,365298336
50	15,973	873	147	1,107237439

Окрім значень обраних метрик в таблиці 2.1 наведені значення квадрату відстані Махаланобіса (SMD). Для кожної багатовимірної точки даних, значення SMD можна знайти за формулою:

$$d_i^2 = (X_i - \bar{X})^T S_N^{-1} (X_i - \bar{X}),$$

де X_i – і-та точка багатовимірних даних негаусівського вектору X; $\bar{X}$– вектор вибіркових середніх; S_N – вибіркова коваріаційна матриця. В нашому випадку розглядається 3 компонентний вектор, N – кількість елементів у вибірці.

При створенні моделі множинної регресії може виникнути проблема мультиколінеарності факторів. Тому перед побудовою нелінійної регресійної моделі для прогнозування розміру (в тисячах рядків коду) веб-застосунків, створених за допомогою Next.js, необхідно проаналізувати предиктори на наявність мультиколінеарності.

Мультиколінеарність — це проблема у регресійному аналізі, яка ускладнює оцінку внеску кожного окремого фактора у залежну змінну. Вона виникає, коли два чи більше факторів у моделі мають сильний кореляційний зв'язок.

Для визначення мультиколінеарності використовують коефіцієнти розширення дисперсії (Variance Inflation Factors, VIFs). У випадку множинної лінійної регресії з k предикторами X_i (де $i = \overline{1,k}$), VIFs — це діагональні елементи зворотної коваріаційної матриці розміром k×k для цих предикторів [18].

Для предикторів "Components" і "Functions", зазначених у таблиці 2.1, значення VIF становлять 2,315 для обох метрик. Оскільки ці значення менші за 5, це свідчить про відсутність мультиколінеарності у даних.

Перевірка вихідних даних показала, що вони не відповідають нормальному розподілу. Тому їх необхідно нормалізувати для виявлення та вилучення викидів. Процес нормалізації передбачає використання спеціальних перетворень, таких як десятковий логарифм, та обчислення квадрату відстані Махаланобіса (SMD), як описано у джерелах [19, 20]. Для кожної точки нормалізованих даних і (де $i = \overline{1, N}$) застосовується така формула:

$$d_i^2 = (Z_i - \bar{Z})^T S_N^{-1} (Z_i - \bar{Z}),$$

де Z_i – і-та точка багатовимірних даних нормалізованого вектору Z; $\bar{Z}$ – вектор вибіркових середніх; S_N – вибіркова коваріаційна матриця для нормалізованих даних. В роботі розглядаємо 3 компонентний вектор.

$$S_N = \frac{1}{N} \sum_{i=1}^{N} (Z_i - \bar{Z})(Z_i - \bar{Z})^T ,$$

Щоб отримати вектор багатовимірних нормализованих даних слід використати (1.1) та нормалізуюче перетворенням десяткового логарифму. В цьому випадку за (1.1) отримаємо перетворення, що має вигляд $\psi = \{\lg Y, \lg X_1, \lg X_2 \}^T$.

Одним з відомим підходів до перевірки даних на викиди є порівняння значення статистики T_{s_i} з квантілем F-розподілу з рівнем значимості α ($F_{m,N-m,\alpha}$)

$$T_{s_i} = \frac{N(N-m)d_i^2}{m(N^2-1)},$$

Якщо для певного рядка значення статистики T_{s_i} перевищує критичне значення критерію Фішера $F_{m,N-m,\alpha}$ або значення SMD перевищує квантиль розподілу χ^2, цей рядок слід вважати викидом [21]. Процес ітеративного вилучення викидів триває доти, доки всі аномальні значення не будуть усунуті.

Після вилучення викидів було виключено 10 проєкти. При розрахунку значень квадрату відстані Махаланобіса (SMD) та статистики T_{s_i} було виявлено, що обидва методи ідентифікують однакові набори даних як викиди.

Для побудови нелінійної регресійної моделі оцінки розміру (в тисячах рядків коду) веб-застосунків, розроблених із використанням фреймворку Next.js, буде використано нормалізований набір даних, що складається з 40 проєктів після видалення викидів.

Як уже згадувалося, для коректного застосування лінійних регресійних моделей необхідно, щоб залишки регресії або залежна змінна відповідали нормальному розподілу. У випадку великих вибірок (понад 30 елементів) доцільно використовувати критерій Пірсона для перевірки нормальності [22].

Значення критерія Пірсона χ^2 визначають як

$$x^2 = \sum_{j=1}^{m} \frac{(n_j - np_j)^2}{np_j}, \quad (2.1)$$

де m – кількість підінтервалів (часток, кліток), на які розбивається інтервал $[x_{min}, x_{max}]$; x_{min}, x_{max} – відповідно мінімальне і максимальне значення випадкової величини X; n_j – абсолютна частота в j-му підінтервалі (кількість значень випадкової величини, які попадають у j-ий підінтервал); p_j – ймовірність того, що значення випадкової величини X попадають у j-ий підінтервал.

Значення кількості підінтервалів m можна розрахувати за формулою Старджеса або формулою Брукса-Карузена. В даній роботі використовується формула Старджеса - $m = \log_2 n + 1 = 3{,}3 \lg n + 1$. Для наступного розрахунку критерія Пірсона візьмемо $m = 6$.

Розрахунок ймовірностей потрапляння значень випадкової величини у j-ий підінтервал можна провести за допомогою наступної формули:

$$p_j = \int_{x_{j-1}}^{x_j} f(x)dx,$$

де x_{j-1} та x_j – ліва і права границі j-го підінтервалу. $f(x)$ – функція щільності ймовірності.

Функція щільності ймовірності нормального розподілу має вигляд

$$f(x) = \frac{1}{\sigma_x\sqrt{2\pi}} * e^{-\frac{(x-m_x)^2}{2\sigma_x^2}},$$

де m_x – математичне сподівання випадкової величини, σ_x – середьоквадратичне відхилення випадкової величини. Після цього в залежності від рівня значимості α та кількості ступенів вільності v за таблицею верхніх 100α %-вих точок розподілу χ^2 знаходять значення $\chi^2_{\text{кр}}$. Якщо $\chi^2 \leq \chi^2_{\text{кр}}$, то з ймовірністю $1 - \alpha$ можна прийняти гіпотезу про те, що закон розподілу результатів спостережень є нормальним, якщо $\chi^2 > \chi^2_{\text{кр}}$ – цю гіпотезу потрібно відкинути.

Значення v – кількість ступенів вільності можна визначити як $v = m - k - 1$,

де k – кількість параметрів, від яких залежить закон розподілу. Для нормального розподілу k = 2.

Для вибірки значень випадкової величини значення статистики дорівнює 16,50 за формулою (2.1). Для 2 ступенів вільності та рівня значимості 0,05 критичне значення становить 5,99. Оскільки розраховане значення перевищує критичне, гіпотезу про нормальність розподілу випадкової величини необхідно відкинути.

Цей результат свідчить про відсутність теоретичних підстав для використання моделі лінійної регресії, яка представляється у вигляді

$$\hat{Y} = \hat{b}_0 + \hat{b}_1 X_1 + \hat{b}_2 X_2 + \hat{b}_3 X_3 + \varepsilon$$

для оцінювання розміру (в тисячах рядків коду) web-застосунків, створених за допомогою фреймворку Next.js. Таким чином, виникає необхідність побудови нелінійної регресійної моделі для оцінки розміру (в тисячах рядків коду) web-застосунків на базі Next.js, використовуючи дані, наведені в таблиці 2.1.

2.2 Побудова нелінійної регресійної моделі для оцінювання розміру веб-застосунків, що створюються з використанням фреймворку NextJS.

Для створення нелінійної регресійної моделі, що прогнозує розміру (в тисячах рядків коду) веб-застосунків на основі фреймворку Next.js, використовуватимемо очищений набір даних без викидів. На основі цих нормалізованих даних побудуємо двофакторну лінійну регресійну модель, яка матиме наступну структуру:

$$Z_Y = \hat{Z}_Y + \varepsilon = \widehat{b_0} + \widehat{b_1} Z_1 + \widehat{b_2} Z_2 + \varepsilon, \quad (2.2)$$

Де $Z_Y = lgY;\ Z_i = lgZ_i, i = \overline{1,2}$

Для визначення параметрів моделі (2.2) застосовується метод найменших квадратів. У результаті його використання отримано наступні значення параметрів: $\widehat{b_0}$ = −1,96457, $\widehat{b_1} = 0{,}15446$, $\widehat{b_2} = 0{,}96852$.

Розрахуємо значення критерію Пірсона для залишків ε лінійної регресії на нормалізованих даних, щоб обґрунтувати застосування лінійних регресійних моделей. Використовуючи формулу (2.1), значення критерію Пірсона χ^2 для лінійної регресії становить 2,783. Критичне значення χ^2 при двох ступенях свободи та рівні значущості 0,05 дорівнює 5,99.

Оскільки $\chi^2 < \chi^2_{\text{кр}}$, нульова гіпотеза про нормальність розподілу залишків ε приймається на рівні значущості 0,05.

Оцінка математичного сподівання залишків ε лінійної регресії для нормалізованих даних становить -3,05e^{-15}, а середньоквадратичне відхилення — 0,0743.

На основі рівняння лінійної регресії (2.8) і зворотного нормалізуючого перетворення побудована нелінійна регресійна модель для оцінки кількості рядків коду веб-застосунків, створених із використанням Next.js:

$$Y = 10^{\varepsilon + \hat{b}_0} X_1^{\hat{b}_1} X_2^{\hat{b}_2},$$

Отриману модель необхідно оцінити за якістю. Для цього використовуються такі метрики, як множинний коефіцієнт детермінації (R^2), середнє значення відносної помилки (MMRE), а також відсоток прогнозів, для яких відносна помилка менша за 0,25 (PRED(0,25)).

Допустимі межі показників:

- MMRE має бути не більше 0,25;
- PRED(0,25)— не менше 0,75;
- R^2 — якомога ближче до 1 [23, 24].

Розраховані значення показників якості моделі:

- R^2=0,9914,
- MMRE=0,1317,
- PRED(0,25)= 0,8000.

Для порівняння також була побудована двофакторна лінійна регресійна модель за умови нормальності розподілу даних.

$$Y = \hat{Y} + \varepsilon = \widehat{b_0} + \widehat{b_1}X_1 + \widehat{b_2}X_2 + \widehat{b_3}X_3 + \varepsilon, \quad (2.3)$$

Для визначення параметрів моделі (2.3), як і для моделі (2.3) було використано метод найменших квадратів та отримані наступні значення параметрів: $\widehat{b_0}$ = −0.131, $\widehat{b_1} = 0.0103$, $\widehat{b_2} = 0.0646$

Для отриманої лінійної моделі були розраховані показники якості, які мають такі значення: R^2 = 0.966, MMRE = 0.1357, PRED(0.25) = 0.712.

2.3 Висновки до розділу 2

У цьому розділі була розроблена нелінійна регресійна модель для оцінювання розміру (в тисячах рядків коду) веб-застосунків, створених із використанням фреймворку Next.js, із застосуванням нормалізуючого перетворення десяткового логарифму.

Також у розділі виконано:

- Пошук проєктів та збір метрик для них за допомогою платформи GitHub і програми SonarQube.
- Перевірку вихідних даних на нормальність, яка показала, що дані не відповідають нормальному розподілу.
- Аналіз на наявність мультиколінеарності серед предикторів, у результаті якого мультиколінеарності не виявлено.
- Пошук і видалення викидів за методом, що базується на нормалізуючих перетвореннях і розрахунку квадрату відстані Махаланобіса. Було ідентифіковано 10 проєктів як викиди.
- Побудову двофакторної нелінійної регресійної моделі для 40 проєктів, які залишилися після видалення викидів, із використанням нормалізуючого перетворення десяткового логарифму.
- Створення двофакторної лінійної регресійної моделі для порівняння з нелінійною моделлю.
- Порівняння моделей: для нелінійної регресійної моделі (2.2) отримані такі показники якості: R^2=0,9914, MMRE=0,1317, PRED(0.25)= 0,8000; для лінійної моделі (2.3): R^2=0.966, MMRE=0.1357, PRED(0.25)= 0.712. Нелінійна модель демонструє покращення значення R^2 та PRED(0.25).

3 РОЗРОБКА ПРОЕКТУ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ОЦІНЮВАННЯ РОЗМІРУ ВЕБ-ЗАСТОСУНКІВ, ЩО СТВОРЮЮТЬСЯ З ВИКОРИСТАННЯМ ФРЕЙМВОРКУ NEXTJS

3.1 Постановка задачі на розробку програмного забезпечення для оцінювання розміру веб-застосунків на основі Next.js

Процес розробки програмного забезпечення включає низку етапів, таких як планування, проєктування та реалізація. Одним із ключових питань є вибір технологій та мови програмування для створення програми. Важливо враховувати, чи потребуватиме користувач додаткового налаштування середовища для запуску застосунку.

Оптимальним рішенням є використання мови Python, яка забезпечує сумісність із популярними операційними системами (Windows, Linux, MacOS). Python — це високорівнева мова з динамічною типізацією та відкритим вихідним кодом. Серед основних переваг вибору цієї мови можна виділити:

- Велику кількість готових рішень завдяки її популярності.
- Широкий набір бібліотек, включаючи NumPy і SciPy для математичних та статистичних обчислень.
- Можливість створення графічного інтерфейсу за допомогою бібліотеки Qt.
- Широкий вибір інструментів для розробки та редагування коду.

Враховуючи зазначені переваги, у межах цього проєкту для розробки програмного забезпечення використовуватиметься Python.

Створення програми для оцінювання розміру (в тисячах рядків коду) веб-застосунків на основі Next.js буде виконано відповідно до технічного завдання (Додаток А). Процес включатиме розробку ескізного, технічного та робочого проєктів.

Вимоги до програми.

Вимоги функціональних характеристик.

Вимоги до складу функцій, що виконуються.

Програма має виконувати такі функції:

1) Вводити з інтерфейсу програми дві метрики – кількість компонентів та функцій, за якими потрібно оцінити розмір ПЗ web застосунків, що створюються з використанням фреймворку NextJS в тисячах строк коду.

2) Виконувати обчислення розміру web застосунка, що створюється з використанням фреймворку NextJS.

Вимоги до складу й параметрів технічних засобів.

