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.

КВАЛІФІКАЦІЙНАРОБОТА
на здобуття ступеня вищої освіти «магістр»

на тему: Нелінійна регресійна модель для оцінювання трудомісткості розробки логістичних web-сервісів на Python та розробка програми для її реалізації

Виконав: студент 6151м групи
Заславський Д.А.
(підпис, ПІБ)

Керівник роботи:
доцент, к.т.н., доцент
(посада, науковий ступень вчене звання)

Устенко І.В.
(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 Заславському Дмитру Андрійовичу
(Прізвище, ім'я, по батькові)

1. Тема роботи: Нелінійна регресійна модель для оцінювання трудомісткості розробки логістичних web-сервісів на Python та розробка програми для її реалізації _____
Керівник роботи Устенко Ірина Валеріївна, к.т.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: 1.Актуальність теми, 2. Мета і завдання дослідження, 3.Об'єкт і предмет дослідження, 4. Методи дослідження, 5. Наукова новизна одержаних результатів, 6. Практичне значення одержаних результатів, 7. Апробація результатів досліджень, 8. Розподіли випадкових величин до нормалізації, 9. Розподіли випадкових величин після нормалізації. 10. Нелінійна регресійна модель для оцінювання трудовитрат на розробку логістичних web-сервісів на Python, 11. Довірчий інтервал, інтервал передбачення та рівняння регресії для ненормалізованих даних, 12. Довірчий інтервал, інтервал передбачення та рівняння регресії для нормалізованих даних, 13. Порівняння значень метрик точності, 14.Діаграма варіантів використання, 15. Діаграма діяльності,

16. Концептуальна модель даних, 17. Діаграма класів, 18. Діаграма послідовності для варіанту використання "Ввести дані", 19. Інтерфейс користувача, 20.Висновки.

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*
[illegible]	[illegible]	[illegible]	[illegible]
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 _____ Заславський Д.А.
(підпис) (ПІБ)

Керівник роботи _____ Устенко І.В.
(підпис) (ПІБ)

РЕФЕРАТ

Заславський Дмитро Андрійович

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

Обсяг роботи: 107 стор., 23 таблиць, 13 рисунків, 23 використаних джерел, 5 додатків.

Актуальність теми. Оцінювання трудовитрат розробки ПЗ є важливою складовою у визначенні термінів реалізації програмного проекту та його здійсненності. Без адекватної та достовірної оцінки трудовитрат неможливо забезпечити чітке планування та управління проектом. Існуючі підходи до оцінювання мають свої переваги та недоліки, залежать від класу оцінюваних проектів, мови програмування, Нелінійна регресійна модель для оцінювання трудомісткості розробки логістичних web-сервісів на Python та розробка програми для її реалізації платформи. Так як не було знайдено моделі для оцінювання трудомісткості розробки логістичних web-сервісів на Python, то актуальною є тема визначення трудовитрат такої розробки.

Мета та задачі дослідження. Мета дослідження полягає в підвищенні достовірності оцінки трудомісткості розробки логістичних web-сервісів на Python. Сформульована мета визначила наступні задачі: виконати аналіз підходів до оцінювання трудовитрат, удосконалити математичну модель для оцінювання трудовитрат на розробку логістичних web-сервісів на Python та розробити ПЗ для реалізації моделі.

Об'єктом досліджень є процес оцінювання трудовитрат на створення логістичних web-сервісів на Python.

Предметом досліджень є нелінійна модель регресії для оцінювання трудовитрат створення логістичних web-сервісів на Python.

Методи дослідження. Поставлені задачі вирішувалися з допомогою методів регресійного аналізу, теорії ймовірностей та математичної статистики.

Наукова новизна одержаних результатів: удосконалено модель регресії для оцінювання трудовитрат на розробку логістичних web-сервісів на Python з застосуванням для нормалізації десяткового логарифму, що дозволяє отримати більш точну оцінку в порівнянні з лінійною моделлю.

Практичне значення одержаних результатів полягає в тому, що була розроблена програма, яка дає можливість оцінити трудовитрати на розробку логістичних web-сервісів на Python, зменшити час на розрахунки та підвищити достовірність.

Апробація результатів досліджень. Отримані результати були представлені XIII Міжнародній науково-практичній конференції «Free and Open Source Software» (м. Харків, 16-18 листопада 2021 р.).

Ключові слова: ТРУДОМІСТКІСТЬ РОЗРОБКИ, ЛОГІСТИЧНИЙ WEB-СЕРВІС, PYTHON, МОДЕЛЬ НЕЛІНІЙНОЇ РЕГРЕСІЇ

ABSTRACT

Dmytro Zaslavskyi

Nonlinear regression model for estimating the complexity of developing logistics web-services in Python and developing the software for its implementation

Major's thesis for gaining the educational level of major in 121 – «Software engineering».Admiral Makarov National University of Shipbuilding. Mykolaiv, 2021.

The work contains: 107 pages, 23 tables, 13 figures, 23 references, 5 appendices.

Relevance of the topic: Estimating the labor costs of software development is an important component in determining the timing of the program project and its feasibility. Without an adequate and reliable estimate of labor costs, it is impossible to ensure clear project planning and management. Existing approaches to evaluation have their advantages and disadvantages, depending on the class of projects being evaluated, programming language, platform. Since no model has been found to assess the complexity of developing logistics web-services in Python, the topic of determining the labor costs of such development is relevant.

Purpose and tasks of the research. The purpose of the study is to increase the reliability of the assessment of the complexity of developing logistics web-services in Python. The formulated purpose identified the following tasks: to analyze the approaches to labor cost estimation, to improve the mathematical model for labor cost estimation for the development of logistics web-services in Python and to develop software for the implementation of the model.

The object of research is the process of estimating the labor costs for the creation of logistics web-services in Python.

The subject of research is a nonlinear regression model for estimating the labor costs of creating logistics web-services in Python.

Research methods. The tasks were solved using regression analysis methods, probability theory and mathematical statistics**.**

Scientific novelty of the obtained results: improved regression model for estimating labor costs for the development of logistics web-services in Python with the use of decimal logarithm for normalization, which allows to obtain a more accurate estimate compared to the linear model..

The practical significance of the results is that a program has been developed that allows you to estimate the labor costs for the development of logistics web-services in Python, reduce the time for calculations and improve their quality.

Approbation of research results. The obtained results were presented at the XIII International Scientific and Practical Conference "Free and Open Source Software" (Kharkiv, November 16-18, 2021).

Keywords: HARDWARE OF DEVELOPMENT, LOGISTICS WEB-SERVICE, PYTHON, NONLINEAR REGRESSION MODEL

ЗМІСТ

ВСТУП

Актуальність теми. Оцінювання трудомісткості розробки програмного забезпечення (ПЗ) є важливою складовою у визначенні термінів реалізації програмного проекту та його здійсненності. Моделі та методи оцінювання трудомісткості використовуються для розробки бюджету проекту, аналізу ступеня ризику та вибору компромісного рішення, планування та управління проектом. Без адекватної та достовірної оцінки трудовитрат неможливо забезпечити чітке планування та управління проектом [1].

Існує безліч рішень щодо оцінювання трудомісткості розробки програмного забезпечення [2-10]. Одним із перших дослідників у цій галузі був Нельсон (1965-1966р.). В 1981 Баррі Боем створив принципово нову модель COCOMO. 1997 року з'явилася модель COCOMO II, яка являється спадкоємницею моделі COCOMO. Дослідники продовжують активно працювати, удосконалюючи вже існуючі підходи та створюючи нові [4-5].

У результаті склалися цілі напрямки оцінювання трудомісткості програмного забезпечення, серед яких можна виділити: експертні, алгоритмічні, комбіновані та інші методи [1-3].

Більшість моделей для визначення трудомісткості розробки програмного забезпечення може бути зведена до функцій п'яти основних параметрів [1]: розміру кінцевого проекту; особливостей процесу, що використовується для отримання кінцевого проекту; можливостей персоналу; середовища, що складається з інструментів та методів; необхідної якості проекту. Найвпливовіший чинник оцінювання трудомісткості у цих моделях – розмір програмного проекту.

Великий розвиток отримали методи та моделі оцінювання трудовитрат на створення ПЗ, які основані на аналізі статистичних даних про раніше виконані проекти, при цьому емпірично визначається залежність

трудомісткості проекту від якогось кількісного показника, що характеризує програмний продукт (ПП). Проводиться оцінювання цього показника для даного проекту, після чого на основі моделі прогнозуються майбутні трудовитрати.

Незважаючи на значне опрацювання цих питань, неможливо виділити єдиний універсальний підхід до оцінювання трудомісткості розробки програмного забезпечення, який в усіх випадках буду давати високу точність. Кожне з існуючих рішень має свої переваги та недоліки, залежить від класу оцінюваних проектів, мови програмування, платформи розробки. Тобто існує необхідність удосконалювати існуючі та розробляти нові методи та моделі для підвищення точності оцінювання трудомісткості розробки ПЗ. Тому завдання оцінювання трудомісткості розробки логістичних web-сервісів на Python є актуальним.

Мета та задачі дослідження. Мета дослідження полягає в підвищенні достовірності оцінки трудомісткості розробки логістичних web-сервісів на Python. Сформульована мета визначила наступні задачі: виконати аналіз підходів до оцінювання трудовитрат, удосконалити математичну модель для оцінювання трудовитрат на розробку логістичних web-сервісів на Python та розробити ПЗ для реалізації моделі.

Об’єктом досліджень є процес оцінювання трудовитрат на створення логістичних web-сервісів на Python.

Предметом досліджень є нелінійна модель регресії для оцінювання трудовитрат створення логістичних web-сервісів на Python.

Методи дослідження. Поставлені задачі вирішувалися з допомогою методів регресійного аналізу, теорії ймовірностей та математичної статистики.

