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

_____
«___» _____ 2021 р.

КВАЛІФІКАЦІЙНА РОБОТА
на здобуття ступеня вищої освіти «магістр»

на тему: Математична модель для оцінювання розміру веб-застосунків, що створюються за допомогою платформи ASP.Net, та розробка програмного забезпечення для її реалізації

Виконав: студент 6151м групи
Устенко А.С.
(підпис, ПІБ)

Керівник роботи:
Доцент кафедри ПЗАС, к.ф.-м.н., доцент
(посада, науковий ступень вчене звання)
Латанська Л.О.
(підпис, ПІБ)

Рецензент:
Доцент кафедри ПЗАС, к.т.н., доцент
(посада, науковий ступень вчене звання)
Макарова Л.М.
(підпис, ПІБ)

Миколаїв – 2021 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)
«___» _____ 2021 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ
на здобуття ступеня вищої освіти «магістр»

Студенту Устенку Андрію Сергійовичу
(Прізвище, ім'я, по батькові)

1. Тема роботи: Математична модель для оцінювання розміру веб-застосунків, що створюються за допомогою платформи ASP.Net, та розробка програмного забезпечення для її реалізації

Керівник роботи Латанська Л.О. к.ф.-м.н., доцент

Затверджені наказом ректора № 1239уч від «13» _____10_____ 2021 р.

2. Термін подання роботи: 06.12.2021 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; Проект 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

6. Консультанти розділів роботи

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

7. Дата видачі завдання 13.10.2021 р.

К А Л Е Н Д А Р Н И Й П Л А Н

Номер	Назва етапів роботи	Терміни виконання	Примітка
1.	Підготовка розділу Вступ	18.10.2021	НС*
2.	Підготовка розділу з огляду літератури та обґрунтування необхідності проведення досліджень за обраною темою	19.10.2021	НС*
3.	Підготовка розділу (ів) з результатів власних досліджень	22.10.2021	НС*
4.	Підготовка розділу з проекту програмного забезпечення	16.11.2021	
5.	Підготовка організаційно-економічного розділу	19.11.2021	
6.	Підготовка розділу з охорони праці	22.11.2021	
7.	Підготовка розділу з охорони навколишнього середовища	24.11.2021	
8.	Підготовка розділу Висновки	25.11.2021	
9.	Оформлення списку використаних джерел та додатків	29.11.2021	
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.)	06.12.2021	не пізніше, ніж за 14 діб до захисту (згідно п.4.1 зазначеного Порядку
11.	Підготовка презентації та доповіді	09.12.2021	
12.	Попередній захист роботи на засіданні кафедри ПЗАС	10.12.2021	
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	20.12.2021	

* - за результатами наукового стажування (НС), яке було з 01.09.2021 до 10.10.2021 р.

Студент _____ Устенко А.С.
(підпис) (ПІБ)

Керівник роботи _____ Латанська Л.О.
(підпис) (ПІБ)

РЕФЕРАТ

Устенко Андрій Сергійович

Математична модель для оцінювання розміру веб-застосунків, що створюються за допомогою платформи ASP.Net, та розробка програмного забезпечення для її реалізації

Кваліфікаційна робота на здобуття освітнього рівня магістра зі спеціальності 121 – «Інженерія програмного забезпечення». Національний університет кораблебудування імені адмірала Макарова. Миколаїв, 2021 р.

Обсяг роботи: 113 сторінок друкованого тексту, містить 19 рисунків, 22 таблиці, 4 додатки та список використаних джерел з 25 найменувань.

Актуальність теми: На сьогоднішній день існує декілька методів та моделей оцінювання розміру програми, але вони не є універсальними та більшість з них не можуть бути використаними внаслідок труднощів, пов'язаних зі збором даних по проектам. Крім того, мова та технології, що використовуються для розробки програмних проектів, суттєво впливають на модель, яка створюється. Таким чином, актуальність теми полягає в отриманні ефективної математичної моделі для оцінювання розміру веб-застосунків, що розробляються мовою C# за допомогою платформи ASP.NET. Розв'язання цієї проблеми є важливим, оскільки, саме від цього залежить кількість витрачених ресурсів і коштів на розробку проекту.

Мета і завдання дослідження. Метою дослідження є підвищення достовірності оцінювання розміру веб-застосунків, які створюються мовою програмування C# із застосуванням платформи ASP.NET. Для вирішення поставленої мети необхідно вирішити наступні задачі:

– провести аналіз існуючих методів та моделей для оцінювання розміру веб-застосунків;

–обґрунтувати необхідність розробки математичної моделі для оцінювання розміру веб-застосунків, які створюються мовою програмування C# із застосуванням платформи ASP.NET;

–розробити математичну модель для оцінювання розміру веб-застосунків, які створюються мовою програмування C# із застосуванням платформи ASP.NET.

Об'єктом досліджень є процес оцінювання розміру програмного забезпечення, яке створюється мовою програмування C# за допомогою платформи ASP.NET.

Предметом дослідження є математична регресійна модель для оцінювання кількості рядків коду веб-застосунків, реалізованих мовою C# за допомогою платформи ASP.NET.

Наукова новизна полягає в удосконаленні регресійної моделі для оцінювання розміру веб-застосунків, реалізованих мовою програмування C# за допомогою платформи ASP.NET, за рахунок застосування нормалізуючого перетворення на основі натурального логарифму, що дозволило збільшити достовірність оцінювання розміру програмного забезпечення в порівнянні з лінійною моделлю.

Практична цінність полягає у розробці програми, яка реалізує створену математичну модель для оцінювання розміру веб-застосунків та може бути використана при проектування програмних проектів для економії ресурсів та часу.

Апробація результатів досліджень. Основні положення роботи були оприлюднені на Міжнародній науково-практичній конференції "Free and open source software" (FOSS-2021) (м. Харків, 16-18 листопада 2021). (FOSS - Free and Open Source Software (kn-it.info)).

Ключові слова: ОЦІНЮВАННЯ РОЗМІРУ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ, МОВА ПРОГРАМУВАННЯ C#, ПЛАТФОРМА ASP.NET, РЕГРЕСІЙНА МОДЕЛЬ.

ABSTRACT

Ustenko Andrii

A mathematical model for estimating the size of web applications created using the ASP.Net platform and developing software for its implementation

Qualification work for obtaining a master's degree in specialty 121 - "Software Engineering". Admiral Makarov National University of Shipbuilding. Mykolaiv, 2021.

Volume of work: 113 pages of printed text, contains 19 figures, 25 tables, 4 appendices and a list of used sources from 25 titles.

Relevance: There are currently several methods and models for estimating program size, but most of them cannot be used due to difficulties in collecting project data. In addition, the language and technology used to develop software projects significantly affect the model being created. The urgency of the problem is to obtain an effective mathematical model for estimating the size of web applications developed in C # using the ASP.NET platform. Solving this problem is important, because it depends on the amount of resources and funds spent on project development.

The purpose and objectives of the study. The aim of the study is to increase the reliability of estimating the size of web applications created in the C # programming language using the ASP.NET platform. To solve this goal it is necessary to solve the following tasks:

–analyze existing methods and models for estimating the size of web applications;

–justify the need to develop a mathematical model for estimating the size of web applications created in the C # programming language using the ASP.NET platform;

–develop a mathematical model for estimating the size of web applications created in the C # programming language using the ASP.NET platform.

The object of research is the process of estimating the size of web appliactions created in the C # programming language using the ASP.NET platform.

The subject of the study is a mathematical regression model for estimating the number of lines of code for web applications implemented in C # using the ASP.NET platform.

The scientific novelty is to improve the regression model for estimating the size of web applications implemented in the C # programming language using the ASP.NET platform, through the use of normalizing transformations based on natural logarithm, which increased the reliability of software size estimation.

The practical value lies in the development of a program that implements a mathematical model for estimating the size of web applications, which can be used in the design of software projects to save resources and time.

Approbation of research results. The main provisions of the work were announced at the International Scientific and Practical Conference "Free and open source software" (FOSS-2021) (Kharkov, November 16-18, 2021). (FOSS - Free and Open Source Software (kn-it.info)).

Keywords: ESTIMATING THE SIZE OF WEB-APPLICATIONS, PROGRAMMING LANGUAGE C#, ASP.NET PLATFORM, REGRESSION MODEL.

ЗМІСТ

ВСТУП

Актуальність теми. Створення веб-застосунків є одним з найбільш затребуваних напрямів сучасної IT-індустрії. C# є однією з популярних та таких, що стрімко розвивається, мов програмування для їх розробки. Поєднання мови програмування C# з сучасною платформою ASP.NET є найбільш доцільним, тому що надає розробнику багато переваг.

При розробці програмних проектів існує серйозна проблема, що пов’язана зі зривом термінів їх виконання, що в свою чергу, призводить до незадоволення замовників, а іноді взагалі до повного призупинення проектів. Тому при розробці програмних проектів вже на початкових етапах необхідно достовірно оцінити їх трудомісткість.

Розмір програмних продуктів, зокрема, веб-застосунків, що виражений кількістю рядків коду, істотно впливає на їх трудомісткість та тривалість виконання, тому що основна праця розробників вкладається в розробку текстів програм. Виходячи з цього, достовірне оцінювання розміру є одним із важливих критеріїв успіху розробки програмного проекту.

На сьогоднішній день існує декілька методів та моделей оцінювання розміру програми, але більшість з них не можуть бути використаними внаслідок труднощів, пов’язаних зі збором даних по проектам або відсутністю експертного досвіду. Крім того, мова та технології, що використовуються для розробки програмних проектів, суттєво впливають на модель, яка створюється.

У джерелах [1 – 5] були розглянуті питання побудови моделі для оцінювання розміру веб-застосунків, але вони розроблені для інших мов програмування.

Таким чином, актуальність теми полягає в отриманні ефективної математичної моделі для оцінювання розміру веб-застосунків, що розробляються мовою C# за допомогою платформи ASP.NET. Розв’язання цієї проблеми є важливим, оскільки, саме від цього залежить кількість витрачених ресурсів і коштів на розробку проекту.

Мета і завдання дослідження. Метою дослідження є підвищення достовірності оцінювання розміру веб-застосунків, які створюються мовою програмування C# із застосуванням платформи ASP.NET. Для вирішення поставленої мети необхідно вирішити наступні задачі:

– провести аналіз існуючих методів та моделей для оцінювання розміру веб-застосунків;

– обґрунтувати необхідність розробки математичної моделі для оцінювання розміру веб-застосунків, які створюються мовою програмування C# із застосуванням платформи ASP.NET;

– розробити математичну модель для оцінювання розміру веб-застосунків, які створюються мовою програмування C# із застосуванням платформи ASP.NET.

Об’єктом досліджень є процес оцінювання розміру програмного забезпечення, яке створюється мовою програмування C# за допомогою платформи ASP.NET.

Предметом дослідження є математична регресійна модель для оцінювання кількості рядків коду веб-застосунків, реалізованих мовою C# за допомогою платформи ASP.NET.

Наукова новизна полягає в удосконаленні регресійної моделі для оцінювання розміру веб-застосунків, реалізованих мовою програмування C# за допомогою платформи ASP.NET, за рахунок застосування нормалізуючого перетворення на основі натурального логарифму, що дозволило збільшити достовірність оцінювання розміру програмного забезпечення в порівнянні з лінійною моделлю.

Практична цінність полягає у розробці програми, що реалізує удосканалену математичну модель для оцінювання розміру веб-застосунків, яка може бути використана при проектуванні програмних проектів для економії ресурсів та часу.

Особистий внесок здобувача. Кваліфікаційна (магістерська) робота є самостійно виконаною працею. Усі результати, викладені у роботі, отримані автором особисто. У тезах доповіді [6], написаних у співавторстві, здобувачу

належить побудова нелінійної регресійної моделі для оцінювання розміру програмного забезпечення на C# за допомогою платформи ASP.NET.

Апробація результатів досліджень. Основні положення роботи були оприлюднені на XIII-ій Міжнародній науково-практичній конференції “Free and open source software” (FOSS-2021) (м. Харків, 16-18 листопада 2021). (FOSS - Free and Open Source Software (kn-it.info)).

Публікації. Основні результати магістерської роботи викладено у 1 науковій праці – тезах конференції «Free and open source software».

1 АНАЛІЗ МЕТОДІВ І МОДЕЛЕЙ ДЛЯ ОЦІНЮВАННЯ РОЗМІРУ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

1.1 Застосування платформи ASP.NET у програмних проектах на C#

У якості мови програмування для проектів, які досліджуються та розмір яких оцінюється математичною моделлю, було обрано C# з використанням ASP.NET.

C# – сучасна об’єктно-орієнтована мова, що містить мовні конструкції, які дозволяють розробникам програмного забезпечення створювати надійні, ефективні, функціональні та безпечні веб-застосунки.

Використання C# дозволяє створювати найрізноманітніші типи застосунків, як стосовно розміру, так і функціонального навантаження.

Найбільш доцільним є застосування мови C# у поєднанні з платформою ASP.NET – однієї з популярних сучасних платформ для розробки програмного забезпечення у вигляді веб-застосунків.

Застосування C# у поєднанні з платформою ASP.NET надає розробникам програмного забезпечення багато переваг, основні з яких наведено нижче:

1. Завдяки компіляції замість інтерпретації веб-застосунки виконуються набагато швидше.

2. При потребі оновлення даних, що використовуються застосунком, перезапуск серверу не є необхідним.

3. Для поширення можливостей API Windows надається доступ до програмної платформи .NET Framework.

4. Застосування ASP.NET полегшує роботу при розробці програмного забезпечення завдяки тому, що повертає готовий код об’єктів. Тобто, аналогічно з Windows Forms, достатньо вказати властивості об’єкта, а HTML код генерується платформою автоматично.

1.2 Вплив розміру програмних проектів на розрахунок їх трудомісткості

Достовірне оцінювання трудомісткості програмних продуктів – один з найважливіших критеріїв успіху. Відомо, що майже всі програмні продукти створюються в умовах обмежених ресурсів, як фінансових, так і часових.

У такій ситуації дуже важливим моментом є достовірне оцінювання трудомісткості проекту вже на початкових етапах його розробки. Це дозволяє спрогнозувати складність та розмір продукту, економічні витрати. Завищені або нереалістичні очікування учасників часто призводять до провалу проекту, крім того, досить складною задачею є створення реального плану-графіку за відсутністю оцінки трудомісткості. Це зумовлено неправильним прогнозуванням, а не відсутністю професійного досвіду розробників.

Великий вплив на трудомісткість проекту має розмір текстів програм, що виражений у кількості рядків коду. Дуже складно, майже неможливо, оцінити трудомісткість за відсутністю оцінки розміру програмного забезпечення.

Доведено, що розмір програмних продуктів, зокрема веб-застосунків, безпосередньо впливає на їх трудомісткість, тому оцінка розміру є не менш важливим завданням.

1.3 Основні метрики для оцінювання розміру програмного забезпечення

На сьогоднішній день існують різні метрики, для оцінювання розміру програмного забезпечення. Серед них виділяють наступні:

1. Кількість рядків коду – це розміро-орієнтована метрика, що розраховується як кількість рядків коду вихідного тексту [7].
2. Метрики Холстеда – основані на аналізі кількості рядків та синтаксичних елементів коду, що дозволяє врахувати можливість реалізації однієї функціональності різною кількість рядків вихідного коду та операторів.
3. Функціональні точки – даний метод дозволяє оцінити об’єм програмного продукту за його функціональною моделлю, тобто оцінюється об’єм функцій системи, що розроблюється [7].

4. Об’єктні точки – для проектів з об’єктно-орієнтованим підходом відбувається підрахунок об’єктних точок. Тобто, кожному унікальному об’єкту чи класу проекту відповідає одна така точка.

5. Метод UCP (Use case points) – представляє собою оцінювання розміру програмних проектів на основі діаграм UML та методології RUP [7].

Серед усіх факторів, що впливають на оцінку трудомісткості, розмір програмного продукту, який виражений у кількості рядків вихідного коду, є найбільш важливим показником. Часто розрахунки функціональності та її застосування не мають сенсу, оскільки не враховуються багато факторів. У таких проектах для оцінювання розміру програмного проекту доцільніше використовувати кількість рядків вихідного коду (LOC) [8]. Переваги даної метрики полягають у наступному:

– зв’язок з розміром кінцевого проекту;
– оцінювання відбувається за фізичним розміром програмного продукту;
– вдосконалення проекту співвідноситься з реальним розміром.

1.4 Основні методи та моделі оцінювання розміру програмного забезпечення

На сьогоднішній день існує декілька методів та моделей для оцінювання розміру веб-застосунків. Серед них можна виділити наступні:

1. Метод експертних оцінок – це метод прогнозування, що ґрунтується на основі припущень та думок експертів, за допомогою яких можна побудувати достовірну модель майбутнього об’єкта.

2. Регресійні моделі використовуються для аналізу залежностей регресору і факторів та їх впливу на очікуваний результат. Регресійні моделі застосовуються для отримання математичних моделей для оцінювання розміру програмного забезпечення, зокрема, веб-застосунків. Серед регресійних моделей виділяють наступні:

а) за моделями, які використовуються в основі регресії:

– лінійні регресії – дозволяють встановити взаємозв’язок між регресом та факторами за допомогою лінійних моделей. Вони легко

моделюються та є інтуїтивно зрозумілими, але вони є дуже чутливими до викидів. Тому, головне завдання при розробці лінійної регресійної моделі є ретельна перевірка даних на наявність викидів;

– нелінійні регресії – в даних методах для встановлення взаємозв'язку застосовуються нелінійні моделі. Вони є гнучкими та дозволяють знаходити більш складні взаємозв'язки;

б) за кількістю факторів що впливають на регресор:

– однофакторні – містять в собі один фактор, що впливає на значення регресора;

– багатофакторні – містять в собі два або більше факторів, які впливають на значення регресора. Лінійність або нелінійність регресії досягається завдяки використанню лінійної або нелінійної моделі [9].

В знайдених джерелах були розглянуті питання створення нелінійних регресійних моделей з використанням нормалізуючих перетворень [10 – 12]. Також було досліджено моделі із застосуванням нормалізуючого перетворення Джонсона для оцінювання розміру програмних проектів, розроблених на мовах програмування Java [13 – 15], PHP [1 – 4], PHP з використанням фреймворку Larawell [5] та для мобільних застосунків [16].

Оскільки підвищення достовірності оцінювання розміру веб-застосунків є основною метою даної роботи, тому був проведений ретельний аналіз джерел, що стосуються їх розробки :

– у джерелі [1] було розглянуто питання побудови нелінійної регресійної моделі для оцінювання розміру веб-застосунків, розроблених на мові програмування PHP, на основі багатовимірного нормалізуючого перетворення;

– джерела [2 – 4] присвячені побудові трьохфакторних нелінійних регресійних моделей, розроблених на мові програмування PHP, з використанням нормалізуючого перетворення Джонсона;

– у джерелі [5] були розглянуті питання побудови нелінійних регресійних моделей, розроблених на мові програмування PHP з використанням фреймворку

Larawell, на основі нормалізуючих перетворень, таких як: десятковий логарифм, Box-Cox та перетворення Джонсона.

Аналіз даних джерел показав, що побудова моделі пов’язана з виконанням значної кількості доволі складних розрахунків. Крім того, було виявлено, що модель для оцінювання розміру програмних застосунків є досить чутливою до мови, якою створюються веб-застосунки.

В результаті аналізу дійшли висновку, що є необхідність в удосконаленні математичної моделі для оцінювання розміру веб-застосунків, розроблених мовою C# з використанням платформи ASP.NET у зв’язку з відсутністю моделей, що дозволяють вирішити поставлену мету.

3. Нейронні мережі – які в своїй основі мають штучний інтелект, що навчається за аналогічними дослідженнями. Нейронна мережа – це метод прогнозування. Вона будує розв’язок ймовірних значень, які може прийняти послідовність в даний момент, на основі значень, які передують розрахункам [17]. Нейронні мережі можуть бути ефективно застосовані для оцінювання зовнішніх та внутрішніх ресурсів програмних проектів, зокрема, їх розміру, кількості дефектів, часу випуску та інших. Основними перевагами застосування нейронних мереж до оцінювання є наступні:

– дослідження отриманих відношень безпосередньо з розрахованих значень;
– вони є не параметричними та адаптованими [18].

При оцінюванні розміру програмних проектів нейронні мережі демонструють гарні показники при роботі з даними. В деяких випадках дають можливість виявити нелінійні відношення, які не були знайдені за допомогою регресії.

4. Композитні методи – методи, в яких використовуються два або більше методів та моделей для оцінювання.

1.5 Деякі підходи до вилучення викидів із статистичних даних

Першим етапом необхідно виконати передобробку статистичних даних, а саме позбавлення їх від викидів.

Розглянемо деякі підходи до вилучення викидів із статистичних даних.

Першим розглянемо метод квадратів відстані Махаланобіса. Відстань Махалонобіса – це міра відстані від багатовимірного спостереження до вибіркового середнього набору даних, що нормована за допомогою коваріційної матриці. Відстань Махаланобіса розраховується за формулою:

$$DM(Z) = \sqrt{(Z - \overline{Z})TS - 1(Z - \overline{Z})}.$$

Для кожної ітерації доцільно використовувати наступну формулу

$$d_i^2 = (Z_i - \overline{Z})^T S_N^{-1} (Z_i - \overline{Z}), \quad (1.1)$$

де d_i^2 – i -ий елемент D^2

Z_i – i - ий елемент (має координати Z_{x_i}, Z_{y_i}),

N – кількість елементів у виборці,

$\overline{Z}$ – середній вектор вибірки (середні по нормалізованому x и по нормалізованому y),

S_N – коваріційна матриця:

$$S_N = \frac{1}{N} \sum_{i=1}^{N} (Z - \overline{Z}) (Z - \overline{Z})^T. \quad (1.2)$$

Після обчислення d_i^2 можна застосовувати або критерій Пірсона χ^2 або критерій Фішера F. Для використання критерію Фішера потрібно додатково розрахувати статистику для кожного d_i^2

$$T_{S_i} = (N - k) N d_i^2 / ((N^2 - 1)k), \quad (1.3)$$

де N – всього елементів у вибірці,

k – кількість параметрів.

Якщо отримане значення відстані Махаланобіса більше критичного значення Фішера $F_{k,\,N-k,\alpha}$, що відповідає обраному рівню значимості α, то такі точки є викидами. Їх потрібно видалити з вибірки та заново повторити розрахунок.

Ще одним методом є відстань Кука. Відстань Кука – це метод, який дуже часто використовується для того, щоб оцінити вплив змінної на модель. Вона визначається як різниця між значеннями розрахованої регресії та значеннями регресії, яка була побудована без спостережуваного видаленого і-го значення. В адекватній моделі ці значення мають бути однаковими, але якщо спостерігається відмінність у даних, то вони є викидами [19].

Повна формула визначається як:

$$CD_i = \frac{\sum_{i=1}^{n}(y-y_i)^2}{kS_e^2}.$$

Наступним критерієм для знаходження викидів є критерій Фішера. Критерій Фішера – це критерій, який зводиться до знаходження відношень дисперсій вибірок, якщо отримане значення виконує нульову гіпотезу, то кажуть, що дані мають розподіл Фішера [20].

Розраховане значення називається F-статистика. Воно дорівнює:

$$F = \frac{S_x^2}{S_y^2}.$$

1.6 Оцінки якості регресійних моделей

Оцінювання отриманих математичних моделей є одним з найважливіших завдань, оскільки від отриманих оцінок залежить, можна використовувати дану модель для подальших проектів чи ні. До характеристик якості математичної моделі відносять:

– *коефіцієнт детермінації*

Згідно з джерелом [21] коефіцієнт показує, який відсоток дисперсії можна пояснити впливом величини X. Даний коефіцієнт змінюється у діапазоні від 0 до 1. У випадку, коли коефіцієнт дорівнює 0, це означає, що зв'язок між регросором та факторами регресійної моделі відсутній.

Еквівалентність зі значенням 1 визначає ідеальну модель, тобто всі точки лежать точно на лінії регресії, оскільки сума квадратів їх відхилень дорівнює 0.

Згідно з джерелом [21] коефіцієнт детермінації розраховується за формулою

$$R^2 = 1 - (\sum_{i=1}^{n} (y_i - \hat{y}_i)^2 / \sum_{i=1}^{n} (y_i - \bar{y})^2), \quad (1.3)$$

де $\hat{y}$ – передбачуване значення змінної *y.*

– *величину відносної похибки MRE (Magnitude of Relative Errors)*

Для лінійного та нелінійного рівнянь регресії знайдемо наступним чином:

$$MRE_i = \left| \frac{y_i - \hat{y}_i}{y_i} \right|,$$

де $\hat{y}_i$ – значення y, підраховане за моделлю регресії;

y_i – фактичне значення випадкової величини y.

– *середню величину відносної похибки MMRE (Mean of MRE)*

можна розрахувати за формулою [22]:

$$MMRE = \frac{1}{N} \sum_{i=1}^{N} MRE_i . \quad (1.4)$$

– *рівень прогнозування (PRED(l))*

можна розрахувати за формулою:

$$PRED(l) = \frac{k}{N}, \quad (1.5)$$

де k – кількість значень з MRE ≤ l;

N – кількість проектів.