Програма має експлуатуватись на обладнанні з параметрами не нижчими за наступні: 64-бітній процесор на архітектурі x86 або ARM, оперативна пам’ять 4Гб (2 Гб мінімум), простір на жорсткому диску 100 Мб мінімум.

Вимоги до інформаційної та програмної сумісності.

Для запуску та роботи з програмним забезпеченням необхідна операційна система Windows 8/10/11, Ubuntu 18/19/20 або Mac OS 10.12/10.13/10.14

На підставі цієї постановки задачі розроблено технічне завдання, яке наведено у додатку Г.

3.2 Розробка ескізного проєкту програмного забезпечення для оцінювання розміру веб-застосунків на Next.js

На основі діаграм варіантів використання та діяльності буде створено концептуальну модель програмного забезпечення.

3.2.1 Побудова моделі варіантів використання застосунку «Next.js Size Estimator»

Діаграма варіантів використання дозволяє відобразити основні функції, які доступні користувачам (акторам) програми. Фокус робиться на описі можливих операцій без деталізації способів їх реалізації.

Оскільки передбачається один тип користувача, діаграма міститиме одного актора. Основні сценарії використання наведено в таблиці 3.1.

Таблиця 3.1 – Виявлені варіанти використання

Основні актори	Найменування	Формулювання
Користувач	Ввести кількість компонентів	Користувач вводить кількість компонентів застосунку
Користувач	Ввести кількість функцій	Користувач вводить кількість функцій застосунку
Користувач	Оцінити розмір (в тисячах рядків коду)	Користувач отримує оцінку розміру застосунку

Діаграму варіантів використання наведено на рисунку 3.1.

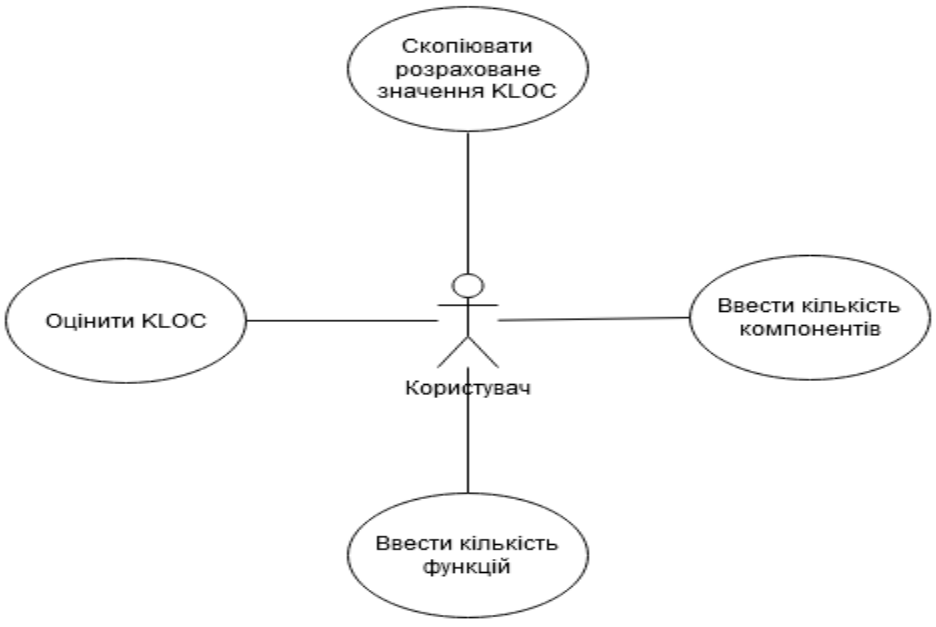

Рисунок 3.1 – Діаграма варіантів використання

Отриману діаграму варіантів використання використовуватимемо в подальшому в проектуванні в ескізному, технічному та робочому проектах.

3.2.2 Специфікація варіантів використання для програмного застосунку «Next.js Size Estimator»

Розглянемо детальні специфікації основних варіантів використання:

1. **Назва:** Введення кількості компонентів

   Актори: Користувач

   Зв'язки з іншими варіантами використання: Відсутні

   Короткий опис: Цей варіант дозволяє користувачеві вказати кількість компонентів веб-застосунку для подальшого оцінювання.

   Основний потік подій: Користувач вводить кількість компонентів у відповідне поле інтерфейсу.

   Спеціальні вимоги:

   - Значення повинно бути натуральним числом.

Подібні специфікації застосовуються для варіантів використання «Введення кількості функцій» та «Введення кількості компонентів». Вхідними даними є натуральні числа.

2. **Назва:** Оцінка розміру

   Актори: Користувач

   Зв'язки з іншими варіантами використання: Відсутні

   Короткий опис: Цей варіант відповідає за виконання розрахунків і надання користувачеві оцінки розміру веб-застосунку. Результат відображається на екрані інтерфейсу.

   Основний потік подій:

   1. Користувач вводить дані (кількість компонентів, функцій, рівень довіри).

2. Система обчислює і показує оцінку розміру веб-застосунку.

Спеціальні вимоги: Відсутні.

3.2.3 Моделювання сценарію роботи застосунку «Next.js Size Estimator» за допомогою діаграми діяльності

Діаграма діяльності відображає послідовність виконання процесів у програмі, акцентуючи увагу на станах і переходах між ними. Вона демонструє алгоритм виконання операцій, де кожен елемент — це набір дій чи обчислень, які виконує система.

Основні елементи діаграми:

- **Стан:** Відображає конкретний етап виконання програми після певної дії.
- **Перехід між станами:** Вказує на зміну стану в результаті виконання дії.

Послідовність операцій для оцінки розміру веб-застосунку, створеного з використанням Next.js, представлена на діаграмі (рисунок 3.2).

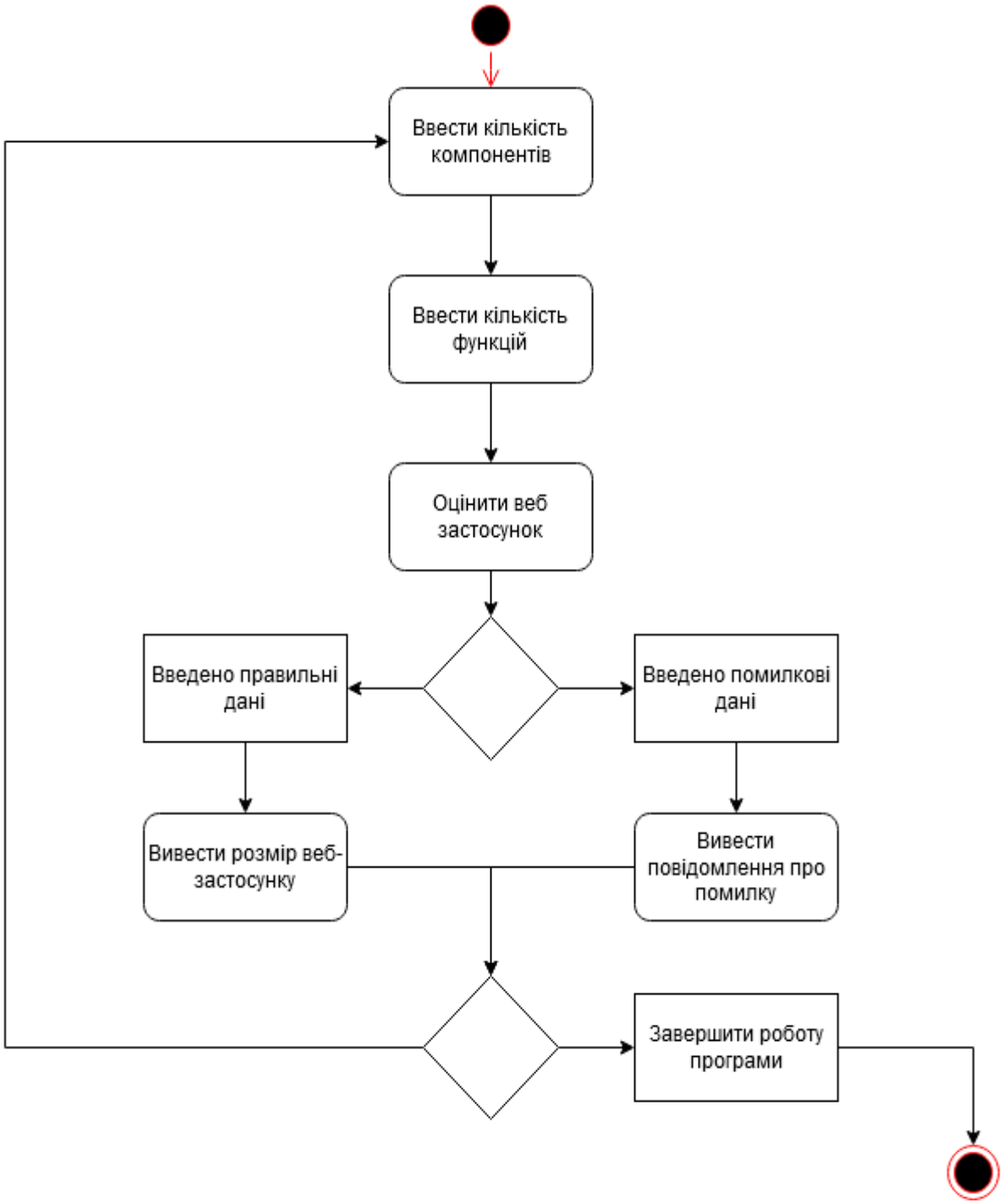

Рисунок 3.2 – Діаграма діяльності

Отримана діаграма в подальшому буде використана при реалізації алгоритмів виконання програмного забезпечення.

3.3 Розробка технічного проекту програмного забезпечення оцінювання розміру веб-застосунків, що створюються з використанням фреймворку NextJS

3.3.1 Статична модель програмного застосунку «NextJS KLOC estimation» у вигляді діаграми класів

Перехід від концептуального проєктування до детальної реалізації системи передбачає створення діаграми класів. Ця діаграма відображає статичну структуру застосунку, демонструючи класи, їхні атрибути та методи, а також взаємозв'язки між ними, пакети та інтерфейси.

На рисунку 3.3 представлена діаграма класів для застосунку, призначеного для оцінювання розміру веб-застосунків, створених за допомогою фреймворку Next.js.

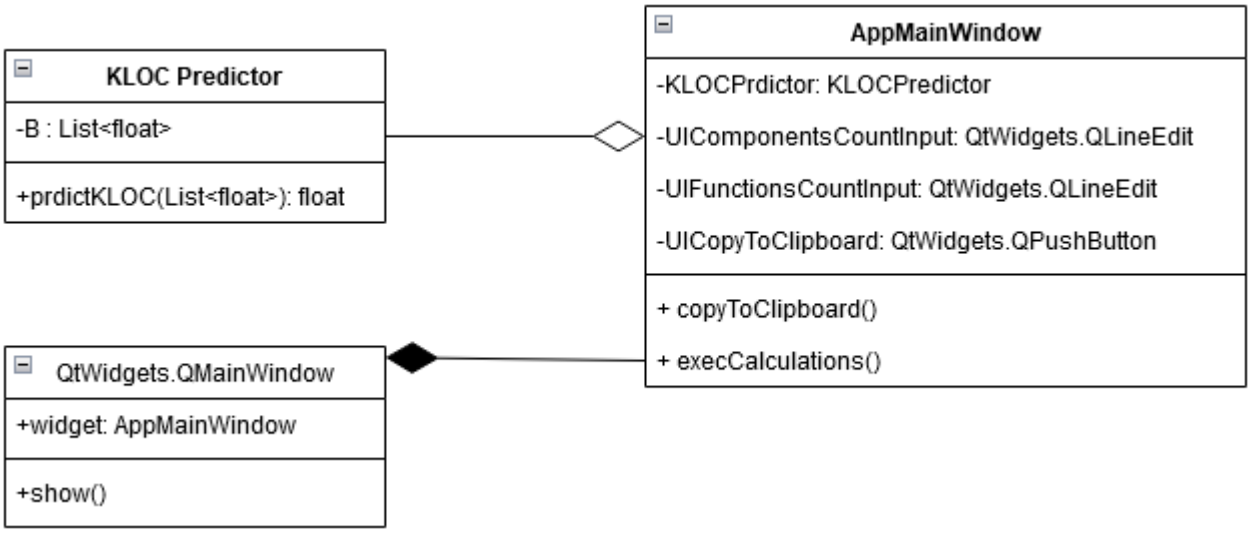

Рисунок 3.3 – Діаграма класів

Побудована діаграма слугуватиме для кодогенерації та безпосередньому написанні програмного коду на етапі реалізації програмного забезпечення.

3.3.2 Специфікації класів програмного застосунку «Next.js KLOC Estimation»

Основні класи програмного застосунку можна описати так:

- AppMainWindow — клас-контролер, що відповідає за обробку дій користувача та введення даних у графічному інтерфейсі.
- QtWidget — базовий клас для компонентів інтерфейсу користувача, побудованих з використанням бібліотеки Qt.
- KLOCPredictor — реалізує логіку оцінювання розміру для застосунку, створеного на основі Next.js.

3.3.3 Динамічна модель програмного застосунку у вигляді діаграм послідовностей

Діаграма послідовностей показує порядок виконання операцій та обмін повідомленнями між об'єктами. Основними компонентами є:

- Об’єкти, що взаємодіють між собою;
- Фокус управління, що показує активність об'єктів;
- Повідомлення, які передаються між об'єктами.