Наукова новизна одержаних результатів: удосконалено модель регресії для оцінювання трудовитрат на розробку логістичних web-сервісів на Python з застосуванням для нормалізації десяткового логарифму, що дозволяє отримати більш точну оцінку в порівнянні з лінійною моделлю.

Практичне значення одержаних результатів полягає в тому, що була розроблена програма, яка дає можливість оцінити трудовитрати на розробку логістичних web-сервісів на Python, зменшити час на розрахунки та підвищити їх достовірність.

Особистий внесок здобувача. Кваліфікаційна (магістерська) робота є самостійно виконаною працею. Усі результати, викладені у роботі, отримані автором особисто. У тезах доповіді [11], написаних у співавторстві, здобувачу належить аналіз переваг використання Python для веб-розробок.

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).

Публікації. Основні результати магістерської роботи викладено у 1 науковій праці – тезах конференції.

1 ОСОБЛИВОСТІ ВИКОРИСТАННЯ PYTHON ДЛЯ WEB-РОЗРОБОК ТА АНАЛІЗ СУЧАСНИХ ПІДХОДІВ ДО ОЦІНЮВАННЯ ТРУДОМІСТКОСТІ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

1.1 Переваги та недоліки використання Python для web-розробок

Професія web-розробника ще довго залишатиметься затребуваною на ринку IT-фахівців. Існують різні мови та технології створення web-продуктів. Frontend-розробники використовують HTML, CSS та JavaScript, Backend-розробники – PHP, Python та Ruby. Немає однозначного рішення, яка мова краща для веб-розробок. У кожної зі згаданих мов є свої переваги та недоліки. Вибір фахівця залежить від поточних завдань, поставлених цілей, складності проекту та власних уподобань [12].

Зазначимо переваги та недоліки використання Python для веб-розробок. Python – мова програмування загального призначення, яку широко застосовують фахівці, на спеціалістів по якій великий попит, яку підтримують могутні компанії в IT-промисловості (Google, Facebook, Dropbox, Spotify, Quora, Netflix), яка має велике співтовариство користувачів; одна із найбільш швидко зростаючих мов по рейтингу TIOBE (TIOBE programming community index) [13].

Python застосовується для: створення мобільних додатків, прикладних програм, скриптів, вбудованих систем для різних пристроїв, ігор; системного адміністрування; інтелектуального аналізу даних; організації автоматизованого тестування.

Також Python використовують веб-розробники.

Розглянемо деякі плюси мови для web-розробників [14]:

– легко піддається вивченню;

– логічна, лаконічна та зрозуміла;

– гнучка;

– ефективна для візуалізації;

– має підтримку асинхронного коду;

– масштабована;

– має код меншого розміру;

– має значну кількість бібліотек.

Недоліками Python для web-розробників є [14]:

– не велика швидкодія;

– споживає більше пам'яті, ніж подібні мови;

– не реалізована багатопоточність в повному обсязі.

Хоча недоліки Python досить суттєві при web-розробках, але сильні сторони мови все ж беруть перевагу над недоліками. Це підтверджується тим, що Python використовували при розробці таких систем, як YouTube, Survey Monkey, Pinterest, Instargam, Uber.

При роботі з Python застосовують фреймворки Pylons, Tornado, TurboGears, Flask, CherryPy Django, Pyramid, FastAPI.

Також розроблені і движки для веб-програмування на Python: Abilian SBE, Saleor, Wagtail, Ella, Django-CMS.

Python полюбляють і розробники допоміжних інструментів, які використовуються при роботі з Інтернетом, в тому числі і скраперів, які займаються збиранням інформації на замовлення з різних сторінок інтернету.

1.2 Методи та моделі для оцінювання трудомісткості розробки програмного забезпечення

В епоху інформаційних технологій, що швидко розвиваються, зростаючої кількості розроблюваних програмних проектів, високої конкуренції у галузі розробки програмного забезпечення, дуже важливим стає вміння оцінити на ранніх етапах можливі вигоди та збитки від проекту, проаналізувати можливі сценарії розвитку подій. Помилка, недооцінка складнощів, з якими прийдеться зіткнутися в процесі розробки, переоцінка

сил команди розробників, неприйняття до розрахунку тих чи інших факторів часто призводить до фінансових втрат і навіть банкрутства IT-компаній.

За статистикою лише чверть усіх розпочатих проектів завершується своєчасно та в повному обсязі, чверть взагалі скасовується, і половина проектів завершується з перевищенням бюджетних витрат або із запізненням.

Аналіз причин можливих помилок оцінки при плануванні IT-проектів показав, що найбільш значущими та частими причинами помилок оцінювання є:

– невизначеність на ранніх стадіях проектування;

– недосконалість організації процесу розробки;

– неповні технічні вимоги;

– оптимістична та суб'єктивна оцінка трудомісткості задач.

Існує безліч рішень щодо оцінювання трудомісткості розробки програмного забезпечення [2-10]. Одним із перших дослідників у цій галузі був Нельсон (1965-1966р.). В 1981 Баррі Боем створив принципово нову модель COCOMO. 1997 року з'явилася модель COCOMO II, яка являється спадкоємницею моделі COCOMO [4-5]. Дослідники продовжують активно працювати, удосконалюючи вже існуючі підходи та створюючи нові.

У результаті склалися цілі напрямки оцінювання трудомісткості програмного забезпечення, серед яких можна виділити: експертні, регресійні, нейронні мережі, динамічні, модельні, комбіновані та інші [1-3].

Експертні методи базуються на оцінках експертів, які мають значний досвід у розробці програмного забезпечення в заданій сфері.

Модельні методи основані на математичній моделі, за допомогою якої відповідно до вхідних параметрів обчислюється трудомісткість проекту.

Регресійні методи базуються на емпіричних даних по завершеним проектам та містять різні види регресії. Вони використовуються у поєднанні з методами оцінювання, основаними на моделях.

Нейронні мережі – це методи оцінювання з можливістю навчання за аналогією за допомогою штучного інтелекту.

Динамічні методи використовують динаміку зміни таких показників як витрати, навички, зусилля.

Комбіновані – є комбінацією двох або більше методів для формування найбільш відповідної системи оцінювання.

Також існує інша класифікація методів та моделей оцінювання трудомісткості розробки програмного забезпечення, по якій всі вони поділяються на алгоритмічні та неалгоритмічні.

До неалгоритмічних відносяться методи Price-to-win, оцінювання за Паркінсоном, експертне оцінювання, оцінювання за аналогією.

До алгоритмічних належать модель Путнема SLIM, сімейство моделей COCOMO, сімейство моделей COCOMO II, регресійні моделі.

Більшість моделей для визначення трудомісткості розробки програмного забезпечення може бути зведена до функцій п'яти основних параметрів [1]: розміру кінцевого проекту; особливостей процесу, що використовується для отримання кінцевого проекту; можливостей персоналу; середовища, що складається з інструментів та методів; необхідної якості проекту. Найвпливовіший чинник оцінювання трудомісткості у цих моделях – розмір програмного проекту.

В оцінюванні трудомісткості ПЗ використовують дві одиниці розміру:

– кількість рядків коду Line of Code (LOC) чи її похідні [4]

– кількість функціональних точок Function Point (FP) [15].

Розглянемо переваги і недоліки використання LOC для вимірювання розміру програмного забезпечення. Перевагою LOC є простота. Недоліками LOC є наступні: неможливість точного визначення розміру до завершення проекту; залежність від мови програмування; не врахування якості коду; важкість застосування для проектів на декількох мовах програмування. В даний час існує крім LOC ряд похідних метрик.

Функціональні точки були запропоновані в якості альтернативи LOC [2] досить давно А. Дж. Альбректом. В компанії IBM у середині 70-х років минулого століття виникла потреба в проведенні оцінювання трудовитрат на

розробку ПЗ, яке б не залежало від мови та середовища розробки. Недолік функціонально-орієнтованих метрик: результати основані на суб'єктивній інформації; використовуються непрямі виміри.

Оцінка розміру ПЗ базується на аналізі вимог до системи. Розглянемо основні способи отримання такої оцінки:

1. Обчислення розміру за певними алгоритмами, аналізуючи емпіричні дані.

2. Використовуючи метод оцінювання за аналогією. В цьому випадку також використовуються емпіричні дані стосовно подібних проектів, на основі яких роблять висновок про новий проект.

Проблеми оцінювання розміру програмного проекту:

– відсутність емпіричних даних;
– неточності в емпіричних даних;
– не дотримання стандартів оцінювання;
– відсутність кваліфікованого спеціаліста.

З аналізу сучасних підходів до розрахунку трудомісткості розробки випливає, на даний момент існує велика кількість методів та моделей для її оцінювання. Але частина з цих підходів вже застаріла. З іншого боку прогнози щодо точності отриманих оцінок існуючими методами та моделями, зазвичай, справджуються для проектів того ж класу, що й проекти, на яких методи та моделі були побудовані. Тобто відсутні універсальні рекомендації по отриманню оптимальних оцінок при різних мовах, технологіях і середовищах.

В наш час великий розвиток отримали методи та моделі оцінювання трудовитрат на створення ПЗ, які аналізують статистичні дані про уже завершені проекти. В цьому випадку будується функціональна залежність трудомісткості від обраної метрики (метрик) програмного продукту. Проводиться оцінювання метрики, а потім з використанням моделі, передбачується трудомісткість розроблюваного проекту.

Тому існує необхідність удосконалювати існуючі та розробляти нові методи та моделі оцінки трудомісткості розробки програмного забезпечення, в тому числі і для оцінювання трудомісткості розробки логістичних web-сервісів на Python, які будуть враховувати негаусівський розподіл емпіричних даних та нелінійність зв'язків між факторами.

1.3 Перевірка якості регресійних моделей для оцінювання трудомісткості розробки програмного забезпечення

Розглянемо показники для оцінювання якості регресійної моделі.