1.7 Обґрунтування необхідності проведення досліджень

У першому розділі було проведено аналіз методів та моделей для оцінювання розміру програмних проектів. Відомо, що розмір програмного забезпечення, який вимірюється в рядках коду, суттєво впливає на оцінювання трудомісткості розробки програмного забезпечення. Саме тому створення моделі для достовірного оцінювання кількості рядків коду, як важливої складової визначення трудомісткості, є актуальним завданням. Крім того, аналіз показав, що мова, на якій створюються застосунки впливає на модель, що розроблюється. Тому доцільно створити регресійну модель для оцінювання розміру веб-застосунків на мові C# з використанням технології ASP.NET.

2 УДОСКОНАЛЕННЯ МАТЕМАТИЧНОЇ МОДЕЛІ ДЛЯ ОЦІНЮВАННЯ РОЗМІРУ ВЕБ-ЗАСТОСУНКІВ, СТВОРЕНИХ МОВОЮ C# ЗА ДОПОМОГОЮ ПЛАТФОРМИ ASP.NET

2.1 Аналіз даних для побудови математичної моделі для оцінювання розміру веб-застосунків

Для побудови регресійної моделі необхідно визначити залежну та незалежну змінні. Для програмних проектів, що досліджуються та написані мовою C# в якості незалежної змінної, було обрано «Кількість класів» в якості залежної – «Розмір програми».

Після обрання критеріїв був відібраний список реалізованих проектів на електронному ресурсі Github. До вхідних даних було обрано 60 проектів. Шляхом їх аналізу, з них було зібрано інформацію яка містить у собі кількість класів та кількість рядків коду проекту.

Для аналізу вихідних даних та подальшої побудови регресійних моделей їх необхідно перевірити на нормальність розподілу. Перевірку будемо здійснювати критерієм χ^2 при рівні значимості $\alpha = 0{,}05$ та кількості ступенів вільності $\nu = 2$. Для цього необхідно розрахувати значення χ^2 за формулою (2.1) та порівняти його зі статистичним значенням:

$$\chi^2 = \sum_{i=1}^{m} \frac{(n_i - np_i)^2}{np_i}. \quad (2.1)$$

Якщо розраховане значення більше ніж статистичне значення, то дані не належать нормальному розподілу. Тому їх потрібно нормалізувати. Зазвичай емпіричні дані для обчислення розміру програмного забезпечення мають нормальний розподіл лише в окремих випадках. У даному випадку для нормалізації використовується перетворення на основі натурального логарифму.

$$z = \ln x. \tag{2.2}$$

Після процесу нормалізації перевіримо отримані дані на викиди за допомогою методу квадрату відстані Махаланобіса. Для нормалізованих даних без викидів побудуємо лінійну регресійну модель, яка має вигляд:

$$z_y = a + bz_x, \tag{2.3}$$

де a і b знаходяться за формулами:

$$b = \frac{n\sum_{i=1}^{n} y_i x_i - \sum_{i=1}^{n} y_i * \sum_{i=1}^{n} x_i}{n\sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2}; \tag{2.4}$$

$$a = \frac{\sum_{i=1}^{n} y_i - b\sum_{i=1}^{n} x_i}{n}. \tag{2.5}$$

Для отриманої моделі побудуємо довірчий інтервал та інтервал передбачення [23, 24] за наступними формулами:

$$\hat{Y} \pm t_{\alpha/2,n-2} * S_y * \sqrt{\frac{1}{n} + \frac{(X-\bar{x})^2}{\sum_{i=1}^{n} (x_i-\bar{x})^2}}; \tag{2.6}$$

$$\hat{Y} \pm t_{\alpha/2,n-2} * S_y * \sqrt{1 + \frac{1}{n} + \frac{(X-\bar{x})^2}{\sum_{i=1}^{n} (x_i-\bar{x})^2}}, \tag{2.7}$$

де $\hat{Y}$ – значення, розраховані за моделлю регресії;

$t_{\alpha/2,n-2}$ – квантиль розподілу Стьюдента;

α – статистична значимість;

n – кількість елементів у виборці;

$\bar{x}$ – середнє значення;

$S_y = \sqrt{\frac{1}{n-2} * \sum_{i=1}^{n} (y_i - \hat{y}_i)^2}$.

Зворотнє перетворення для переходу до вихідних даних має вигляд:

$$x = e^z. \tag{2.8}$$

На основі значень, що були отримані за допомогою лінійної регресійної моделі (2.3), з використанням зворотнього нормалізуючого перетворення (2.8) перейдемо до нелінійної регресійної моделі:

$$y = e^{a+\varepsilon} * x^{b}. \quad (2.9)$$

Довірчий інтервал та інтервал прогнозування для отриманого нелінійної регресійної моделі знаходиться за допомогою зворотнього перетворення (2.8) для даних, отриманих з лінійної моделі.

2.2 Побудова нелінійної моделі регресії та знаходження довірчого інтервалу та інтервалу передбачення

За допомогою перевірки даних критерієм χ^2 Пірсона (2.1) було встановлено, що вони не належать нормальному закону розподілу. З використанням нормалізуючого перетворення (2.2) були отримані нормалізовані дані. Для них було проведено перевірку на викиди. Даний процес є ітераційним. Після перевірки даних методом квадрату відстані Махаланобіса за допомогою формул (1.1) – (1.3) на першій ітерації було виявлено три викиди. Необхідно, щоб після проходження ітерацій у даних не було викидів. Щоб повністю позбавитись від даних, що є викидами, було проведено три ітерації. В результаті, з вибірки у 60 проектів, залишилось 55.

Отримані емпіричні та нормалізовані дані без викидів можна побачити у табл. 2.1.

Таблиця 2.1 – Емпіричні та нормалізовані дані без викидів

№	Y кількість рядків коду	X кількість класів	Zy нормаліз. Y	Zx нормаліз. X
1	2	3	4	5
1	543	47	6,297	3,850
2	587	50	6,375	3,912
3	804	50	6,690	3,912
4	383	46	5,948	3,829
5	682	43	6,525	3,761
6	946	52	6,852	3,951
7	226	38	5,421	3,638
8	524	47	6,261	3,850
9	498	45	6,211	3,807

Продовження табл. 2.1

1	2	3	4	5
10	804	41	6,690	3,714
11	625	40	6,438	3,689
12	803	43	6,688	3,761
13	722	39	6,582	3,664
14	1005	52	6,914	3,951
15	528	45	6,269	3,807
16	1154	53	7,051	3,970
17	882	41	6,782	3,714
18	894	48	6,796	3,871
19	965	55	6,872	4,007
20	743	51	6,611	3,932
21	247	35	5,509	3,555
22	844	43	6,738	3,761
23	347	36	5,849	3,584
24	487	45	6,188	3,807
25	1261	54	7,140	3,989
26	223	38	5,407	3,638
27	583	46	6,368	3,829
28	489	49	6,192	3,892
29	342	42	5,835	3,738
30	623	49	6,435	3,892
31	1126	53	7,026	3,970
32	1093	52	6,997	3,951
33	459	46	6,129	3,829
34	506	51	6,227	3,932
35	993	52	6,901	3,951
36	1028	56	6,935	4,025
37	1269	54	7,146	3,989
38	548	45	6,306	3,807
39	932	52	6,837	3,951
40	248	35	5,513	3,555
41	1160	55	7,056	4,007
42	1137	56	7,036	4,025
43	656	48	6,486	3,871
44	1198	55	7,088	4,007
45	885	44	6,786	3,784
46	1285	54	7,159	3,989
47	662	39	6,495	3,664
48	689	44	6,535	3,784
49	1210	54	7,098	3,989
50	929	44	6,834	3,784
51	443	40	6,094	3,689
52	459	41	6,129	3,714

Завершення табл. 2.1

1	2	3	4	5
53	579	46	6,361	3,829
54	579	47	6,361	3,850
55	1295	54	7,166	3,989

Гістограми з отриманими нормалізованими даними без викидів можна побачити на рис. 2.1 і 2.2.

Рисунок 2.1 – Нормалізовані X

Рисунок 2.2 – Нормалізовані Y

Для побудови лінійної регресійної моделі потрібно знайти коефіцієнти регресії a та b. Їх знаходять за допомогою методу найменших квадратів за формулами (2.4) та (2.5). За знайденими коефіцієнтами будуємо лінійну регресійну модель:

$$y = -4{,}13 + 2{,}77x + \varepsilon.$$

Також для отриманої моделі необхідно розрахувати значення довірчого інтервалу (2.6) та інтервалу передбачення (2.7).

В ході удосконалення математичної моделі були розраховані значення регресійних залишків ε_i за формулою:

$$\varepsilon_i = \widehat{y_i} - y_i,$$

які показали, що дані знаходяться у нормальному розподілі при рівні значимості $\alpha = 0{,}05$ та кількості ступенів вільності $\nu = 2$.

Математичне сподівання розрахованих залишків дорівнює:

$$M(\varepsilon) = 0.$$

Дисперсія розрахованих залишків дорівнює:

$$D(\varepsilon) = 0{,}1.$$

Для побудови нелінійної регресійної моделі переходимо до вихідних даних з використанням зворотнього нормалізуючого перетворення (2.8). Нелінійна регресія згідно з (2.9) має вигляд:

$$y = e^{-4{,}13+\varepsilon} * x^{2{,}77}.$$

За формулами зворотнього перетворення (2.8) розраховуємо границі довірчого інтервалу та інтервалу прогнозування для побудованої нелінійної регресійної моделі. На рис. 2.3 представлено побудовану нелінійну регресійну модель, довірчий інтервал, інтервал прогнозування, а також емпіричні дані.

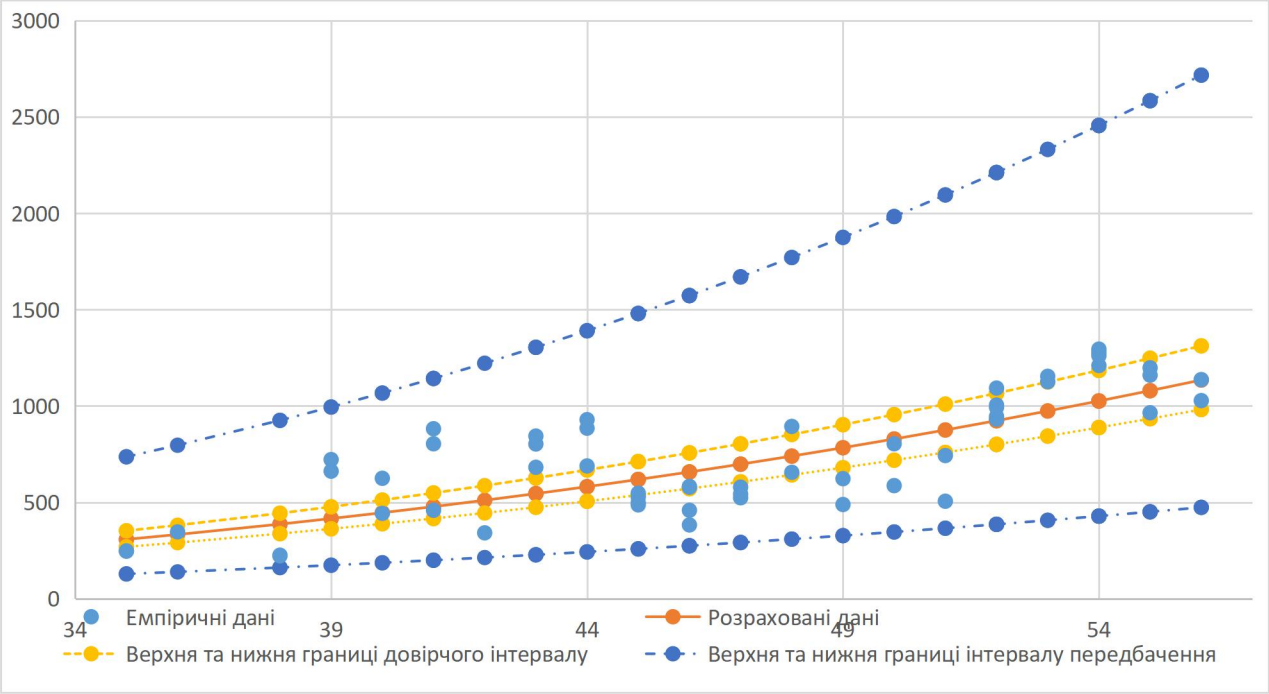


Рисунок 2.3 – Зображення нелінійної регресії, її інтервалів та емпіричних даних на графіку

Значення інтервалів для нелінійної моделі наведено у табл. 2.2

Таблиця 2.2 – Значення інтервалів для нелінійної моделі

№	Розра-хункове значення	Ліва границя довірчого інтервалу	Права границя довірчого інтервалу	Ліва границя інтервалу прогнозув.	Права границя інтервалу прогнозув.
1	2	3	4	5	6
1	698,099	606,424	803,633	291,860	1669,783
2	828,775	718,909	955,431	346,412	1982,808
3	828,775	718,909	955,431	346,412	1982,808
4	657,682	571,590	756,742	274,984	1572,988
5	545,498	474,780	626,749	228,131	1304,372
6	924,001	800,752	1066,218	386,155	2210,973
7	387,185	337,819	443,765	161,986	925,462

Продовження табл. 2.2

1	2	3	4	5	6
8	698,099	606,424	803,633	291,860	1669,783
9	618,793	538,051	711,652	258,744	1479,862
10	478,005	416,443	548,667	199,936	1142,808
11	446,369	389,072	512,103	186,718	1067,090
12	545,498	474,780	626,749	228,131	1304,372
13	416,104	362,871	477,146	174,072	994,662
14	924,001	800,752	1066,218	386,155	2210,973
15	618,793	538,051	711,652	258,744	1479,862
16	974,121	843,790	1124,583	407,069	2331,083
17	478,005	416,443	548,667	199,936	1142,808
18	740,070	642,576	852,357	309,383	1770,310
19	1079,502	934,195	1247,411	451,036	2583,662
20	875,560	759,132	1009,845	365,939	2094,901
21	308,230	269,333	352,745	128,984	736,570
22	545,498	474,780	626,749	228,131	1304,372
23	333,275	291,072	381,597	139,454	796,481
24	618,793	538,051	711,652	258,744	1479,862
25	1025,947	888,264	1184,970	428,693	2455,292
26	387,185	337,819	443,765	161,986	925,462
27	657,682	571,590	756,742	274,984	1572,988
28	783,620	680,064	902,945	327,563	1874,631
29	511,039	445,005	586,872	213,737	1221,880
30	783,620	680,064	902,945	327,563	1874,631
31	974,121	843,790	1124,583	407,069	2331,083
32	924,001	800,752	1066,218	386,155	2210,973
33	657,682	571,590	756,742	274,984	1572,988
34	875,560	759,132	1009,845	365,939	2094,901
35	924,001	800,752	1066,218	386,155	2210,973
36	1134,813	981,603	1311,936	474,109	2716,251
37	1025,947	888,264	1184,970	428,693	2455,292
38	618,793	538,051	711,652	258,744	1479,862
39	924,001	800,752	1066,218	386,155	2210,973
40	308,230	269,333	352,745	128,984	736,570
41	1079,502	934,195	1247,411	451,036	2583,662
42	1134,813	981,603	1311,936	474,109	2716,251
43	740,070	642,576	852,357	309,383	1770,310
44	1079,502	934,195	1247,411	451,036	2583,662
45	581,407	505,788	668,332	243,130	1390,345
46	1025,947	888,264	1184,970	428,693	2455,292
47	416,104	362,871	477,146	174,072	994,662
48	581,407	505,788	668,332	243,130	1390,345

Завершення табл. 2.2

1	2	3	4	5	6
49	1025,947	888,264	1184,970	428,693	2455,292
50	581,407	505,788	668,332	243,130	1390,345
51	446,369	389,072	512,103	186,718	1067,090
52	478,005	416,443	548,667	199,936	1142,808
53	657,682	571,590	756,742	274,984	1572,988
54	698,099	606,424	803,633	291,860	1669,783
55	1025,947	888,264	1184,970	428,693	2455,292

Для дослідження моделі необхідно розрахувати характеристики якості отриманої регресії, а саме: коефіцієнт детермінації (R^2), середню величину відносної похибки (MMRE), рівень прогнозування (PRED (0,25)).

Для розрахунку коефіцієнту детермінації необхідно застосовувати формулу (1.3).

Для розрахунку середньої величини відносної похибки необхідно використовувати формулу (1.4).

Для розрахунку рівня прогнозування необхідно застосовувати формулу (1.5).

Для отриманої нелінійної моделі, були розраховані характеристики адекватності:

$$\mathrm{R}^2 = 0{,}61;$$

$$\mathrm{MMRE} = 0{,}249;$$

$$\mathrm{PRED}\ (0{,}25) = 0{,}618.$$

Для порівняння розробленої нелінійної регресійної моделі побудуємо лінійну модель у припущенні про нормальність розподілу емпіричних даних. Знайдемо для неї коефіцієнти за формулами (2.4) та (2.5). Отримана лінійна регресійна модель має вигляд:

$$\mathrm{y} = -1138{,}922 + 40{,}221\mathrm{x} + \varepsilon.$$

Побудуємо довірчий інтервал та інтервал передбачення для лінійної моделі у припущенні про нормальність розподілу. Емпіричні дані, значення лінійної моделі та інтервали показано на рис 2.4.

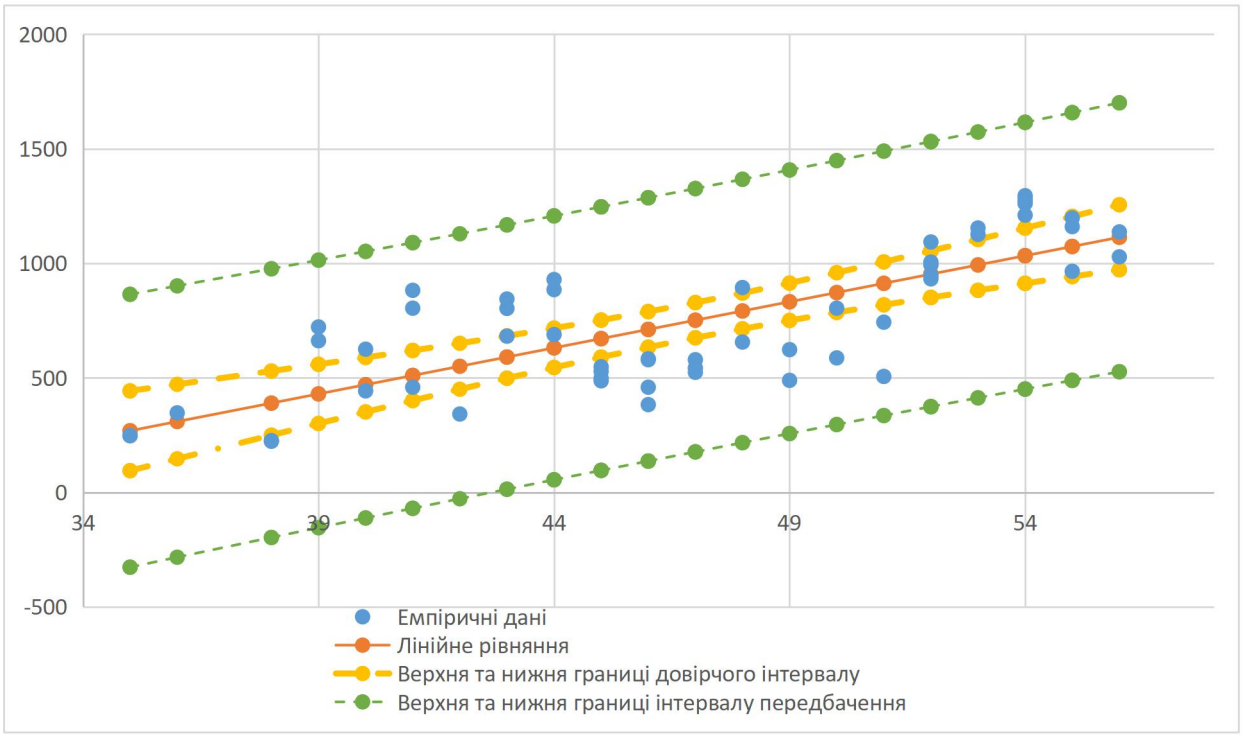


Рисунок 2.4 – Зображення лінійної регресії у припущенні про нормальність розподілу, її інтервалів та емпіричних даних на графіку

В результаті за формулами (1.3), (1.4) та (1.5) були розраховані оцінки адекватності моделі:

$$R^2 = 0{,}59;$$

$$\text{MMRE} = 0{,}267;$$

$$\text{PRED}\,(0{,}25) = 0{,}581.$$

Інтервали приведено у табл. 2.3.

Таблиця 2.3 – Значення інтервалів для лінійної моделі у припущенні про нормальність розподілу

№	Розрахо-ване значення	Ліва границя довірчого інтервалу	Права границя довірчого інтервалу	Ліва границя інтервалу прогнозув.	Права границя інтервалу прогнозув.
1	2	3	4	5	6
1	751,511	658,352	844,671	174,422	1328,601
2	872,177	778,070	966,285	294,934	1449,420
3	872,177	778,070	966,285	294,934	1449,420
4	711,289	618,448	804,130	134,251	1288,327
5	590,623	498,743	682,503	13,739	1167,508
6	952,621	857,887	1047,355	375,275	1529,967
7	389,513	299,257	479,769	-187,115	966,141
8	751,511	658,352	844,671	174,422	1328,601
9	671,067	578,545	763,589	94,080	1248,054
10	510,179	418,945	601,413	-66,603	1086,961
11	469,957	379,048	560,866	-106,773	1046,688
12	590,623	498,743	682,503	13,739	1167,508
13	429,735	339,152	520,318	-146,944	1006,414
14	952,621	857,887	1047,355	375,275	1529,967
15	671,067	578,545	763,5892	94,080	1248,054
16	992,843	897,797	1087,889	415,446	1570,240
17	510,179	418,945	601,413	-66,603	1086,961
18	791,733	698,256	885,210	214,592	1368,874
19	1073,287	977,620	1168,954	495,787	1650,787
20	912,399	817,978	1006,820	335,104	1489,694
21	268,847	179,580	358,114	-307,627	845,321
22	590,623	498,743	682,503	13,739	1167,508
23	309,069	219,472	398,667	-267,456	885,595
24	671,067	578,545	763,589	94,080	1248,054
25	1033,065	937,708	1128,422	455,617	1610,513
26	389,513	299,257	479,769	-187,115	966,141
27	711,289	618,448	804,130	134,251	1288,327
28	831,955	738,162	925,748	254,763	1409,147
29	550,401	458,843	641,959	-26,432	1127,234
30	831,955	738,162	925,748	254,763	1409,147
31	992,843	897,797	1087,889	415,446	1570,240
32	952,621	857,887	1047,355	375,275	1529,967
33	711,289	618,448	804,130	134,251	1288,327
34	912,399	817,978	1006,820	335,104	1489,694
35	952,621	857,887	1047,355	375,275	1529,967

Продовження табл. 2.3

1	2	3	4	5	6
36	1113,509	1017,533	1209,484	535,958	1691,060
37	1033,065	937,708	1128,422	455,617	1610,513
38	671,067	578,545	763,589	94,080	1248,054
39	952,621	857,887	1047,355	375,275	1529,967
40	268,847	179,580	358,114	-307,627	845,321
41	1073,287	977,620	1168,954	495,787	1650,787
42	1113,509	1017,533	1209,484	535,958	1691,060
43	791,733	698,256	885,210	214,592	1368,874
44	1073,287	977,620	1168,954	495,787	1650,787
45	630,845	538,643	723,047	53,909	1207,781
46	1033,065	937,708	1128,422	455,617	1610,513
47	429,735	339,152	520,318	-146,944	1006,414
48	630,845	538,643	723,047	53,909	1207,781
49	1033,065	937,708	1128,422	455,617	1610,513
50	630,845	538,643	723,047	53,909	1207,781
51	469,957	379,048	560,866	-106,773	1046,688
52	510,179	418,945	601,413	-66,603	1086,961
53	711,289	618,447	804,130	134,251	1288,327
54	751,511	658,352	844,671	174,422	1328,601
55	1033,065	937,708	1128,422	455,617	1610,513

Таким чином була побудована нелінійна модель регресії та лінійна модель у припущенні про нормальність розподілу емпіричних даних.

2.3 Результати проведених досліджень

Для оцінювання розміру програмних проектів, як складової трудомісткості, за метрику було обрано кількість класів проекту.

Виходячи з отриманих результатів усі оцінки нелінійної регресійної моделі мають кращі показники, ніж лінійна модель. У нелінійній регресійній моделі ліва границя інтервалу передбачення не містить від’ємних значень на відміну від лінійної моделі у припущенні про нормальність. Для побудованої нелінійної моделі було зменшено ширини довірчих інтервалів та інтервалів передбачення для більшості точок, на відміну від лінійної моделі, побудованої у припущенні про нормальність розподілу.

За отриманими покращеними показниками можна сказати, що удосконалена математична модель може використовуватись для оцінювання розміру веб-застосунків, створених мовою C# за допомогою платформи ASP.NET.