На рисунку 3.4 зображена діаграма послідовності для варіанту використання «Введення кількості компонентів»:

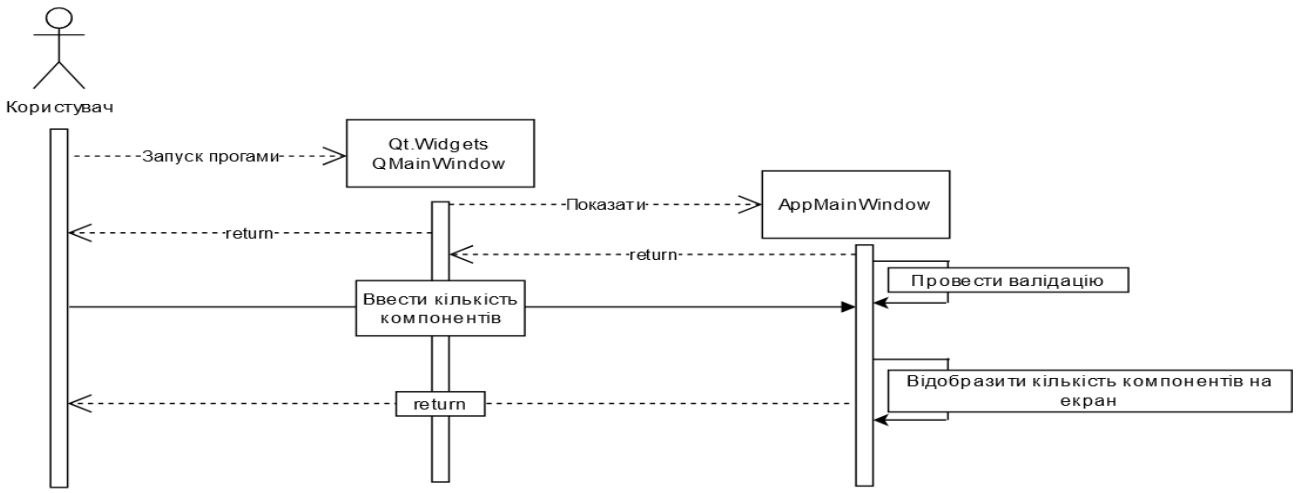

Рисунок 3.4 – Діаграма послідовності для варіанту використання «Ввести кількість компонентів»

Діаграму послідовності для варіанту використання «Оцінити розмір» наведено на рисунку 3.5:

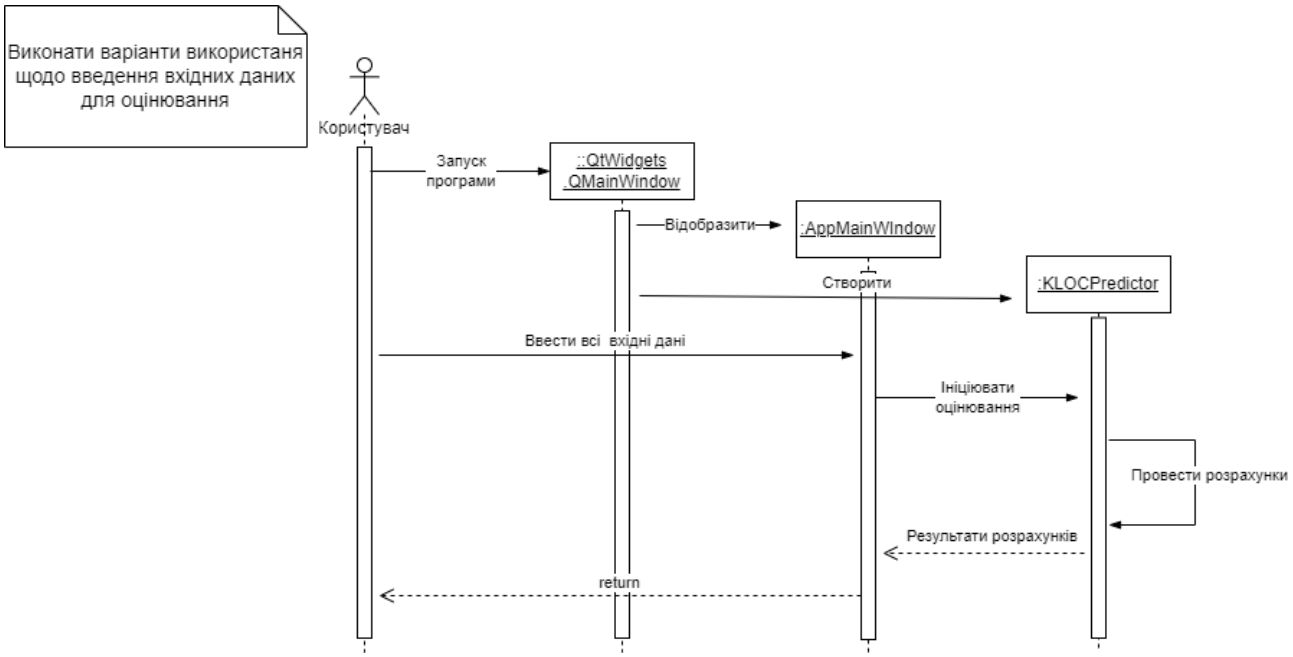

Рисунок 3.5 – Діаграма послідовності для варіанту використання «Оцінити розмір»

Проведене моделювання поведінки для основних варіантів використання полегшить роботу під час написання коду для їх реалазації.

3.4 Реалізація робочого проєкту для оцінки розміру Next.js-веб-застосунків

3.4.1 Кодування програмного застосунку

Основною мовою реалізації обрано **Python**. Нижче наведено приклад основного класу, що виконує необхідні розрахунки:

```
class locPredictor:
    B = [
```

```
        -1.96457,
        0.15446,
        0.96852,
    ]

    def KLOCPredict(self, X):
        return (X[0] ** self.B[1]) * (X[1] ** self.B[2]) *
(10 ** self.B[0])
```

Повний текст програми наведено в додатку Г.

3.4.2 Тестування програмного застосунку

Після завершення розробки коду проведено тестування та підготовлено технічну документацію відповідно до вимог проєкту. Відповідно до документа «Програма та методика тестування» (див. Додаток Б), було здійснено перевірку працездатності розробленого застосунку.

Перелік функціональних можливостей, які тестувалися:

- Введення кількості компонентів для застосунку.
- Введення кількості функцій у застосунку.
- Розрахунок розміру (в тисячах рядків коду) застосунку.

Результати тестування показали, що програмне забезпечення виконує всі функції коректно. На рисунках 3.6 та 3.7 представлено вікно застосунку до та після розрахунку розміру застосунку, створеного з використанням фреймворку Next.js.

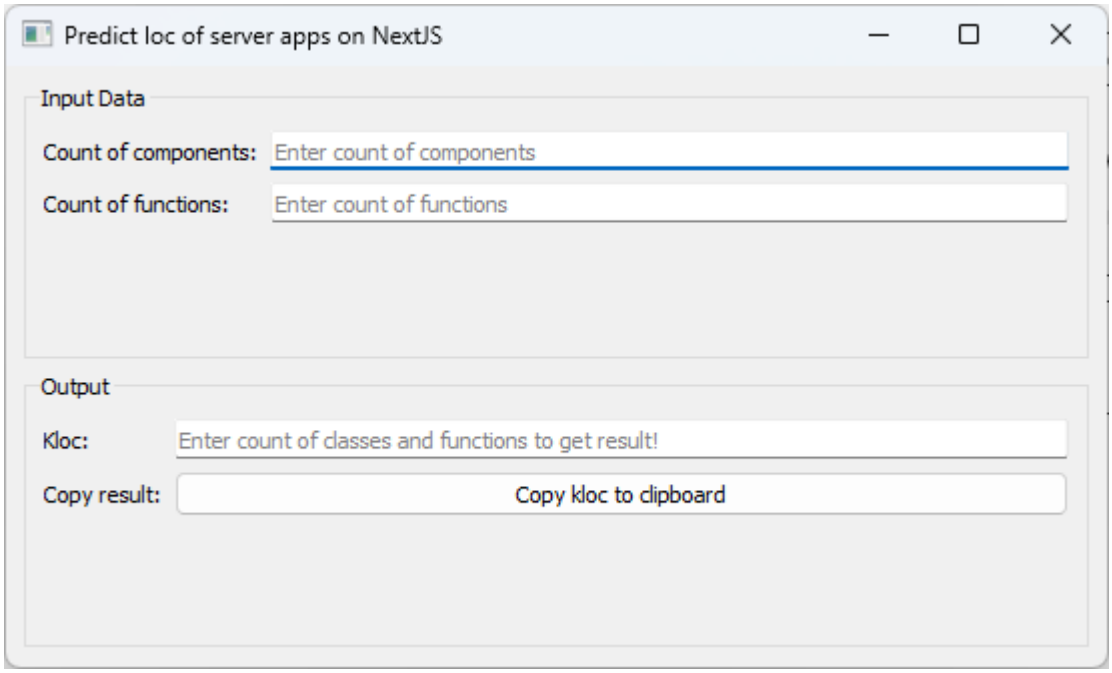

Рисунок 3.6 – Вікно програми до виконання оцінювання

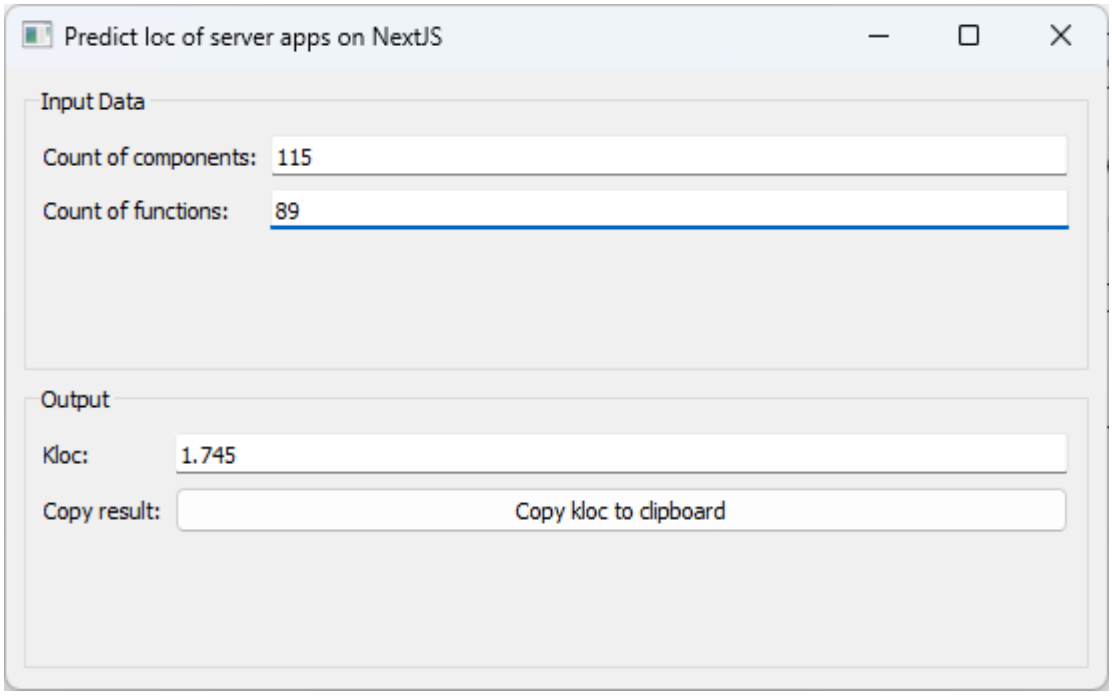

Рисунок 3.7 – Результат оцінювання розміру застосунку, що створюється з використанням фреймворку NextJS

3.5 Висновки до розділу 3

У цьому розділі було виконано проєктування та реалізацію програмного забезпечення для оцінки розміру (в тисячах рядків коду) веб-застосунків, створених за допомогою Next.js. Основні досягнення розділу включають:

- **Вибір інструментів розробки:** Обґрунтовано вибір мови програмування (Python) та інших необхідних технологій.
- **Створення ескізного проєкту:** Визначено основні сценарії використання системи та побудовано відповідні діаграми варіантів використання з описом специфікацій.
- **Розробка технічного проєкту:** Побудовано діаграму класів для представлення статичної структури застосунку. До кожного класу розроблено специфікації, а також створено діаграми послідовностей для ключових сценаріїв.
- **Реалізація робочого проєкту:** Виконано програмування основних компонентів, тестування функціональності та верифікацію результатів. Тестування проводилось із застосуванням методики еквівалентного розбиття.

4 РОЗРАХУНОК ЕКОНОМІЧНОЇ ЕФЕКТИВНОСТІ РОЗРОБКИ І ВПРОВАДЖЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ОЦІНЮВАННЯ РОЗМІРУ ВЕБ-ЗАСТОСУНКІВ, ЩО СТВОРЮЮТЬСЯ З ВИКОРИСТАННЯМ ФРЕЙМВОРКУ NEXTJS

Створення різноманітних інформаційних технологій завжди супроводжується значними капітальними вкладеннями. До них належать придбання обладнання, розробка проєктів, планування робіт, а також підготовка кваліфікованих спеціалістів. Тому важливо проводити економічне обґрунтування доцільності впровадження інформаційних систем. Оцінка ефективності автоматизованих комп'ютерних технологій та систем є ключовим етапом цього процесу.

4.1 Розрахунок витрат на створення й використання програмного забезпечення

Витрату на розробку ПЗ складають витрати на заробітню плату розробника, на амортизацію комп’ютера, на якому розроблюється система, на експлуатацію цього комп’ютера, на інструменти розробки та витратні матеріали і комплектуючі частини комп’ютера.

Розробка ПЗ виконується розробником, місячний оклад якого складає 15000 грн. Додаткова заробітна плата складає 20% від основної. На основі цих даних, основна і додаткова плата становлять 18000 грн. Вартість сучасного комп’ютера складає 20000 грн. (Приведена вартість комп’ютера на базі AMD Ryzen 3 4100).

Вартість розробки ПЗ розраховується за наступною формулою:

$$Спр = (Ззп + Зсз + Ззг + Зк) * Т + Зм$$

де Т – тривалість розробки, міс.;

Ззп – основна і додаткова заробітня плата персоналу, грн.;

Зсз – відрахування на соціальні заходи (38% від основної і додаткової заробітної плати), грн.;

Ззн – загальногосподарські витрати (10% від основної заробітної плати), грн.;

Зе – витрати на електроенергію;

Зм – витрати на основні і допоміжні матеріали.

Зе при споживаній потужності 0,5 кВт, тривалості роботи на місяць, рівної 22*6 = 132 години, вартості кіловат-години електроенергії 4,32 грн(за рішенням кабінету міністрів від 1 вересня 31.05.2024 року) складає:

Зе = 132 * 4,32 * 0,5 = 285,12 грн.

Витрати щодо допоміжних матеріалів приведені в таблиці 4.1.

Таблиця 4.1 – Витрати на допоміжні матеріали

Витрата	Вартість, грн.
Папір	120,00
Картридж принтера	290,00
Непердбачені витрати	300,00
Загальна сума	710,00

Витрати щодо розробки програмного забезпечення наведено в таблиці 4.2.