– Коефіцієнт детермінації R^2:

$$R^2 = 1 - (\sum_1^n (y_i - \hat{y}_i)^2 / \sum_1^n (y_i - \bar{y})^2), \quad (1.1)$$

де y_i – вихідне значення;

$\hat{y}_i$ – розраховане значення;

$\bar{y}$ – середнє значення;

n – кількість вихідних значень.

Коефіцієнт детермінації змінюється в діапазоні від 0 до 1. Якщо він дорівнює 0, це означає, що зв'язок між змінними регресійної моделі відсутній і замість неї для оцінки значення вихідної змінної можна використовувати просте середнє її вихідних значень. Навпаки, якщо коефіцієнт детермінації дорівнює 1, це відповідає ідеальній моделі, коли всі точки спостережень лежать точно на лінії регресії, тобто сума квадратів відхилень дорівнює 0.

Насправді, якщо коефіцієнт детермінації близький до 1, це свідчить про те, що модель працює дуже добре (має високу значимість), а якщо до 0, це означає низьку значимість моделі, коли вхідна змінна погано «пояснює» поведінку вихідної, т. б. лінійна залежність між ними відсутня. Очевидно, що така модель матиме низьку якість[16, 17,18].

– Сума квадратів відхилень S_Y :

$$S_Y = \sum_1^n (y_i - \hat{y}_i)^2 , \quad (1.2)$$

де y_i – вихідне значення;

$\hat{y}_i$ – розраховане значення.

Сума квадратів відхилень є мірою якості регресійної моделі. Чим менше значення S_Y, тим краща якість побудованої моделі.

– Величина відносної похибки MRE:

$$MRE = \left|\frac{y_i - \hat{y}_i}{y_i}\right| , \quad (1.3)$$

де y_i – вихідне значення;

$\hat{y}_i$ – розраховане значення.

Через MRE розраховуються такі важливі показники якості регресійних моделей, як середня величина відносної похибки $MMRE$ та рівень прогнозування $PRED(l)$.

– Середня величина відносної похибки $MMRE$:

$$MMRE = \frac{1}{n} \cdot \sum_1^n \left|\frac{y_i - \hat{y}_i}{y_i}\right| . \quad (1.4)$$

де y_i – вихідне значення;

$\hat{y}_i$ – розраховане значення;

n – кількість вихідних значень.

Прийнятне значенням $MMRE \leq 0{,}25$. Кращим вважається менше значення $MMRE$.

– Рівень прогнозування $PRED(l)$:

$$PRED(l) = \frac{k}{n}, \tag{1.5}$$

де k – кількість значень $\left|\frac{y_i - \hat{y}_i}{y_i}\right| \leq l$ $(i = \overline{1,n})$.

Кращою є регресійна модель, для якої значення PRED(l) є більшим. При обчисленні зазвичай використовують PRED(0,25) .

Всі вищенаведені показники використовують також для порівняння моделей між собою.

1.4 Обгрунтування необхідності проведення досліджень

Із вищевикладеного випливає, що на сьогодні існує велика кількість підходів до оцінювання трудомісткості розробки програмного забезпечення. Їх використання є одним з головних аргументів при техніко-економічному обгрунтуванні вартості розробки.

Але частина з цих підходів має низьку точність оцінювання, не враховує нелінійність зв’язків між факторами, призначена для даних, що підпорядковуються нормальному закону розподілу. Також відомо, що не існує універсальних методів чи моделей, які є оптимальними для всіх типів програмного забезпечення і середовищ розробки. Тому існує необхідність удосконалювати існуючі та розробляти нові методи та моделі оцінювання трудомісткості розробки програмного забезпечення обраних типів, зокрема логістичних web-сервісів на Python. Це дозволить підвищити точність оцінювання трудомісткості розробки логістичних web-сервісів на Python з урахуванням нелінійних зв’язків між даними, що не підпорядковуються нормальному закону розподілу.

2 НЕЛІНІЙНА РЕГРЕСІЙНА МОДЕЛЬ ДЛЯ ОЦІНЮВАННЯ ТРУДОМІСТКОСТІ РОЗРОБКИ ЛОГІСТИЧНИХ WEB-СЕРВІСІВ НА PYTHON

2.1 Алгоритм побудови нелінійної регресійної моделі для оцінювання трудомісткості розробки логістичних web-сервісів на Python

При побудові нелінійної регресійної моделі для оцінювання трудомісткості розробки логістичних web-сервісів на Python будемо використовувати наступні дані:

- кількість рядків коду (X);
- трудомісткість розробки (Y).

Вихідні дані для оцінювання трудомісткості розробки мають нормальний розподіл лише в окремих випадках і зв'язки між ними не завжди є лінійними. Перевірку на нормальність вихідних даних будемо виконувати за допомогою критерію χ^2 Пірсона. У випадку нормально розподілених даних необхідно використовувати лінійну регресійну теорію.

При негаусівських даних для побудови нелінійної регресійної моделі для оцінювання трудомісткості будемо використовувати метод нормалізуючих перетворень [19, 20].

В якості нормалізуючого перетворення можна обрати перетворення на основі натурального чи десяткового логарифму, перетворення Бокса-Кокса, перетворення Джонсона та інші. В кваліфікаційній роботі в якості нормалізуючого перетворення обрано десятковий логарифм:

$$Z_X = \log_{10} X, \tag{2.1}$$

$$Z_Y = \log_{10} Y, \tag{2.2}$$

де X, Y – вихідні значення,

Z_X, Z_Y – нормалізовані значення.

Нормалізовані дані необхідно перевірити на викиди. Для цього використаємо квадрат відстані Махаланобіса [21] :

$$d_i^2 = (Z_i - \overline{Z})^T S_N^{-1} (Z_i - \overline{Z}), \quad (2.3)$$

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

N – кількість елементів;

$\overline{Z}$ – середнє значення;

S_N –матриця коваріацій:

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

Викиди будемо знаходити з використанням критерію Фішера. Розрахуємо статистики

$$T_{S_i} = (N - k) N d_i^2 / ((N^2 - 1)k), \quad (2.4)$$

де N – кількість елементів,

k – кількість параметрів.

Елементи (Z_{x_i}, Z_{y_i}), у яких T_{S_i} перевищує критичне значення Фішера $F_{k,N-k,\alpha}$ при обраному рівні значущості α (0,05), являються викидами. Їх необхідно вилучати з вихідних даних та повторювати розрахунок.

Після закінчення ітерацій по вилученню викидів будується лінійна регресійна модель:

$$\hat{Z}_y = \mathrm{b}_0 + \mathrm{b}_1 \cdot Z_x + \varepsilon, \quad (2.5)$$

де $\mathrm{b}_0, \mathrm{b}_1$ – коефіцієнти;

ε – випадкова помилка.

У випадку нормально розподіленої випадкової помилки ε для отриманої регресійної моделі будуються довірчі інтервали та інтервали прогнозування.

Довірчим інтервалом називається інтервал, через який вихідна функція проходить з певною ймовірністю (часто використовуване значення 95%). Мінімальне та максимальне значення інтервалу довіри для побудованої регресії визначається по співвідношенню [22]:

$$\hat{Z}_Y \pm t_{\alpha/2,n-2} \cdot S_{Z_Y} \sqrt{\frac{1}{n} + \frac{(Z_X - \overline{Z_X})^2}{\sum_n^1 (z_{xi} - \overline{Z_X})^2}}, \quad (2.6)$$

де $\hat{Z}_y$ – розраховані значення;

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

α – рівень статистичної значущості;

n – кількість елементів в вибірці;

$$S_{Z_Y} = \sqrt{\frac{1}{n-2} \cdot \sum_1^n \left(z_{y_i} - \hat{z}_{y_i}\right)^2}.$$

Інтервалом передбачення називається інтервал, який буде містити нові точки нові точки з певною ймовірністю (часто використовуване значення 95%). Мінімальне та максимальне значення інтервалу передбачення для побудованої регресії визначається по співвідношенню (2.7):

$$\hat{Z}_Y \pm t_{\alpha/2,n-2} \cdot S_{Z_Y} \sqrt{1 + \frac{1}{n} + \frac{(Z_X - \overline{Z_X})^2}{\sum_n^1 (z_{x_i} - \overline{Z_x})^2}}. \quad (2.7)$$

Перехід до нелінійної моделі виконаємо через зворотне перетворення:

$$\hat{Y} = 10^{b_0 + \varepsilon} \cdot X^{b_1}. \quad (2.8)$$

Аналогічно через зворотне перетворення отримаємо межі довірчих інтервалів та інтервалів передбачення.

Якість отриманого нелінійного рівняння регресії перевіримо через підрахунок коефіцієнта детермінації R^2 (1.1), середньої величини відносної похибки $MMRE$ (1.4), та рівня прогнозування $PRED(l)$ (1.5).

2.2 Побудова нелінійної регресійної моделі для оцінювання трудомісткості розробки логістичних web-сервісів на Python

Емпіричні дані для побудови нелінійної регресійної моделі для оцінювання трудомісткості розробки логістичних web-сервісів на Python були отримані на GitHub у кількості 34 проектів. Вихідні дані перевірили на нормальність розподілу за допомогою критерію χ^2 Пірсона. Дані не підпорядковувалися нормальному закону розподілу. Згідно (2.1) та (2.2) виконали нормалізацію вихідних даних. Перевірили нормалізовані дані на викиди за допомогою квадрату відстані Махаланобіса (2.3). Після вилучення викидів залишилося 30 наборів даних.