2.4 Постановка задачі на розробку програмного забезпечення для оцінювання розміру веб-застосунків, які створюються мовою програмування C# з використанням платформи ASP.NET

Провівши аналіз методів, моделей та готових рішень було зроблено висновок, що існуючі програмні продукти є досить складними у розрахунках та не враховують проекти, які були реалізовані мовою програмування C# з використанням платформи ASP.NET.

Виходячи з цього, було прийнято рішення створити власне програмне забезпечення для оцінювання розміру веб-застосунків, створених на мові програмування C# з використанням платформи ASP.NET. Таким чином, сформулюємо наступну постановку задачі – розробити програмне забезпечення для оцінювання розміру веб-застосунків, створених на мові програмування C# з використанням платформи ASP.NET.

Функціональні вимоги програмного забезпечення

Розроблене програмне забезпечення повинно виконувати наступні функції:

– внесення даних по проектам;
– нормалізація даних та видалення викидів;
– розрахунок коефіцієнтів регресії;
– побудови нелінійної регресійної моделі з використанням зворотнього нормалізуючого перетворення;
– побудови лінійної регресійної моделі у припущенні про нормальність розподілу емпіричних даних;
– перевірка адекватності отриманих моделей.

Вимоги до вхідних та вихідних даних

Вхідним файлом програмного забезпечення є файл з даними, необхідними для подальшого дослідження. Дані, що містяться у файлі, повинні бути оформлені у правильному форматі без помилок та зберігатися у файлі формату .csv.

Вихідні дані повинні виводитися в таблиці поруч із вхідними даними.

Вимоги до складу та параметрів технічних засобів

Система користувача повинна задовольняти вказаним мінімальним вимогам для коректного запуску програми:

– CPU: Intel Pentium 4405Y (1.2 ГГц);
– RAM: 1 GB;
– 500 Mb вільного місця на диску.

Вимоги до інформаційної та програмної сумісності

Для повноцінного функціювання системи необхідно використовувати: ОС Windows версії не нижче 7. Для коректної роботи програми оцінювання розміру проекту необхідна наявність офісного пакету MS Excel 2007 або вище.

3 ПРОЕКТ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ОЦІНЮВАННЯ РОЗМІРУ ПРОГРАМНИХ ПРОЕКТІВ

В ході створення програмного забезпечення для оцінювання розміру веб-застосунків для створення моделей було обрано мову моделювання UML.

3.1 Ескізний проект програмного забезпечення

В ході розробки ескізного проекту було розроблено:

– діаграму прецедентів та специфікації до неї;
– діаграми діяльності, що стосуються основних прецедентів;
– інтерфейс користувача.

3.1.1 Діаграма прецедентів

Для визначення вимог до розробки програмного забезпечення для оцінювання розміру веб-застосунків, було створено діаграму прецедентів. Вона зображена на рисунку 3.1

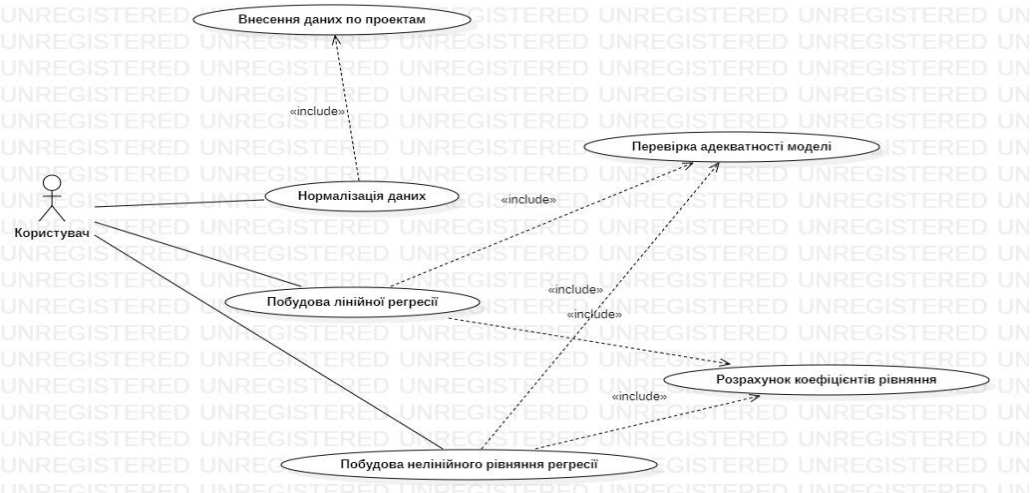


Рисунок 3.1 – Діаграма прецедентів

Для детального сприйняття даної моделі необхідно привести їх специфікації.

Специфікація прецеденту *Внесення даних по проектам*

Назва прецеденту

Внесення даних по проектам.

Опис прецеденту

Прецедент ініціюється активним суб'єктом Користувач та дає змогу внести дані по проектам.

Дійові особи прецеденту

Користувач.

Зв’язок з іншими прецедентами

Відсутні.

Передумови

Запуск програми.

Базовий потік

Користувач натискає кнопку «Обрати файл» після чого відкривається провідник, де обирається шлях до файлу у форматі .csv. Користувач обирає потрібний файл та підтверджує вибір.

Альтернативні потоки

1. Невірний формат файлу. Виникає у разі, якщо користувач обирає неправильний формат файлу. У цьому разі система виводить помилку про невірний формат.
2. Файл пустий. Виникає у разі, коли користувач обирає файл, який не містить у собі даних. У цьому випадку система виводить помилку про те, що обраний файл є пустим.

Постумови

При успішному завершенні прецеденту обраний файл буде доданий до проекту.

Важливість

Висока.

Особливості

Користувач не зможе працювати доки не обере потрібний файл.

Специфікація прецеденту *Нормалізація даних*

Назва прецеденту

Перегляд програми поточної події.

Опис прецеденту

Дана функція розраховує такі параметри для вихідних вибірок: (кількість значень вибірки; вибіркове середнє; дисперсія; середньоквадратичне відхилення) та призначений для перевірки нульовою гіпотезою нормального розподілу та подальшої нормалізації вихідних даних.

Дійові особи прецеденту

Користувач.

Зв’язок з іншими прецедентами

Має зв’язок стереотипом include із варіантом використання «Внесення даних по проектам».

Передумови

Користувач вибрав коректний формат файлу.

Базовий потік

1. Користувач натискає кнопку «Нормалізувати».
2. Програма розраховує вихідні дані вибірок.
3. Програма перевіряє дані на нормальність.

Альтернативні потоки

Файл з даними пустий. Виникає у разі, коли користувач обирає файл який не містить у собі даних. У цьому випадку система виводить помилку про те, що обраний файл є пустим.

Постумови

Система записує нормалізовані дані у файл.

Важливість

Висока.

Особливості

Відсутні.

Специфікація прецеденту *Розрахунок коефіцієнтів рівняння*

Назва прецеденту

Розрахунок коефіцієнтів рівняння.

Опис прецеденту

Призначений для розрахунку коефіцієнтів рівняння за даними з файлу.

Дійові особи прецеденту

Користувач.

Зв’язок з іншими прецедентами

Відсутній.

Передумови

Успішне завантаження файлу з нормалізованими даними.

Базовий потік

1. Програма розраховує коефіцієнти.
2. Система виводить розраховані коефіцієнти.

Альтернативні потоки

Файл пустий. Виникає у разі, коли користувач обирає файл який не містить у собі даних. У цьому випадку система виводить помилку про те, що обраний файл є пустим.

Постумови

Система записує розраховані коефіцієнти у файл.

Важливість

Висока.

Особливості

Відсутні.

Специфікація прецеденту *Побудова рівняння лінійної регресії*

Назва прецеденту

Побудова рівняння лінійної регресії.

Опис прецеденту

Призначений для побудови рівняння лінійної регресії за розрахованими коефіцієнтами.

Дійові особи прецеденту

Користувач.

Зв'язок з іншими прецедентами

Має зв'язок стереотипом include із варіантами використання «Розрахунок коефіцієнтів рівняння» та «Перевірка адекватності моделі».

Передумови

Завантаження файлу з розрахованими коефіцієнтами.

Базовий потік

1. Користувач натиснув кнопку «Побудувати лінійне рівняння регресії».
2. Відкривається нове вікно програми.
3. Користувач натиснув кнопку «Побудувати» на вкладці «Побудувати лінійне рівняння регресії».

Альтернативні потоки

Файл пустий. Виникає у разі, коли користувач обирає файл який не містить у собі даних. У цьому випадку система виводить помилку про те, що обраний файл є пустим.

Постумови

Запис даних лінійного рівняння у файл.

Важливість

Висока.

Особливості

Відсутні.

Специфікація прецеденту *Перевірка адекватності моделі*

Назва прецеденту

Перевірка адекватності моделі.

Опис прецеденту

Призначений для перевірки адекватності моделі за допомогою розрахунку параметрів та побудови довірчого інтервалу та інтервалу передбачення.

Дійові особи прецеденту

Користувач.

Зв’язок з іншими прецедентами

Відсутні.

Передумови

Побудова моделі регресії.

Базовий потік

Система виводить параметри рівняння регресії, будує графік для рівняння регресії, довірчого інтервалу та інтервалу прогнозування.

Альтернативні потоки

Файл пустий. Виникає у разі, коли користувач обирає файл який не містить у собі даних. У цьому випадку система виводить помилку про те, що обраний файл є пустим.

Постумови

Вивід розрахованої інформації та графіка, запис інформації у вихідний файл.

Важливість

Висока.

Особливості

Користувач має змогу переглядати лише власні матеріали.

Специфікація прецеденту *Побудова рівняння нелінійної регресії*

Назва прецеденту

Побудова рівняння нелінійної регресії.

Опис прецеденту

Призначений для побудови рівняння нелінійної регресії за розрахованими коефіцієнтами та перетвореними даними.

Дійові особи прецеденту

Користувач.

Зв’язок з іншими прецедентами

Має зв’язок стереотипом include із варіантами використання «Розрахунок коефіцієнтів рівняння» та «Перевірка адекватності моделі».

Передумови

Завантаження файлів з розрахованими коефіцієнтами та перетвореними даними.

Базовий потік

1. Користувач натиснув кнопку «Побудувати нелінійне рівняння регресії»;
2. Відкривається нове вікно програми;
3. Користувач натиснув кнопку «Побудувати» на вкладці «Побудувати нелінійне рівняння регресії».

Альтернативні потоки

Файли пусті. Виникає у разі, коли користувач обирає файл який не містить у собі даних. У цьому випадку система виводить помилку про те, що обраний файл є пустим.

Постумови

Запис даних нелінійного рівняння у файл.

Важливість

Висока.

Особливості

Відсутні.

В даному підрозділі було побудовано діаграму прецедентів та її специфікації.

3.1.2 Діаграми діяльності проекту

Для візуалізації деяких прецедентів були розроблені діаграми діяльності, які дозволяють детально зобразити сценарії виконання прецедентів. Вони зображені на рис 3.2-3.4.

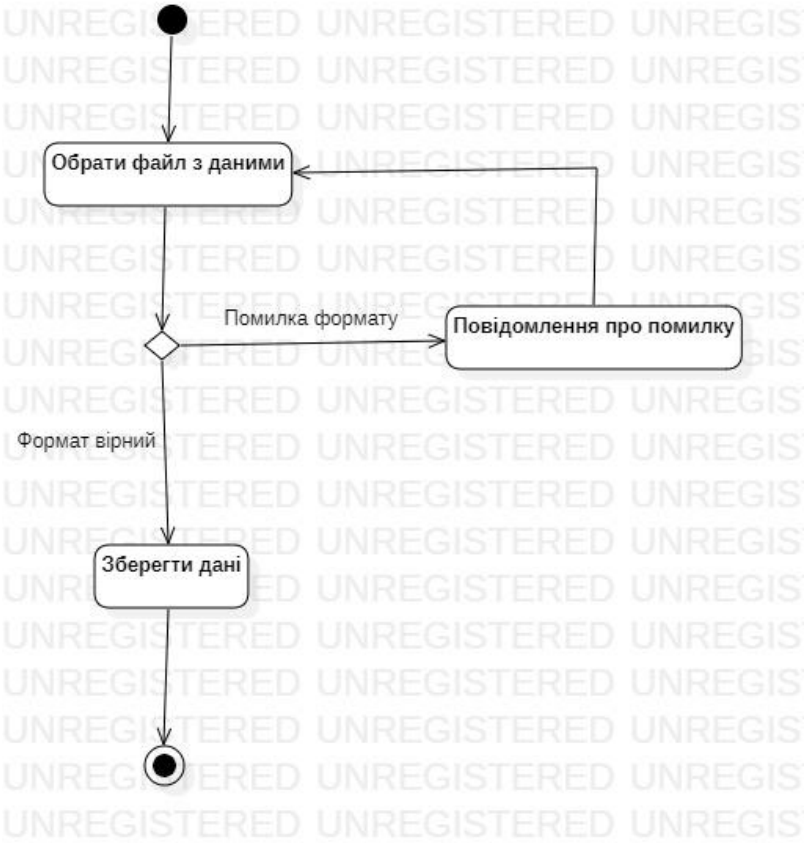


Рисунок 3.2 – Діаграма діяльності для прецеденту

”Внесення даних по проекту”

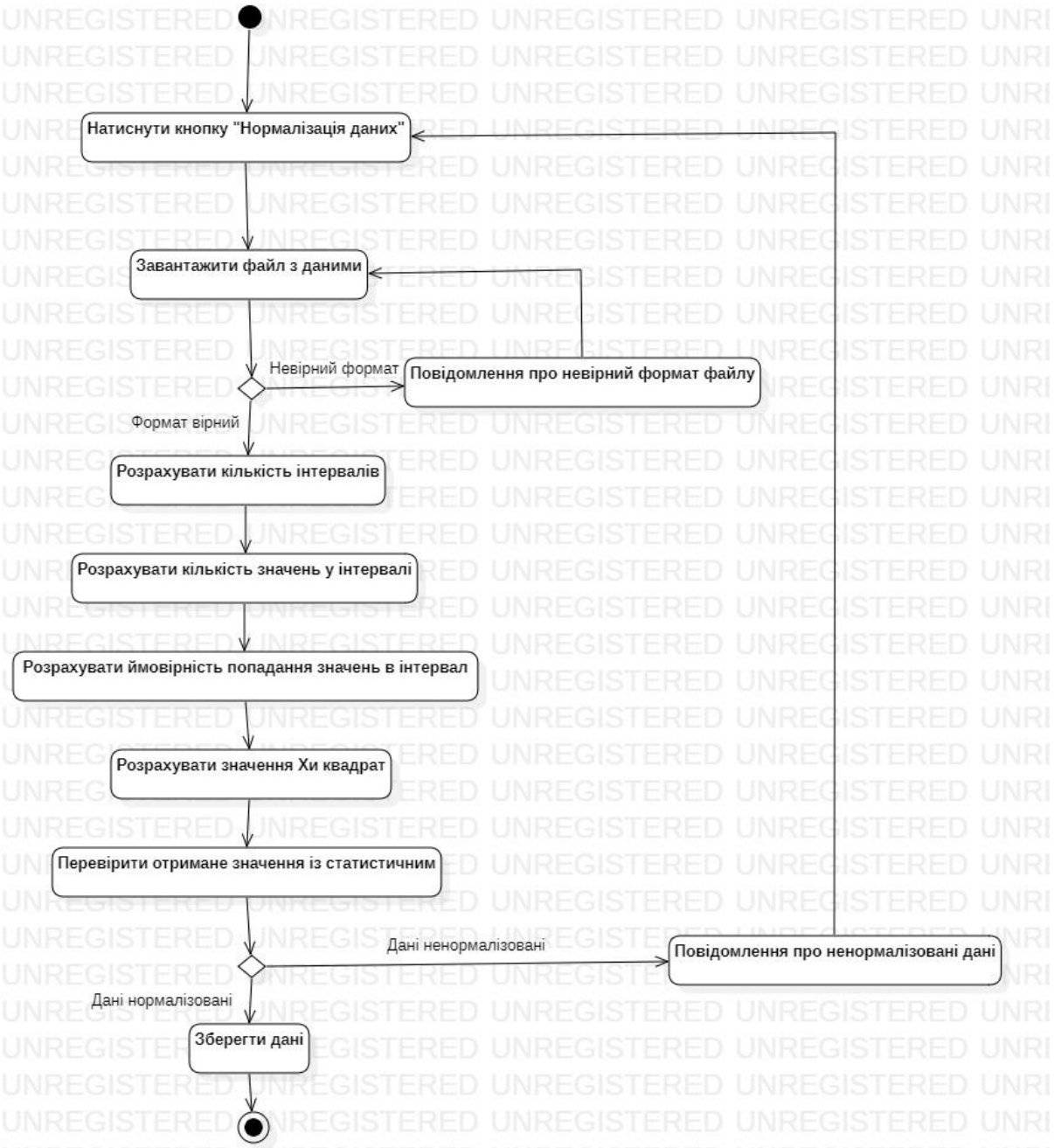


Рисунок 3.3 – Діаграма діяльності для прецеденту ”Нормалізація даних”

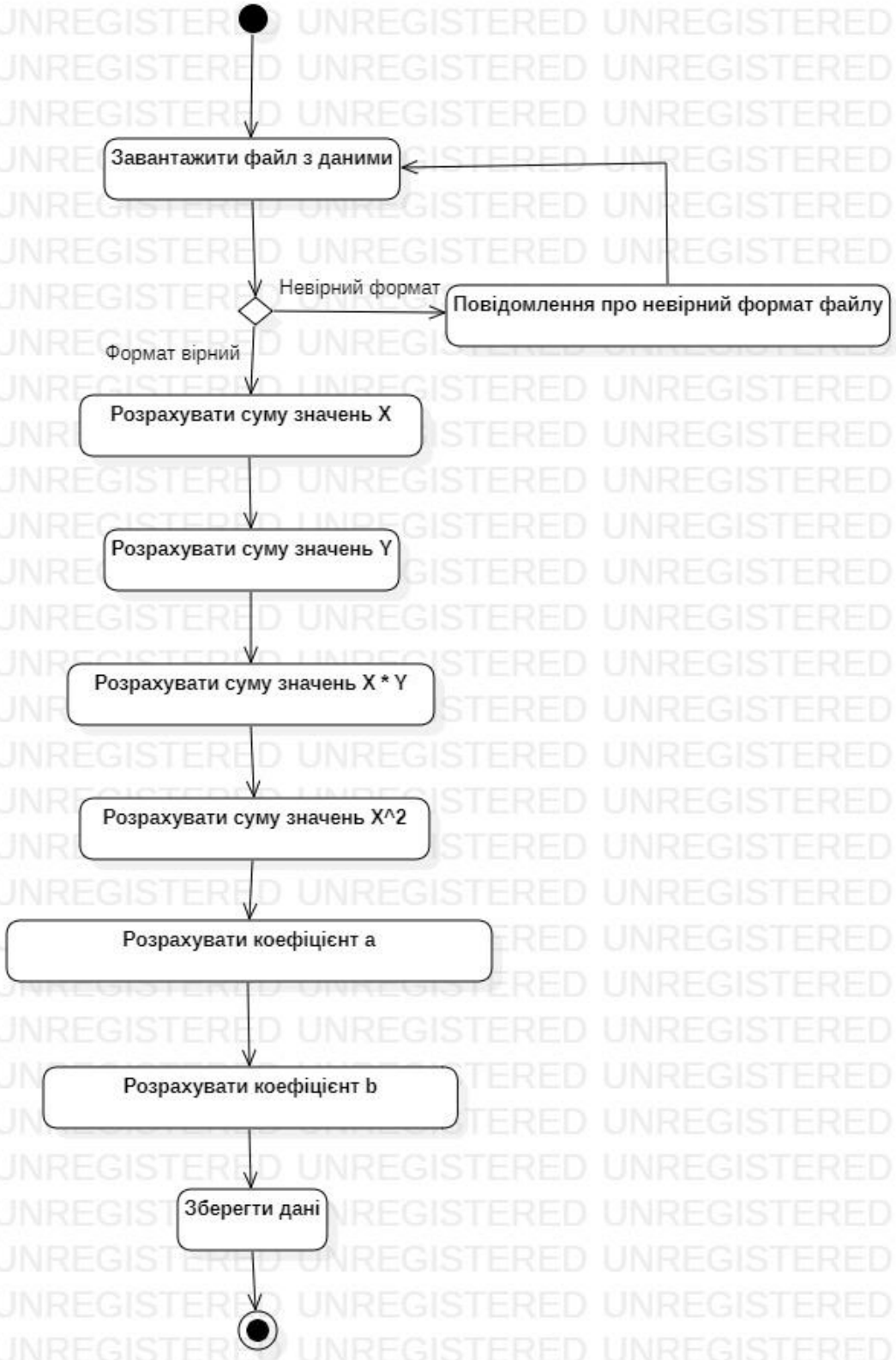


Рисунок 3.4 – Діаграма діяльності для прецеденту

”Розрахунок коефіцієнтів рівняння”

В даному підрозділі були побудовані діаграми діяльності для деяких прецедентів.

3.1.3 Розробка інтерфейсу користувача

Дуже важливим етапом при розробці програмного забезпечення є розробка інтуїтивно зрозумілого інтерфейсу користувачу, який полегшить роботу із застосунком.

На рис. 3.5-3.8 зображено основні елементи інтерфейсу

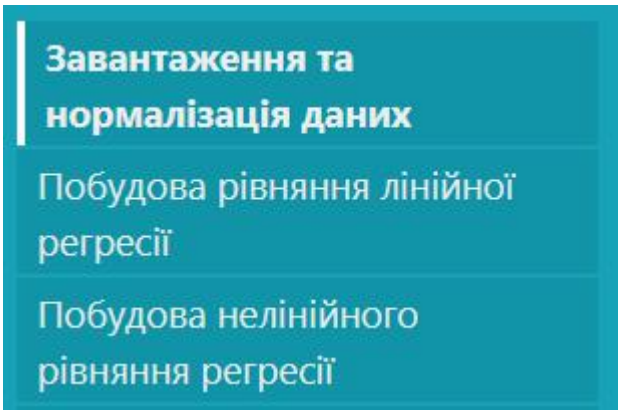


Рисунок 3.5 – Основні функції програмного забезпечення

Рисунок 3.6 – Сторінка функції “Завантаження та нормалізація даних”

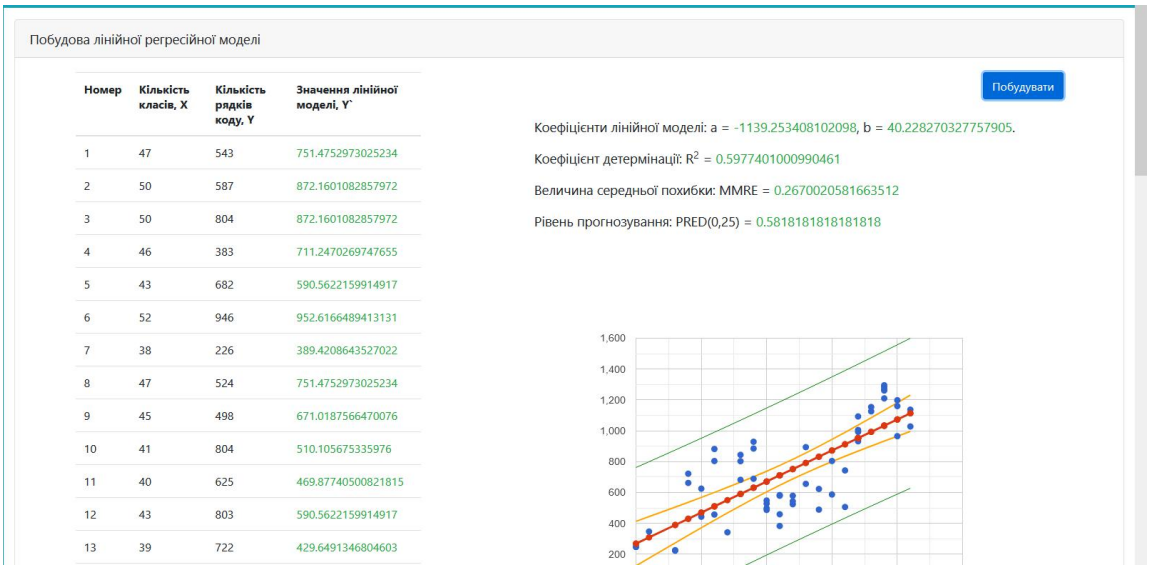


Рисунок 3.7 – Сторінка функції “Побудова лінійної регресійної моделі”

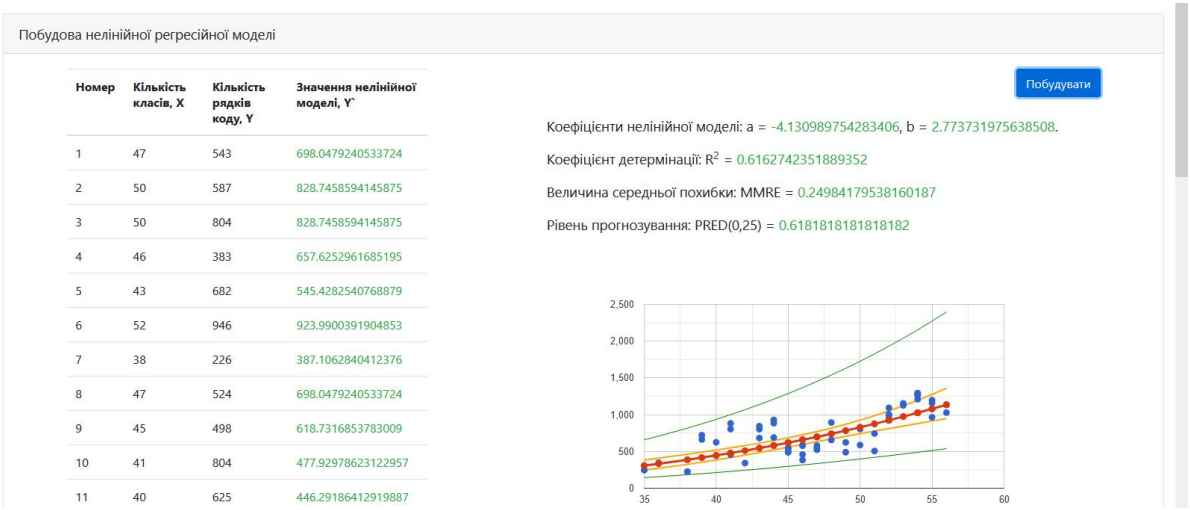


Рисунок 3.8 – Сторінка функції “Побудова нелінійної регресійної моделі”

В даному підрозділі було розроблено інтерфейс користувача для роботи з програмним забезпеченням.