Таблиця 4.2 – Витрати на розробку системи

Витрата	Кількість\вартість	Одиниця виміру
Тривалість розробки	1	Міс.
Основна та додаткова заробітна плата	18000,00	Грн.
Відрахування на соціальні заходи	6840,00	Грн.
Загальногосподарські витрати	1500,00	Грн.
Витрати на допоміжні матеріали	710,00	Грн.
Витрати на електроенергію	285,12	Грн.

Відповідно до формули, вартість розробки ПЗ оцінюється: Спр = (18000 + 6840 + 1500 + 285,12) * 1 + 710 = 27 335,12грн.

Амортизаційні відрахування на устаткування складають 60% балансової вартості в рік:

Аок = 20000 * 0,6 = 12000 грн.

У масштабах підприємств річні витрати на основні та допоміжні матеріали визначаються у розмірі 5% вартості основного устаткування:

Вм = 20000 * 0,05 = 1000 грн.

Річний обсяг робіт комп’ютера у годинах визначається у наступний спосіб:

Фм = 264,5 * Тз

Де Тз – це середнє місячне завантаження устаткування(приблизно 4 години); 265,5 – середня кількість робочих днів у році.

Отже, річний обсяг роботи комп’ютера складає: Фм = 264,5 * 4 = 1058 годин.

Витрати на електроенргію Зе при 1058 годинах роботи устаткування в рік складуть:

Зе = 1058 * 4,32 * 0,5 = 2 285,28 грн.

Експлуатаційні витрати для комп’ютера за рік складуть: Ззр = 12000 + 1000 + 2 285,28 = 15 285,28 грн.

Отже, у перший рік витрати на розробку й експлуатацію програмного забезпечення складуть:

Зсе = 27 335,12 + 15 285,28 = 42 620,40 грн.

4.2 Економічна ефективність розробки і впровадження програмного забезпечення

Основним показником економічної ефективності функціонування системи є підвищення ефективності керування інформацією у вигляді зниження витрат на керування при одночасному збільшенні швидкості і якості одержання необхыдного результату.

Крім багатьох інших негативних ефектів, ручна обробка інформації спричиняє наступні негативні економічні ефекти:

- високі витрати на складування паперових документів (сейфи, шафи, папки й ін.);
- підвищені витрати на канцтовари;
- витрати, пов'язані з корегуванням раніше допущених помилок (людський фактор).

До числа основних факторів, що визначають приріст прибутку в зв'язку з впровадженням ПЗ, відносяться:

- підвищення продуктивності праці;
- вивільнення робочого часу.

Також, не піддається прямій грошовій оцінці підвищення оперативності керування, якість одержуваних результатів, поліпшення організації праці і т.д. Обов'язковою умовою визначення економічної ефективності ПЗ є порівнянність усіх показників у часі, за цінами й іншими нормами, використовуваними для визначення показників, за змістом і колом елементів витрат.

Визначимо пряму економічну ефективність, ґрунтуючись на тому, що впровадження ПЗ вивільняє 0,6 працівника (за експертною оцінкою фахівців). Зарплата 0,6 працівника в рік складає:

(15000 * 12) * 0,6 = 108000 грн.

Річний економічний ефект розраховується по формулі: Эгод = ΔCn = En * k

Де ΔCn = вивільнені кошти після впровадження програмного забезпечення мінус експлуатаційні витрати, тобто 94111,28 грн.

En – коефіцієнт ефективності(0,6).

k – одноразові витрати на впровадження продукту(41049,60 грн.) . Эгод = 94111,28 – 42 620,40 * 0,6 = 68 539,04 грн.

Строк окупності системи розрахувається наступним чином:

$$\mathbf{T = \frac{42620{,}40}{94111{,}28} = 0{,}45}$$

Отже можна стверджувати, що строк окупності системи складає приблизно 5 місяців.

5. ОХОРОНА ПРАЦІ

5.1 Визначення, цілі, завдання та актуальність питань, пов'язаних з охороною праці

Охорона праці – це система законодавчих актів соціально-економічних, організаційно-технічних і санітарно-гігієнічних заходів, що забезпечують безпеку, збереження здоров'я і працездатність людини в процесі праці []. Загальне керівництво роботою по охороні праці на підприємстві покладається на директора і головного інженера, а безпосереднє керівництво й організацію цієї роботи здійснює особисто головний інженер, а на великих підприємствах – його заступник по охороні праці. У їхньому підпорядкуванні знаходиться відділ охорони праці, на який покладаються наступні основні функції:

- контроль за дотриманням керівниками цехів, відділів і інших структурних підрозділів діючого законодавства, стандартів, правил і норм по охороні праці; за виконанням розпоряджень контролюючих органів і наказів по підприємству;
- розробка заходів щодо створення здорових і безпечних умов праці в цехах, відділах, на споруджуваних об'єктах;
- проведення вступного інструктажу для людей, які надходять на підприємство і контроль за своєчасним і якісним проведенням інструктажу на робочих місцях;
- участь у роботі комісій з перевірки знань інженерно-технічних працівників в області техніки безпеки і виробничої санітарії, по розслідуванню причин аварій і нещасних випадків, по розгляду проектів будівництва, капітального ремонту цехів і устаткування;

- проведення паспортизації санітарно-технічного стану цехів;
- облік потерпілих від нещасних випадків на виробництві, проведення аналізу і складання звітів про виробничий травматизм, контроль за освоєнням коштів і впровадженням заходів, спрямованих на поліпшення умов праці;
- організація виставок і стендів по охороні праці. Поліпшення умов праці сприяє:
- підвищенню ефективності виробництва;
- підтримці працездатності людини на високому рівні і його якнайшвидшому відновленні;
- скороченню виробничого травматизму;
- зменшенню професійних захворювань.

5.2 Аналіз небезпечних і шкідливих факторів, що впливають на людину при роботі з персональним комп'ютером

На багатьох заводах і фабриках виробництво пов'язане з постійним впливом на працівників несприятливих умов. Шкідливі та небезпечні виробничі фактори нерозривно пов'язані між собою. ВПФ – це ті фактори, які в результаті свого тривалого або короткочасного впливу на людину призводять до погіршення стану його здоров'я або до травми. На виробництвах з такими умовами праці різні нещасні випадки відбуваються досить часто.

ВПФ – це фактори, які, діючи на працівника, знижують його працездатність або призводять до різних захворювань, їх часто ще називають професійними хворобами. Варто зазначити, що межа між цими двома групами факторів досить умовна. При деяких умовах шкідливі виробничі фактори

можуть стати небезпечними. Наприклад, підвищена вологість відноситься до несприятливих умов праці, вона може викликати різні захворювання дихальної системи. Якщо людині доводиться в таких умовах працювати з електричним струмом, то це стає вже занадто небезпечно, а не просто шкідливо.

Всі фактори на будь-якому підприємстві можуть мати різне походження. Часто можна стикатися з несприятливими умовами праці, які виникають з вини керівництва. Це питання потребує особливої уваги з боку перевіряючих органів. Хочеться сподіватися, що велика частина небезпечних факторів має природне походження, і людині просто необхідно вжити всі заходи, щоб їх вплив був мінімальним. Всі шкідливі виробничі фактори ГОСТ поділяє на наступні групи:

- Фізичні
- Хімічні
- Біологічні
- Психофізіологічні, до яких можна віднести важеі та напружені умови праці.

Персональні комп’ютери, периферійні пристрої, інше устаткування (апарати управління, контрольно-вимірювальні прилади, світильники), електропроводи та кабелі за виконанням і ступенем захисту мають відповідати класу зони, мати апаратуру захисту від струму короткого замикання та інших аварійних режимів. Під час монтажу та експлуатації ліній електромережі необхідно повністю унеможливити виникнення електричного джерела загоряння внаслідок короткого замикання та перевантаження проводів, обмежувати застосування проводів з легкозаймистою ізоляцією і, за можливості, застосовувати негорючу ізоляцію. Лінія електромережі для живлення персональних комп’ютерів і периферійних пристроїв виконується як окрема групова трипровідна мережа шляхом прокладання фазового, нульового робочого та нульового захисного провідників. Нульовий захисний провідник використовується для заземлення (занулення) електроприймачів.

Не допускається використовувати нульовий робочий провідник як нульовий захисний провідник. Нульовий захисний провідник прокладається від стійки групового розподільного щита, розподільного пункту до розеток електроживлення. Не допускається підключати на щиті до одного контактного затискача нульовий робочий та нульовий захисний провідники. Площа перерізу нульового робочого та нульового захисного провідника в груповій трипровідній мережі має бути не менше площі перерізу фазового провідника. Усі провідники мають відповідати номінальним параметрам мережі та навантаження, умовам навколишнього середовища, умовам розподілу провідників, температурному режиму та типам апаратури захисту. У приміщенні, де одночасно експлуатуються понад п’ять персональних комп’ютерів і периферійних пристроїв, на помітному та доступному місці встановлюється аварійний резервний вимикач, який може повністю вимкнути електричне живлення приміщення, крім освітлення. Персональні комп’ютери і периферійні пристрої повинні підключатися до електромережі тільки за допомогою справних штепсельних з’єднань і електророзеток заводського виготовлення. У штепсельних з’єднаннях та електророзетках, крім контактів фазового та нульового робочого провідників, мають бути спеціальні контакти для підключення нульового захисного провідника. Їхня конструкція має бути такою, щоб приєднання нульового захисного провідника відбувалося раніше, ніж приєднання фазового та нульового робочого провідників. Порядок роз’єднання при відключенні має бути зворотним. Не допускається підключати персональні комп’ютери та периферійні пристрої до звичайної двопровідної електромережі, в тому числі з використанням перехідних пристроїв. Електромережі штепсельних з’єднань та електророзеток для живлення персональних комп’ютерів та периферійних пристроїв потрібно виконувати за магістральною схемою, по 3-6 з’єднань або електророзеток в одному колі. Штепсельні з’єднання та електророзетки для напруги 12В та 42В за своєю конструкцією мають відрізнятися від штепсельних з’єднань для напруги 127В та 220В. Штепсельні з’єднання та електророзетки, розраховані на напругу 12В та 42В, мають візуально (за кольором) відрізнятися від кольору штепсельних

з’єднань, розрахованих на напругу 127В та 220В. Індивідуальні та групові штепсельні з’єднання та електророзетки необхідно монтувати на негорючих або важкогорючих пластинах. Електромережу штепсельних розеток для живлення персональних комп’ютерів і периферійних пристроїв при розташуванні їх уздовж стін приміщення прокладають по підлозі поруч зі стінами приміщення, як правило, в металевих трубах і гнучких металевих рукавах, а також у пластикових коробах і пластмасових рукавах з відводами відповідно до затвердженого плану розміщення обладнання та технічних характеристик обладнання. При розміщенні в приміщенні до п’яти персональних комп’ютерів і периферійних пристроїв допускається прокладання трипровідникового захищеного проводу або кабелю в оболонці з негорючого чи важкогорючого матеріалу по периметру приміщення без металевих труб та гнучких металевих рукавів. Не допускається в одній трубі прокладати ланцюги до 42В та вище 42В. При організації робочих місць операторів електромережу штепсельних розеток для живлення персональних комп’ютерів, периферійних пристроїв і у центрі приміщення прокладають у каналах або під знімною підлогою в металевих трубах або гнучких металевих рукавах. При цьому не допускається застосовувати провід і кабель в ізоляції з вулканізованої гуми та інші матеріали, які містять сірку.

5.3 Розрахунок загального рівномірного освітлення люмінісцентними лампами у виробничому приміщенні для роботи за комп’ютером

Приміщення офісу складається з 2 кімнат. Розміри приміщення складають $a \times b \times H = 15{,}0 \times 25{,}0 \times 3{,}0$м. У приміщенні влаштовано 4 металопластикових вікна (з подвійним склопакетом) розмірами $c \times d = 2{,}0 \times 2{,}5$м. Приміщення зовнішньою стіною спрямовано за азимутом в діапазоні Аз = 46...135 град. Приміщення має сучасний офісний інтер’єр та меблі. Стеля приміщення виконана у вигляді гладкої поверхні пофарбованої у білий колір. Стіни мають гладку поверхню білого кольору із елементами синього. Підлога має покриття із плитки,

що імітує паркет світло-сірого кольору. Вікна приміщення обладнанні світлозахисними пристроями у вигляді нерегульованих вертикальних жалюзі.