Вихідні значення та результати нормалізації без викидів представлені в табл. 2.1

Таблиця 2.1 – Вихідні значення та результати нормалізації

№	Y трудомісткість люд/год	X кількість рядків коду	Z_Y нормаліз. Y	Zx нормаліз. X
1	2	3	4	5
1	14968	76896	4,17516	4,88590
2	62888	105083	4,79857	5,02153
3	54192	109199	4,73394	5,03822
4	29952	111308	4,47643	5,04653
5	70200	128981	4,84634	5,11053
6	61064	129414	4,78579	5,11198
7	58216	133332	4,76504	5,12493
8	92456	133735	4,96594	5,12625
9	56280	134529	4,75035	5,12882

Продовження табл. 2.1

1	2	3	4	5
10	86352	146912	4,93627	5,16706
11	62072	147417	4,79290	5,16855
12	102720	150112	5,01166	5,17642
13	62160	152631	4,79351	5,18364
14	73160	154631	4,86427	5,18930
15	97408	160236	4,98859	5,20476
16	39808	160841	4,59997	5,20640
17	41088	165215	4,61372	5,21805
18	102216	170332	5,00952	5,23130
19	79872	170344	4,90239	5,23133
20	66576	171392	4,82332	5,23399
21	121120	195062	5,08322	5,29017
22	118224	198378	5,07271	5,29749
23	125128	207758	5,09735	5,31756
24	110328	218917	5,04269	5,34028
25	103012	222454	5,01289	5,34724
26	45416	233508	4,65721	5,36830
27	101240	234231	5,00535	5,36964
28	175200	253345	5,24353	5,40371
29	191480	306478	5,28212	5,48640
30	329920	429988	5,51841	5,63346

На рисунках 2.1.а та 2.1.б представлені відповідно вихідний та нормалізований розподіли Y.

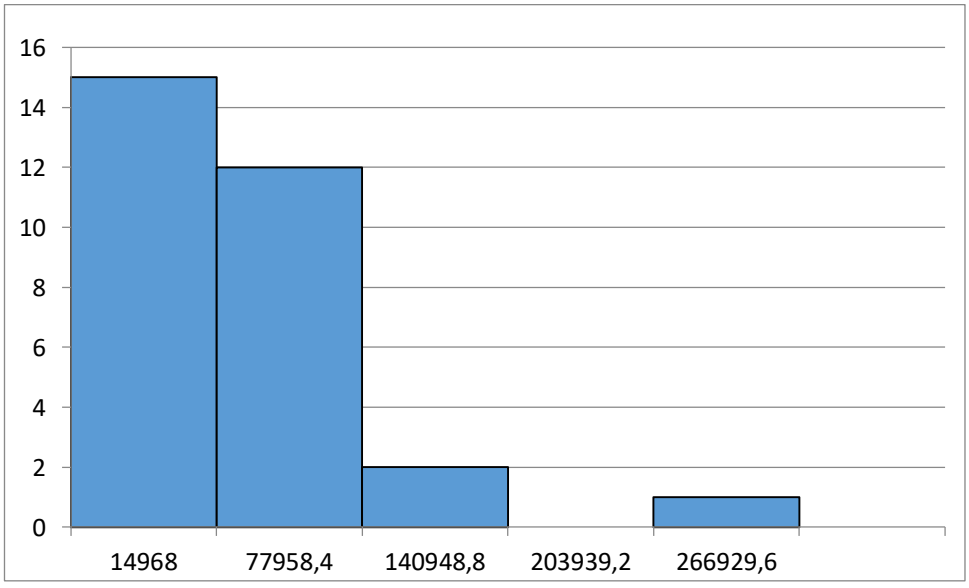

Рисунок 2.1.а – Емпіричний розподіл для вибірки Y

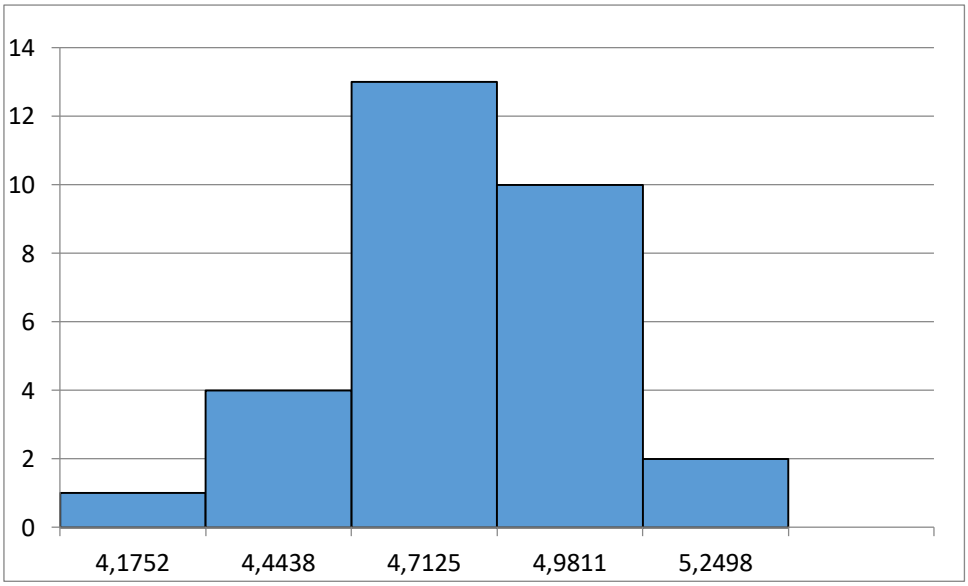

Рисунок 2.1.б – Нормалізований розподіл для вибірки Y

Емпіричний розподіл для вибірки X представлено на рис. 2.2.а, а її нормалізований розподіл представлений на рис. 2.2.б.

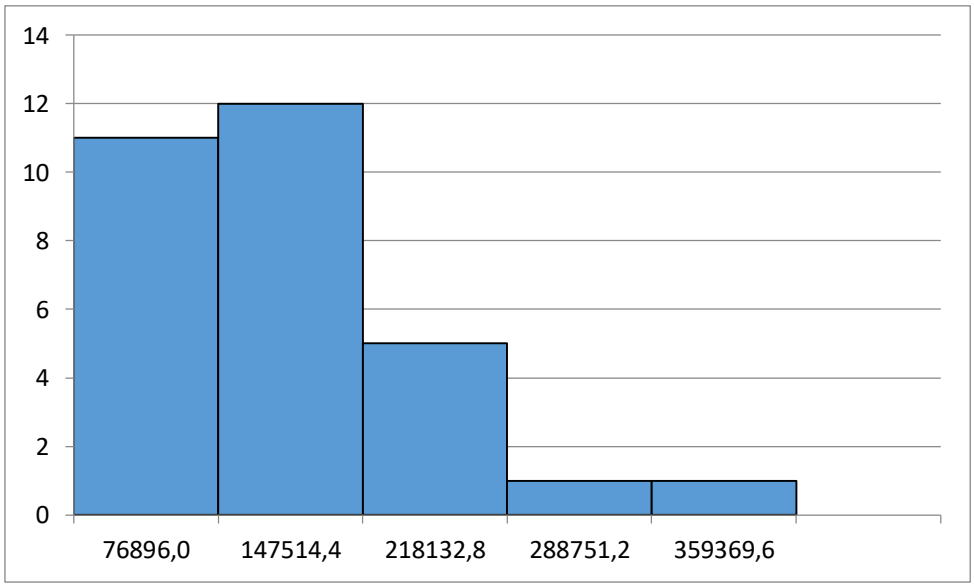

Рисунок 2.2.а – Емпіричний розподіл для вибірки X

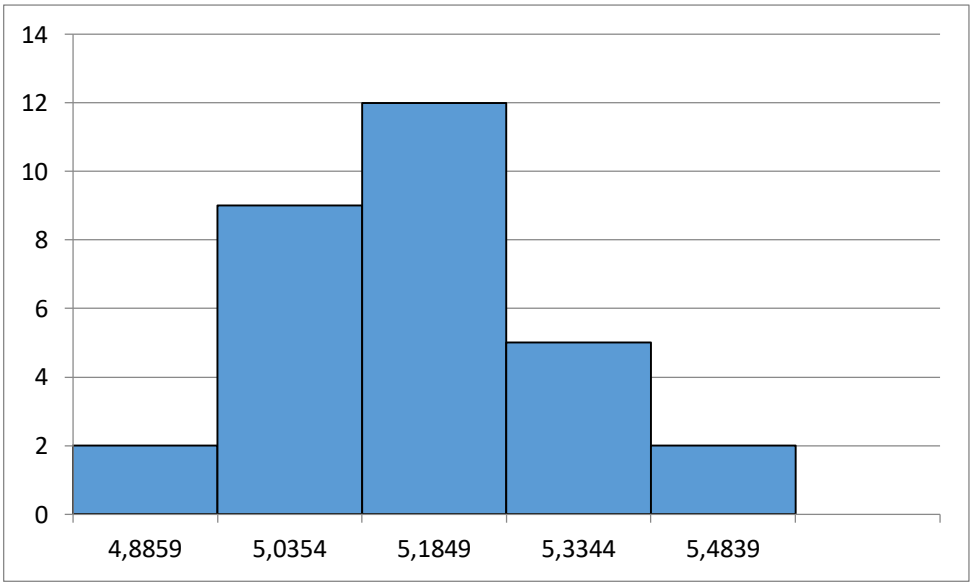

Рисунок 2.2.б – Нормалізований розподіл для вибірки X

Лінійна регресійна модель для нормалізованих даних Z_X, Z_Y побудована згідно(2.5).

Значення коефіцієнтів розраховані за методом найменших квадратів: b_0 = -2,3372, b_1 = 1,3835.

Отримали рівняння:

$$\hat{Z}_Y = -2{,}3372 + 1{,}3835 Z_X + \varepsilon.$$

Для отриманого лінійного рівняння знайшли залишки. Було перевірено, що залишки нормально розподілені.