3.2 Технічний проект програмного забезпечення

У процесі розробки технічного проекту було створено статичну модель, що зображена у вигляді діаграми класів, та динамічну модель – діаграми послідовності.

3.2.1 Діаграма класів

Для подання статичної моделі ПЗ використаємо діаграму класів. Вона використовується для відображення зв’язків між об’єктами та описує їх поведінку.

На рис. 3.9 зображено діаграму класів.

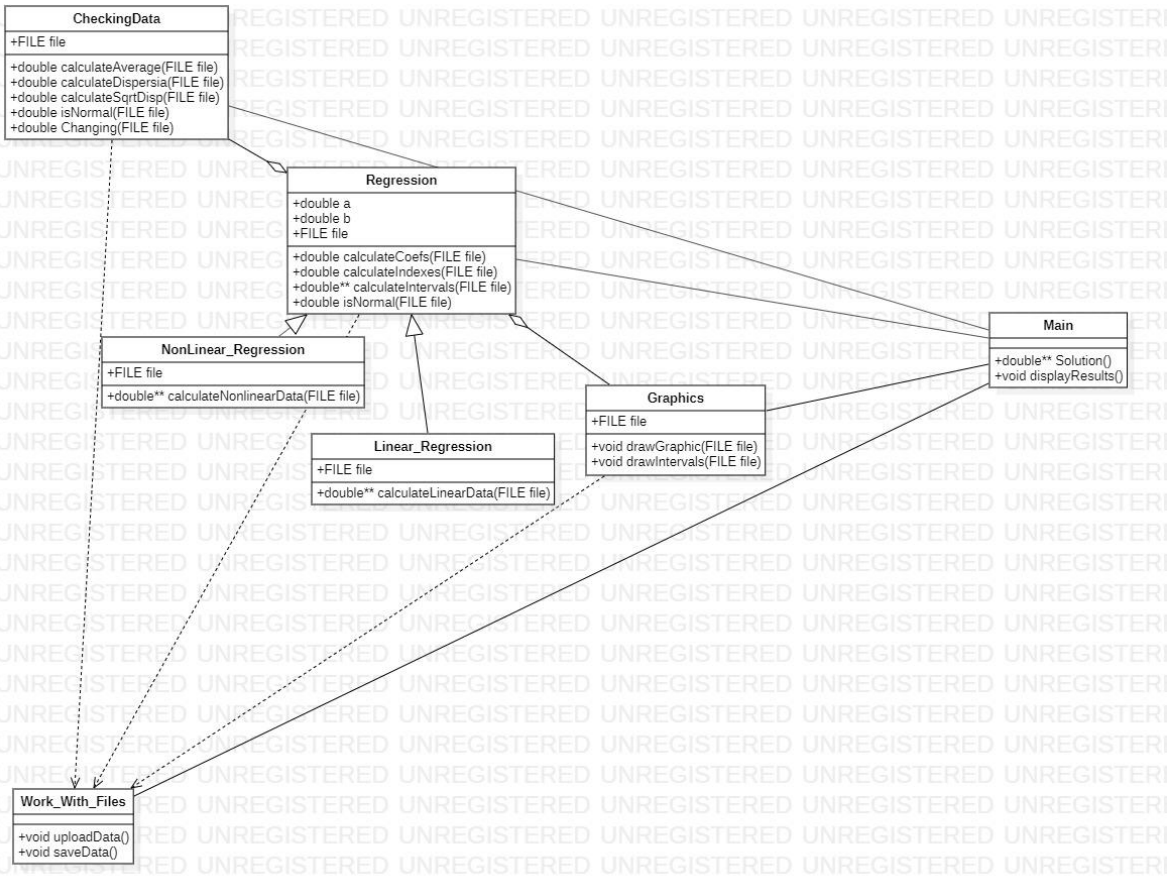


Рисунок 3.9 – Статична модель проекту

Нижче у таблицях 3.1-3.7 наведено специфікації класів проекту.

Таблиця 3.1 – Клас Main

Параметр	Значення
Коментар	Головний клас проекту, у цьому класі відбуваються основні взаємодії
Атрибути	Відсутні
Операції	Solution() – запускає основну програму. displayResults() – виводить результати програми

Таблиця 3.2 – Клас CheckingData

Параметр	Значення
Коментар	Перевіряє дані на нормальність
Атрибути	FILE file
Операції	calculateAverage() – розраховує математичне очікування. calculateDispersia() – розраховує дисперсію. calculateSqrtDisp() – розраховує середньоквадратичне відхилення. isNormal() – перевіряє отримані дані. Changing() – виконує зворотнє перетворення.

Таблиця 3.3 – Клас Work_With_Files

Параметр	Значення
Коментар	Клас для роботи з файлами
Атрибути	Відсутні
Операції	uploadData() – завантажує файл за даними saveData() – зберігає файл з даними

Таблиця 3.4 – Клас Regression

Параметр	Значення
Коментар	Клас для розрахунку атрибутів моделі
Атрибути	double a double b FILE file
Операції	calculateCoefs() – розраховує коефіцієнти рівнянь. calculateIndexes() – розраховує оцінки моделі. calculateIntervals() – розраховує інтервал передбачення та довірчий інтервал. isNormal() – перевіряє адекватність моделі.

Таблиця 3.5 – Клас Linear_Regression

Параметр	Значення
Коментар	Клас для розрахунку даних лінійної регресії
Атрибути	FILE file
Операції	calculateLinearData() – розраховує дані лінійної регресії.

Таблиця 3.6 – Клас NonLinear_Regression

Параметр	Значення
Коментар	Клас для розрахунку даних нелінійної регресії
Атрибути	FILE file
Операції	calculateNonlinearData() – розраховує дані нелінійної регресії.

Таблиця 3.7 – Клас Graphics

Параметр	Значення
Коментар	Клас для побудови отриманих графіків та інтервалів
Атрибути	FILE file
Операції	drawGraphic() – надає графічне зображення отриманих графіків. drawIntervals() – надає графічне зображення отриманих інтервалів.

У даному підрозділі була наведена статична модель проекту та її специфікації.

3.2.2 Динамічна модель проекту

На основі розробленої діаграми прецедентів та статичної моделі у вигляді діаграми класів було створено динамічну модель у вигляді діаграми послідовності. Дану модель зображено на рис. 3.10

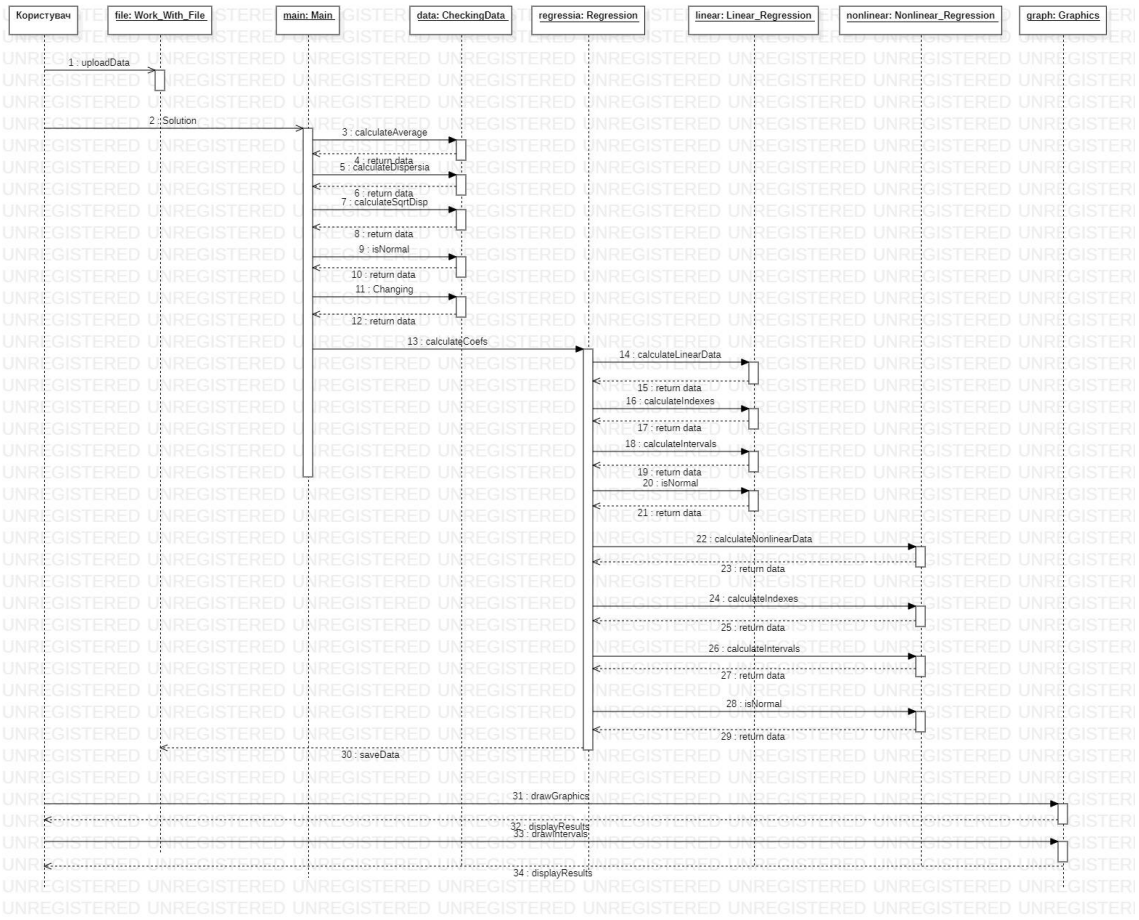


Рисунок 3.10 – Динамічна модель проекту

Таким чином, для проекту, що розроблюється, була побудована динамічна модель проекту у вигляді діаграми послідовності.

3.3 Робочий проект програмного забезпечення

На етапі робочого проекту було обґрунтовано вибір мови програмування та середовища розробки; побудовано діаграму компонентів проекту; проведено кодування, тестування та випробування створеного програмного забезпечення.

3.3.1 Обґрунтування вибору мови програмування та середовища розробки

Для розробки програмного забезпечення, що реалізує удосконалену математичну модель для оцінювання розміру веб-застосунків, розроблених на C# з використанням платформи ASP.NET була обрана мова програмування PHP8 та

середовище розробки Phpstorm. PHP є однією з найпопулярніших скриптових мов завдяки своїй простоті, швидкості виконання, багатій функціональності і розповсюдженню початкових кодів на основі ліцензії PHP.

Підставою для вибору середовища стало те, що Phpstorm –кросплатформене інтегроване середовище розробки для мови програмування PHP, яке поєднує в собі всі інструменти необхідні для створення сучасного програмного забезпечення.

Для відображення результатів виконання програми в графічному вигляді був застосован Google Chart Tools API – багатофункціональний набір інструментів для візуалізації даних, його застосування надає можливість використовувати різноманітні графічні засоби при розробці програмного забезпечення

3.3.2 Діаграма компонентів програмного забезпечення для оцінювання розміру веб-застосунків, створених мовою C# з використанням платформи ASP.NET.

Діаграму компонентів, яка описує фізичне представлення програмного забезпечення, представлено на рис. 3.11

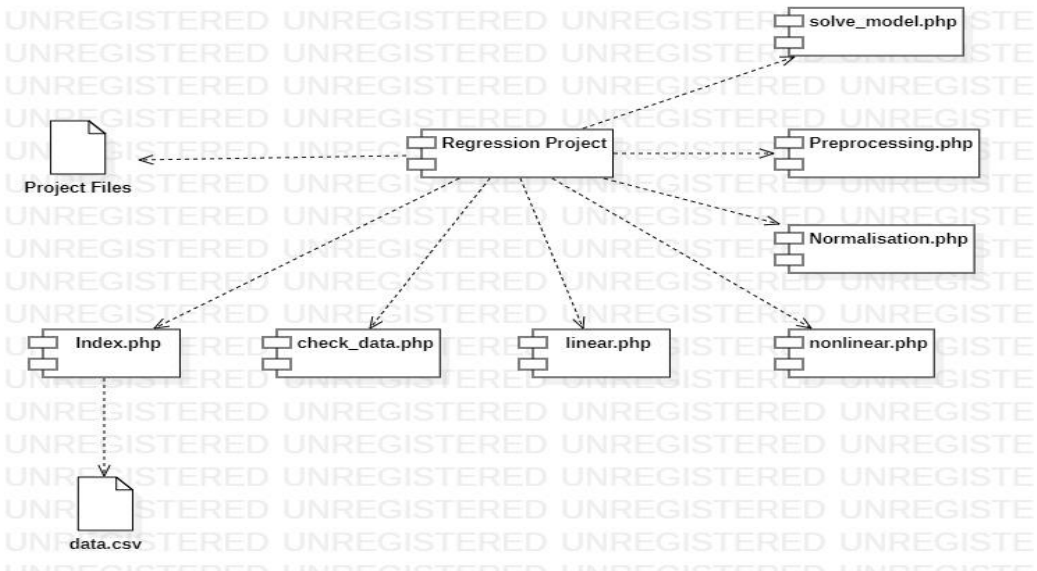


Рисунок 3.11 – Діаграма компонентів проекту

В ході розробки робочого проекту, була побудована діаграма компонентів проекту.

3.3.3 Кодування

Кодування програми проводиться на основі специфікації класів, представлених у підрозділі 3.2.1.

Нижче наведено файл з текстом основного класу веб-застосунку (index.php):

```php
<?php
    session_start();

    require_once "include/check_data.php";

    $menu_item = [
        [
            'title' => 'Завантаження та нормалізація даних',
            'url' => 'preprocessing.php',
        ],
        [
            'title' => 'Побудова рівняння лінійної регресії',
            'url' => 'linear.php',
        ],
        [
            'title' => 'Побудова нелінійного рівняння регресії',
            'url' => 'nonlinear.php',
        ],
        [
            'title' => 'Порівняння результатів побудови',
            'url' => 'd.php',
        ],
    ];

    function showMenu(&$active, &$enabled, $menu_item) {
        $active = (int)$active;
        $enabled = (int)$enabled;
        if ($active >= count($menu_item) || $active < 0)
            $active = 0;
//        if (!$active)
//            $enabled = 0;
        if ($enabled >= count($menu_item))
            $enabled = 0;
```

```
        echo '<div class="menu">';
        foreach ($menu_item as $key => $item) {
            echo "<a class=\"menu-item" .
                 ($key == $active ? ' active' : '') .
                 ($key > $enabled ? ' disable' : '') .
                 ($key && $key != $active && $key <= $enabled ?
"\" href=\"/?active=$key&enabled=-1" : '') .
                 "\"" .
                 (!$key && $enabled > 0 ? "
onclick=\"$('#clearModal').modal();\"" : '') .
                 ">{$item['title']}</a>";

        }
        echo '</div>';
    }

    $active = $_POST['active'] ?? ($_GET['active'] ?? 0);
    $enabled = $_POST['enabled'] ?? ($_GET['enabled'] ?? -1);
    $mode = $_POST['mode'] ?? 'none';
    switch ($mode) {
        case 'file':
            $lines = checkFile($_FILES['custom_file']);
            if ($lines === false)
                $file_message = "Не вдалося завантажити файл";
            break;
    }

?>
<html>
<head>
    <link rel="stylesheet" href="/vendor/css/bootstrap.min.css">
    <script src="/vendor/js/jquery-3.6.0.min.js"></script>
    <script src="/vendor/js/bootstrap.min.js"></script>
<!--    <script src="/vendor/js/google_chart.js"></script>-->
    <script type = "text/javascript" src =
"https://www.gstatic.com/charts/loader.js">
    </script>
    <script type = "text/javascript">
        google.charts.load('current', {packages: ['corechart']});
    </script>
    <style>
        body {
            font-size: 14pt !important;
        }
```

```
        .menu-item {
            display: block;
            cursor: pointer;
            color: #fff;
            background-color: #17b2c9;
            padding: 3px 10px;
            margin: 2px 0px;
        }
        .menu-item:hover {
            text-decoration: none;
            color: #fff;
        }
        .menu-item.active {
            border-left: white 4px solid;
            cursor: default;
            font-weight: bold;
            color: #fff;
        }
        .menu-item.disable {
            cursor: not-allowed;
            background-color: #1294a8;
            color: #eee;
        }
        .menu-item:not(.disable,.active):hover {
            background-color: #18c8e0;
        }
    </style>
</head>
<body>
    <div class="container-fluid d-flex flex-column h-100">
        <div class="row">
            <div class="col-12 bg-info p-4 text-white text-
center text-uppercase">
                Математична модель для оцінювання розміру веб-
застосунків, створених за допомогою платформи ASP.Net, та
розробка програмного забезпечення для її реалізації
            </div>
        </div>
        <div class="row flex-fill">
            <div class="col-2 bg-info">
                <?= showMenu($active,$enabled, $menu_item); ?>
            </div>
            <div class="col-10">
                <?php require_once
```

```
"include/{$menu_item[$active]['url']}"; ?>
            </div>
        </div>
    </div>
    <div class="modal" id="clearModal" tabindex="-1"
role="dialog">
        <div class="modal-dialog" role="document">
            <div class="modal-content">
                <div class="modal-header">
                    Попередження
                    <button type="button" class="close" data-
dismiss="modal" aria-label="Close">
                        <span aria-hidden="true">&times;</span>
                    </button>
                </div>
                <div class="modal-body">При виборі цього пункту
всі дані та розрахунки будуть знищені. Продовжити?</div>
                <div class="modal-footer">
                    <button class="btn btn-outline-danger"
type="button" data-dismiss="modal">Ні</button>
                    <button class="btn btn-success"
id="clear_conf">Так</button>
                </div>
            </div>
        </div>
    </div>
    <script>
        $('#clear_conf').click(() => {
            location = '/';
        });
    </script>
</body>
</html>
```

Повний текст програми наведено у Додатку А.

3.3.4 Тестування

Тестування проводилось для варіантів використання, таких як «Завантаження даних» та «Розрахунок коефіцієнтів рівняння».

Тестування варіанта використання «Завантаження нормалізованих даних».

Для тестування варіанта використання «Завантаження нормалізованих даних» був використаний метод еквівалентної розбивки. Виділені класи еквівалентності, представлені в таблиці 3.8. Результати тестувань представлені в таблицях 3.9 та 3.10.

Таблиця 3.8 – Класи еквівалентності

Вхідні данні	Правильні класи	Неправильні класи
Файл	Завантажено файл з правильним форматом (1)	Неправильний формат файлу(2), Файл пустий(3)

Таблиця 3.9 – Тести правильних класів еквівалентності

№ тесту	Покритий клас	Вхідні дані	Вихідні дані	Результат тестування
1	1	Завантажено файл: data.csv	Продовження роботи програми	Тест пройдено

Таблиця 3.10 – Тести неправильних класів еквівалентності

№ тесту	Покритий клас	Вхідні дані	Вихідні дані	Результат тестування
1	2	Завантажено файл: data.txt	Повідомлення про неправильний формат файлу	Тест пройдено
2	3	Завантажений файл пустий	Повідомлення про пустий файл	Тест пройдено

Тестування варіанта використання «Розрахунок коефіцієнтів рівняння».

Для тестування варіанта використання «Розрахунок коефіцієнтів рівняння» був використаний метод еквівалентної розбивки. Виділені класи еквівалентності, представлені в таблиці 3.11. Результати тестувань представлені в таблицях 3.12 та 3.13.

Таблиця 3.11 – Класи еквівалентності

Вхідні данні	Правильні класи	Неправильні класи
Файл	Завантажено файл з правильним форматом (1)	Неправильний формат файлу(2), Файл пустий(3)
Дані	Правильний формат даних (4)	Неправильний формат даних (5)

Таблиця 3.12 – Тести правильних класів еквівалентності

№ тесту	Покритий клас	Вхідні дані	Вихідні дані	Результат тестування
1	1	Завантажено файл: data.txt	Продовження роботи програми	Тест пройдено
2	4	Дані: 1,23	Продовження роботи програми	Тест пройдено

Таблиця 3.13 – Тести неправильних класів еквівалентності

№ тесту	Покритий клас	Вхідні дані	Вихідні дані	Результат тестування
1	2	Завантажено файл: data.txtr	Повідомлення про неправильний формат файлу	Тест пройдено
2	3	Завантажений файл пустий	Повідомлення про пустий файл	Тест пройдено
3	5	Неправильний формат даних: 1.23	Повідомлення про неправильний формат даних	Тест пройдено

Тестування показало, що варіанти використання працюють правильно.

3.3.5 Випробування програмного забезпечення

Згідно з програмою та методикою випробувань, яку наведено у Додатку В, було проведено випробування розробленого програмного забезпечення. Воно полягало у введені даних.

Випробування можливості додавання даних до програми полягає у наступному:

– натиснути на кнопку «Вибрати файл»;
– обрати шлях до файлу;
– файл відкривається і виводить дані, що містяться всередині.

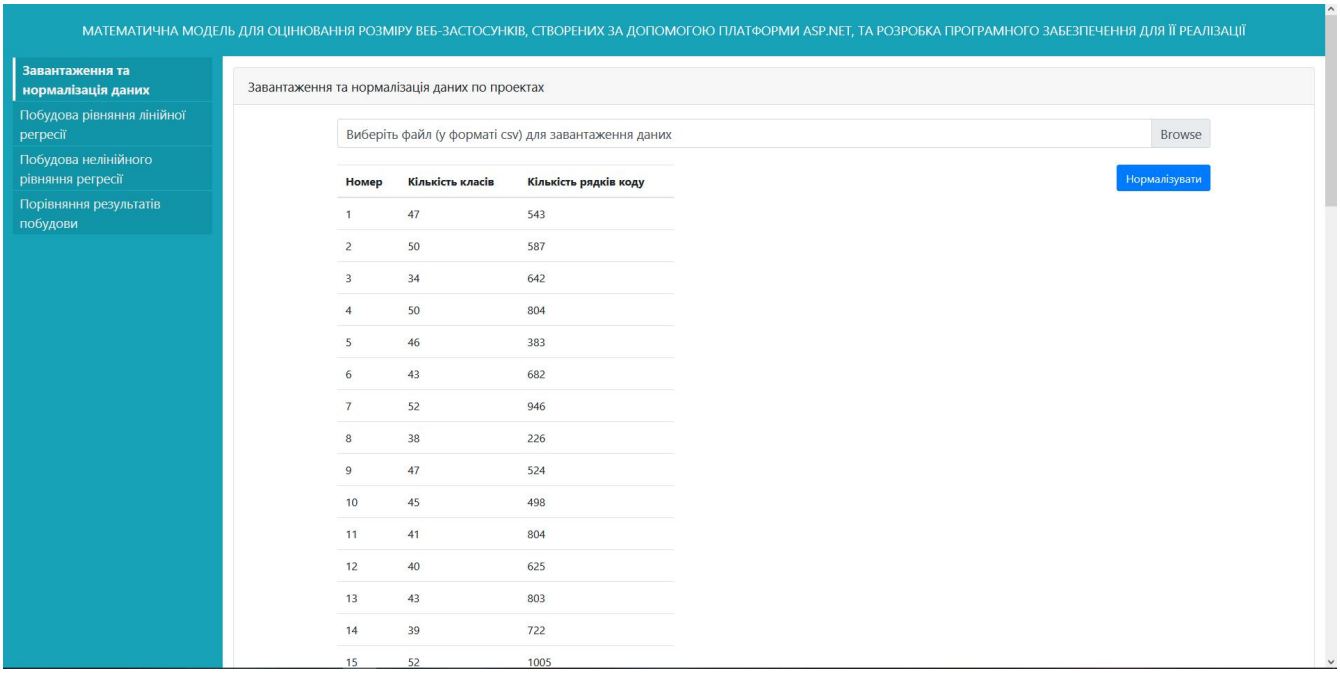


Рисунок 3.12 – Сторінка додавання файлу з даними

Додавання файлу з вихідними даними відбулось коректно і без помилок, отже, ця перевірка пройшла вдало.