Приміщення (рис. 5.1) [23] нараховує 21 робоче місце, обладнаних сучасними персональними комп'ютерами та мабільними лептопами із необхідними перефірійними пристроями для роботи, один лазерний принтер, 1 шредер. Для зберігання робочої документації та науково-технічної літератури передбачена відкрита шафа у вигляді полиць спеціального виготовлення. Також за дозволом адміністрації використовується побутова елекстротехніка (обігрівальна лампа, холодильник, електричний чайник, кавоварка, мікрохвильова піч)

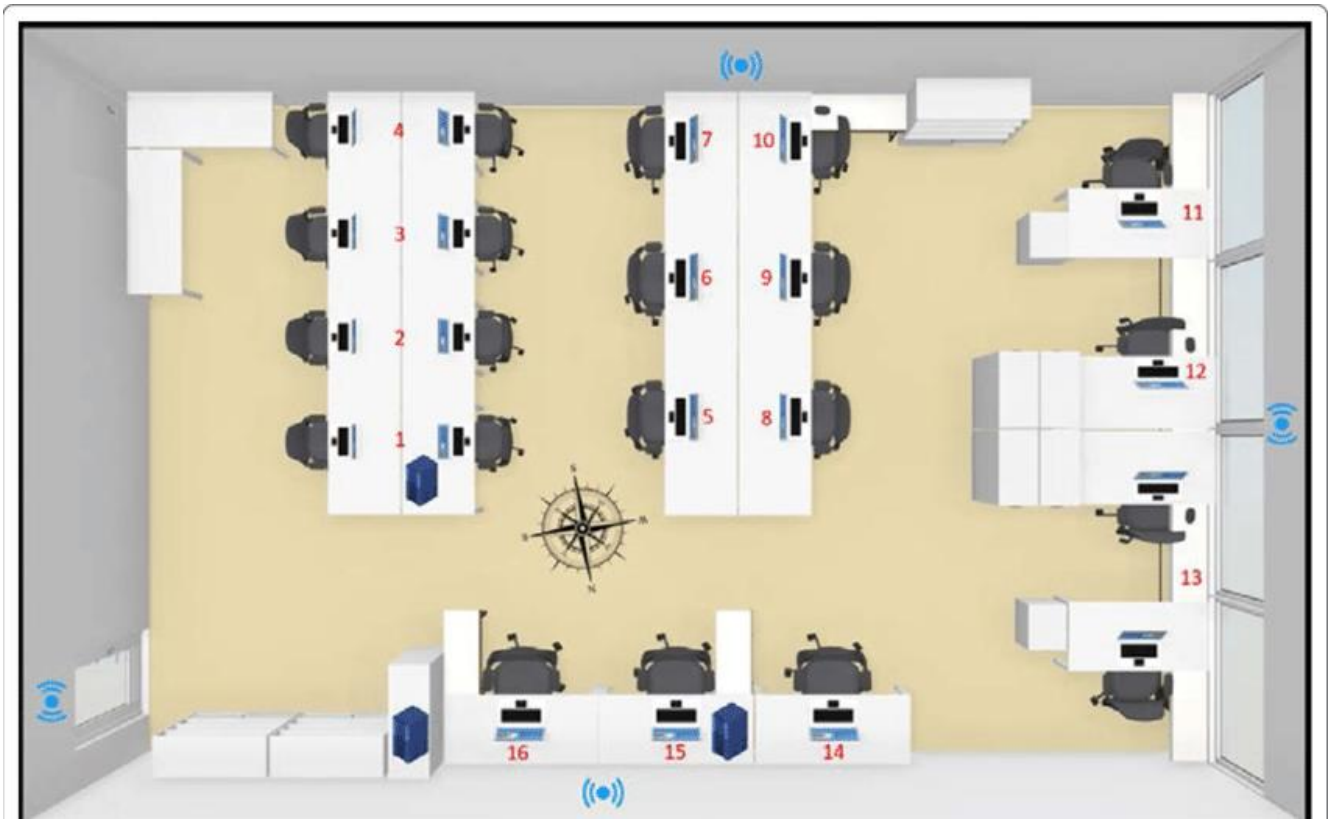

Рисунок 5.1 – Ескіз виробничого приміщення

Напруга джерела живлення електроспоживної техніки – 220 В. Електромережа виконана у вигляді розімкненого типу з дотриманням усіх вимог нормативних документів. За небезпекою ураження електричним струмом приміщення відноситься до приміщень без підвищеної небезпеки ураження електричним струмом. Доступ електропостачання забезпечений майже з будь якої

точки виробничого приміщення для забезпечення найбільш комфортних умов праці співпрацівників компанії.

Мікрокліматичні умови у літній період (частково у перехідний) забезпечується системою кондиціонування, потужності якої може не вистачати для забезпечення комфортних умов праці у період активної спеки та підвищеної температури. У зимовий період опалення здійснюється центральною системою, яка цілком задовольняє умови комфортної праці і виконує поставлену задачу на відмінно.

Деякий дискомфорт, пов'язаний із виробничим освітленням, може відчуватися у певні часи робочого дня. Оскільки вікна виробничого приміщення розвернуті до сонячної сторони деякі робочі місця можуть бути не дуже комфортними для роботи із монітором комп’ютеру

Пожежна безпека в обраному виробничому приміщенні відповідає вимогам нормативних документів, зокрема НПАОП 0.00-1.28-10 (Про затвердження правил охорони праці під час експлуатації електронно- обчислювальних машин).

Вентиляція виробничого приміщення виконується за рахунок штучної робочої комбінованої вентелюючої системи офісів. Рівень шуму системи не перевищує норм та не відволікає від роботи. Повітря циркулює із помірною швидкістю. Вентеляційна система є пожежо та вибухобезпечними, мають просте облаштування та надійну конструкцію.

За даними контрольних обстежень, виконанням необхідних вимірів, а також експертних оцінок здійснена оцінка умов праці (за фактором: виробниче освітлення) в обраному виробничому приміщенні. У таблиці 5.1 наведено відомості, що являються вихідними даними для виконання подальних розрахункових робіт даного розділу кваліфікаційної роботи магістра.

Таблиця 5.1 – Вихідні дані для оцінки умов праці у виробничому приміщенні за фактором: виробниче освітлення

№	Параметри	Розмірність	Позна-чення
1	Довжина приміщення	15	м
2	Ширина приміщення	25	м
3	Висота приміщення	3	м
4	Ширина вікна	2	м
5	Висота вікна	2,5	м
6	Кількість вікон	4	шт.
7	Висота верхнього краю вікна відносно робочої поверхні	3,8	м
8	Відстань розрахункової точки (робочої поверхні) до зовнішньої стіни	2	м
9	Висота карнизу протилежної будівлі відносно Підвіконня	1,5	м
10	Відстань до затіняючої будівлі	60	м
11	Орієнтація світлових прорізів (ОСП) за сторонами горизонту	ОСП	–
12	Точність зорових робіт	Т	–
13	Вид світлопропускного матеріалу	–	–
14	Вид віконної рами	–	–
15	Сонцезахисні пристрої	–	–
16	Стан стелі	–	–
17	Стан стін	–	–
18	Стан поверхні підлоги	–	–
19	Кількість робочих місць	25	шт.

Для перевірочного розрахунку необхідно виконати підготовку даних та обрахунок значень.

Нормований коефіцієнт природного освітлення для відповідної (заданої) категорії зорової роботи eН, %. зорові роботи середньої точності (IV розряд); при цьому l_{min} = 0,5...1 мм і e_H = 1,5 %

Коефіцієнт світлового клімату *m*. Для Миколаївської області (як і для решти територій України, окрім Одеської обл. та АР Крим) при орієнтації світлових прорізів у зовнішніх стінах будівель на північ (ПН) *m* = 0,9.

Нормований коефіцієнт природного освітлення для розглянутих умов праці.

$e_N = e_н \times m$.

e_N – нормований коефіцієнт природного освітлення для розглянутих умовпраці; $e_н$ – нормований коефіцієнт природного освітлення для відповідної (заданої) категорії зорової роботи;

m – коефіцієнт світлового клімату.

Коефіцієнт запасу, що приймається за розрахунками природного освітлення. Для приміщень з нормальними умовами праці (кабінети та робочі приміщення, лабораторії, навчальні приміщення і т. ін.) при боковому освітленні і рекомендованій кількості чищень $n_ч$ = 1, коефіцієнт запасу складає $к_з$ = 1,2.

Геометричні співвідношення, що характеризують виробниче приміщення та розташування робочого місця в ньому: *a/b, b/h, l/b*.

a/b = 1,6;

b/h = 5;

l/b = 0,13;

a – довжина приміщення; b – ширина приміщення; h – висота приміщення;

l – відстань розрахункової точки (робочої поверхні) до зовнішньої стіни.

Світлова характеристика вікна. Виконується за допомогою двумірної лінійної інтерполяції.

$$h_в = f\,(a/b,\ b/h,\ l/b),$$

Коефіцієнт світлопропускання матеріалу, коефіцієнт, що враховує втрату світла у віконних рамах, коефіцієнт, що враховує втрати світла у несучих

конструкціях, коефіцієнт, що враховує втрату світла у сонцезахисних пристроях відповідно:

$\tau 1 = 0.8$;

$\tau_2 = 0{,}78$;

$\tau_3 = 0{,}75$;

$\tau_4 = 1$;

Загальний коефіцієнт світлопропускання. Розраховується на основі попередніх коефіцієнтів.

$$t_{заг} = t1 \times t2 \times t3 \times t4,$$

t1–коефіцієнт світлопропускання матеріалу;

t 2 –коефіцієнт, що враховує втрату світла у віконних рамах;

t3 –коефіцієнт, що враховує втрати світла у несучих конструкціях;

t 4 –коефіцієнт, що враховує втрату світла у сонцезахисних пристроях.

Розрахункове значення середньозваженого коефіцієнта відбивання внутрішніх поверхонь виробничого приміщення $\rho_{сер} = 0.3$, оскільки досліджується виробниче приміщення.

Площа підлоги виробничого приміщення

$$S_{підл} = a \times b,$$

a – довжина приміщення; b – ширина приміщення.

Коефіцієнт, що враховує підвищення коефіцієнта природного освітлення за рахунок світла, яке відбивається від внутрішніх поверхонь приміщення.

$$r1 = f(\rho сер, a/b, b/h, l/b),$$

a – довжина приміщення; b – ширина приміщення; h – висота приміщення;

l – відстань розрахункової точки (робочої поверхні) до зовнішньої стіни.

$P_{сер}$ – розрахункове значення середньозваженого коефіцієнта відбивання внутрішніх поверхонь виробничого приміщення.

Відношення відстані між протилежними будівлями до висоти карнизу протилежного будинку над підвіконням.

$$D/H^1,$$

D – відстань до затіняючої будівлі;

H^I – висота карнизу протилежної будівлі відносно підвіконня.

Коефіцієнт, що враховує вплив протилежної будівлі на освітленість у виробничому приміщенні

$$Кбуд = f\,(D/H^I),$$

Площа вікон, що необхідна для забезпечення нормованої природної освітленості у виробничому приміщенні

$$S_{\text{в}} = \frac{e_N * k_{\text{з}} * n_{\text{в}} * S_{\text{підл}} * k_{\text{буд}}}{\tau_{\text{заг}} * r_1 * 100},$$

e_N – нормований коефіцієнт природного освітлення для розглянутих умовпраці;

$k_{\text{з}}$ – Коефіцієнт запасу, що приймається за розрахунками природного освітлення;

$n_{\text{в}}$ – кількість чищень;

$S_{\text{підл}}$ – площа підлоги;

$k_{\text{буд}}$– коефіцієнт, що враховує впливпротилежної будівлі на освітленість увиробничому приміщенні;

$\tau_{\text{заг}}$ – загальний коефіцієнт світлопропускання;

r_1– коефіцієнт світлопропускання матеріалу.

Після завершення підрахунків необхідної площи вікон для виробничого приміщення необхідно порівняти її із фактичною площею. Фактична площа складає 5 м^2 , а розрахункова складає 2 м^2. Виходячи з цього можна стверджувати, що фактична площа відповідає нормованій і навіть перевищує її 2,5 рази. Для запобігання такого перевищення у виробничому приміщенні можна використовувати більш плотні штори або жалюзі.

5.2 Розрахунок загального рівномірного освітлення люмінісцентними лампами у виробничому приміщенні

Оскільки приміщення має гарну природню освітлюватність варто разрахувати систему штучного освітлення, тобто потужність джерел світла. За основу взято метод коефіцієнту використання світлового потоку.

Висота світильника над підлогою. Оскільки стеля є підвісною то зовнішня площина світильника співпадає з площиною стелі(рисунок 5.2).

$h_0 = H - h_c$

Н – висота стелі;

h_c – зовнішня площина світильника;

h_0 – висота світильника над підлогою.

Висота світильника над робочою поверхнею

$$h = h_0 - h_p \; h = 3{,}27 \text{ м},$$

h_0 – висота світильника над підлогою;

h_p – висота робочої поверхні

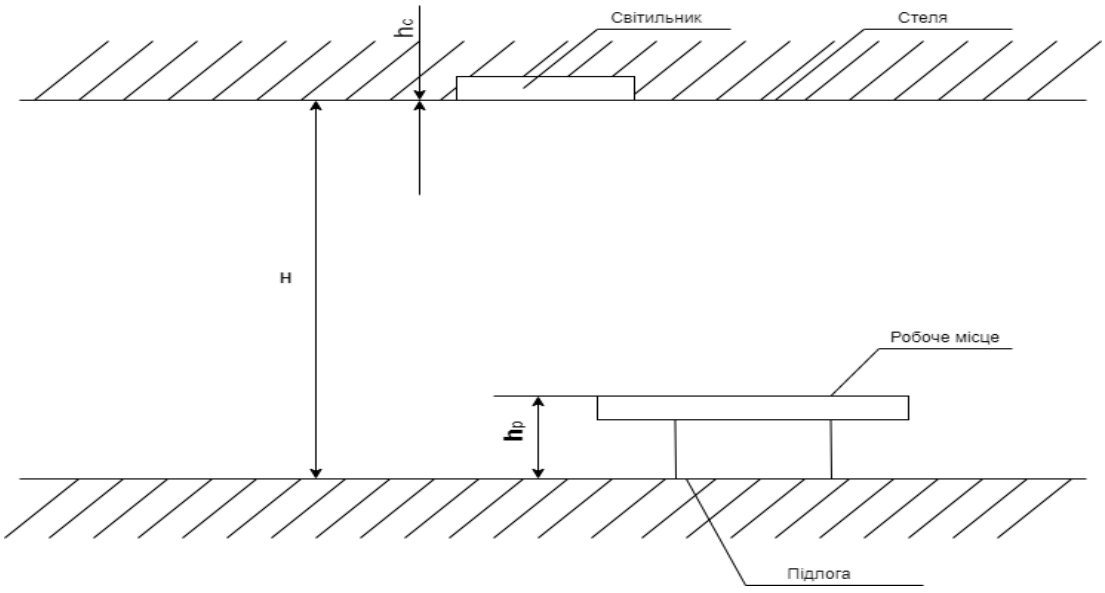

Рисунок 5.2 – Схема для розрахунку штучного освітлення

Коефіцієнт відбиття внутрішніх поверхонь виробничого приміщення.

$$\rho_{стелі} = 85\%; \rho_{стін} = 42\%.$$

Осереднене значення коефіцієнтів відбиття стелі та стін.

$r = 0,5 \times (r_{стелі} + r_{стін})$,

r = 45%,

Коефіцієнт використання світлового потоку.

h = *f* (*i*, r),

h = 64%,

i – характеристика приміщення;

r - осереднене значення коефіцієнтів відбиття стелі та стін.

Нормоване значення освітленості. $E_H = 200$.

Коефіцієнт запасу, що використовується при розрахунку штучного освітлення. Для приміщень з нормальними умовами праці (кабінети, лабораторії, та навчальні приміщення і т. ін.) при рекомендованій кількості чищень $n_ч = 1–2$ коефіцієнт запасу складає: $к_з = 1,4$.