Для нормалізованих даних після побудови регресійного рівняння будуємо довірчий інтервал згідно (2.6) та інтервал передбачення згідно (2.7).

Через зворотне перетворення (2.8) будуємо нелінійну регресійну модель для оцінювання трудомісткості розробки логістичних web-сервісів на Python. $\hat{Y}$ набуде вигляду:

$$\hat{Y} = 10^{-2{,}3372+\varepsilon} * X^{1{,}3835}.$$

Для порівняльного аналізу побудовано лінійну регресійну модель для оцінювання трудомісткості розробки логістичних web-сервісів на Python для вихідних даних у припущенні про нормальність їх розподілу. Вона має вигляд

$$\hat{Y} = -46144{,}67 \ + 0{,}7745 + \varepsilon.$$

Для цієї моделі також були побудовані довірчі інтервали та інтервали прогнозування.

Функція нелінійної регресії, її довірчі інтервали та інтервали прогнозування наведені в таблиці 2.2.

Таблиця 2.2 – Функція нелінійної регресії, її довірчі інтервали та інтервали прогнозування

№ п/п	Нелінійна регресія	Довірчий інтервал (ліва)	Довірчий інтервал (права)	Інтервал прогноз. (ліва)	Інтервал прогноз. (права)
1	2	3	4	5	6
1	26456,5974	18069,4097	38736,8242	10580,0658	66157,5797
2	40754,8507	31476,5452	52768,1117	17030,3545	97529,2589
3	42979,8633	33655,9279	54886,8733	18031,7989	102445,0558
4	44132,5418	34790,6211	55982,9397	18549,8747	104997,0024
5	54113,2313	44680,1856	65537,8164	23009,1159	127264,4204
6	54364,7273	44928,8993	65782,2385	23120,7763	127829,7721
7	56654,9889	47187,6785	68021,7351	24135,8582	132988,3418
8	56892,0424	47420,7256	68255,0604	24240,7403	133523,3347
9	57359,8923	47880,1888	68716,4636	24447,6334	134579,7853
10	64791,2097	55061,5523	76240,1473	27715,0134	151466,6727
11	65099,5443	55353,5630	76561,4794	27849,7923	152171,7154
12	69847,4865	59774,4624	81617,9883	29917,0546	163073,2512
13	68926,5089	58928,6540	80620,6031	29517,2639	160952,0325
14	69547,9994	59500,0699	81292,7486	29787,1131	162383,1153
15	73059,8502	62676,8465	85162,8953	31306,9441	170497,0535
16	73441,7718	63016,8189	85591,3381	31471,7147	171382,2682
17	76219,2950	65455,7440	88752,8057	32666,9631	177836,5778
18	79504,5763	68263,5547	92596,6672	34073,7936	185508,4800
19	79512,3258	68270,0791	92605,8682	34077,1032	185526,6250
20	80189,9148	68838,7704	93412,8021	34366,3252	187114,0544
21	95906,7034	81108,6498	113404,6219	40986,1413	224419,6565
22	98169,7081	82746,9194	116467,0741	41925,4981	229867,0742
23	104649,4210	87292,7162	125457,2180	44596,4526	245568,8891
24	112505,2652	92555,5122	136755,0607	47798,2238	264809,7292
25	115027,8952	94195,8948	140467,0203	48818,0910	271035,1102
26	123010,5286	99250,8912	152457,9776	52019,5984	290882,4867
27	123537,7858	99578,1275	153262,4173	52229,7043	292201,2431
28	137699,8903	108106,5198	175394,2299	57812,5200	327978,4346
29	179197,8679	130985,0151	245156,8664	73556,3855	436561,3623
30	286278,9953	181988,7994	450333,5559	110868,4430	739215,4246

Функція лінійної регресії, її довірчі інтервали та інтервали прогнозування наведені в таблиці 2.3.

Таблиця 2.3 – Функція лінійної регресії, її довірчі інтервали та інтервали прогнозування

№ п/п	Лінійна регресія	Довірчий інтервал (ліва)	Довірчий інтервал (права)	Інтервал прогноз. (ліва)	Інтервал прогноз. (права)
1	2	3	4	5	6
1	13407,7381	-7817,2088	34632,6850	-56841,9137	83657,3900
2	35237,2689	17892,5068	52582,0310	-32987,8180	103462,3558
3	38424,9210	21600,0247	55249,8173	-29555,8569	106405,6988
4	40058,2442	23493,8525	56622,6358	-27802,5856	107919,0740
5	53745,1671	39176,5149	68313,8194	-13252,5841	120742,9184
6	54080,5057	39555,8890	68605,1223	-12899,3326	121060,3439
7	57114,8159	42976,3398	71253,2919	-9710,0692	123939,7009
8	57426,9208	43326,8627	71526,9788	-9382,7567	124236,5982
9	58041,8371	44016,7369	72066,9374	-8738,2778	124821,9520
10	67631,8992	54637,9001	80625,8983	1243,6838	134020,1146
11	68022,9984	55065,0638	80980,9330	1647,9891	134398,0077
12	73982,4208	61508,9463	86455,8954	7781,4989	140183,3428
13	72835,4547	60278,5560	85392,3533	6604,9954	139065,9140
14	73609,9085	61109,8702	86109,9468	7399,6013	139820,2158
15	77950,7224	65727,6258	90173,8189	11837,3451	144064,0997
16	78419,2670	66221,7009	90616,8330	12314,7231	144523,8108
17	81806,7281	69767,5004	93845,9559	15756,5821	147856,8741
18	85769,6085	73855,4373	97683,7797	19761,9551	151777,2619
19	85778,9019	73864,9461	97692,8577	19771,3214	151786,4825
20	86590,5296	74693,9603	98487,0988	20588,8236	152592,2355
21	104921,8523	92667,2115	117176,4932	38797,5370	171046,1677
22	107489,9413	95074,6091	119905,2736	41309,4967	173670,3860
23	114754,3185	101753,6422	127754,9948	48363,6542	181144,9827
24	123396,4490	109476,4368	137316,4613	56657,5363	190135,3618
25	126135,6923	111880,5064	140390,8783	59264,3757	193007,0090
26	134696,5052	119279,4554	150113,5550	67344,0396	202048,9708
27	135256,4354	119757,9761	150754,8947	67868,9786	202643,8921
28	150059,3464	132215,4878	167903,2050	81593,5735	218525,1193
29	191208,4030	165637,4828	216779,3233	118297,5674	264119,2387
30	286861,1987	240982,8594	332739,5380	197228,4445	376493,9528

Якість отриманих рівнянь регресії перевірили через підрахунок коефіцієнтів детермінації R^2 (1.1), середньої величини відносної похибки $MMRE$ (1.4) та рівня прогнозування $PRED(l)$ (1.5).

Результати перевірки якості нелінійної моделі: коефіцієнт детермінації R^2 = 0,8122; середня величина відносної похибки $MMRE$ = 0,2915; рівень прогнозування $PRED(0{,}25)$ = 0,7333.

Результати перевірки якості лінійної моделі: коефіцієнт детермінації R^2 = 0,7962; середня величина відносної похибки $MMRE$ = 0,2881; рівень прогнозування $PRED(0{,}25)$ = 0,6667.

Порівнюючи результати, бачимо, що по переважній більшості показників кращу якість має нелінійна модель.

Будуємо довірчий інтервал, інтервал прогнозування та рівняння регресії для ненормалізованих даних (рис. 2.3).

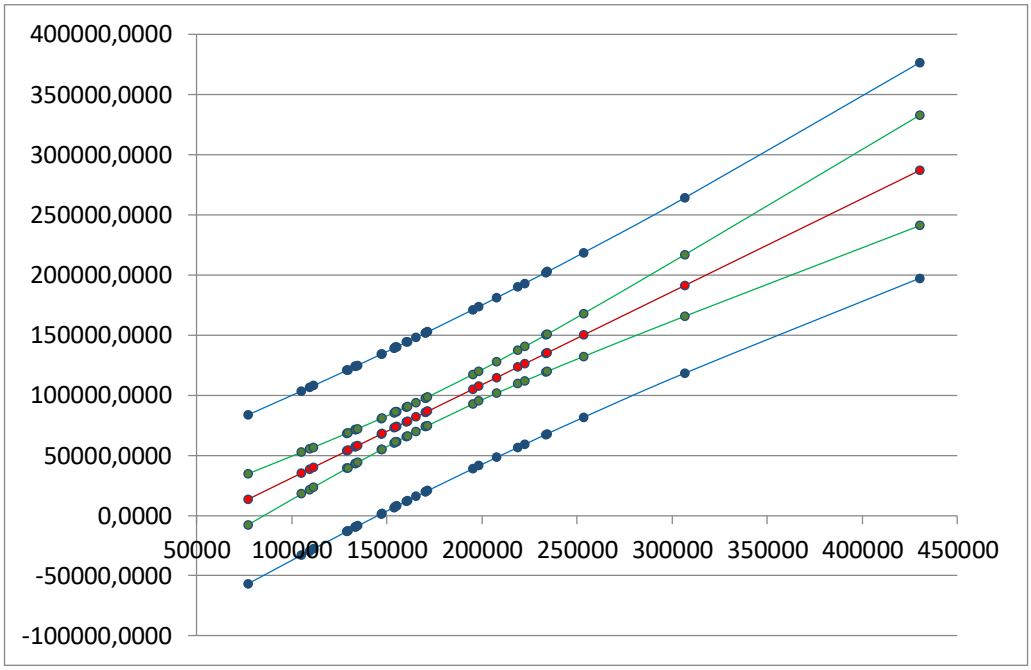

Рисунок 2.3 – Довірчий інтервал, інтервал прогнозування та рівняння регресії для ненормалізованих даних

Будуємо довірчий інтервал, інтервал прогнозування та рівняння регресії для нормалізованих даних (рис. 2.4).