Випробування можливості побудови лінійного рівняння регресії полягає у наступному:

– прочитати дані з файлу;
– натиснути на кнопку «Побудувати»;

– нові значення та їх графіки з’являються на екрані;

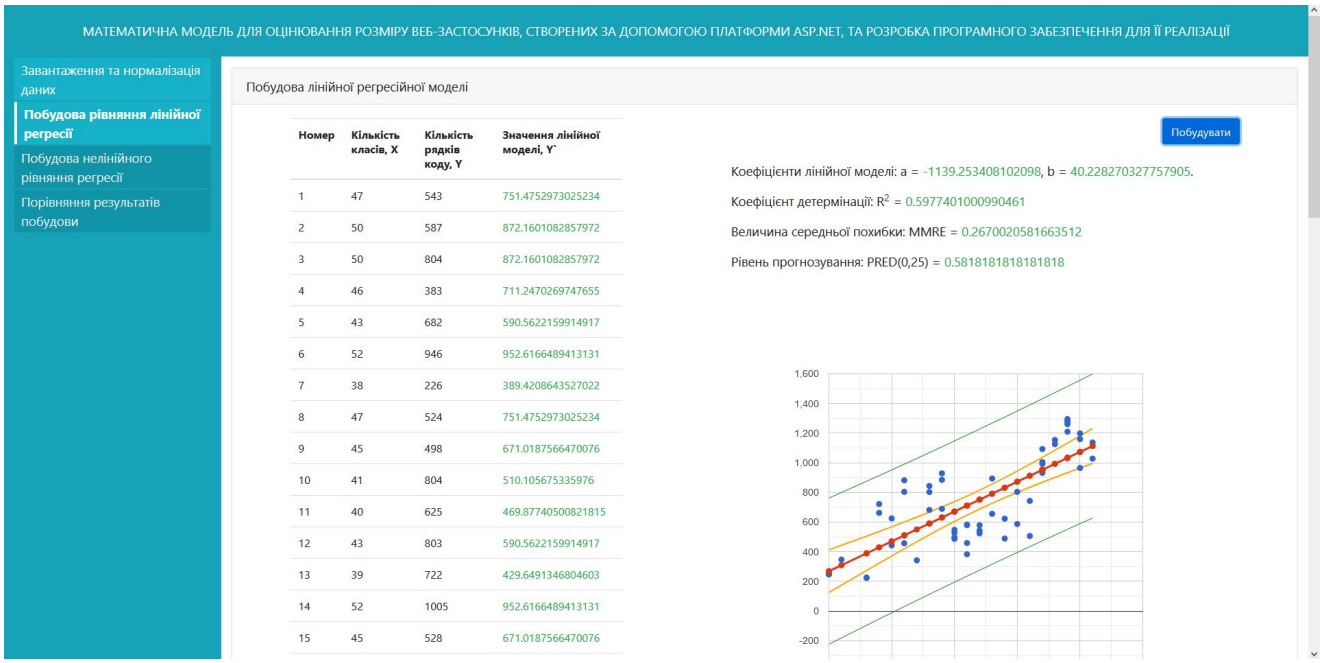


Рисунок 3.13 – Сторінка розрахунку значень лінійної моделі

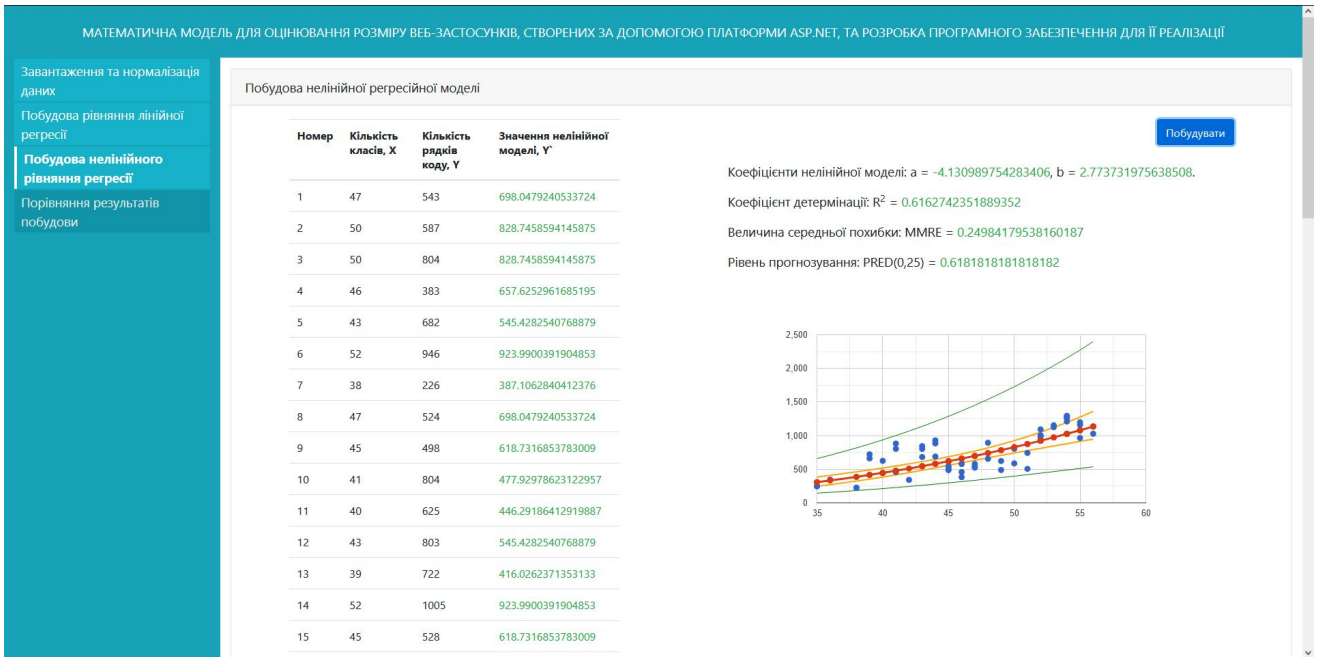
Розрахунок нових значень відбувся коректно і без помилок, отже, ця перевірка пройшла вдало.

Випробування можливості побудови нелінійного рівняння регресії полягає у наступному:

– прочитати дані з файлу;

– натиснути на кнопку «Побудувати»;

– нові значення та їх графіки з’являються на екрані;



Номер	Кількість класів, X	Кількість рядків коду, Y	Значення нелінійної моделі, Y`
1	47	543	698.0479240533724
2	50	587	828.7458594145875
3	50	804	828.7458594145875
4	46	383	657.6252961685195
5	43	682	545.4282540768879
6	52	946	923.9900391904853
7	38	226	387.1062840412376
8	47	524	698.0479240533724
9	45	498	618.7316853783009
10	41	804	477.92978623122957
11	40	625	446.29186412919887
12	43	803	545.4282540768879
13	39	722	416.0262371353133
14	52	1005	923.9900391904853
15	45	528	618.7316853783009

Рисунок 3.14 – Сторінка розрахунку значень нелінійної моделі

Розрахунок нових значень відбувся коректно і без помилок, отже, ця перевірка пройшла вдало.

Проведені випробування показали, що програмне забезпечення працює коректно.

4 РЕЗУЛЬТАТИ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

В процесі виконання кваліфікаційної роботи було розроблено програмне забезпечення щодо реалізації розробленої математичної моделі для оцінювання розміру веб-застосунків розроблених за допомогою мови програмування C# з використанням платформи ASP.NET.

Система розроблена для web-сервера Apache мовою програмування PHP та за допомогою HTML[25].

Тестування та випробування даного програмного продукту показало, що програмне забезпечення відповідає вимогам згідно з постановкою задачі та успішно справляється з поставленими перед ним задачами, а саме:

– внесенням даних по проектам;
– перевірки нормальності розподілу даних та видалення викидів;
– розрахунком коефіцієнтів рівняння;
– побудови нелінійної регресійної моделі з використанням зворотнього нормалізуючого перетворення;
– побудови лінійної регресійної моделі у припущенні про нормальність розподілу емпіричних даних;
– перевірки адекватності отриманих моделей.

Робота із програмою здійснюється з використанням зручного інтерфейсу користувача й розрахована на роботу в режимі діалогу з користувачем.

Розроблений дизайн і інтерфейс сайту для web-серверу містить 7 HTML-файлів з розширенням PHP, що містять PHP-скрипти, JavaScript та HTML код. Головну сторінку застосунку зображено на рис.4.1

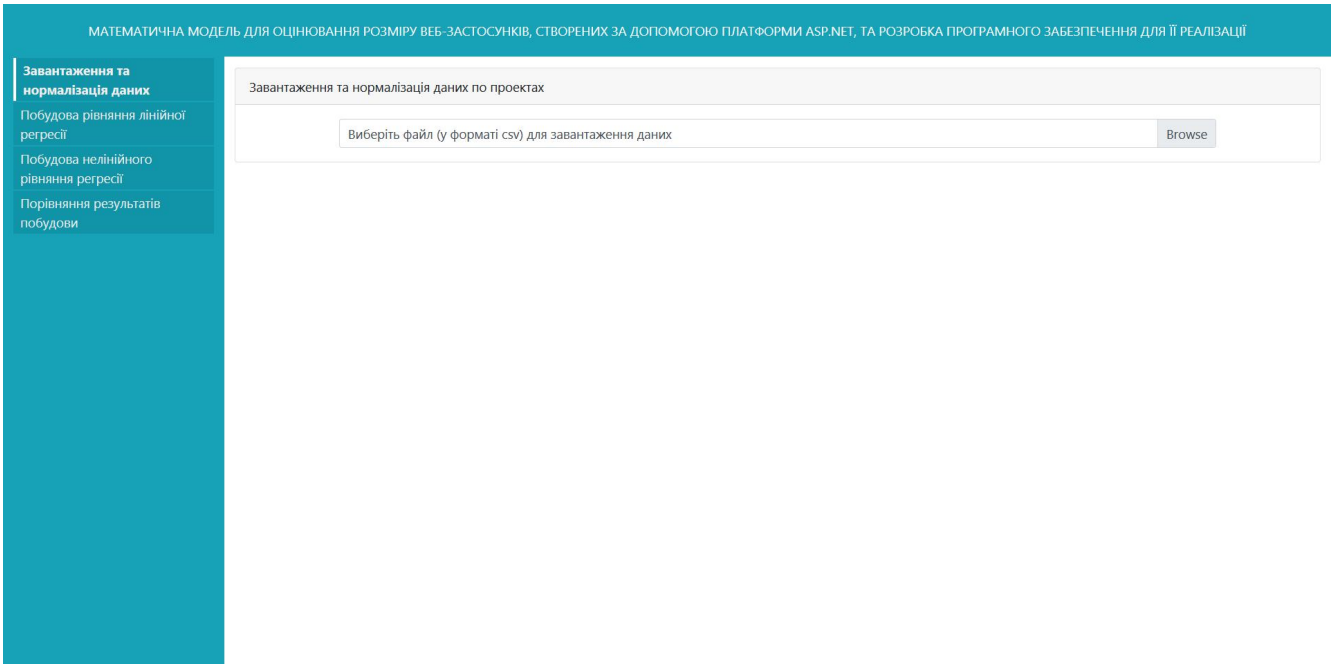


Рисунок 4.1 – Головна сторінка проекту

Програмне забезпечення сервера вимагає встановлення наступних програмних продуктів:

– Web-сервер Apache 1.3.x і вище;

– PHP інтерпретатор.

– Програмно-апаратне забезпечення для клієнта:

– будь-яка операційна система, що забезпечує мережеву взаємодію по протоколу TCP/IP;

– будь-який web-браузер, бажано працюючий із зображеннями і JavaScript, переважно Mozilla Firefox, Internet Explorer 10.0 і вище;

– підключення до Internet;

Текст програми наведено в Додатку А, опис програми наведено в Додатку Б, інструкцію користувача для роботи з розробленою системою наведена в Додатку В, методику випробувань наведено в Додатку Г.

5 РОЗРАХУНОК ЕКОНОМІЧНОЇ ЕФЕКТИВНОСТІ ВІД РОЗРОБКИ ТА ВПРОВАДЖЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Вступ

Дана робота присвячена розробці програмного забезпечення для оцінювання розміру програмного забезпечення яке створюється мовою програмування C# за допомогою платформи ASP.NET.

Ефективність інформаційних систем визначається ступенем їх позитивного впливу на підприємство, що управляється або процес в результаті використання засобів обчислювальної техніки, програмного забезпечення, зміни інформаційних потоків та організаційної структури управління.

Створення програмного забезпечення вимагає одноразових витрат на розробку, придбання необхідних технічних засобів і поточних витрат на функціонування продукту. Економія від функціонування ПЗ визначається з урахуванням витрат на його експлуатацію. Відношення цієї економії до витрат на створення програмного забезпечення характеризує економічну ефективність капітальних вкладень. Економічні показники визначаються за діючими на момент розрахунку оптовими цінами, тарифами і ставками заробітної плати.

5.1 Розрахунок витрат на створення та впровадження програмного забезпечення

Витрати на розробку програмного забезпечення складаються з витрат на заробітну плату розробника, на амортизацію та експлуатацію ЕОМ, на якій виконується розробка, на засоби розробки та витрати на матеріали і комплектуючі.

Розробка програмного забезпечення виконується спеціалістом, місячний оклад якого складає 18000 грн. Додаткова заробітна плата складає 10% від основної. Виходячи з цього, основна і додаткова заробітна плата розроблювача системи 19800 грн./міс, а вартість сучасної ПЕОМ складає 13000 грн. Вартість кіловат- години електроенергії дорівнює 1,68 грн. Витрати на допоміжні матеріали наведені в табл. 5.1

Таблиця 5.1 – Витрати на допоміжні матеріали

Пункти витрат	Сума, грн.
Папір	120
Заправлення картриджа до принтера	300
Непередбачені витрати	200
Разом матеріали і комплектуючі вироби.	620

Вартість розробки системи розраховується за формулою:

$$С_{пр} = (З_{зп} + З_{сз} + З_{зг} + З_{е}) * Т + З_{м}, \quad (5.1)$$

де Т – тривалість розробки, міс.

$З_{зп}$ – основна і додаткова заробітна плата обслуговуючого персоналу, грн

$З_{сз}$ – відрахування на соціальні заходи (38% від основної і додаткової заробітної плати), грн.;

$З_{зг}$ – загальногосподарські витрати (10% від заробітної плати), грн.; $З_{е}$ – витрати на електроенергію;

$З_{м}$ – витрати на основні і допоміжні матеріали.

Розрахуємо $З_{е}$ при споживанні потужності 0,3 Вт, тривалості роботи на місяць рівної 21* 8 = 168 годин і вартості кіловат-години 1,44 грн. до 250кВт включно та 1,68 після 250кВт, отримаємо

$$З_{е} = 168 * 1{,}44 * 0{,}3 \ = 72{,}58 \text{ грн.}$$

Представимо всі поточні витрати на розробку програмного забезпечення в таблиці 5.2.

Таблиця 5.2 – Витрати на розробку програмного забезпечення

Найменування витрат	Одиниця виміру	Кількість
Тривалість розробки	міс.	3
Основна і додаткова заробітні плати	грн.	19800
Відрахування на соціальні заходи	грн.	7524
Загальногосподарські витрати	грн.	1980
Витрати на допоміжні матеріали	грн.	620
Витрати на електроенергію	грн.	72,58

Відповідно до формули (5.1) вартість розробки ПЗ складає:

$$С_{пр} = (19800 + 7524 + 1980 + 72{,}58) * 3 + 620 = 88749{,}74 \text{ грн.}$$

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

$$Аоб = 13000 * 0{,}6 = 7800 \text{ грн.}$$

У масштабах підприємства річні витрати на основні і допоміжні матеріали визначаються в розмірі 5% вартості основного устаткування:

$$Вм = 13000 * 0{,}05 = 650 \text{ грн.}$$

Річний обсяг робіт ПЕОМ у годинах визначається в такий спосіб:

$$Фм = 253{,}3 * Тз,$$

де $Т_з$ – це середнє місячне завантаження устаткування (близько 7 годин),

253,3 – середня кількість робочих днів у році:

$$Фм = 253{,}3 * 7 = 1773{,}1 \text{ год.}$$

Витрати на електроенергію $З_е$ при 1773,1 годинах роботи устаткуванняв рік складуть:

$$Зе = 1773,1 * 0,3 * 1,44 = 765,98 \text{ грн.}$$

Експлуатаційні витрати для ПЕОМ за рік складуть:

$$Ззр = 7800 + 650 + 765,98 = 9215,98 \text{ грн.}$$

В перший рік витрати на створення й експлуатацію програмного забезпечення складуть:

$$Зсе = 88749,74 + 9215,98 = 97965,72 \text{ грн.}$$

5.2 Розрахунок економічної ефективності розробки

Визначимо пряму економічну ефективність, ґрунтуючись на тому, що впровадження програмного забезпечення вивільняє одного працівника (за експертною оцінкою фахівців підприємства).

Зарплата одного працівника в рік складає:

$$18000 * 12 * 1 = 216000 \text{грн.}$$

Річний економічний ефект розраховується по формулі:

$$Э_{рік} = \Delta C_n - E_n \cdot k \ , \qquad (5.2)$$

де ΔC_n – вивільнені кошти після впровадження системи (216000 грн.) мінус експлуатаційні витрати (9215,98 грн.);

E_n – коефіцієнт ефективності(дорівнює коефіцієнту амортизації (0,6));

k – одноразові витрати на впровадження продукту (97965,72).

$$Э_{рік} = 216000 - 9215,98 - 97965,72 * 0,6 = 148004,59 \text{ грн.}$$

Строк окупності системи розраховується по формулі:

$$T = \frac{k}{\Delta C} = \frac{97965{,}72}{216000 - 9215{,}98} \approx 0{,}47$$

Отже строк окупності програмного забезпечення складає 5,6 місяців.

Висновки

У цьому розділі були здійснені розрахунки на створення й експлуатацію програмного забезпечення, а також розрахунки економічної ефективності та строку окупності програмного забезпечення. Виходячи з розрахунків, впровадження такого програмного забезпечення окупить себе протягом першого року експлуатації за 5,6 місяців. Отже, можна зробити висновок, що дане програмне забезпечення економічно вигідне та обґрунтоване.

6 ОХОРОНА ПРАЦІ

Вступ

Охорона праці – це система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних та лікувально-профілактичних заходів та засобів, спрямованих на збереження здоров'я і працездатності людей в процесі праці (ст. 1 Закону «Про охорону праці»). Як правовий інститут охорона праці – це чималий комплекс правових норм. Можна визначити охорону праці також як систему забезпечення безпеки, збереження здоров'я і працездатності людини в процесі праці, засновану на сукупності законодавчих актів і відповідних їм соціально-економічних, технічних, гігієнічних і організаційних заходів.

Умови праці на робочому місці, безпека технологічних процесів, машин, механізмів, устаткування та інших засобів виробництва, стан засобів колективного та індивідуального захисту, що використовуються працівником, а також санітарно-побутові умови повинні відповідати вимогам нормативних актів про охорону праці.

Власник або уповноважений ним орган повинен впроваджувати сучасні засоби техніки безпеки, які запобігають виробничому травматизму, і забезпечувати санітарно-гігієнічні умови, що запобігають виникненню професійних захворювань працівників.

На власника або уповноважений ним орган покладається систематичне проведення інструктажу (навчання) працівників з питань охорони праці, протипожежної охорони.

Охорона праці – один з центральних інститутів трудового права. Він має виключно практичне значення. Недодержання вимог охорони праці створює небезпеку для здоров'я і життя працівників. У свою чергу і ті, кого законодавець називає власником або уповноваженим ним органом, несуть сувору, у тому числі й кримінальну відповідальність за порушення правил охорони праці.

6.1 Аналіз шкідливих факторів у відділі розробки програмного забезпечення для IT компанії

Небезпечним називається виробничий фактор, вплив якого на працюючу людину в певних умовах приведе до травми або іншому раптовому різкому погіршенню здоров'я. Якщо ж виробничий фактор приведе до захворювання або зниження працездатності, то його вважають шкідливим. Залежно від рівня й тривалості впливу шкідливий виробничий фактор може стати небезпечним.

На офісного працівника при роботі впливають дві основні групи шкідливих факторів:

1) Фізичні, до яких відносять фактори, що породжуються безпосереднім впливом оточуючого середовища (електромагнітне випромінювання, освітленість робочого місця, температура повітря, шум).

2) Психофізіологічні, до яких відносять фізичне та нервово-психічне перевантаження організму (зорове напруження, статичні навантаження, нервово-психічне навантаження).

Освітлення – отримання, розподіл та використання світлової енергії для забезпечення нормальних умов праці. Світло, крім забезпечення фізичної можливості розрізняти предмети праці, також впливає на психічний стан людини та перебіг фізіологічних процесів в організмі. Раціонально влаштоване освітлення виробничих приміщень позитивно впливає на працівників, підвищує ефективність та безпеку праці, знижує втому та травматизм, забезпечує високу працездатність.

За вимогами санітарних норм освітлення повинно бути достатньо яскравим, рівномірним, не засліплювати очі та не створювати відблисків на робочій поверхні. За спектральним складом освітлення має наближатися до сонячного світла. Гігієнічними нормами вимагається максимально використовувати природне освітлення, оскільки денне світло краще сприймається органами зору. Штучне освітлення у виробничих та адміністративно-громадських приміщеннях має здійснюватися системою загального рівномірного освітлення, допускають застосування системи комбінованого освітлення. Значення освітленості на поверхні робочого столу в зоні розміщення документів має становити 300-500лк.

як джерела світла для штучного освітлення мають застосовувати переважно люмінесцентні лампи типу ЛБ.

Одним з найнесприятливіших факторів, що зменшує працездатність та уважність робітників, створює передумови до травматизму та захворювань, є шум. Інтенсивний шум та вібрації негативно впливають не лише на слух, а і на нервову, серцево-судинну, та більшість інших систем організму, викликаючи зниження слуху, гіпертонію, психічні розлади тощо.

Гігієнічними нормативами (СН 3222-85, ГОСТ 12.1.003-85, ГР 2411-81) встановлені допустимі рівні звуку різних частот для робітників певних професій. У табл. 6.1 наведені нормативи для людей, що працюють з ЕОМ.

Таблиця 6.1 – Допустимі рівні шуму

Вид трудової діяльності	Рівні звукового тиску в дБ октавних смугах із середньо геометричними частотами, ГЦ									
	31,5	63	125	250	500	1000	2000	4000	8000	Рівні звуку, еквівалентні рівні звуку, дБа/дБАекв
Програмісти ЕОМ	86	71	61	54	49	45	42	40	38	50
Оператори в залах обробки інформації на ЕОМ та оператори комп’ютерного набору	96	83	74	68	63	60	57	55	54	65
В приміщеннях для розташування шумних агрегатів ЕОМ	103	91	83	77	73	70	68	66	64	75

Виходячи з даних норм, при проведенні комплексу заходів зі зменшення шуму усувається не будь-яка віброакустична активність обладнання, а тільки та, яка перевищує встановлений граничний рівень та шкідливо впливає на організм людини. Усунення промислового шуму в першу чергу відбувається шляхом вдосконалення технологічних процесів та обладнання. При цьому в першу чергу усуваються найбільш сильні джерела шуму. Також необхідно знешумлювати усе обладнання, яке цього потребує, оскільки знешумлення окремих одиниць при наявності великої кількості обладнання, що продовжує випромінювати шум, недоцільно.

Самопочуття та працездатність людини також сильно залежать від теплового стану організму, на який впливають такі фізичні фактори виробничого середовища, як температура повітря, вологість, швидкість руху повітря, атмосферний тиск, доля кисню у повітрі тощо. Наприклад, якщо кількість кисню в повітрі зменшиться до 12% то утруднюється дихання. В таких умовах людина напружує дихальний апарат, дихає частіше; такий стан людина витримує до 0,5 години. Перегрівання організму відбувається за умов надлишкового конвективного випромінювання тепла нагрітих поверхонь. При перегріванні вступають в активну реакцію пристосувальні функції організму. При цьому активізується робота серцево-судинної та дихальної систем, відбувається інтенсивне потовиділення, котре сягає 5л за зміну. З потом втрачається велика кількість мінеральних солей та вітамінів. Вологість повітря суттєво впливає на терморегуляцію людського оргазму. Підвищення відносної вологості повітря у виробничому приміщенні ускладнює терморегуляцію, зменшення тепловиділення організмом. Фізіологічно оптимальною є відносна вологість в межах 40..60%.

Сукупність описаних показників стану навколишнього середовища називають мікрокліматом. Для приміщень з ЕОМ встановлені норми мікроклімату приведені у табл. 6.2.

Таблиця 6.2 – Норми мікроклімату для приміщень з ЕОМ

Пора року	Категорія робіт	Температура повітря, С, не більше	Відносна вологість повітря, %	Швидкість руху повітря, м/с
Холодна	Легка – 1а	22-24	40-60	0,1
	Легка – 1б	21-23	40-60	0,1
Тепла	Легка – 1а	23-25	40-60	0,1
	Легка – 1б	22-24	40-60	0,2

Робота комп'ютерів і периферійних пристроїв можлива завдяки електричному струмові, до джерел якого вони підключаються, тому існує небезпека ураження користувача струмом. Щоб уникнути нещасних випадків варто строго дотримуватися правил електробезпечності.