Коефіцієнт нерівномірності світлового потоку. Згідно з рекомендаціями при освітленні люмінесцентними лампами: Z = 1,1.

Площа підлоги виробничого приміщення.

Sпідл = *a* × *b*,

Sпідл = 375 м^2

a – довжина приміщення, b – ширина приміщення

Світловий потік однієї лампи $Ф_л = 2200$ лм.

Необхідна кількість світильників у виробничому приміщенні прикількості *n* (шт.) ламп у кожному *N*, шт.

$$N = \frac{E * k_з * S_{підл} * Z * 100}{n * Ф_з * h},$$

N = 10,24 шт.

E – нормоване значення освітленості;

$к_з$ – коефіцієнт запасу;

Sпідл – площа підлоги виробничого приміщення; Z – коефіцієнт нерівномірності світлового потоку;

h – коефіцієнт використання світлового потоку;

$Ф_з$ – світловий потік однієї лампи;

n – кількість ламп у світильнику.

Оскільки отриманий результат прийнято округляти до більшого числа, а при розташуванні світильників у два ряди до більшого парного числа, то за результатами підрахунків виробничому приміщенню необхідно 12 світильників для забезпечення найкращої штучної освітлюваності робочих місць при умові, що потужність однієї лампи 40 Вт, а кількість ламп у світильнику складає 8 шт.

Потужність загального штучного освітлення люмінесцентними лампами у виробничому приміщенні.

$$Рзаг = Рл \times N \times n,$$

$P_л$ – потужність однієї лампи світильника;

N – кількість світильників;

n – кількість лам у одному світильнику.

$Р_{заг}$ = 3840Вт

Після розрахунку необхідної кількості світильників для виробничого приміщення підраховано потужність загального штучного освітлення люмінісцентними лампами, що складає 3840 Вт при умові, що у приміщенні 12 світильників.

6. ОХОРОНА НАВКОЛИШНЬОГО СЕРЕДОВИЩА

6.1 Вплив людини на навколишнє середовище

Охорона навколишнього природного середовища є системою державних і суспільних заходів, направлених на відтворювання збереження і раціональне використання природних ресурсів, поліпшення полягання природного середовища, і є частиною прикладної екології. Відносини у галузі охорони навколишнього природного середовища в Україні регулюються Законом України № 1268-XII від 26.06.91р. «Про охорону навколишнього природного середовища» [38], а також розроблюваними відповідно до нього земельним, водним, лісовим законодавством, законодавством про надра, про охорону атмосферного повітря, про охорону і використання рослинного і тваринного світу та іншим спеціальним законодавством. В міру прискорення темпів науково-технічного прогресу дія людей на природу стає все більш потужною. І в даний час воно вже сумарно із дією природних чинників, що приводить до якісної зміни співвідношення сил між суспільством і природою. На сучасному етапі людство поставлено перед чинником виникнення в природі незворотних процесів, нових шляхів переміщення і перетворення енергії та речовини. В природу втручається все більше і більше нових речовин, чужих їй, часом сильно токсичних для організмів. Частина з них не включається в природний колообіг і накопичується в біосфері, що приводить до небажаних екологічних наслідків. Накопичення промислових відходів сприяє підвищенню захворюваності людей і тварин, прискоренню корозії машин і металевого устаткування, зниженню врожайності сільськогосподарських культур і продуктивності тваринництва, прискореному і нераціональному використовуванню природних ресурсів і енергії, погіршенню багатьох властивостей екологічних систем, загибелі деяких унікальних природних територіальних комплексів, зникненню окремих видів тварин і рослин. Основну загрозу для навколишнього середовища в межах галузі розробки програмного забезпечення становлять комплекси електронно-

обчислювальних машин (надалі ЕОМ), а саме електромагнітні випромінювання від них. Дослідження останніх років показали, що електромагнітні випромінювання вищезгаданих електронних пристроїв містять торсіонову компоненту, що несе інформацію про процеси, які відбуваються в тому чи іншому електронному пристрої. Торсіонові поля мають високу проникаючу здатність і їх неможливо екранувати. Останніми роками при проектуванні (конструюванні), виготовленні (будівництві) і експлуатації технічних і систем управління в різних сферах діяльності надзвичайно широко застосовуються персональні комп'ютери і всілякі комп’ютерні програми.

Робота з комп'ютерами розробників, операторів і інших користувачів пов'язана з додатковими шкідливими і небезпечними чинниками, що негативно впливають на організм. Наприклад, шкідлива дія на зір надає не при тримання стандартних візуальних ергономічних параметрів екрану, розмір мінімального елементу відображення, мерехтіння зображення, відбивна здатність (відблиски) та ін. Працюючий комп’ютер створює електромагнітне поле, що шкідливо діє на організм людини. Це поле може викликати радіоперешкоди, тобто заважати роботі радіо і телеаппаратури, що призводить до зниження надійності технічної системи або системи управління, до збільшення ризиків виникнення аварійної ситуації у виробничому середовищі. Для забезпечення безпеки роботи з комп’ютером розроблені і повинні всюди застосовуватися стандарти на монітори, вимоги до приміщень для експлуатації комп'ютерів, ЕОМ загалом і до організації і устаткування робочих місць.

6.2 Утилізація відходів електронної та комп’ютерної техніки

Використання комп'ютерів вимагає вирішення таких важливих питань, як утилізація відходів (мікросхеми з вмістом кольорових металів, плати, дискети). При утилізації старих комп'ютерів відбувається їх розділ на сім груп: метали, пластмаси, штекери, дроти, батареї, стекло. Жодна деталь не йде для повторного використання, оскільки не можна гарантувати їх надійність, але у формі

вторинної сировини вони йдуть на виготовлення нових комп'ютерів або інших пристроїв та ЕОМ.

Детально розглянемо декілька прикладів переробки відходів обчислювальної техніки. Гадолінієво-галлеві гранати (ГГГ) використовуються у виробництві компонентів пристроїв, що запам'ятовують. В ході обробки біля 80% вихідного матеріалу перетворюється на відходи або є браком. ГГГ мають високу вартість і їх виділення з відходів представляє інтерес з економічної точки зору.

При здобутті досить чистих продуктів можливі повторне їх використання як вихідного матеріалу. При цьому значно підвищується економічність виробництва заготівель з ГГГ. Під терміном відходи маються на увазі кристалічні залишки, а також дрібний порошок, що виходить при різанні, шліфовці, поліровці кристалів граната або подібних матеріалів. Переробка цих відходів протягом ряду останніх років викликає труднощі і не вирішена до сьогодні. Всі попередні спроби були безуспішними із-за низької розчинності цих складних оксидів.

Вдосконалений процес, розроблений Е.Гуссетом (патент США 4-198231 від 15 квітня 1980 року, фірма «Свісс Алюміній Лтд.», Швейцарія), призначений для виділення галію і гадолінія з відходів, що містять обидва ці елементи у вигляді оксидів або з'єднань, що перекладаються в оксиди. Відходи дрібно подрібнюються і потім розчиняються в сильних мінеральних кислотах. Гадоліній 61 осідає з очищених розчинів у вигляді оксалата, галій виділяється в металевому вигляді електролітично. Електролітичне виділення галію може проводитися до виділення гадолінія у вигляді оксалата з розчину.

Розглянемо приклад проведення такого процесу. Відходи піддаються переробці, є залишки завантаження в пристрої для зростання кристалів, роздроблені частини кристалів зі всіх стадій переробки, дрібнозернисті порошки і пудри після операції різання, шліфовки і поліровки гранатів GdGaO. Подрібнений порошок після обробки кристалів ГГГ висушується при 1200 С і

потім нагрівається при 6000 С протягом декількох годин для розкладання летких забруднень. Дрібнозернисті відходи в кількості 1000 г розміром менше

40 мкм., що містять 34% галію і 46% гадолінія кип'ятяться із зворотним холодильником протягом двох годин в 2100 мл 35%-ої соляної кислоти. Це відповідає 99% завантаження. Після кип'ятіння частина відходів, що не розчинилася, фільтрується та промивається. Після висушування залишок важить 20 р. Залишки такого типа об'єднуються і піддаються повторній обробці кип'ятінням. Фільтрат, що включає промивні води, об'ємом 2300 мл, обробляється 4000 г металевого галію при 500 С в течії 45 хвилин. Метал має максимально диспергувати. В ході цієї операції благородніші елементи, присутні в розчині, виділяються і частково розчиняються в галії до його насичення і далі охолоджуються у вигляді інтерметалевих включень або в елементарній формі. Відділення можна проводити з меншою кількістю галію. Метал може періодично замінюватися на нову порцію до досягнення необхідної міри очищення. В результаті процесу очищення виходить розчин з вмістом галію 140 г/л і гадолінія 190 г/л.

Встановлюється величина ph= 1 шляхом додавання 900 мл 4%-ного розчину перекису водню для окислення домішок заліза. Осадження гадолінія проводиться при 500 С шляхом додавання 1500 г кристалічної технічної щавлевої кислоти СОН*2НО; суспензія акуратно перемішується 12 годин для повторного осадження у вигляді оксалата гадолінія. 62 Оксалат гадолінія Gd(CO)*10HO відділяється центрифугуванням, промивається 20 мл розбавленої щавлевої кислоти (6 г/л) і висушується при 1300 С; перетворення на оксид гадолінія досягається підігрівом при 8000С. Подальше очищення може проводитися розчиненням в кислоті і осіданням у вигляді окасалата гадолінія. До рідини після центріфугованія і промивань осаду (3300 мл) додають 2300 г КОН при інтенсивному перемішуванні до рн= 12,6000 мл розчину з вмістом галію 55 г/л і гадолінія 1 г/л піддається електролізу при 600 Із з використанням катода з нержавеющей сталії і графітового анода при щільності струму близько 0,1 А/кв.см. Після 48 годин осідає 325 г галію і залишається розчин з вмістом

0,4 г/л, що не піддається подальшій переробці. Збагачений метал має чистоту 99,99% і може бути безпосередньо перетворений в оксид. При експлуатації персонального комп'ютера витрачаються наступні ресурси:

- електроенергія;
- папір для принтера;
- картріджи з фарбувальною стрічкою.

Для раціонального використання електроенергії не слід залишати включеним комп'ютер і принтер, якщо відсутня потреба у їх використані на тепершній момент часу. Друкувати можна з двох сторін. Витрати на папір навряд чи вдасться скоротити удвічі, проте економія буде вельми істотною. Також можна спробувати спеціальну бумагу, що виготовляється с пластику, але є більш дорогою за ціною. Проблему з утилізацією паперових відходів може вирішити вторинна переробка. Для повторного використання картріджів з фарбувальною стрічкою необхідно купити пристрій для просочення стрічок і тоді картріджи можна буде використовувати від 20 до 35 разів.

Сучасна технологія виготовлення елементів засобів обчислювальної техніки (СВТ) дозволяє досягти дуже низького рівня відмов елементів під час експлуатації. У зв'язку з цим відпадає необхідність проведення ремонтних робіт на місці експлуатації сучасних засобів обчислювальної техніки і як наслідок не утворюються зайві відходи (несправні мікросхеми), що містять дорогоцінні і рідкоземельні метали. Природно, в сервісних центрах, що спеціалізуються на ремонті і технічному обслуговуванні СВТ, має бути організований збір і облік матеріалів, що містять коштовні метали, з подальшою обробкою цих матеріалів на спеціалізованих підприємствах з метою їх вилучення. У зв'язку з тим, що виробництво сучасних компонентів інформаційних технологій сьогодні знаходиться в на початковому рівні, СВТ складаються з парку імпортних машин та іншого устаткування. Через відсутність інформації про вміст коштовних металів в елементах устаткування, суворий облік не представляється можливим і має бути покладений на фахівців експортних фірм. В умовах ринкової економіки підприємства мають бути самі зацікавлені у вторинній переробці, що

містять дорогоцінні метали вузлів і елементів за умови неможливості їх використання. Сумарна вага дорогоцінних металів в комп'ютері типа IBM PC/XT, використовуваному в розробці даної кваліфікаційної роботи складає: золото – 0,22958 г; срібло – 5, 091346 г.

Технологічний процес відділення дорогоцінних металів з друкарських плат здійснюється за наступною схемою. Друкарські плати сортуються по переважанню в них кількості дорогоцінних металів, роздроблюються і подрібнюються, обпалюються і плавляться. В процесі випалення перолітичному розкладанню піддається пластмасова основа, а основа дорогоцінних металів у вигляді металевих залишків відновлюється до оксидів. Металевий залишок подрібнюється, гранулюється, обробляється через магнітну сепарацію і відбувається відділення магнітних від немагнітних часток. Отриманий таким чином порошок, розділений по видах дорогоцінних металів, у вигляді гранул розплавляється в індукційних плавильних печах з подальшим розділенням кожного металу окремо

ВИСНОВКИ

У результаті кваліфікаційної роботи було підвищено точність оцінювання розміру веб веб-застосунків, створених за допомогою фреймворку Next.js, завдяки розробці нелінійної регресійної моделі. У процесі дослідження досягнуто таких результатів:

– Проведено аналіз наявних регресійних моделей для оцінювання розміру програмного забезпечення на мовах Java, PHP та відповідних фреймворках. Виявлено, що для JavaScript та його веб-фреймворків, таких як Next.js, подібні моделі відсутні.

– Виконано пошук проєктів з відкритим кодом на GitHub, що використовують Next.js. Для збору необхідних метрик застосовано інструмент SonarQube.

– Перевірено вихідні дані на наявність викидів. Оскільки дані не підпорядковуються нормальному розподілу, використано метод нормалізуючих перетворень і обчислення квадрата відстані Махаланобіса для усунення викидів.

– Розроблено двофакторну нелінійну регресійну модель для оцінювання розміру (в тисячах рядків коду) веб-астосунків на Next.js. Модель продемонструвала високі показники якості (R^2, MMRE, PRED(0,25)).

– Розроблено програмне забезпечення для оцінювання розміру (в тисячах рядків коду) веб-застосунків на основі Next.js. Тестування показало, що програма відповідає всім вимогам.

– Виконано оцінку економічної ефективності впровадження створеного ПЗ. Результати підтвердили доцільність його використання.