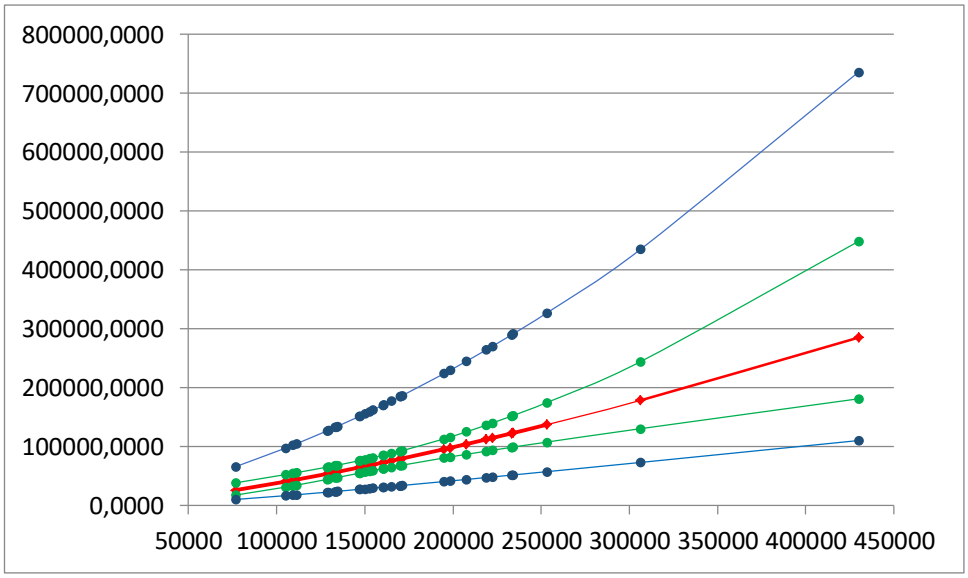

Рисунок 2.4 – довірчий інтервал, інтервал прогнозування та рівняння регресії для нормалізованих даних

Таким чином, прогнозування для нормалізованих даних дає кращий результат, ніж для ненормалізованих.

2.3 Постановка задачі на розробку ПЗ для оцінювання трудомісткості розробки логістичних web-сервісів на Python

За результатами розробки нелінійної регресійної моделі сформулюємо задачу на розробку програмного забезпечення для оцінювання трудомісткості розробки логістичних web-сервісів на Python, яке буде реалізовувати побудовану модель.

Необхідно розробити програмне забезпечення з наступним функціоналом:

1) Введення зібраних даних після перед обробки;

2) Нормалізація зібраних даних;

3) Розробка лінійної моделі для отриманих гаусівських даних;

4) Знаходження довірчих інтервалів для лінійної моделі;

5) Знаходження інтервалів прогнозування для лінійної моделі;

6) Побудова нелінійної моделі для вихідних даних, її довірчих інтервалів та інтервалів прогнозування;

7) Перевірка якості побудованої моделі;

8) Відображення інтервалів довіри та прогнозування;

9) Вилучення з екрану інтервалів довіри та прогнозування;

10) Прогнозування даних.

3 ПРОЕКТ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ОЦІНЮВАННЯ ТРУДОМІСТКОСТІ РОЗРОБКИ ЛОГІСТИЧНИХ WEB-СЕРВІСІВ НА PYTHON

Проаналізувавши предметну галузь і взявши до уваги вимоги, що ставляться до розроблюваного програмного забезпечення, перейдемо до проектування і реалізації поставлених завдань.

При розробці проекту ПЗ використовується об'єктно-орієнтована методологія та мова моделювання UML. В якості CASE-засобу застосовувалась платформа LucidChart.

В даний час UML є візуальною мовою моделювання, яка дозволяє системним архітекторам представити своє бачення системи в стандартній і легкій для розуміння формі. Також UML надає можливість спільно використовувати проектні рішення і взаємодіяти розробникам між собою та з замовниками.

3.1 Ескізній проект програмного забезпечення для оцінювання трудомісткості розробки логістичних web-сервісів на Python

В ескізному проекті представлені діаграма варіантів використання, специфікації варіантів використання, діаграми діяльності для основних варіантів використання, проект користувальницького інтерфейсу а також концептуальна модель даних предметної галузі.

3.1.1 Модель варіантів використання ПЗ для оцінювання трудомісткості розробки логістичних web-сервісів на Python

Щоб зрозуміти і правильно спроектувати майбутню систему, потрібно насамперед визначити, що вона робитиме, тобто необхідно створити

проекцію, яка називається функціональною моделлю. Першим кроком в описі функціональності системи є моделювання вимог до неї.

Метою аналізу та моделювання вимог є:

– досягнення угоди між розробниками, замовниками та користувачами про те, що має робити ПЗ;

– досягнення кращого розуміння розробниками поведінки ПЗ;

– обмеження системної функціональності;

– створення базису для планування розробки проекту;

– визначення інтерфейсу користувача.

Досягнення цієї мети забезпечується за допомогою побудови діаграми варіантів використання (прецедентів). Діаграма прецедентів являється концептуальним представленням системи при її проектуванні та розробці. Діаграма містить акторів, які взаємодіють із системою, що розробляється, через варіанти використання. В якості акторів можуть виступати об’єкти, суб’єкти чи інші системи, які являються зовнішніми по відношенню до модельованої. З іншого боку, прецедент – це сервіс, який надає модельована система актору. Однак діаграма варіантів використання не надає інформації про те, як буде реалізований варіант використання.

Для визначення та аналізу вимог до програмного забезпечення для оцінювання трудомісткості розробки логістичних web-сервісів на Python розробимо діаграму варіантів використання.

Побудову діаграми почнемо з визначення акторів та варіантів використання.

Актором в даній системі буде особа, що працює з програмним забезпеченням. Її названо «Користувач». Опис актора представлено в таблиці 3.1.

Таблиця 3.1 – Опис актора програмного забезпечення для оцінювання трудомісткості розробки логістичних web-сервісів на Python

Актор	Сервіси модельованої системи
Користувач	Користувачу надаються наступні сервіси: імпорт даних, нормалізація вихідних даних, обчислення параметрів регресійної моделі, побудова регресійної моделі, аналіз регресійної моделі, обчислення показників якості регресійної моделі, обчислення інтервалів прогнозу та довіри, оцінювання трудомісткості.

Розглянемо більш детально сервіси модельованої системи, які на діаграмі представляються варіантами використання (табл. 3.2).

Таблиця 3.2 – Короткий сервісів модельованої системи

Актор	Сервіс	Короткий опис сервісу
Користувач	Імпорт даних	Сервіс реалізує введення вихідних даних з файлу в систему
Користувач	Нормалізація вихідних даних	Сервіс реалізує приведення вихідних даних до нормальної форми
Користувач	Обчислення параметрів регресійної моделі	Сервіс відповідає за обчислення параметрів регресійної моделі
Користувач	Побудова регресійної моделі	Сервіс відповідає за побудову регресійної моделі
Користувач	Аналіз регресійної моделі	Сервіс відповідає за аналіз регресійної моделі
Користувач	Обчислення показників якості регресійної моделі	Сервіс відповідає за обчислення коефіцієнту детермінації, середньої величини відносної похибки, рівня прогнозування
Користувач	Обчислення інтервалів прогнозу та довіри	Сервіс відповідає за побудову інтервалів довіри та передбачення для побудованої регресійної моделі
Користувач	Оцінювання трудомісткості	Сервіс відповідає за оцінювання трудомісткості розробки ПЗ

Таким чином варіанти використання призначені в першу чергу для визначення функціональних вимог до системи і керують усім процесом розробки. Діаграма варіантів використання представлена на рис. 3.1.

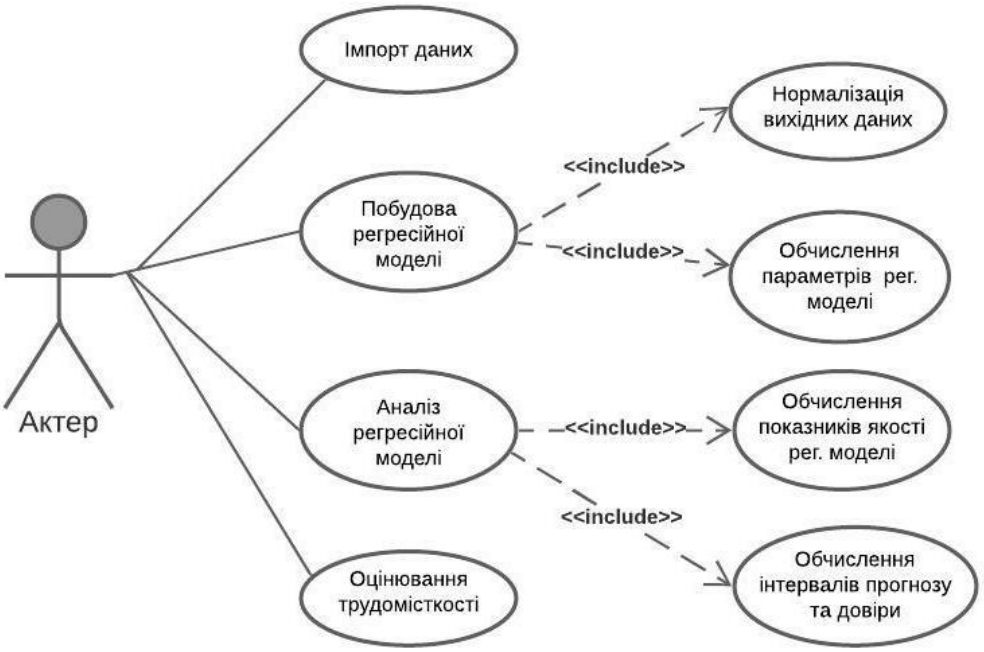

Рисунок 3.1 – Модель варіантів використання ПЗ для оцінювання трудомісткості розробки логістичних web-сервісів на Python

У структурному підході аналогом діаграми варіантів використання є діаграми IDEF0 та DFD, варіантів використання – роботи (IDEF0) та процеси (DFD), акторів – зовнішні сутності (DFD).