Небезпека електричного струму на відміну від інших небезпек збільшується тим, що людина не в змозі без спеціальних приладів виявити напругу дистанційно, а також швидкоплинністю поразки – небезпека виявляється, коли людина вже уражена. Ураження струмом приводить до різних порушень в організмі, викликаючи як місцеву поразку тканин і органів, так і загальне ураження організму.

Людина відчуває дратівну дію змінного струму промислової частоти силою 0,6 -1,5 мА і постійного струму 5-7 мА. Ці струми не представляють серйозної небезпеки для організму людини, тому що при такій силі струму можливе самостійне звільнення людини від контакту зі струмоведучими частинами. Для перемінного струму промислової частоти сила невідпускаючого струму знаходиться в межах 6-20 мА. Постійний струм не викликає невідпускаючого ефекту, але приводить до сильних болючих відчуттів, сила такого струму 15-80 мА і більше.

6.2 Розробка заходів щодо зменшення впливу шкідливих факторів

Сьогодні фахівці в області ергономіки вже зрозуміли, що не можна знайти ідеальне положення, у якому можна перебувати і працювати протягом усього

робочого дня. Для більшості людей комфортабельним робочим місцем повинне бути таке, котре можна пристосувати не менш чим для двох позицій, при цьому положення крісла, монітора повинні щораз відповідати виконуваній роботі і звичкам. Багато хто вважають, що для роботи на комп'ютері більше підходять вертикальне і злегка похиле положення. Можливо, щоб крісло було злегка нахилене вперед.

Схеми розміщення робочих місць із персональним комп'ютером повинні ураховувати відстані між робочими столами з відеомоніторами (в напрямку тилу поверхні одного відеомонітора й екрана іншого відеомонітора), які повинні бути не менш 2 м, а відстань між бічними поверхнями відеомоніторів – не менш 1,2 м.

Робочий стіл повинен мати простір для ніг висотою не менш 600 мм, шириною не менш 500 мм, глибиною на рівні колін не менш 450 мм і на рівні витягнутих ніг не менш 650 мм.

Дисплей. Положення тіла звичайно відповідає напрямку погляду; дисплеї, розташовані занадто низько чи під неправильним кутом, є основними причинами сутулості. Відстань від дисплея до очей може варіюватися в залежності від характеру виконуваної роботи – 40-70 см. При роботі з текстом відстань від екрана до очей повинна лише небагато перевищувати відстань між книгою й очима і складати 40-45 см. Необхідно давати відпочинок очам і час від часу їх просто закривати. Не можна допускати, щоб очі увесь час були сфокусовані на одній відстані, працюючи з текстом, рекомендується установлювати великий шрифт. Дуже важливо, щоб екран монітора не мерехтів. Частота регенерації зображення повинна бути не менш 72 Гц, тому що при більш низькій частоті помітне мерехтіння, хоча люди з особливо чуттєвим зором можуть помітити його і при більш високій частоті. Їм доведеться підібрати графічну плату і монітор з частотою регенерації 85 Гц і вище.

Необхідно уникати того, щоб монітор був звернений убік вікна, оскільки інтенсивна освітленість області зору може затопити потоками світла очі і розмити зображення на сітківці. Якщо приходиться сидіти поруч з вікном, то треба розташуватися під прямим кутом до нього, причому екран дисплея повинний бути перпендикулярний шибці – цим виключаються відблиски на екрані.

Для зниження впливу низькочастотного електромагнітного випромінювання монітора на ЭЛТ рекомендується вибирати монітор, що задовольняє стандарту MPR II, чи TCO 92-95. У шншому випадку необхідно установити на екран захисний фільтр, найбільш сучасні моделі, затримують до 99 % електромагнітного випромінювання.

Форма спинки крісла повинна повторювати форму спини працюючого. Крісло необхідно установити на такій висоті, щоб не почувався тиск на куприк (крісло розташоване занадто низько) на стегна (крісло розташоване занадто високо). Фахівці з ергономіки, вважали, що кут між стегнами і хребтом повинний складати 90°, однак недавно проведені дослідження показали, що більшість людей переважно сидять трохи відкинувшись.

Клавіатуру необхідно установити так, щоб не треба було до неї тягтися і нахилятися вперед. При зміні положення тіла (наприклад, з вертикального на похиле) обов'язково треба перемінити положення клавіатури і дисплея. Може виявитися корисною регульована підставка клавіатури, завдяки якій можна без усякої напруги працювати з маніпулятором "миша". Можна поставити клавіатуру і на коліна. Руки при роботі повинні бути прямими в зап'ястях і зігнуті в ліктях приблизно під прямим кутом. Пальці також повинні бути злегка зігнуті. Удари по клавішах не повинні бути занадто сильними.

Зручна висота столу особливо важлива в тому випадку, коли на ньому розташовується клавіатура. Якщо в клавіатури немає підставки, а висоту столу не можна змінити (і він занадто високий), то треба вище підняти сидіння крісла, а під ноги підставити лавочку: чи що-небудь інше. Якщо стіл занадто низький, необхідно підкласти що-небудь під його ніжки.

Для попередження передчасної втоми рекомендується чергувати роботу з ЕОМ і без неї. Якщо режим роботи цього не дозволяє, тоді при інтенсивній роботі з ПЕОМ перерви встановлюють через кожні 45-60 хв роботи на 10-15 хв. У нічний час (з 22 до 6 год) тривалість регламентованих перерв збільшують на 30%.

Таблиця 6.3 – Сумарний час регламентованих перерв при роботі з ЕОМ

Категорія роботи з ПЕОМ	Рівень навантаження за робочу зміну при видах робіт з ПЕОМ			Сумарний час регламентованих перерв, хв	
	група А, кількість знаків	група Б, кількість знаків	група В, год.	при 8-годинній зміні	при 12-годинній зміні
I	до 20000	до 15000	до 2	50	80
II	до 40000	до 30000	до 4	70	110
III	до 60000	до 40000	до 6	90	140

Внутрішньозмінні режими праці і відпочинку містять додаткові нетривалі перерви в періоди, що передують появі стомлення і зниження працездатності.

Впродовж робочої зміни мають передбачатися:

– перерви для відпочинку і вживання їжі (обідні перерви);

– перерви для відпочинку і особистих потреб (згідно з трудовими нормами)

– додаткові перерви, що вводяться для окремих професій з урахуванням особливостей трудової діяльності.

У всіх випадках, коли виробничі обставини не дозволяють застосувати регламентовані перерви, тривалість безперервної роботи з ВДТ не повинна перевищувати 4 години.

При 12-годинній робочій зміні регламентовані перерви повинні встановлюватися в перші 8 годин роботи аналогічно перервам при 8-годинній робочій зміні, а протягом останніх чотирьох годин роботи, незалежно від характеру трудової діяльності, через кожну годину тривалістю 15 хвилин.

З метою зменшення негативного впливу монотонності є доцільним застосовувати чергування операцій обробки тексту і числових даних (зміна змісту роботи), чергування вводу даних та редагування текстів. Для зниження нервово-емоційного напруження, стомлення зорового аналізатора, поліпшення мозкового кровообігу, подолання несприятливих наслідків гіподинамії, запобігання втомі доцільні деякі перерви використовувати для виконання комплексу вправ, приклади яких наведено нижче.

Комплекс вправ для очей:

1) вихідне положення - сидячи. Швидко моргати очима протягом 15 с;

2) вихідне положення - сидячи на віддалі 30-35 см од вікна обличчям до нього. Дивитися на позначку на шибці протягом 5 с, потім перевести погляд на більш віддалений об'єкт за вікном і дивитися ще протягом 5 с. Повторити 10 разів;

3) вихідне положення - сидячи. Швидко перевести погляд по діагоналі: праворуч вгору - ліворуч униз. Потім дивитися прямо у далеч протягом 6 с. Швидко перевести погляд по діагоналі: ліворуч вгору - праворуч униз. Потім дивитися прямо у далеч протягом 6 с. Повторити 4-5 разів.

Комплекс вправ для поліпшення мозкового кровообігу:

1) вихідне положення - стоячи або сидячи, руки на поясі. На рахунок "раз-два" коловим рухом відвести праву руку назад з поворотом тулуба і голови праворуч, на рахунок "три-чотири" - те саме ліворуч. Повторити 4-6 разів у повільному темпі;

2) вихідне положення - стоячи або сидячи, руки в сторони, долоні вперед, пальці розведені. На рахунок "раз" обхопити себе за плечі руками якомога міцніше і далі, на рахунок "два" повернутися у Вихідне положення Повторити 4-6 разів у швидкому темпі;

3) вихідне положення - сидячи на стільці, руки на поясі. На рахунок "раз" повернути голову праворуч, на рахунок "два" - Вихідне положення Те саме - ліворуч. Повторити 6-8 разів у повільному темпі.

Комплекс вправ для рук:

1) руки, не напружуючи, простягнути вперед на ширину плечей. Повільно згинати й розгинати пальці. Потім з того самого положення повільно згинати і розгинати руки в зап'ястках;

2) руки простягнути вперед на ширину плечей долонями догори. Згинати і розгинати руки в ліктьових суглобах;

3) руки опущені вздовж тулуба долонями всередину, пальці без напруження стиснути в кулак. Обертати кулаки за годинниковою стрілкою і проти. З того самого положення згинати і розгинати руки в зап'ястках

4) підняти руки в сторони до рівня плечей, потім опустити. Підняти руки в сторони до рівня плечей і обертати їх у плечових суглобах спочатку назад, потім - вперед.

Виконання приведених рекомендацій дозволяє скоротити вплив шкідливих виробничих факторів на організм людини.

7 ОХОРОНА НАВКОЛИШНЬОГО СЕРЕДОВИЩА

Вступ

Під навколишнім середовищем розуміють цілісну систему взаємопов'язаних природних і антропогенних об'єктів і явищ, під впливом і при безпосередньому використанні яких відбувається праця, побутова діяльність, відпочинок людей. Поняття «навколишнє середовище» включає соціальні, природні і штучно створені фізичні, хімічні та біологічні фактори, тобто все те, що впливає на життя і діяльність людини. Складовою частиною навколишнього середовища є природне середовище. Перед сучасним суспільством стоїть завдання не тільки зберегти природу, а й запобігти негативним наслідкам господарської діяльності людини в майбутньому.

Економічна діяльність у всіх її проявах здійснює забруднення навколишнього середовища. У процесі цієї діяльності забруднюються і стають дефіцитними ресурси повітря, води, територій, що здавалися нескінченними. Нині рівень забруднення досяг загрозливих розмірів, набувши по суті кризового характеру.

Цей процес посилюється розвитком виробничих сил і збільшенням маси речовин, що залучаються в господарський обіг. Через це в навколишнє середовище надходить все більше й більше різноманітних речовин, які йому чужі, а часом токсичні. Значна частина з них не включається в природний кругообіг, накопичується в біосфері і зумовлює небажані екологічні наслідки. Відомо, що екологія – це наука взаємовідносин між живими організмами і сферою їх перебування, тому наслідки промислової і господарської діяльності людства можуть завдати непоправні збитки біосфері і велику шкоду людині.

Забруднюючі речовини, що потрапляють в природне середовище здатні переміщуватись на досить великі відстані, а закономірність цих процесів вивчена ще недостатньо. Ці речовини мігрують у великих кількостях в контурах окремих складових біосфери. Так в атмосфері вони переносяться повітряними течіями. Ступінь їх розсіювання формується швидкістю і напрямом переміщення

повітряних мас і залежить від метеорологічних умов. потрапивши у воду, забруднюючі речовини окиснюються мікроорганізмами або адсорбуються частинками речовин, які є у воді.

Охорона навколишнього середовища являє собою форму відносин між суспільством і природою. Вона здійснюється різними засобами: економічними, правовими, науково-технічними, санітарно-гігієнічними, біологічними та іншими.

В загальному випадку проблема охорони навколишнього середовища зводиться до вирішення двох завдань:

1) організації раціонального природокористування;
2) забезпечення чистоти природних (екологічних) систем.

При здійсненні різних видів економічної діяльності суб'єкти господарювання використовують різноманітні природні ресурси: землю, воду, корисні копалини тощо. Проте ресурси ці обмежені. Обмеженість природних ресурсів була і залишається головною і дуже жорсткою умовою, що накладається на розвиток економіки і відповідно зростання суспільного добробуту.

Наслідком обмеженості природних ресурсів є конкуренція за їх застосування, тобто суперництво між альтернативними цілями використання ресурсів. Адже майже всі ресурси можуть використовуватися для задоволення найрізноманітніших потреб.

Раціональне природокористування означає розробку та здійснення концепції і конкретних заходів щодо раціонального використання і відтворення природних ресурсів, гармонічну взаємодію суспільства і природи, людини і навколишнього природного середовища.

Завдання організації раціонального природокористування вирішується шляхом:

1) оптимального розподілу ресурсів між різними господарськими цілями;
2) використання технологій, що зберігають ресурси;
3) проведення заходів щодо поповнення природних ресурсів.

Іншим, не менш важливим, завданням охорони навколишнього середовища є забезпечення чистоти природних екологічних систем, тобто водного середовища,

повітряного басейну, ґрунтових покровів тощо, з тим, щоб забезпечити населення екологічно чистими продуктами харчування, водою, повітрям і, в остаточному підсумку, зберегти високий рівень здоров'я населення та його активного довголіття.

7.1 Забруднення навколишнього середовища IT компаніями

IT компанія – це офісна компанія, яка займається наданням послуг з розробки та супроводження програмного забезпечення. Хоча компанія не займається промисловим виробництвом з викидом шкідливих речовин у атмосферу, воду та ґрунт, це не виключає існування джерел забруднення довкілля. До таких джерел, насамперед, належать:

- дизельна електростанція;
- твердопаливні котли;
- побутове сміття;
- електромагнітне випромінювання.

Одночасно в офісі компанії можуть працювати до 300 комп’ютерів, стаціонарних телефонів та велика кількість серверної техніки, яка обслуговує роботу цього обладнання, аварійне чи планове відключення електроенергії призводить до значних перешкод у роботі персоналу. Для того, щоб уникнути таких ситуацій, на території компанії знаходиться дизельна електростанціє. Вона працює як в аварійному, так і в резервному режимі.

Результатом роботи дизельної електростанції є продукти згоряння дизельного палива та шумове забруднення.

Твердопаливні коти використовуються в холодну пору року в системі опалення офісних приміщень. Для опалення використовуються деревні пелети. Хоча таке джерело тепла має менший відсоток шкідливих викидів в атмосферу, ніж, наприклад, вугілля чи мазут, цей вид опалення не можна вважати екологічно чистим. Порівняльні характеристики продуктів згорання палива наведені у таблиці 7.1.

Таблиця 7.1 – Рівні викидів забруднюючих речовин в атмосферу при спалюванні різних видів палива

Вид палива	Викиди забруднюючих речовин в атмосферне повітря без систем очищення, тонн на 1 тис. тонн нат. палива				
	CO2	NO_2	SO_2	Тверді частинки (пил неорг.)	Разом
Природний газ	1,18	3,52	0,00	0,00	4,70
Древні брикети, пелети	4,68	9,31	0,28	4,11	17,69
Деревина дров'яна	4,9	9,4	0,3	4,3	18,9
Тирса деревна	5,0	9,6	0,5	5,0	20,0
Деревні відходи, обрізки	5,2	9,9	0,4	5,2	20,7
Швидкозростаюча деревина	4,8	9,5	0,0	8,4	22,7
Тріска, сучки, кора	5,6	11,4	0,8	13,4	31,3
Мазут	5,20	5,20	35,30	0,30	45,90
Брикет торф'яний	8,04	26,81	3,00	13,02	50,87
Кам'яне вугілля	9,58	63,56	9,20	65,32	147,66

Побутові відходи – це залишки речовин і предметів, які утворюються в результаті побутової та господарської діяльності людини і які не можуть бути використані на місці утворення, а їх накопичення і зберігання порушують санітарний стан навколишнього середовища.

Всі побутові відходи, які утворюються в процесі функціонування компанії, можна розділити на рідкі та тверді. До рідких побутових відходів належать нечистоти з вигребів туалетів, помиї (від миття посуду, підлоги, тощо) і стічні води (побутові, зливові). До твердих побутових відходів можна віднести сміття

(побутові відходи) та кухонні відходи. Зокрема, твердими відходами є папір, картон, скло, пластмаса, продукти харчування.

Генератором електромагнітного забруднення є, в першу чергу, велика кількість комп'ютерної техніки на території офісу. Більшість із них працює 8 годин на добу, а деякі не вимикаються цілодобово. Випромінювачами в даному випадку є і процесор, і монітор. Випромінювання останнього значно вищі, особливо його бічні і задні стінки, адже вони не мають спеціального захисного покриття, як у лицьовій частині монітора. Крім того, в офісі є дві кухні, кожна з яких обладнана 5 мікрохвильовими печами.

Електромагнітне випромінювання має несприятливий вплив на організм людини. Його наслідками можуть бути головний біль, порушення сну, перевтома і, навіть, у деяких випадках стенокардія.

7.2 Розробка заходів щодо зменшення забруднення

Для того, щоб зменшити рівень забруднення навколишнього середовища, варто вжити певних заходів, які мінімізують цей вплив, якщо його неможливо усунути цілком. Серед таких заходів:

1) Очищення димових газів у мокрих золоуловлювачах.

Електрофільтри на електростанціях застосовуються для досягнення найбільш глибокого очищення димових газів в основному на великих енергоблоках потужністю 300 МВт та більше. Мокрі золоуловлювачі, які працюють при маленьких питомих витратах води та невеликих перепадах тиску, встановлюються за котлоагрегатами середньої паропродуктивності. У мокрих золоуловлювачах уловлювання часток золи на плівці води, яка стікає по його стінках, здійснюється за рахунок відцентрової сили, яка діє на частки. Ефективність апарата не перевищує 90%.

2) Використання природного газу для опалення офісних приміщень замість деревних пелетів.

Згідно з показниками, наведеними в таблиці 6.1, рівень шкідливих викидів від згорання природного газу є найнижчим серед перерахованих

видів палива. Проте зміну джерела енергії слід проводити, враховуючи економічну ефективність, оскільки вартість цих видів палива не є однаковою.

3) Раціональне позбавлення від побутових відходів.

Деякі побутові відходи можна безпечно спалювати для отримання енергії. Вторинну сировину (макулатуру, скло, пластмаси) можна відправляти на переробку, а не вивозити на сміттєзвалища. Крім того, зменшити кількість побутових відходів допоможе повторне їх використання (наприклад, скляних чи пластикових пляшок та контейнерів для їжі), скорочення споживання та використання меншої кількості упаковки.

4) Зменшити рівень електромагнітного забруднення та захиститися від його надмірного впливу.

Зрозуміло, що неможливо повністю обійтися без електроприладів та комп’ютерів, проте можна дещо зменшити їх негативний вплив. Для цього слід вимикати з електромережі комп’ютери, телефони та обладнання, з якими ніхто не працює. Також варто час від часу виходити на прогулянки і робити перерви в роботі з комп'ютером.

Що стосується мікрохвильових печей, то їх потужність може змінюватись, тому час від часу треба звертатися до майстра, щоб контролювати рівень випромінювання.

ВИСНОВКИ

Мета, яка ставилася на початку виконання роботи, була досягнута. Отримана математична модель підвищила достовірність оцінювання розміру веб-застосунків, розроблених мовою програмування C# з використанням платформи ASP.NET за рахунок удосконалення нелінійної регресійної моделі.

Для досягнення поставленої мети були виконані наступні завдання:

– було проведено аналіз існуючих методів та моделей, що використовуються при розрахунку розміру програмного забезпечення;

– обгрунтовано необхідність розробки математичної моделі для оцінювання розміру веб-застосунків, розроблених мовою програмування C# з використанням платформи ASP.NET;

– розроблено нелінійну регресійну модель з використанням нормалізуючого перетворення, побудовано довірчий інтервал та інтервал передбачення, розраховані оцінки адекватності отриманої моделі;

– розроблено лінійну регресійну модель у припущенні про нормальність розподілу емпіричних даних, побудовано довірчий інтервал та інтервал передбачення, розраховані оцінки адекватності отриманої моделі;

– виконано порівняння отриманих оцінок для отриманих моделей та зроблено висновки щодо застосування розробленої математичної моделі.

Результати, отримані у даній роботі, в подальшому можуть бути використані при розробці веб-застосунків, призначених для оцінювання програмних проектів розроблених на C# за допомогою платформи ASP.NET.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Приходько Н.В., Приходько С.Б. Нелінійне регресійне рівняння для оцінювання розміру програмного забезпечення інформаційних систем з відкритим кодом на php, м. Сєвєродонецьк, 2018р. С. 135–139.

2. Prykhodko, Sergiy & Prykhodko, Natalia & Farionova, T.A. & Vorona, M.V.. (2020). Three-factor non-linear regression model to estimate the size of open source php-based applications (in Ukrainian). 31(70). 124–131. 10.32838/2663-5941/2020.1-1/23.

3. Prykhodko, Sergiy & Prykhodko, Natalia & Tatiana, Smykodub. (2018). Constructing the non-linear regression equation to estimate the software size of open source PHP-based information systems, Problems of information technologies, 2018, № 1 (23), pp.118–125..

4. Prykhodko, Sergiy & Vorona, Mykhaylo. (2021). Estimating the size of open-source php-based apps by nonlinear regression models with various factors. Collection of Scientific Publications NUS. 92–98. 10.15589/znp2021.1(484).13.

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, ІТКІ, вип. 50, вип. 1, с. 115–121, Квіт 2021.

6. Латанська Л.О., Устенко А.С., Розробка регресійної моделі для оцінювання розміру програмного забезпечення, створеного за допомогою платформи ASP.NET. Матеріали XIII Міжнародної науково-практичної конференції “Free and open source software” (FOSS-2021). м. Харків, 16-18 листопада 2021р. Харків, 2021. С 53–54.

7. Інтернет-ресурс [електроний ресурс]: Выбор метрики размера проекта в модели оценки трудоемкости разработки програм. – Режим доступу: WWW.URL: https://cyberleninka.ru/article/n/vybor-metriki-razmera-proekta-v-modeli-otsenki-trudoemkosti-razrabotki-programm/viewer.

8. Липаев В.В. Экономика программной инженерии заказных программных продуктов, 2014. 148 с.

9. Інтернет-ресурс [електроний ресурс]: 5 видов регрессии и их свойства. – Режим доступу: WWW.URL: https://medium.com/nuances-of-programming/5-видов-регрессии-и-их-свойства-f1bb867aebcb.

10. Prykhodko N., Prykhodko S. (2018). Constructing the non-linear regression models on the basis of multivariate normalizing transformations, Kyiv, 12-14 September 2018, Kyiv, 2021. P. 217-220.

11. Prykhodko S., Prykhodko N., Prykhodko K. (2018). Building the equations, confidence and prediction intervals of non-linear regressions on the basis of multivariate normalizing transformations, Ivano-Frankivsk, 3-5 April 2018, Ivano-Frankivsk, 2018. P. 16.

12. Prykhodko S. Prykhodko N., Smykodub T. (2020). Four-factor non-linear regression model to estimate the size of open source java-based applications. Scientific notes of Taurida National V.I. Vernadsky University. Series: Technical Sciences. 1. 157–162. 10.32838/2663-5941/2020.2-1/25.

13. Макарова Л.М., Приходько Н.В., Кудін О.О. Побудова нелінійної регресійної моделі для оцінювання розміру веб-додатків, реалізованих мовою java, м. Херсон, 5 липня 2019р. Херсон, 2019. С. 145–153.

14. Приходько Н.В., Приходько С.Б. Нелінійна регресійна модель для оцінювання розміру програмного забезпечення промислових інформаційних систем на java, С. 81–88.

15. Prykhodko S. Prykhodko N. & Makarova L. (2018). Building the Non-Linear Regression Equations on the Basis of Multivariate Normalizing Transformations. 15. 10.1109/SAIC.2018.8516742.

16. Prykhodko, Sergiy, Prykhodko, Natalia and Knyrik, Kateryna. Estimating the Efforts of Mobile Application Development in the Planning Phase Using Nonlinear Regression Analysis Applied Computer Systems, 2020, vol.25, no.2, pp.172–179.

17. Інтернет-ресурс [електроний ресурс]: Применение нейронных сетей для решения задач прогнозирования. Режим доступу: WWW.URL: https://zhurnal.ape.relarn.ru/articles/2006/136.pdf

18. Dolado J. A validation of the component-based method for software size estimation. IEEE Trans. SE. 2000. Vol. 26, N 10. P. 1006–1021.