– Розроблено рекомендації з охорони праці та захисту навколишнього середовища.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Trendowicz A., Jeffery R. Software project effort estimation. foundations and best practice guidelines for success. Springer International Publishing, 2014. 491 p. DOI: https://doi.org/10.1007/978-3-319-03629-8.
2. Boehm B. W. Software engineering economics. Englewood Cliffs, NJ: PrenticeHall. 1981. 768 p.
3. Приходько С. Б., Приходько А. С., Шутко І. С. Нелінійні регресійні моделі для оцінювання кількості строк коду web-застосунків, що створюються за допомогою PHP-фреймворків. *Інформаційні технології: моделі, алгоритми, системи* : зб. тез II всеукраїнської науково-практичної інтернет конференції, 26-28 жовтня. 2021 р. С 11–12.
4. Ворона М. В., Приходько А. С., Шутко І. С. Трьохфакторні нелінійні регресійні моделі для оцінювання розміру web-застосунків, що створюються з використанням php фреймворків. *Інформатика, інформаційні системи та технології* : зб. тез доповідей вісімнадцятої всеукраїнської конференції студентів і молодих науковців. Одеса, 23 квітня 2021 р. Одеса, 2021. С. 111–113.
5. Приходько С. Б., Приходько Н. В., Ворона М. В., Бєловол І. О. Нелінійна регресійна модель для оцінювання розміру Web-застосунків, що створюються з використанням фреймворку Laravel. *Інформаційні технології та комп'ютерна інженерія.* 2021. Т. 50, № 1. С. 115–121.
6. Приходько С. Б., Приходько Н. В., Фаріонова Т. А., Ворона М. В. Трьохфакторна нелінійна регресійна модель для оцінювання розміру Php застосунків з відкритим кодом. *Вчені записки Таврійського національного університету імені В. І. Вернадського. Серія: Технічні науки*. 2020. № 4 (56). С. 124– 131.
7. Vorona M. V. Comparison o f the evaluation results of the size of open source php-based apps by nonlinear regression models. *German International Journal of Modern Science*. 2021. №6. P 43–47.
8. Приходько С. Б., Грицань В.Ю. НЕЛІНІЙНА РЕГРЕСІЙНА МОДЕЛЬ ДЛЯ ОЦІНЮВАННЯ РОЗМІРУ ВЕБ-ЗАСТОСУНКІВ, ЩО СТВОРЮЮТЬСЯ З

ВИКОРИСТАННЯМ ФРЕЙМВОРКУ NEXTJS. *Сучасні інформаційні технології в освіті і науці» («УМАНСЬКИЙ ДЕРЖАВНИЙ ПЕДАГОГІЧНИЙ УНІВЕРСИТЕТ ІМЕНІ ПАВЛА ТИЧИНИ», м. Умань, 14–15 листопада 2024 р.* [Електронний ресурс] //Режим доступу: URL: https://drive.google.com/file/d/1vECN_F1U2nTKJRHmpbGLQj0-cvU8Dlfy/view?usp=sharing

9. Next.js Documentation [Електронний ресурс]. //Режим доступу : URL: https://nextjs.org/docs
10. Kaczmarek J., Kucharski M. Size and effort estimation for applications written in Java. Information and SoftwareTechnology. 2004. Vol. 46, Issue 9. P. 589–601.
11. Montgomery D.C., Peck E. A., Vining G. G. Introduction to Linear RegressionAnalysis : Wiley, 2021. 702 p.
12. Prykhodko S. B., Prykhodko N. V. Constructing the non-linear regression models on the basis of multivariate normalizing transformations. *Electronic modeling*. 2018. T.40. № 6. C. 99–108.
13. 15 компаній, які успішно використовують Node.js у 2021 році [Електронний ресурс] // Режим доступу : URL: https://www.trio.dev/blog/companies- use-node-js
14. Показники використання фреймворків та зацікавленості серед зацікавлених осіб [Електронний ресурс] // Режим доступу : URL: https://stateofjs.com/en-US.
15. The Most Popular JavaScript Frameworks – 2011/2021 [Електронний ресурс]. // Режим доступу : URL: https://statisticsanddata.org/data/the-most-popular-javascript-frameworks-2011-2021/.
16. Github [Електронний ресурс]. //Режим доступу : URL: https://github.com/
17. SonarQube [Електронний ресурс]. //Режим доступу : URL: https://www.sonarqube.org/
18. Frost J. Regression Analysis: An intuitive guide for using and interpreting linear models.by Example. Statistics by Jim Publishing, 2020. 355 p.

19. Prykhodko S. B., Prykhodko N. V., Makarova L. M., Pugachenko K. M. Detecting Outliers in Multivariate Non-Gaussian Data on the basis of Normalizing Transformations. *Proceedings of the 2017 IEEE First Ukraine Conference on Electrical and Computer Engineering (UKRCON)* : «Celebrating 25 Years of IEEE Ukraine Section». Kyiv, Ukraine, May 29-June 2. Kyiv, 2017. P. 846–849.

20. Prykhodko S. B., Prykhodko N. V., Makarova L. M. Pukhalevych A. V. Application of the Squared Mahalanobis Distance for Detecting Outliers in Multivariate Non-Gaussian Data. *Proceedings of 14th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (TCSET)* : Lviv- Slavske, Ukraine, February 20–24. Lviv-Slavske, 2018. P. 962–965.

21. Arkes J. Regression Analysis: A practical introduction. Taylor & Francis Group. 2019. 342 p.

22. Приходько С. Б., Макарова Л. М., Пугаченко К. С. Методичні вказівки та завдання до виконання лабораторних робіт з дисципліни "Обробка експериментальних даних на комп`ютері". Миколаїв : НУК, 2018. 76 с.

23. Prykhodko S. B., Prykhodko N. V. Mathematical Modeling of Non-Gaussian Dependent Random Variables by Nonlinear Regression Models Based on the Multivariate Normalizing Transformations. Mathematical Modeling and Simulation of Systems (MODS'2020). MODS 2020. Advances in Intelligent Systems and Computing. 2020. Vol. 1265. P. 166–174.

24. Foss T. A, Stensrud E., Kitchenham B., Myrtveit I. Simulation study of the model evaluation criterion MMRE. *IEEE Transactions on software engineering*. 2003. Vol. 11 (29). P. 985–995.

25. Rusek, Robert & Frigola, Joaquim & Colomer, Joan. (2021). Influence of User Behavior on Energy Consumption and Its Relation With Comfort. A Case Study Based on Sensor and Crowd-Sensed Data. 10.21203/rs.3.rs-163089/v1. [Електронний ресурс]
// Режим доступу : URL: https://www.researchgate.net/figure/Overview-of-the-72m-2-office-lab-where-the-experiment-took-place_fig1_349129596

ДОДАТОК А – ТЕХНІЧНЕ ЗАВДАННЯ

Вступ

Найменування програми: «NextJS KLOC estimation».

1 Підстави для розробки

Підставою для розробки є наказ на затвердження тем кваліфікаційних робіт магістрів.

2 Призначення розробки

2.1 Функціональне призначення

Функціональним призначенням програми є надання користувачу можливості оцінити розміру (в тисячах рядків коду) веб-застосунків, що створюються з використанням фреймворку NextJS.

2.2 Експлуатаційне призначення

Програма призначена для експлуатації на персональних комп’ютерах користувачів для оцінювання розміру (в тисячах рядків коду) веб-застосунків, що створюються з використанням фреймворку NextJS.

3 Вимоги до програми або програмного продукту

3.1 Вимоги для функціональних характеристик

3.1.1 Вимоги до переліку виконуваних функцій

Розроблена програма має надавати можливість виконувати наступні функції:

– Оцінювання розміру (в тисячах рядків коду) веб-застосунків, що створюються з використанням фреймворку NextJS, у тисячах строк коду.

3.1.2 Вимоги для організації вхідних та вихідних даних

Введення вхідних даних відбувається за рахунок заповнення полів графічного інтерфейсу користувача.

Результати виводяться у відповідні поля графічного інтерфейсу користувача.
Вхідними даними для програми є наступні: кількість компонентів, кількість функцій для проведення оцінювання
Кількість компонентів та кількість функцій веб-застосунку – це натуральні числа.
Вихідними даними є отримана оцінка розміру (в тисячах рядків коду) веб-застосунку у тисячах рядків коду.

3.2 Вимоги до надійності

Надійне (стійке) функціонування програмного забезпечення повинно бути забезпечено виконанням сукупності організаційно-технічних заходів, виконанням валідації вхідних даних згідно вимог та відображення повідомлень про виявлення некоректних даних з закликом ввести коректні.

3.3 Умови експлуатації

Кліматичні умови експлуатації, при яких повинні забезпечуватися задані характеристики, повинні задовольняти вимогам, що пред'являються до технічних засобів в частині умов їх експлуатації.

3.4 Вимоги до технічної та програмної сумісності

Програма має експлуатуватись на обладнанні з параметрами не нижчими за наступні: 64-бітний процесор на архітектурі x86 або ARM, оперативна пам’ять 2Гб (1 Гб мінімум), простір на жорсткому диску 100 Мб мінімум.
Для запуску та подальшої роботи програмного забезпечення необхідна операційна система Windows 7/8/10/11, Ubuntu 17/18/19/20.

3.5 Вимоги до захисту інформації та програм

Вимоги до захисту інформації та програм не висуваються.

3.6 Вимоги до маркування та упаковки

Вимоги до маркування та упаковки не висуваються.

3.7 Вимоги до транспортування та зберігання

Вимоги до транспортування та зберігання не висуваються.

4 Вимоги до програмної документації

Склад програмної документації має включати в себе:

1) Технічне завдання;
2) Програму та методику випробувань;
3) Інструкцію користувача;
4) Опис програмного забезпечення;
5) Текст програми.

5 Техніко-економічні показники

Економічна ефективність від впровадження програмного забезпечення була розрахована у розділі 4. Виходячи з одержаних результатів, впровадження даного програмного забезпечення є доцільним, приблизний термін окупності – 5 місяці.

6 Стадії та етапи розробки програмного забезпечення

Стадії та етапи виконання розробки програмного забезпечення можна представити наступним чином:

Таблиця А.1 – Стадії та етапи розробки

Стадії розробки	Назва етапу	Термін виконання
Технічне завдання	Розробка технічного завдання	30.09.2024
	Затвердження технічного завдання	10.10.2024
Ескізний проект	Розробка ескізного проекту	25.10.2024
	Затвердження ескізного проекту	01.11.2024
Технічний проект	Розробка технічного проекту	15.11.2024
	Затвердження технічного проекту	30.11.2024
Робочий проект	Розробка програми	01.12.2024
	Розробка програмної документації	14.12.2024
	Випробування програми	20.12.2024

7 Порядок контролю та прийомки

Приймально-здавальні роботи мають проводитись під наглядом Замовникаабо уповноваженої ним особи.

Приймально-здавальні випробування повинні проводитись згідно документу «Програма та методика випробувань», який узгоджується між стороною розробника та стороною замовника.

Хід проведення приймально-здавальних робіт Замовник та Виконавець документують в «Протокол проведення випробувань».

На основі Протоколу проведення випробувань Виконавець та Замовник підписують «Акт прийому-здачі програми у експлуатацію».

ДОДАТОК Б – ПРОГРАМА ТА МЕТОДИКА ВИПРОБУВАНЬ

1 Об’єкт випробувань

Об'єктом дослідження треба вважати програмне забезпечення для оцінювання розміру (в тисячах рядків коду) веб-застосунків, що створюються з використанням фреймворку NextJS.

2 Мета випробувань

Мета випробувань – треба перевірити працездатність програмного застосунку, перевірити відповідність фактичних характеристик програмного застосунку тим вимогам, які було викладено у технічному завданні.

3 Вимоги до застосунку

Під час виконання випробувань треба перевірити функціональні характеристики (вимоги), що були викладені в технічному завданні. Розроблену програму треба перевірити на можливість виконання наступних функцій:

– Оцінювання розміру (в тисячах рядків коду) веб-застосунків, що створюються з використанням фреймворку NextJS у тисячах строк коду.

4 Вимоги до програмної документації

Програмна документація включає в себе наступні документи:

1) технічне завдання;
2) програму та методику випробувань;
3) інструкцію користувача;
4) опис програмного забезпечення.
5) текст програми

5 Засоби і порядок випробувань

5.1 Технічні засоби, що використовуються під час випробувань Склад технічних засобів, що використовувались під час випробувань:

– AMD Ryzen 4 3100 (2.1 - 4.0 ГГц).

– Оперативна пам'ять: 16 Гбайт.

– Накопичувач даних: SSD 512 Гбайт.

5.2 Програмні засоби, що використовувались під час випробувань Операційна система Windows 11 Pro

5.3 Порядок проведення випробувань

Порядок проведення випробувань – треба перевірити відповідність функціональних характеристик програмного застосунку, перевірити вимоги до програмної документації.

6 Методи випробувань

6.1 Методика проведення перевірки вимог до програмної документації

За відсутності конкретних вимог до оформлення та подання програмної документації, розроблені програмні документи перевіряються представником замовника візуально. Уповноважена особа перевіряє відповідність розроблених програмних документів з тими, що були вказані у пункті «Склад програмної документації, що пропонується на випробування».

6.2 Методика проведення перевірки вимог до функціональних характеристик програмного продукту

Перевірка працездатності програми виконується згідно з пунктом «Вимоги до функціональних характеристик» технічного завдання.

Перевірку можна вважати успішно завершеною у випадку відповідності виконуваних функції технічному завданню.

Порядок випробувань:

– Перевірка роботи застосунку на моніторах з різною розподільною здатністю.
– Перевірка відображення шрифтів.
– Перевірка кожної форми графічного інтерфейсу на можливість введення даних.
– Перевірка кожної форми графічного інтерфейсу на введення невалідних даних.
– Перевірка можливості виконання оцінювання розміру (в тисячах рядків коду) для веб-застосунків, що створюються з використанням фреймворку NextJS.

7 Результати випробувань

За результатами випробувань треба підтвердити відповідність функціональних можливостей висунутим вимогам до програмного забезпечення.

8 Відомості про відмови, збої та аварійні ситуації

Під час проведення випробувань треба зафіксувати будь-які відмови, збої та аварійні ситуації, якщо вони виникнуть.

9 Відомості про коригування параметрів об’єкта випробувань та технічної документації

Коригування параметрів об’єкта випробувань та технічної документації в процесі виконання випробувань не проводилось.