3.1.2 Специфікації варіантів використання ПЗ для оцінювання трудомісткості розробки логістичних web-сервісів на Python

Для уточнення моделі варіантів використання, представленої на рисунку 3.1, розробимо специфікації варіантів використання.

В таблиці 3.3 наведено специфікацію варіанту використання «Нормалізація вихідних даних».

Таблиця 3.3 – Специфікація варіанту використання «Нормалізація вихідних даних»

Розділ	Опис
Короткий опис	Відбувається приведення вихідних даних до нормальної форми
Передумова	Виконання варіанту використання «Імпорт даних»
Базовий потік подій	Нормалізація емпіричних даних за допомогою нормалізуючого перетворення десятковий логарифм
Альтернативні потоки подій	Не виявлені
Після умови	Не виявлені

В таблиці 3.4 наведено специфікацію варіанту використання «Імпорт даних»

Таблиця 3.4 – Специфікація варіанту використання «Імпорт даних»

Розділ	Опис
Короткий опис	Відбувається введення вихідних даних з файлу в систему.
Передумова	Немає.
Базовий потік подій	Знаходиться та обирається в файловій системі необхідний файл. Зчитуються та аналізуються дані з файлу.
Альтернативні потоки подій	При зчитуванні даних з файлу, у випадку неправильного формату, видається повідомлення про неправильний формат.
Після умова	Якщо формат даних у файлі правильний, то вони вводяться з файлу для подальшої обробки.

В таблиці 3.5 наведено специфікацію варіанту використання «Обчислення параметрів регресійної моделі»

Таблиця 3.5 – Специфікація варіанту використання «Обчислення параметрів регресійної моделі»

Розділ	Опис
Короткий опис	Відбувається обчислення параметрів лінійної регресійної моделі.
Передумова	Введення даних з файлу та їх нормалізація.
Базовий потік подій	Обчислюються параметри лінійної регресії з умови мінімізації суми квадратів відхилень.
Альтернативні потоки подій	Не виявлені.
Після умови	Не виявлені.

В таблиці 3.6 наведено специфікацію варіанту використання «Побудова регресійної моделі»

Таблиця 3.6 – Специфікація варіанту використання «Побудова регресійної моделі»

Розділ	Опис
Короткий опис	Відбувається побудова регресійної моделі.
Передумова	Виконання варіанту використання «Імпорт даних».
Базовий потік подій	Приводяться вихідні дані до нормальної форми. Обчислюються параметри лінійної регресійної моделі. Будується лінійна регресійна модель. Будується нелінійна регресійна модель через зворотне перетворення
Альтернативні потоки подій	Не виявлені
Після умови	Не виявлені.

В таблиці 3.7 наведено специфікацію варіанту використання «Аналіз регресійної моделі»

Таблиця 3.7 – Специфікація варіанту використання «Аналіз регресійної моделі»

Розділ	Опис
Короткий опис	Відбувається аналіз регресійної моделі.
Передумова	Альтернативні потоки подій
Базовий потік подій	Обчислюються показники якості регресійної моделі Обчислюються інтервали прогнозу та довіри
	Не виявлені
Після умови	Не виявлені.

В таблиці 3.8 наведено специфікацію варіанту використання «Обчислення показників якості регресійної моделі»

Таблиця 3.8 – Специфікація варіанту використання «Обчислення показників якості регресійної моделі»

Розділ	Опис
Короткий опис	Відбувається обчислення показників якості регресійної моделі».
Передумова	Виконання варіанту використання «Побудова регресійної моделі»
Базовий потік подій	Обчислюється коефіцієнт детермінації. Обчислюється середня величина відносної похибки. Обчислюється рівень прогнозування. Виводяться результати на екран.
Альтернативні потоки подій	Не виявлені.
Після умови	Не виявлені.

В таблиці 3.9 наведено специфікацію варіанту використання «Обчислення інтервалів прогнозу та довіри»

Таблиця 3.9 – Специфікація варіанту використання «Обчислення інтервалів прогнозу та довіри»

Розділ	Опис
Короткий опис	Відбувається обчислення інтервалів прогнозу та довіри.
Передумова	Виконання варіанту використання «Побудова регресійної моделі».
Базовий потік подій	Будуються інтервали довіри для побудованої лінійної регресійної моделі Будуються інтервали передбачення для побудованої лінійної регресійної моделі Будуються інтервали довіри для побудованої нелінійної регресійної моделі через зворотне перетворення. Будуються інтервали передбачення для побудованої нелінійної регресійної моделі через зворотне перетворення. По запиту користувача інтервали довіри та передбачення виводяться на екран.
Альтернативні потоки подій	Не виявлені.
Після умови	Не виявлені.

В таблиці 3.10 наведено специфікацію варіанту використання «Оцінювання трудомісткості»

Таблиця 3.10 – Специфікація варіанту використання «Оцінювання трудомісткості»

Розділ	Опис
Короткий опис	Виконується оцінювання трудомісткості розробки ПЗ.
Передумова	Виконання варіанту використання «Побудова регресійної моделі».
Базовий потік подій	Вводиться розмір програмного забезпечення для оцінювання трудомісткості розробки. Проводиться оцінювання трудомісткості. Будується інтервал довіри для оцінюваного значення Будується інтервал передбачення для оцінюваного значення .
Альтернативні потоки подій	Не виявлені.
Після умови	Не виявлені.

3.1.3 Сценарії основних варіантів використання ПЗ для оцінювання трудомісткості розробки логістичних web-сервісів на Python

До будь-якого елементу моделі з динамічною поведінкою можемо додати діаграму діяльності – саме для уточнення цієї динаміки. Як приклад, що добре підходить за контекстом, слід згадати можливість застосування діаграм активності для опису варіантів використання.

На рисунку 3.2 представлена діаграма діяльності для варіанту використання «Обчислення показників якості регресійної моделі» програмного забезпечення для оцінювання трудомісткості розробки логістичних web-сервісів на Python.

Рисунок 3.2 – Діаграма діяльності для варіанту використання «Обчислення показників якості регресійної моделі» ПЗ для оцінювання трудомісткості розробки логістичних web-сервісів на Python

На рисунку 3.3 зображена діаграма діяльності для варіанту використання «Імпорт даних» програмного забезпечення для оцінювання трудомісткості розробки логістичних web-сервісів на Python.

Рисунок 3.3 – Діаграма діяльності для варіанту використання «Імпорт даних» ПЗ для оцінювання трудомісткості розробки логістичних web-сервісів на Python

3.1.4 Концептуальна модель даних ПЗ для оцінювання трудомісткості розробки логістичних web-сервісів на Python

У інформаційних моделях даних зазвичай виділяють до трьох рівнів деталізації:

– Концептуальна модель даних – схема найвищого рівня з мінімальною кількістю подробиць. Перевага цього підходу полягає у можливості відобразити загальну структуру моделі та всю архітектуру системи.

– Логічна модель даних містить більш детальну інформацію, ніж концептуальна модель. На цьому рівні визначаються докладніші операційні

та транзакційні сутності. Логічна модель не залежить від технології, в якій вона застосовуватиметься.

– Фізична модель даних: на основі кожної логічної моделі даних будується фізична модель. В останньої має бути достатньо технічних подробиць для складання та впровадження самої бази даних.

В результаті аналізу предметної галузі програмного забезпечення для оцінювання трудомісткості розробки логістичних web-сервісів на Python була побудована концептуальна модель даних, яка наведена на рисунку 3.4.

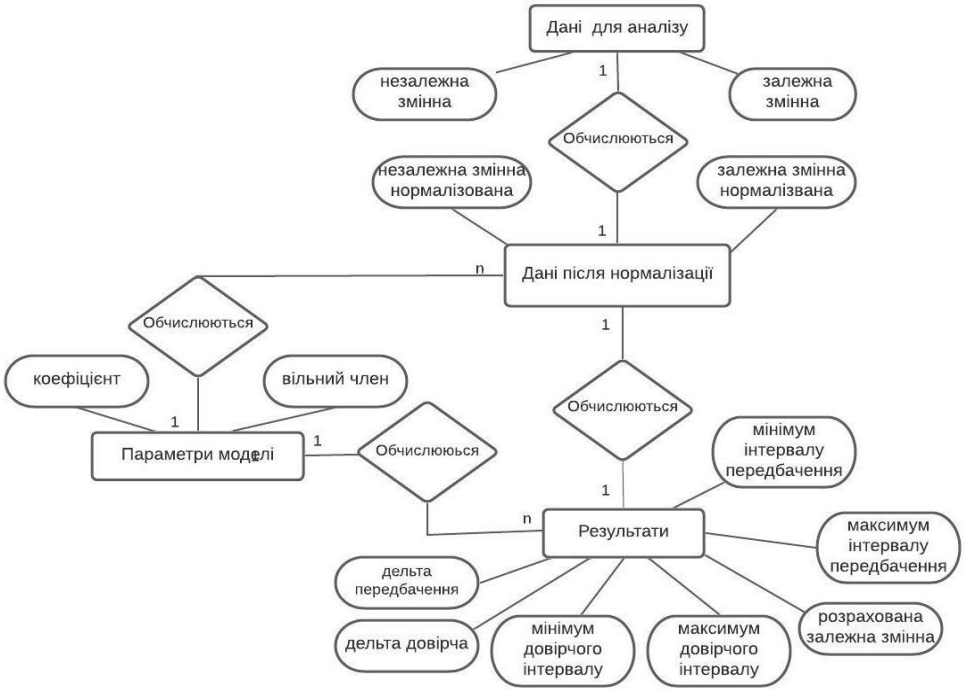

Рисунок 3.4 – Концептуальна модель ПЗ для оцінювання трудомісткості розробки логістичних web-сервісів на Python

3.1.5 Проект інтерфейсу ПЗ для оцінювання трудомісткості розробки логістичних web-сервісів на Python

Інтерфейс користувача – спосіб та засоби взаємодії користувача з програмами. Інтерфейс користувача – це обов'язкова складова сучасного

програмного забезпечення, що орієнтоване на роботу кінцевих користувачів. При розробці інтерфейсу висуваються вимоги як з інженерного напрямку розробки, так і з дизайнерського. Також при розробці інтерфейсу враховують можливості та кваліфікацію кінцевого користувача.