19. Інтернет-ресурс [електроний ресурс]: Расстояние Кука. Режим доступу: WWW.URL:http://www.machinelearning.ru/wiki/index.php?title=Расстояние_Кука

20. Інтернет-ресурс [електроний ресурс]: Критерій Фішера. Режим доступу:WWW.URL:https://wiki.tntu.edu.ua/Критерій_Фішера

21. Інтернет-ресурс [електроний ресурс]: Коэфициент детерминации. Режим доступу: WWW.URL: http://statsoft.ru/home/textbook/glossary/GlossaryTwo/C/CoefficientofDetermination.htm

22. Інтернет-ресурс [електроний ресурс]: Relative error: Definition, Formula, Exapmles. Режим доступу: WWW.URL: https://www.statisticshowto.com/relative-error/

23. Інтернет-ресурс [електроний ресурс]: Що таке довірчий інтервал?. Режим доступу: WWW.URL: https://ua.nesrakonk.ru/confidenceinterval/

24. Інтернет-ресурс [електроний ресурс]: Prediction intervals. Режим доступу: WWW.URL:https://otexts.com/fpp2/prediction-intervals.html

25. Пфаффенбергер, Б. HTML, XHTML и CSS. Библия пользователя [Текст] / [Б. Пфаффенбергер, С. Шафер, Ч. Уайт, Б. Кароу]; [пер. с англ.]. – [3-е изд.] – М.: ООО "И.Д. Вильямс", 2009. – 752 с.

ДОДАТОК А

Текст програми для оцінювання для оцінювання кількості рядків коду веб-застосунків, реалізованих мовою C# з використанням платформи ASP.NET.

Текст файлу index.php

```
<?php
    session_start();

    require_once "include/check_data.php";

    $menu_item = [
        [
            'title' => 'Завантаження та нормалізація даних',
            'url' => 'preprocessing.php',
        ],
        [
            'title' => 'Побудова рівняння лінійної регресії',
            'url' => 'linear.php',
        ],
        [
            'title' => 'Побудова нелінійного рівняння регресії',
            'url' => 'nonlinear.php',
        ],
        [
            'title' => 'Порівняння результатів побудови',
            'url' => 'd.php',
        ],
    ];

    function showMenu(&$active, &$enabled, $menu_item) {
        $active = (int)$active;
        $enabled = (int)$enabled;
        if ($active >= count($menu_item) || $active < 0)
            $active = 0;
//        if (!$active)
//            $enabled = 0;
        if ($enabled >= count($menu_item))
            $enabled = 0;
        echo '<div class="menu">';
        foreach ($menu_item as $key => $item) {
            echo "<a class=\"menu-item" .
                 ($key == $active ? ' active' : '') .
                 ($key > $enabled ? ' disable' : '') .
                 ($key && $key != $active && $key <= $enabled ? "\"
href=\"/?active=$key&enabled=-1" : '') .
                 "\"" .
```

```
                (!$key && $enabled > 0 ? "
onclick=\"$('#clearModal').modal();\"" : '') .
                ">{$item['title']}</a>";

        }
        echo '</div>';
    }

    $active = $_POST['active'] ?? ($_GET['active'] ?? 0);
    $enabled = $_POST['enabled'] ?? ($_GET['enabled'] ?? -1);
    $mode = $_POST['mode'] ?? 'none';
    switch ($mode) {
        case 'file':
            $lines = checkFile($_FILES['custom_file']);
            if ($lines === false)
                $file_message = "Не вдалося завантажити файл";
            break;
    }

?>
<html>
<head>
    <link rel="stylesheet" href="/vendor/css/bootstrap.min.css">
    <script src="/vendor/js/jquery-3.6.0.min.js"></script>
    <script src="/vendor/js/bootstrap.min.js"></script>
<!--    <script src="/vendor/js/google_chart.js"></script>-->
    <script type = "text/javascript" src =
"https://www.gstatic.com/charts/loader.js">
    </script>
    <script type = "text/javascript">
        google.charts.load('current', {packages: ['corechart']});
    </script>
    <style>
        body {
            font-size: 14pt !important;
        }
        .menu-item {
            display: block;
            cursor: pointer;
            color: #fff;
            background-color: #17b2c9;
            padding: 3px 10px;
            margin: 2px 0px;
        }
        .menu-item:hover {
            text-decoration: none;
            color: #fff;
        }
        .menu-item.active {
            border-left: white 4px solid;
            cursor: default;
```

```
            font-weight: bold;
            color: #fff;
        }
        .menu-item.disable {
            cursor: not-allowed;
            background-color: #1294a8;
            color: #eee;
        }
        .menu-item:not(.disable,.active):hover {
            background-color: #18c8e0;
        }
    </style>
</head>
<body>
    <div class="container-fluid d-flex flex-column h-100">
        <div class="row">
            <div class="col-12 bg-info p-4 text-white text-center text-
uppercase">
                Математична модель для оцінювання розміру веб-застосунків,
створених за допомогою платформи ASP.Net, та розробка програмного
забезпечення для її реалізації
            </div>
        </div>
        <div class="row flex-fill">
            <div class="col-2 bg-info">
                <?= showMenu($active,$enabled, $menu_item); ?>
            </div>
            <div class="col-10">
                <?php require_once
"include/{$menu_item[$active]['url']}"; ?>
            </div>
        </div>
    </div>
    <div class="modal" id="clearModal" tabindex="-1" role="dialog">
        <div class="modal-dialog" role="document">
            <div class="modal-content">
                <div class="modal-header">
                    Попередження
                    <button type="button" class="close" data-
dismiss="modal" aria-label="Close">
                        <span aria-hidden="true">&times;</span>
                    </button>
                </div>
                <div class="modal-body">При виборі цього пункту всі дані та
розрахунки будуть знищені. Продовжити?</div>
                <div class="modal-footer">
                    <button class="btn btn-outline-danger" type="button"
data-dismiss="modal">Ні</button>
                    <button class="btn btn-success"
id="clear_conf">Так</button>
                </div>
```

```
            </div>
        </div>
    </div>
    <script>
        $('#clear_conf').click(() => {
            location = '/';
        });
    </script>
</body>
</html>
```

Текст файлу normalisation.php

```
<?php

session_start();

function getF($f)
{
    $F = [
        [40, 3.23],
        [50, 3.18],
        [60, 3.15]
    ];
    foreach ($F as $key => $F_data) {
        if ($f > $F_data[0])
            continue;
        if (!$key)
            return false;
        return $F[$key - 1][1] + ($F_data[1] - $F[$key - 1][1])*($f -
$F[$key - 1][0])/($F_data[0] - $F[$key - 1][0]);
    }
    return false;
}

function iteration($x, $y)
{
    $num = count($x);
    $f = getF($num - 2);
    if ($f === false)
        return false;
    $x = array_map(function ($item) { return log($item); }, $x);
    $y = array_map(function ($item) { return log($item); }, $y);
    $x_mid = array_sum($x) / $num;
    $y_mid = array_sum($y) / $num;
    $X2 = array_map(function ($item) use ($x_mid) { return pow($item -
$x_mid, 2); }, $x);
    $Y2 = array_map(function ($item) use ($y_mid) { return pow($item -
$y_mid, 2); }, $y);
    $XY = array_map(function ($item1, $item2) use ($x_mid, $y_mid) { return
($item1 - $x_mid)*($item2 - $y_mid); }, $x, $y);
```

```
    $cov = [
        [
            array_sum($X2) / $num,
            array_sum($XY) / $num,
        ],
        [
            array_sum($XY) / $num,
            array_sum($Y2) / $num,
        ],
    ];
    $det = $cov[0][0]*$cov[1][1] - $cov[0][1]*$cov[1][0];
    $COV = [
        [
            $cov[1][1] / $det,
            - $cov[1][0] / $det,
        ],
        [
            - $cov[0][1] / $det,
            $cov[0][0] / $det,
        ],
    ];
    $d2 = array_map(function ($item1, $item2) use ($x_mid, $y_mid, $COV) {
        return (($item1 - $x_mid)*$COV[0][0] + ($item2 -
$y_mid)*$COV[1][0])*($item1 - $x_mid) +
            (($item1 - $x_mid)*$COV[0][1] + ($item2 -
$y_mid)*$COV[1][1])*($item2 - $y_mid);
    }, $x, $y);
    $d2 = array_map(function ($item) use ($num) {
        return ($num - 2)*$num*$item/(($num*$num - 1)*2);
    }, $d2);
    $d2_v = array_filter($d2, function ($item) use ($f) {
        return $item > $f;
    });
    return $d2_v;
}

$x = array_map(function ($item) {
    return $item[0];
}, $_POST['lines']);
$y = array_map(function ($item) {
    return $item[1];
}, $_POST['lines']);

if (!empty($_SESSION['x']))
    unset($_SESSION['x']);
if (!empty($_SESSION['y']))
    unset($_SESSION['y']);
$res = $del_lines = [];
while (true) {
    $v = iteration($x, $y);
    if ($v === false)
```

```
        die(json_encode(['error' => "Критерій Фішера неможливо
обчислити"]));
    if (empty($v))
        break;
    $msg = "Відкидаємо:<br/>";
    foreach ($v as $i => $d2) {
        $msg .= ($i + 1) . ": $d2<br/>";
        $del_lines[] = $i;
    }
    $res[] = $msg;
    $X = $x;
    $Y = $y;
    $x = $y = [];
    foreach ($X as $key => $item)
        if (!in_array($key, array_keys($v))) {
            $x[] = $item;
            $y[] = $Y[$key];
        }
}
$_SESSION['x'] = $x;
$_SESSION['y'] = $y;
die(json_encode(['success' => [$del_lines, implode('<br/>', $res)]]));
```

Текст файлу preprocessing.php

```
<?php
/**
 * @var $active int
 * @var $enabled int
 * @var $file_message string
 * @var $lines array
 */
?>
<div class="card my-3">
    <div class="card-header">
        Завантаження та нормалізація даних по проектах
    </div>
    <div class="card-body">
        <div class="row justify-content-center">
            <div class="col-10">
                <form class="mb-0" id="frm_data" enctype="multipart/form-
data" method="post">
                    <input type="hidden" name="MAX_FILE_SIZE"
value="1000000"/>
                    <input type="hidden" name="active" value="<?=
$active; ?>"/>
                    <input type="hidden" name="enabled" value="<?=
$enabled; ?>"/>
                    <input type="hidden" name="mode" value="file"/>
                    <div class="custom-file">
                        <input type="file" class="custom-file-input"
```

```
id="custom_file" name="custom_file" multiple>
                        <label class="custom-file-label" for="custom_file"
id="custom_filename">Виберіть файл (у форматі csv) для завантаження
даних</label>
                    </div>
                </form>
            </div>
        </div>
        <div class="row justify-content-center">
            <div class="col-10">
                <div class="alert alert-danger mt-2 mb-0 p-2<?=
empty($file_message) ? ' d-none' : ''; ?>" id="err_msg"><?= $file_message ??
''; ?></div>
            </div>
        </div>
        <div class="row justify-content-center mt-4<?= empty($lines) ? ' d-
none' : ''; ?>" id="get_data">
            <div class="col-4">
                <table class="table table-hover col mb-0">
                    <thead>
                        <tr>
                            <th>Номер</th>
                            <th>Кількість класів</th>
                            <th>Кількість рядків коду</th>
                        </tr>
                    </thead>
                    <tbody>
                        <?php
                            if (!empty($lines))
                                foreach ($lines as $key => $data)
                                    echo "<tr id='row_$key'><td>" . ($key +
1) . "</td><td>{$data[0]}</td><td>{$data[1]}</td></tr>";
                        ?>
                    </tbody>
                </table>
            </div>
            <div class="col-1"></div>
            <div class="col-5">
                <div class="row justify-content-end">
                    <div class="col-auto">
                        <button class="btn btn-primary"
id="btn_norm">Нормалізувати</button>
                    </div>
                </div>
                <div class="row d-none" id="res_norm"></div>
            </div>
        </div>
    </div>
    <div class="card-footer d-none" id="res_footer">
        <div class="row">
            <div class="col">
```

```
                <button class="btn btn-outline-danger"
id="btn_cncl">Скасувати</button>
            </div>
            <div class="col-auto">
                <form class="m-0" method="post">
                    <input type="hidden" name="active" value="1"/>
                    <input type="hidden" name="enabled" value="1"/>
                    <button class="btn btn-success"
type="submit">Застосувати</button>
                </form>
            </div>
        </div>
    </div>
</div>
<script>
    var lines = <?= json_encode($lines ?? []); ?>;

    $('#custom_file').change(function (e) {
        $('#frm_data').submit();
    });

    $('#btn_norm').click(function () {
        $.ajax({
            url: "/include/normalisation.php",
            data: { lines: lines },
            method: "POST",
            dataType: "json",
            success: function (json) {
                if (json.success !== undefined) {
                    json.success[0].forEach(function (item, index, arr) {
                        $('#row_' + item).prop('classList').add('bg-
danger');
                    });
                    $('#res_norm').html(json.success[1]);
                    $('#res_norm').prop('classList').remove('d-none');
                    $('#res_footer').prop('classList').remove('d-none');
                } else
                    showError(json.error);
            },
            error: function (json) {
                showError("Підключення неможливе");
            }
        });
    });

    $('#custom_file').click(() => {
        $('#custom_filename').html("Виберіть файл (у форматі csv) для
завантаження даних");
        $('#get_data').prop('classList').add('d-none');
        $('#err_msg').prop('classList').add('d-none');
        $('#res_norm').prop('classList').add('d-none');
```

```
        $('#res_footer').prop('classList').add('d-none');
    });

    $('#btn_cncl').click(() => {
        $('#custom_filename').html("Виберіть файл (у форматі csv) для
завантаження даних");
        $('#get_data').prop('classList').add('d-none');
        $('#err_msg').prop('classList').add('d-none');
        $('#res_norm').prop('classList').add('d-none');
        $('#res_footer').prop('classList').add('d-none');
    });

    function showError(msg) {
        $('#err_msg').html(msg);
        $('#err_msg').prop('classList').remove('d-none');
        setTimeout(() => {
            if (!$('#err_msg').prop('classList').contains('d-none'))
                $('#err_msg').prop('classList').add('d-none');
        }, 3000);
    }
</script>
```

Текст файлу solve_model.php

```
<?php

session_start();

function getParams($x, $y)
{
    $sX = array_sum($x);
    $sY = array_sum($y);
    $sX2 = array_sum(array_map(function ($item) { return $item*$item; },
$x));
    $sXY = array_sum(array_map(function ($item1, $item2) { return
$item1*$item2; }, $x, $y));
    $num = count($x);
    $b = ($num*$sXY - $sY*$sX)/($num*$sX2 - $sX*$sX);
    $a = ($sY - $b*$sX)/$num;
    return [$a, $b];
}

function getKoefs($y, $Y)
{
    $num = count($y);
    $Y_mid = array_sum($Y)/$num;
    $Y_1 = array_map(function ($item1, $item2) { return pow($item1 - $item2,
2); }, $y, $Y);
    $Y_2 = array_map(function ($item) use($Y_mid) { return pow($item -
$Y_mid, 2); }, $y);
    $R2 = 1 - array_sum($Y_1)/array_sum($Y_2);
```

```
    $MRE = array_map(function ($item1, $item2) { return abs(($item1 -
$item2)/$item1); }, $y, $Y);
    $MMRE = array_sum($MRE)/$num;
    $MRE_025 = array_filter($MRE, function ($item) {
        return $item <= 0.25;
    });
    $PRED = count($MRE_025)/$num;
    return [$R2, $MMRE, $PRED];
}

function getIntervals($x, $y, $y_new) {
    $st = 2.39939;
    $Y_1 = array_map(function ($item1, $item2) { return pow($item1 - $item2,
2); }, $y, $y_new);
    $num = count($x);
    $Sy = sqrt(array_sum($Y_1)/($num - 2));

    $x_mid = array_sum($x)/$num;
    $Sx = array_sum(array_map(function ($item) use($x_mid) { return
pow($item - $x_mid, 2); }, $x));
    $CIp = array_map(function ($item1, $item2) use($st, $Sy, $Sx, $num,
$x_mid) {
        return $item1 + $st * $Sy * sqrt(1./$num + pow($item2 - $x_mid, 2)
/ $Sx);
    }, $y_new, $x);
    $CIm = array_map(function ($item1, $item2) use($st, $Sy, $Sx, $num,
$x_mid) {
        return $item1 - $st * $Sy * sqrt(1./$num + pow($item2 - $x_mid, 2)
/ $Sx);
    }, $y_new, $x);
    $PIp = array_map(function ($item1, $item2) use($st, $Sy, $Sx, $num,
$x_mid) {
        return $item1 + $st * $Sy * sqrt(1 + 1./$num + pow($item2 - $x_mid,
2) / $Sx);
    }, $y_new, $x);
    $PIm = array_map(function ($item1, $item2) use($st, $Sy, $Sx, $num,
$x_mid) {
        return $item1 - $st * $Sy * sqrt(1 + 1./$num + pow($item2 - $x_mid,
2) / $Sx);
    }, $y_new, $x);
    return [$CIp, $CIm, $PIp, $PIm];
}

$x = $_SESSION['x'];
$y = $_SESSION['y'];
if ($_POST['mode'] != 'linear') {
    $x = array_map(function ($item) { return log($item); }, $x);
    $y = array_map(function ($item) { return log($item); }, $y);
}
[$a, $b] = getParams($x, $y);
$y_new = array_map(function ($item) use($a, $b) { return $a + $b*$item; },
```

```
$x);
if ($_POST['mode'] != 'linear')
    $y_ret = array_map(function ($item) { return exp($item); }, $y_new);
[$R2, $MMRE, $PRED] = getKoefs($_SESSION['y'], $_POST['mode'] != 'linear' ?
$y_ret : $y_new);
[$CIp, $CIm, $PIp, $PIm] = getIntervals($_SESSION['x'], $y, $y_new);
if ($_POST['mode'] != 'linear') {
    $CIp = array_map(function ($item) { return exp($item); }, $CIp);
    $CIm = array_map(function ($item) { return exp($item); }, $CIm);
    $PIp = array_map(function ($item) { return exp($item); }, $PIp);
    $PIm = array_map(function ($item) { return exp($item); }, $PIm);

}

$_SESSION[$_POST['mode']] = [$a, $b, $_POST['mode'] != 'linear' ? $y_ret :
$y_new, $R2, $MMRE, $PRED, $CIp, $CIm, $PIp, $PIm];
die(json_encode(['success' => $_SESSION[$_POST['mode']]]));
```

Текст файлу check_data.php

```
<?php
    function checkFile($file_info) {
        if (empty($file_info) || $file_info['error'])
            return false;
        $filesize = filesize($file_info['tmp_name']);
        if (!$filesize)
            return false;
        $filename = "tmp/{$file_info['name']}";
        if (!move_uploaded_file($file_info['tmp_name'], $filename))
            return false;
        $file_handle = fopen($filename, "r");
        if (!$file_handle)
            return false;
        $lines = [];
        while ($line = fgetcsv($file_handle, 0, ';')) {
            if (count($line) != 2)
                return false;
            $lines[] = $line;
        }
        fclose($file_handle);
        return $lines;
    }
?>
```

Текст файлу linear.php

```
<div class="card my-3">
    <div class="card-header">
        Побудова лінійної регресійної моделі
    </div>
```

```
    <div class="card-body">
        <div class="row justify-content-center">
            <div class="col-11">
                <div class="alert alert-danger mb-2 p-2 d-none"
id="err_msg"></div>
            </div>
        </div>
        <div class="row justify-content-center">
            <div class="col-4">
                <table class="table table-hover col mb-0">
                    <thead>
                    <tr>
                        <th class="align-text-top">Номер</th>
                        <th class="align-text-top">Кількість класів, X</th>
                        <th class="align-text-top">Кількість рядків коду,
Y</th>
                        <th class="align-text-top">Значення лінійної моделі,
Y`</th>
                    </tr>
                    </thead>
                    <tbody>
                    <?php
                        foreach ($_SESSION['x'] as $key => $x)
                            echo "<tr><td>" . ($key + 1) .
"</td><td>$x</td><td>{$_SESSION['y'][$key]}</td><td><span class='text-
success' id='new_$key'>-</span></td></tr>";
                    ?>
                    </tbody>
                </table>
            </div>
            <div class="col-1"></div>
            <div class="col-6">
                <div class="row justify-content-end">
                    <div class="col-auto">
                        <button class="btn btn-primary"
id="btn_linear">Побудувати</button>
                    </div>
                </div>
                <div class="row d-none mt-4" id="res_linear">
                    <div class="col-12">
                        <p>Коефіцієнти лінійної моделі: a = <span
class="text-success" id="coef_a"></span>, b = <span class="text-success"
id="coef_b"></span>.</p>
                        <p>Коефіцієнт детермінації: R<sup>2</sup> = <span
class="text-success" id="coef_r"></span></p>
                        <p>Величина середньої похибки: MMRE = <span
class="text-success" id="coef_mmre"></span></p>
                        <p>Рівень прогнозування: PRED(0,25) = <span
class="text-success" id="coef_pred"></span></p>
                    </div>
                    <div class="col-12" id="grphc"></div>
```

```
                </div>
            </div>
        </div>
    </div>
    <div class="card-footer" id="res_footer">
        <div class="row">
            <div class="col">
                <button class="btn btn-outline-danger"
id="btn_cncl">Скасувати</button>
            </div>
            <div class="col-auto">
                <form class="m-0" method="post">
                    <input type="hidden" name="active" value="2"/>
                    <input type="hidden" name="enabled" value="2"/>
                    <button class="btn btn-success" type="submit"
id="btn_conf" disabled>Застосувати</button>
                </form>
            </div>
        </div>
    </div>
</div>
<script>
    var lines = <?= json_encode(array_map(function ($item1, $item2) {return
[(int)$item1, (int)$item2];}, $_SESSION['x'], $_SESSION['y'])); ?>;

    $('#btn_linear').click(function () {
        $.ajax({
            url: "/include/solve_model.php",
            data: { mode: 'linear' },
            method: "POST",
            dataType: "json",
            success: function (json) {
                if (json.success !== undefined) {
                    json.success[2].forEach(function (item, index, arr) {
                        lines[index].push(item);
                        lines[index].push(json.success[6][index]);
                        lines[index].push(json.success[7][index]);
                        lines[index].push(json.success[8][index]);
                        lines[index].push(json.success[9][index]);
                        $('#new_' + index).html(item);
                    });
                    $('#coef_a').html(json.success[0]);
                    $('#coef_b').html(json.success[1]);
                    $('#coef_r').html(json.success[3]);
                    $('#coef_mmre').html(json.success[4]);
                    $('#coef_pred').html(json.success[5]);
                    $('#res_linear').prop('classList').remove('d-none');
                    google.charts.setOnLoadCallback(drawBarChart);
                    if (json.success[3] >= 0.5)
                        $('#btn_conf').attr('disabled', false);
                } else
```

```
                    showError(json.error);
                },
                error: function (json) {
                    showError("Підключення неможливе");
                }
            });
        });

        $('#btn_cncl').click(() => {
            $('#clearModal').modal();
        });

        function showError(msg) {
            $('#err_msg').html(msg);
            $('#err_msg').prop('classList').remove('d-none');
            setTimeout(() => {
                if (!$('#err_msg').prop('classList').contains('d-none'))
                    $('#err_msg').prop('classList').add('d-none');
            }, 3000);
        }

        function drawBarChart() {
            lines.sort(function (a, b) {
                if (a[0] > b[0])
                    return 1;
                if (a[0] < b[0])
                    return -1;
                return 0;
            });
            var data = new google.visualization.DataTable();
            data.addColumn('number', 'x');
            data.addColumn('number', 'y');
            data.addColumn('number', 'y_new');
            data.addColumn('number', 'CIp');
            data.addColumn('number', 'CIm');
            data.addColumn('number', 'PIp');
            data.addColumn('number', 'PIm');
            data.addRows(lines);
            var options = {
                'legend': 'none',
                'width':'100%',
                'height':700,
                'series': {
                    1: {type: 'line', lineWidth: 3},
                    2: {type: 'line', lineWidth: 2, pointSize: 0, color:
'orange'},
                    3: {type: 'line', lineWidth: 2, pointSize: 0, color:
'orange'},
                    4: {type: 'line', lineWidth: 1, pointSize: 0, color:
'green'},
                    5: {type: 'line', lineWidth: 1, pointSize: 0, color:
```

```
'green'},
            }
        };
        var chart = new
google.visualization.ScatterChart(document.getElementById('grphc'));
        chart.draw(data, options);
    }
</script>
```