ДОДАТОК В – ІНСТРУКЦІЯ КОРИСТУВАЧА

Вступ

Інструкція користувача призначена для ознайомлення користувача з правилами та порядком виконання операцій, які забезпечуються роботою програмного забезпечення.

Ця програма призначена для оцінювання розміру у тисячах строк коду веб-застосунків, що створюються з використанням фреймворку NextJS. Програма є вільним програмним забезпеченням з графічним інтерфейсом користувача з можливістю вводити дані та переглядати результати.

Загальні відомості про програму

Найменування програми: «NextJS LOC estimation».

Використані мови програмування

Програма написана на мові Python, для розробки графічного інтерфейсу користувача було використано сумісну з Python бібліотеку Qt.

Призначення програми

Програма призначена для оцінювання розміру (в тисячах рядків коду) веб-застосунків, що створюються з використанням фреймворку NextJS на основі таких метрик як кількість компонентів та кількість функцій.

Функціональні можливості програми

– Оцінювання розміру (в тисячах рядків коду) веб-застосунків, що створюються з використанням фреймворку NextJS, у тисячах строк коду.

Обмеження області застосування програми

Область застосування програми обмежена галуззю розробки програмного забезпечення.

Умови, необхідні для використання програми

Спеціальні умови для використання програми відсутні

Вимоги до технічної сумісності та програмного середовища

Програма має експлуатуватись на обладнанні з параметрами не нижчими за наступні: 64-бітній процесор на архітектурі x86 або ARM, оперативна пам’ять 2Гб (1 Гб мінімум), простір на жорсткому диску 100 Мб мінімум.

Для запуску та подальшої роботи програмного забезпечення необхідна операційна система Windows 7/8/10/11, Ubuntu 17/18/19/20.

Загальний вигляд стартового вікна програми

Загальний вигляд стартового вікна програми «NextJS LOC estimation» наведено на рисунку В.1.

Стартове вікно містить наступний перелік полів для введення даних:

– Поле «Count of components» дозволяє вказати кількість компонентів оцінюваного застосунку. В якості кількості компонентів можна ввести натуральне число.

– Поле «Count of functions» дозволяє вказати кількість методів в оцінюваному застосунку. В кількості функцій можна ввести натуральне число.

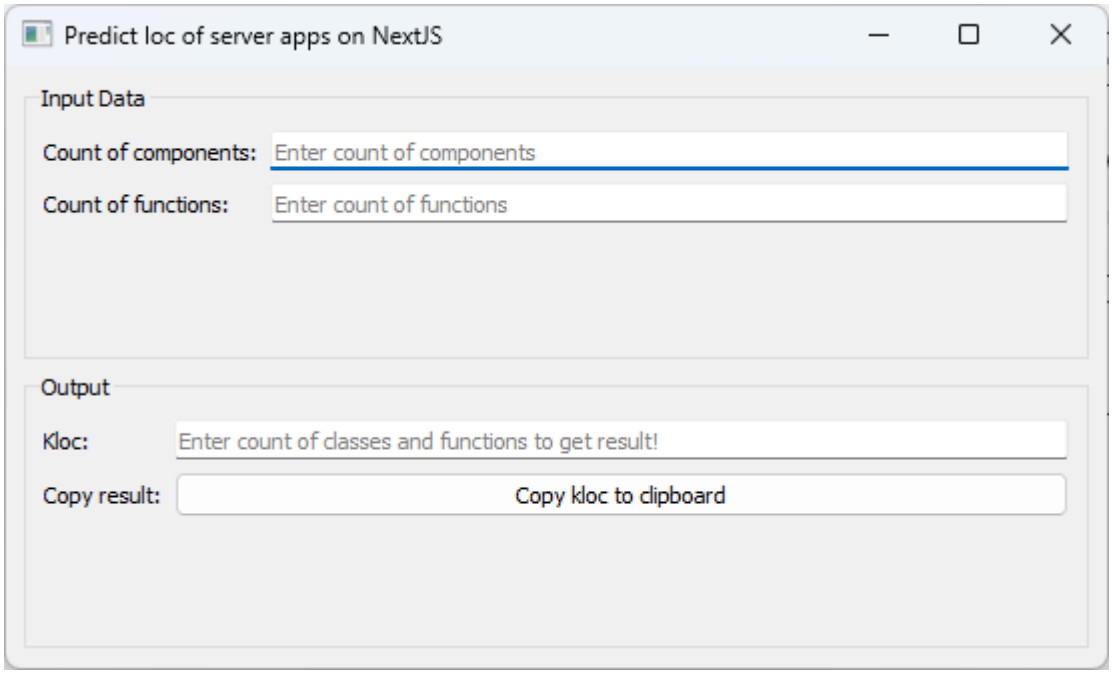

Рисунок В.1 – Вигляд вікна програми до виконання оцінювання

Результати оцінювання розміру веб-застосунків, що створюються з використанням фреймворку NextJS

Після введення вхідних даних програмне забезпечення автоматично виконає розрахунок відповідних оцінок. Результати оцінювання розміру застосунку (його розмір у тисячах строк коду), будуть показані в полях в нижній частині екрану. Результат виконання оцінювання можна побачити на рисунку В.2.

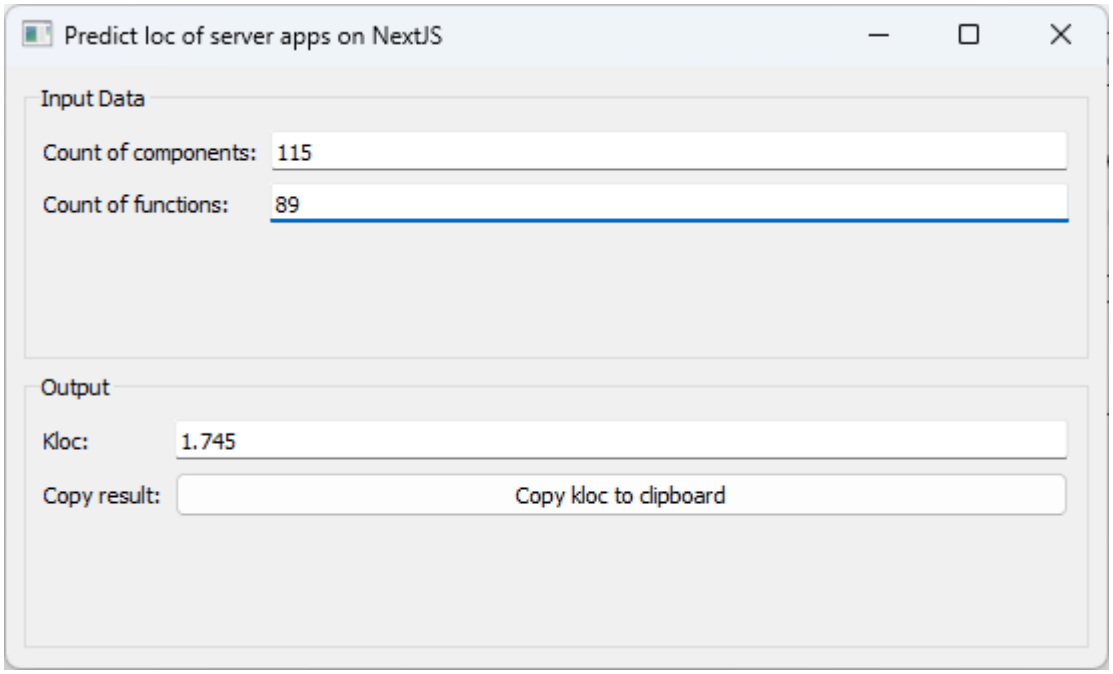

Рисунок В.2 – Результат оцінювання застосунку, що створюється з використанням фреймворку NextJS

ДОДАТОК Г – ТЕКСТ ПРОГРАМИ

```
import sys
from PySide2 import QtWidgets, QtGui, QtCore

class locPredictor:
    B = [
        -1.96457,
        0.15446,
        0.96852,
    ]

    def KLOCPredict(self, X):
        return (X[0] ** self.B[1]) * (X[1] **
self.B[2]) * (10 ** self.B[0])

class AppMainWindow(QtWidgets.QMainWindow):
    def __init__(self):
        super().__init__()
        self.locPredictor = locPredictor()

        UIMainWidget = QtWidgets.QWidget()
        self.setCentralWidget(UIMainWidget)
        UIMainLayout =
QtWidgets.QVBoxLayout(UIMainWidget)

        UIDataInputGroup = QtWidgets.QGroupBox("Input
Data")
        UIMainLayout.addWidget(UIDataInputGroup)
        UIDataInputGroupLayout =
QtWidgets.QFormLayout(UIDataInputGroup)

        self.UIClassesCountInput =
QtWidgets.QLineEdit()

self.UIClassesCountInput.setValidator(QtGui.QIntValida
tor(0, 10000, self))  # Added min value of 0

self.UIClassesCountInput.setPlaceholderText("Enter
count of components")

self.UIClassesCountInput.textChanged.connect(self.exec
Calculations)
        UIDataInputGroupLayout.addRow("Count of
components:", self.UIClassesCountInput)
```

```
        self.UIFunctionsCountInput =
QtWidgets.QLineEdit()

self.UIFunctionsCountInput.setValidator(QtGui.QIntVali
dator(0, 10000, self))  # Added min value of 0

self.UIFunctionsCountInput.setPlaceholderText("Enter
count of functions")

self.UIFunctionsCountInput.textChanged.connect(self.ex
ecCalculations)
        UIDataInputGroupLayout.addRow("Count of
functions:", self.UIFunctionsCountInput)

        UIDataOutputGroup =
QtWidgets.QGroupBox("Output")
        UIMainLayout.addWidget(UIDataOutputGroup)
        UIDataOutputGroupLayout =
QtWidgets.QFormLayout(UIDataOutputGroup)

        self.UIlocInput = QtWidgets.QLineEdit()
        self.UIlocInput.setReadOnly(True)
        self.UIlocInput.setPlaceholderText("Enter
count of components and functions to get result!")
        UIDataOutputGroupLayout.addRow("Kloc:",
self.UIlocInput)

        self.UICopyToClipboardBtn =
QtWidgets.QPushButton("Copy kloc to clipboard")

self.UICopyToClipboardBtn.clicked.connect(self.copyToC
lipboard)
        UIDataOutputGroupLayout.addRow("Copy result:",
self.UICopyToClipboardBtn)

    def copyToClipboard(self):
        clipboard = QtWidgets.QApplication.clipboard()
        clipboard.setText(self.UIlocInput.text())

    def execCalculations(self):
        # Clear any previous state
        self.UIlocInput.clear()

        # Check if both inputs are filled
```

```
        if not (self.UIClassesCountInput.text() and
self.UIFunctionsCountInput.text()):
            return

        # Check for negative numbers
        if
(self.UIClassesCountInput.text().startswith('-') or

self.UIFunctionsCountInput.text().startswith('-')):
            # Show error message
            QtWidgets.QMessageBox.warning(
                self,
                "Input Error",
                "Negative numbers are not allowed.
Please enter positive numbers.",
                QtWidgets.QMessageBox.Ok
            )
            return

        try:
            X = [

float(self.UIClassesCountInput.text().replace(",",
"")),

float(self.UIFunctionsCountInput.text().replace(",",
""))
            ]

            # Validate non-negative numbers
            if X[0] < 0 or X[1] < 0:
                QtWidgets.QMessageBox.warning(
                    self,
                    "Input Error",
                    "Negative numbers are not allowed.
Please enter positive numbers.",
                    QtWidgets.QMessageBox.Ok
                )
                return

            value = self.locPredictor.KLOCPredict(X)
            self.UIlocInput.setText(f'{value:.3f}')
        except (ValueError, TypeError) as e:
            self.UIlocInput.clear()
```

```
def main():
    app = QtWidgets.QApplication.instance()
    if not app:
        app = QtWidgets.QApplication(sys.argv)

    widget = AppMainWindow()
    widget.setWindowTitle("Predict loc of server apps
on NextJS")
    widget.resize(550, 300)
    widget.show()

    sys.exit(app.exec_())

if __name__ == "__main__":
    main()
```

ДОДАТОК Д – ОПИС ПРОГРАМИ

1 Загальні відомості

Найменування: «NextJS LOC estimation».

1.2 Програмне забезпечення, необхідне для функціонування програми

Для запуску та подальшої роботи програмного забезпечення необхідна операційна система Windows 7/8/10/11, Ubuntu 17/18/19/20.

1.3 Мови програмування

Програмне забезпечення для оцінювання розміру (в тисячах рядків коду) веб-застосунків, що створюються з використанням фреймворку NextJS, розроблене з використанням мови програмування Python та використанням бібліотек NumPy та SciPy для виконання математичних обчислень. Для створення графічного інтерфейсу користувача було використано сумісну з Python бібліотеку Qt.

2 Функціональне призначення

Функціональним призначенням програми є надання користувачу можливості оцінити розміру (в тисячах рядків коду) веб-застосунків, що створюються з використанням фреймворку NextJS.

Функції програмного застосунку, які доступні користувачу:

– Оцінювання розміру (в тисячах рядків коду) веб-застосунків, що створюються з використанням фреймворку NextJS у тисячах строк коду.

2.1 Функціональні обмеження

Програмне забезпечення не вимагає використання зовнішніх ресурсів, тому функціональні обмеження відсутні.

3 Технічні засоби, що використовуються

Програма має експлуатуватись на обладнанні з параметрами не нижчими за наступні: 64-бітній процесор на архітектурі x86 або ARM, оперативна пам’ять 2Гб (1 Гб мінімум), простір на жорсткому диску 100 Мб мінімум.

4 Виклик та завантаження

Для запуску та подальшої роботи програмного забезпечення необхідна операційна система Windows 7/8/10/11, Ubuntu 17/18/19/20.
Запуск програмного застосунку відбувається за рахунок натискання на ярлик програмного застосунку.

5 Вхідні дані

Вхідними даними для програми є кількість компонентів, кількість функцій. Кількість компонентів та кількість функцій веб-застосунку – це натуральні числа.

6 Вихідні дані

Вихідними даними є отримані оцінки розміру (в тисячах рядків коду) у тисячах рядків коду. Отримані значення буде відображено у відповідних полях графічного інтерфейсу програми.