Текст файлу nonlinear.php

```
<div class="card my-3">
    <div class="card-header">
        Побудова нелінійної регресійної моделі
    </div>
    <div class="card-body">
        <div class="row justify-content-center">
            <div class="col-11">
                <div class="alert alert-danger mb-2 p-2 d-none"
id="err_msg"></div>
            </div>
        </div>
        <div class="row justify-content-center">
            <div class="col-4">
                <table class="table table-hover col mb-0">
                    <thead>
                    <tr>
                        <th class="align-text-top">Номер</th>
                        <th class="align-text-top">Кількість класів, X</th>
                        <th class="align-text-top">Кількість рядків коду,
Y</th>
                        <th class="align-text-top">Значення нелінійної
моделі, Y`</th>
                    </tr>
                    </thead>
                    <tbody>
                    <?php
                        foreach ($_SESSION['x'] as $key => $x)
                            echo "<tr><td>" . ($key + 1) .
"</td><td>$x</td><td>{$_SESSION['y'][$key]}</td><td><span class='text-
success' id='new_$key'>-</span></td></tr>";
                    ?>
                    </tbody>
                </table>
            </div>
            <div class="col-1"></div>
            <div class="col-6">
                <div class="row justify-content-end">
                    <div class="col-auto">
                        <button class="btn btn-primary"
id="btn_linear">Побудувати</button>
```

```
                    </div>
                </div>
                <div class="row d-none mt-4" id="res_linear">
                    <div class="col-12">
                        <p>Коефіцієнти нелінійної моделі: a = <span
class="text-success" id="coef_a"></span>, b = <span class="text-success"
id="coef_b"></span>.</p>
                        <p>Коефіцієнт детермінації: R<sup>2</sup> = <span
class="text-success" id="coef_r"></span></p>
                        <p>Величина середньої похибки: MMRE = <span
class="text-success" id="coef_mmre"></span></p>
                        <p>Рівень прогнозування: PRED(0,25) = <span
class="text-success" id="coef_pred"></span></p>
                    </div>
                    <div class="col-12" id="grphc"></div>
                </div>
            </div>
        </div>
    </div>
    <div class="card-footer" id="res_footer">
        <div class="row">
            <div class="col">
                <button class="btn btn-outline-danger"
id="btn_cncl">Скасувати</button>
            </div>
            <div class="col-auto">
                <form class="m-0" method="post">
                    <input type="hidden" name="active" value="3"/>
                    <input type="hidden" name="enabled" value="3"/>
                    <button class="btn btn-success" type="submit"
id="btn_conf" disabled>Застосувати</button>
                </form>
            </div>
        </div>
    </div>
</div>
<script>
    var lines = <?= json_encode(array_map(function ($item1, $item2) {return
[(int)$item1, (int)$item2];}, $_SESSION['x'], $_SESSION['y'])); ?>;

    $('#btn_linear').click(function () {
        $.ajax({
            url: "/include/solve_model.php",
            data: { mode: 'nonlinear' },
            method: "POST",
            dataType: "json",
            success: function (json) {
                if (json.success !== undefined) {
                    json.success[2].forEach(function (item, index, arr) {
                        lines[index].push(item);
                        lines[index].push(json.success[6][index]);
```

```
                        lines[index].push(json.success[7][index]);
                        lines[index].push(json.success[8][index]);
                        lines[index].push(json.success[9][index]);
                        $('#new_' + index).html(item);
                    });
                    $('#coef_a').html(json.success[0]);
                    $('#coef_b').html(json.success[1]);
                    $('#coef_r').html(json.success[3]);
                    $('#coef_mmre').html(json.success[4]);
                    $('#coef_pred').html(json.success[5]);
                    $('#res_linear').prop('classList').remove('d-none');
                    google.charts.setOnLoadCallback(drawBarChart);
                    if (json.success[3] >= 0.5)
                        $('#btn_conf').attr('disabled', false);
                } else
                    showError(json.error);
            },
            error: function (json) {
                showError("Підключення неможливе");
            }
        });
    });

    $('#btn_cncl').click(() => {
        $('#clearModal').modal();
    });

    function showError(msg) {
        $('#err_msg').html(msg);
        $('#err_msg').prop('classList').remove('d-none');
        setTimeout(() => {
            if (!$('#err_msg').prop('classList').contains('d-none'))
                $('#err_msg').prop('classList').add('d-none');
        }, 3000);
    }

    function drawBarChart() {
        lines.sort(function (a, b) {
            if (a[0] > b[0])
                return 1;
            if (a[0] < b[0])
                return -1;
            return 0;
        });
        var data = new google.visualization.DataTable();
        data.addColumn('number', 'x');
        data.addColumn('number', 'y');
        data.addColumn('number', 'y_new');
        data.addColumn('number', 'CIp');
        data.addColumn('number', 'CIm');
        data.addColumn('number', 'PIp');
```

```
        data.addColumn('number', 'PIm');
        data.addRows(lines);
        var options = {
            'legend': 'none',
            'width':'100%',
            'height':400,
            'series': {
                1: {type: 'line', lineWidth: 3},
                2: {type: 'line', lineWidth: 2, pointSize: 0, color:
'orange'},
                3: {type: 'line', lineWidth: 2, pointSize: 0, color:
'orange'},
                4: {type: 'line', lineWidth: 1, pointSize: 0, color:
'green'},
                5: {type: 'line', lineWidth: 1, pointSize: 0, color:
'green'},
            }
        };
        var chart = new
google.visualization.ScatterChart(document.getElementById('grphc'));
        chart.draw(data, options);
    }
</script>
```

ДОДАТОК Б

Опис програми для оцінювання кількості рядків коду веб-застосунків, реалізованих мовою C# з використанняv платформи ASP.NET.

1. Загальні відомості

1.1 Позначення і найменування програми

Найменування: Програмне забезпечення для оцінюваннякількості рядків коду веб-застосунків, реалізованих мовою C# з використанням платформи ASP.NET.

Позначення: «Model».

1.2 Програмне забезпечення, необхідне для функціонування програми

Програмне забезпечення «Model» є крос-платформним і основною необхідною вимогою для функціонування виробу є наявність середовища що підтримує HTML код.

1.3 Мови програмування

ПЗ «Model» реалізовано на мові високого рівня PHP. Модулі, реалізовані на інших мовах програмування, не використовуються.

2. Функціональне призначення

2.1 Класи вирішуваних завдань і призначення програми

ПЗ «Model» повинно забезпечувати можливість виконання нижче перерахованих функцій:

– внесення даних про проекти;

– виконання нормалізації внесених даних за допомогою натурального логарифму;

– побудова лінійної регресії для нормалізованих даних;

– побудова нелінійного регресійного рівняння за допомогою зворотного перетворення;

– перевірка адекватності моделі.

2.2 Функціональні обмеження

Файли повинні бути формату CSV.

3. Використовувані технічні засоби

ПЗ «Model» призначене для використання на персональних комп'ютерах, що працюють під управлінням наступних операційних систем: ОС MS Windows XP/7/8/8.1/10

Для ПЗ «Model» пред'являються такі апаратні вимоги до ПЕОМ:

– оперативна пам'ять 512 МБ;

– дисковий простір - не менше 512 МБ вільного місця на диску;

– роздільна здатність екрану - 1280 x 1024 пікселів;

– клавіатура 101/102-х клавішна рус / лат;

Швидкість роботи ПЗ «Model» на конкретному комп'ютері залежить також від характеристик окремих його комплектуючих (процесора, оперативної пам'яті і ін.).

4. Виклик і завантаження

Запуск ПЗ «Model»здійснюється за запуску програмного коду через середовище PHPStorm.

5. Вхідні і вихідні дані

Вхідні дані програми(строки коду та трудомісткість) повинні знаходитись в окремому текстовому файлі у два стовбці. Вихідні дані програми мають відображатися на екрані користувача.

ДОДАТОК В

Програма і методика випробувань програмного забезпечення для оцінювання кількості рядків коду веб-застосунків, реалізованих мовою C# з використанням платформи ASP.NET.

1. Об’єкт випробувань

Об’єктом випробувань є програмне забезпечення для оцінювання кількості рядків коду веб-застосунків, реалізованих мовою C# з використанням платформи ASP.NET.

2. Мета випробувань

Перевірка функціональних можливостей програмного забезпечення на відповідність вимогам технічного завдання.

3. Вимоги до програми

В ході тестування програмного продукту необхідно перевірити, що розроблене ПЗ реалізує наступні функції:

– внесення даних про проекти (кількість строк коду);

– виконання нормалізації внесених даних за допомогою натурального логарифму;

– побудова лінійної регресії для нормалізованих даних;

– побудова нелінійного регресійного рівняння за допомогою зворотного перетворення;

– підрахунок оцінок якості.

4. Вимоги до програмної документації

Склад програмної документації, що надається на випробування:

– текст програми;
– опис програми;
– програма і методика випробувань;
– інструкція користувача.

5. Засоби і порядок випробувань

Випробування проводилось на платформі Windows 10 64bit. Порядок випробувань:

– перевірити можливість додавання даних до програми;
– перевірити можливість розрахунку значень лінійної моделі;
– перевірити можливість розрахунку значень нелінійної моделі.

6. Методика випробувань

– Додавання даних до програми:
  – натиснути на кнопку «Вибрати файл»;
  – вибрати шлях до файлу;
  – файл відкривається і виводить дані, що містяться всередині.

Очікуваний результат: після натискання кнопки «Вибрати файл» файл додається.

– Розрахувати значення лінійної моделі:
  – прочитати дані з файлу;
  – натиснути на кнопку «Побудувати»;
  – нові значення та їх графіки з’являються на екрані.

Очікуваний результат: після натискання кнопки «Побудувати» на головній формі додаються нові значення та їх графіки.

– Розрахувати значення нелінійної моделі:
  – прочитати дані з файлу;

– натиснути на кнопку «Побудувати»;
– нові значення та їх графіки з’являються на екрані.

Очікуваний результат: після натискання кнопки «Побудувати» на головній формі додаються нові значення та їх графіки.

ДОДАТОК Г

Інструкція користувача програмного забезпечення для оцінювання кількості рядків коду веб-застосунків, реалізованих мовою C# з використаннv платформи ASP.NET.

1. Для того, щоб запустити програму, потрібно запустити файл «index.php». При запуску програми з’являється початкова форма (рис. Г.1).

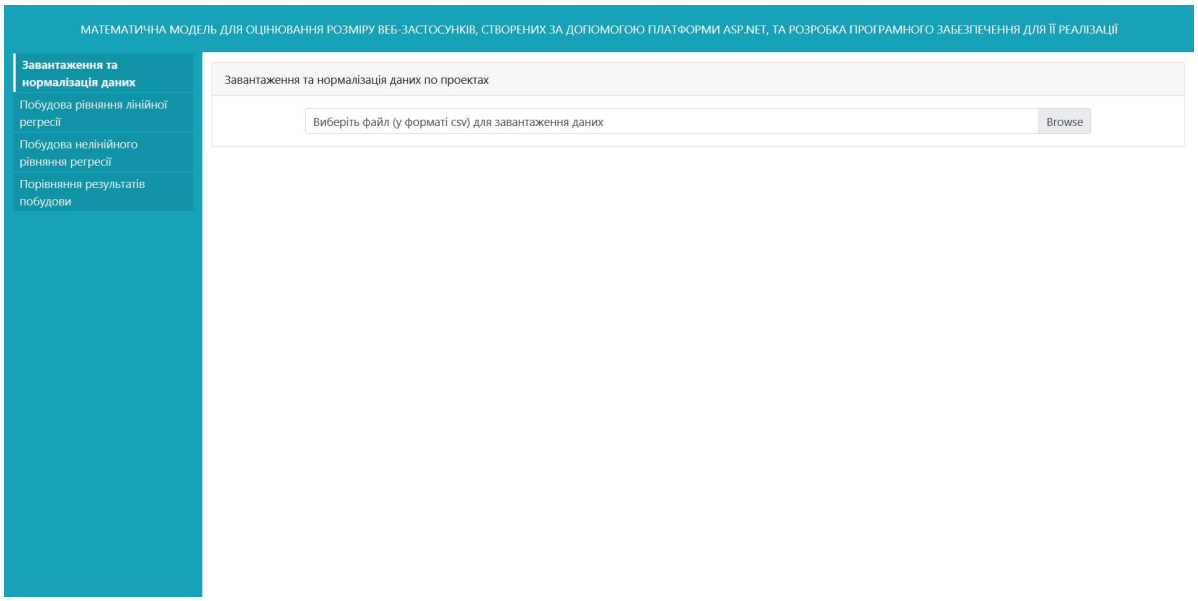


Рисунок Г.1 – Головна форма

2. Для внесення інформації про даних потрібно натиснути на кнопку«Вибрати файл» (рис. Г.2).

Рисунок Г.2 – Додавання даних

3. Потім необхідно у вікні, що відкрилося, знайти файл з даними та запустити його.

4. Після цього на сторінці з’явиться таблиця з отриманими даними (рис. Г.3)

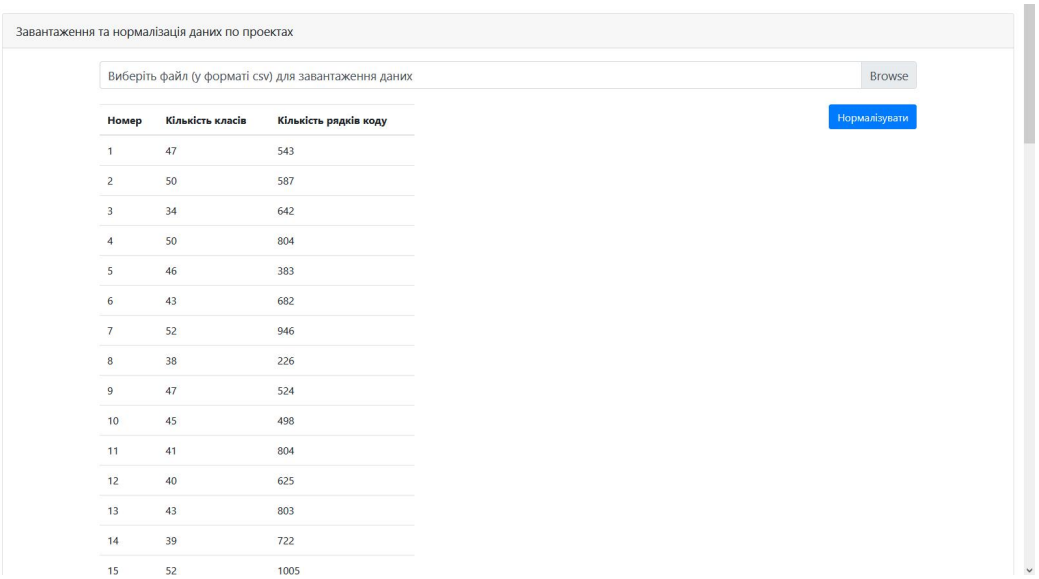
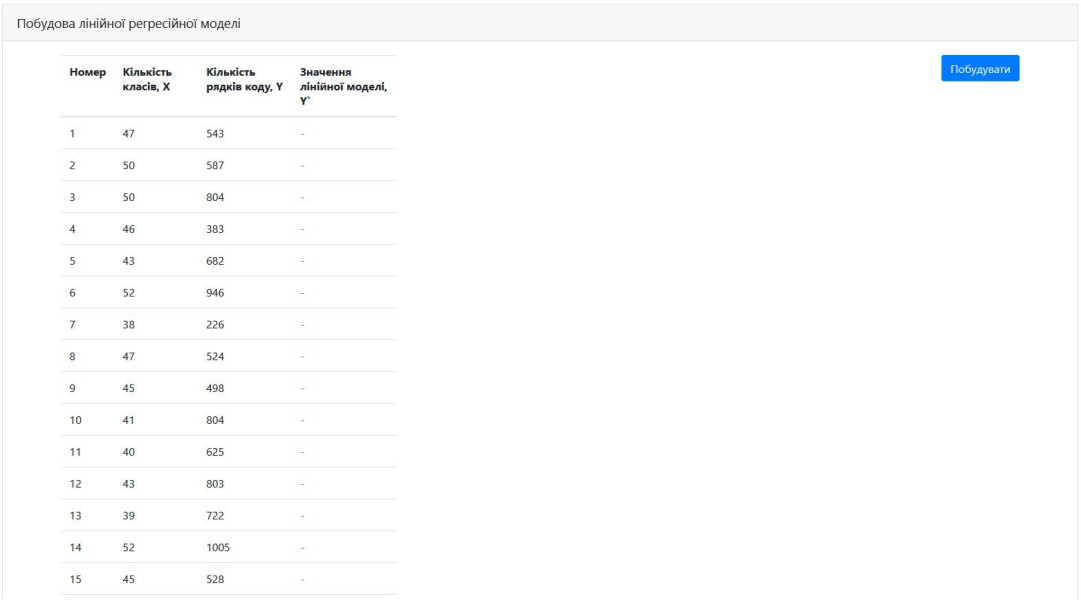


Рисунок Г.3 – Таблиця даних

5. Після цього необхідно натиснути кнопку нормалізувати (рис. Г.4).

Рисунок Г.4 – Нормалізація

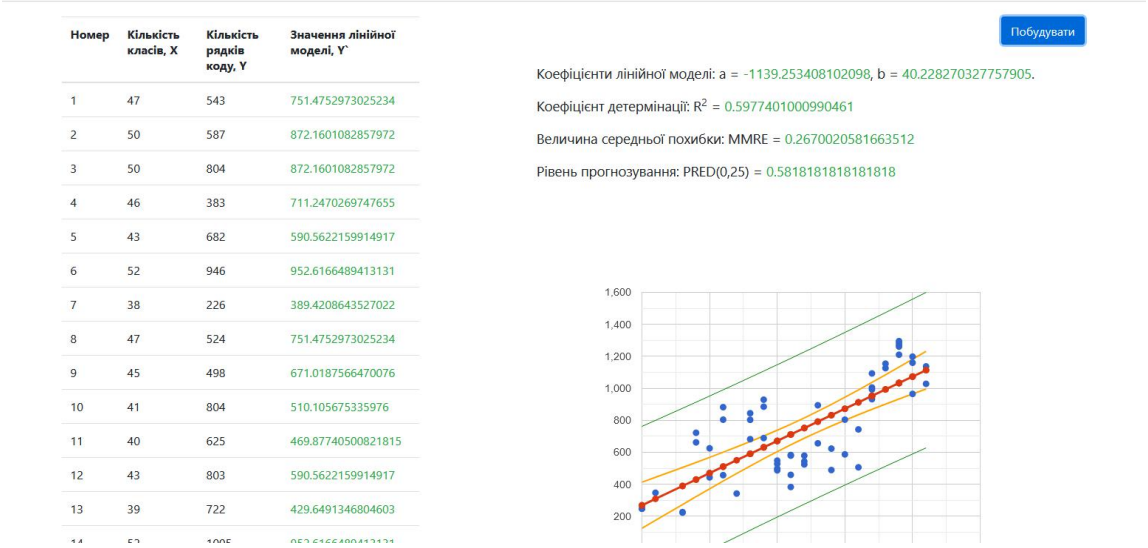
6. Наступним кроком потрібно натиснути кнопку “Застосувати” після чого буде здійснено перехід до побудови лінійної моделі (рис. Г.5).



Номер	Кількість класів, X	Кількість рядків коду, Y	Значення лінійної моделі, Y`
1	47	543	-
2	50	587	-
3	50	804	-
4	46	383	-
5	43	682	-
6	52	946	-
7	38	226	-
8	47	524	-
9	45	498	-
10	41	804	-
11	40	625	-
12	43	803	-
13	39	722	-
14	52	1005	-
15	45	528	-

Рисунок Г.5 – Таблиця нормалізованих даних

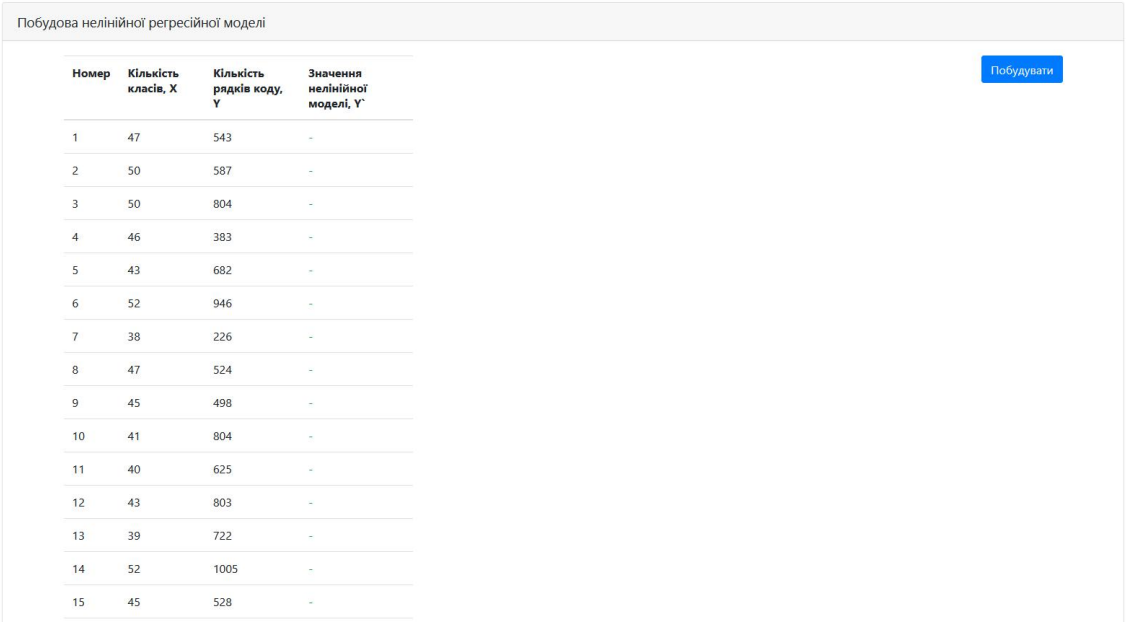
7. Для того щоб побудувати лінійну модель, необхідно натиснути кнопку «Побудувати» (рис. Г.6). В результаті будуть наведені коефіцієнти отриманої моделі, її оцінки адекватності, та отримані інтервали.



Номер	Кількість класів, X	Кількість рядків коду, Y	Значення лінійної моделі, Y`
1	47	543	751.4752973025234
2	50	587	872.1601082857972
3	50	804	872.1601082857972
4	46	383	711.2470269747655
5	43	682	590.5622159914917
6	52	946	952.6166489413131
7	38	226	389.4208643527022
8	47	524	751.4752973025234
9	45	498	671.0187566470076
10	41	804	510.105675335976
11	40	625	469.87740500821815
12	43	803	590.5622159914917
13	39	722	429.6491346804603

Рисунок Г.6 – Отримана лінійна модель

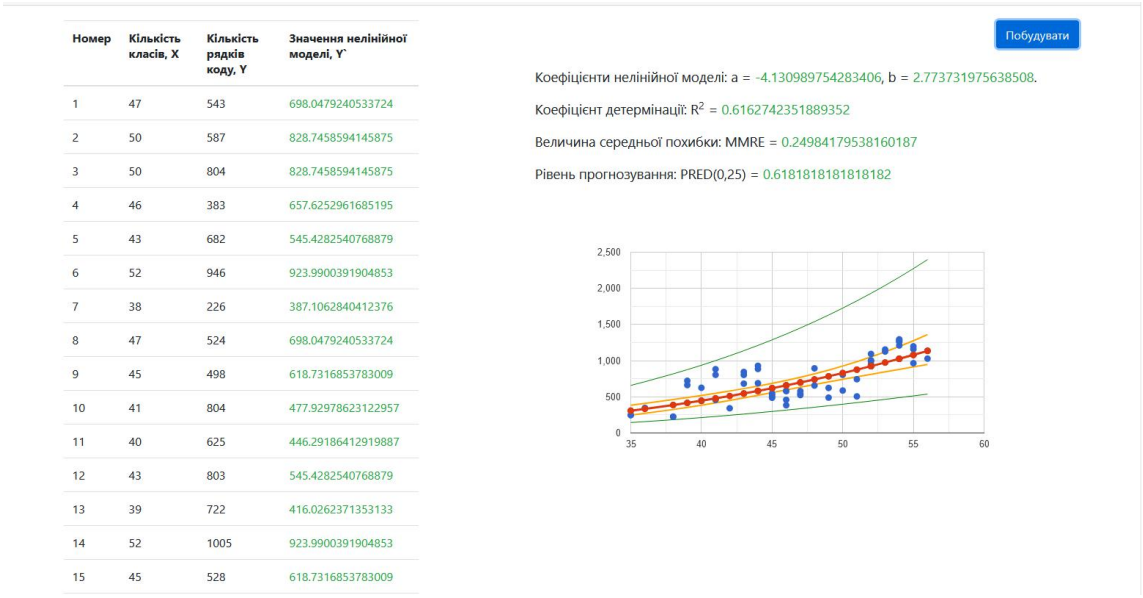
8. Для переходу до нелінійної моделі необхідно натиснути кнопку “Застосувати” після чого буде відкрита форма з нелінійною моделлю (рис. Г.7)



Номер	Кількість класів, X	Кількість рядків коду, Y	Значення нелінійної моделі, Y`
1	47	543	-
2	50	587	-
3	50	804	-
4	46	383	-
5	43	682	-
6	52	946	-
7	38	226	-
8	47	524	-
9	45	498	-
10	41	804	-
11	40	625	-
12	43	803	-
13	39	722	-
14	52	1005	-
15	45	528	-

Рисунок Г.7 – Нелінійна модель

9. Для того щоб побудувати нелінійну модель, необхідно натиснути кнопку «Побудувати» (рис. Г.8). В результаті будуть наведені коефіцієнти отриманої моделі, її оцінки адекватності, та отримані інтервали.



Номер	Кількість класів, X	Кількість рядків коду, Y	Значення нелінійної моделі, Y`
1	47	543	698.0479240533724
2	50	587	828.7458594145875
3	50	804	828.7458594145875
4	46	383	657.6252961685195
5	43	682	545.4282540768879
6	52	946	923.9900391904853
7	38	226	387.1062840412376
8	47	524	698.0479240533724
9	45	498	618.7316853783009
10	41	804	477.92978623122957
11	40	625	446.29186412919887
12	43	803	545.4282540768879
13	39	722	416.0262371353133
14	52	1005	923.9900391904853
15	45	528	618.7316853783009

Рисунок Г.8 – Отримана нелінійна модель

10. При неправильному форматі введених даних програма показує повідомлення про помилку (рис.Г.9)

Рисунок Г.9 – Повідомлення про помилку