Якісно спроектований інтерфейс повинен мати наступні властивості:

– простота;

– гнучкість;

– естетична привабливість;

– узгодженість в межах продукту.

Проект головного вікна програмного проекту наведений на рисунку 3.5.

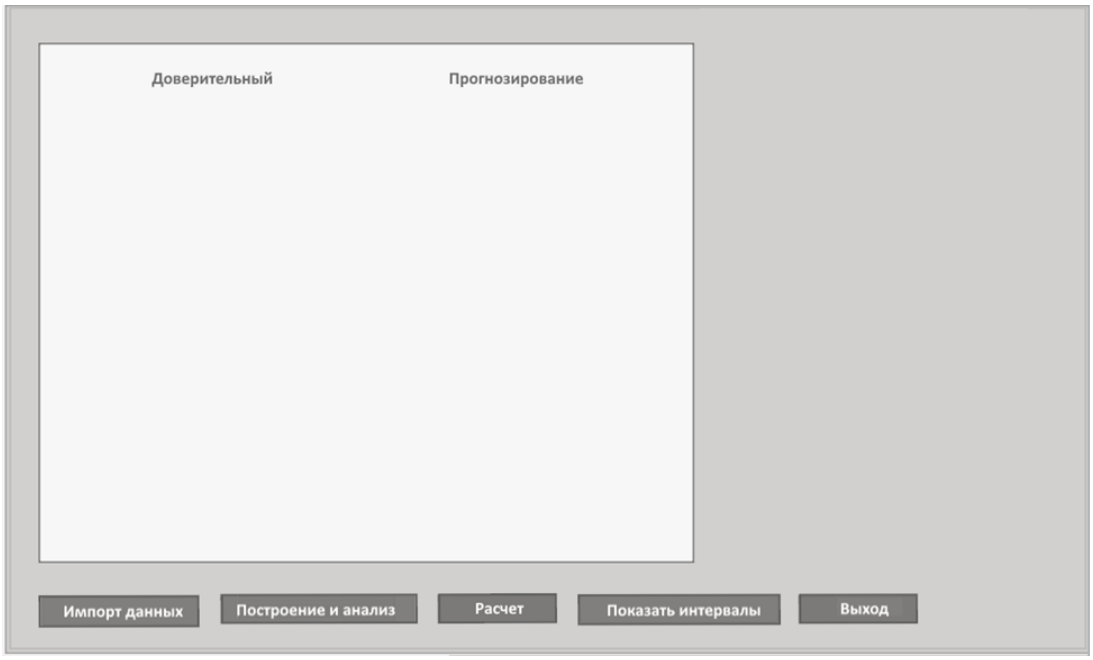

Рисунок 3.5 – Проект інтерфейсу ПЗ для оцінювання трудомісткості розробки логістичних web-сервісів на Python

Вважається, що при професійно розробленому інтерфейсі у користувача повинна скластися уява, що він керує програмним забезпеченням, а не навпаки.

3.2 Технічний проект програмного забезпечення для оцінювання трудомісткості розробки логістичних web-сервісів на Python

При розробці технічного проекту остаточно виявляються технічні рішення, які створюють повне представлення про структуру системи. Технічний проект ПЗ базується на ескізному проекті та включає в себе статичну та динамічну моделі. Статичну модель представляємо діаграмою класів, а динамічну – діаграмою послідовності.

3.2.1 Діаграма класів ПЗ для оцінювання трудомісткості розробки логістичних web-сервісів на Python

З допомогою діаграми класів представляють статичну структуру модельованої системи в термінах класів ООП. З допомогою діаграми класів можна представляти зв’язки між об'єкти та підсистеми, які є сутностями преметної галузі, та описувати їх внутрішню структуру та типи відношень.

На діаграмі класів представляються класи, інтерфейси, кооперації та відношення між ними. Діаграму зображують як безліч вершин і дуг.

На рисунку 3.6 представлено діаграму класів ПЗ для оцінювання трудомісткості розробки логістичних web-сервісів на Python

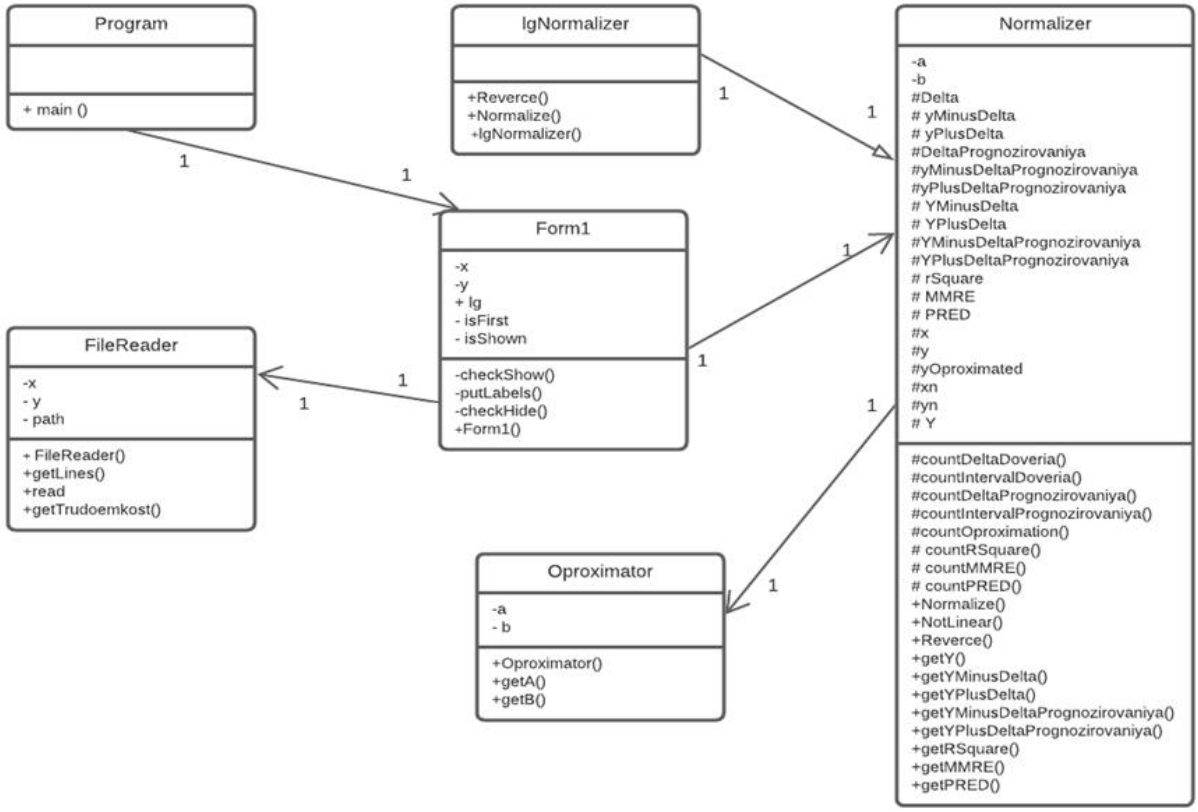

Рисунок 3.6 – Діаграма класів ПЗ для оцінювання трудомісткості розробки логістичних web-сервісів на Python

Діаграма класів не містить інформацію про часові рамки роботи програми. В даному аспекті діаграму класів розглядають як подальший розвиток концептуальної моделі програмного забезпечення, системи.

3.2.2 Специфікації класів ПЗ для оцінювання трудомісткості розробки логістичних web-сервісів на Python

Розглянемо специфікації класів програмного забезпечення.

Клас lgNormalizer

Служить для нормалізації за допомогою десяткового логарифму та зворотнього перетворення вихідних даних.

Методи класу:

lgNormalizer() – конструктор класу

Normalize() – метод нормалізації вихідних даних десятковим логарифмом

Reverse() – метод зворотнього перетворення до вихідних даних

Клас Program

Служить для запуску програми на виконання

Методи класу:

Main() – метод запуску програми на виконання

Клас Form1

Служить для реалізації програмного інтерфейсу.

Атрибути класу:

x – значення розміру програмних проектів (массив)

y – значення трудомісткості програмних проектів (массив)

isShown – ознака відображення таблиці з інтервалами

lg – величина, в якій зберігається екземпляр класу lgNormalizer

Методи класу:

Form1() – конструктор класу

putLabels() – метод відображення показників якості

showIntervals() – метод відображення довірчих інтервалів та інтервалів прогнозування

checkShow() – метод перевірки відображення інтервалів на екрані

checkHide() – метод перевірки не відображення інтервалів на екрані

Клас Oproximator

Служить для знаходження параметрів регресійної моделі з допомогою МНК.

Атрибути класу:

a –коефіцієнт регресії

b – вільний член регресії

Методи класу:

Oproximator() – конструктор класу

getA() – метод визначення коефіцієнту регресії

getB() – метод визначення вільного члену регресії

Клас Normalizer

Служить для виконання розрахунку

Атрибути класу:

x – значення розміру програмних проектів (массив)

y – значення трудомісткості програмних проектів (массив)

yOproximated – апроксимовані y

xn – нормалізовані значення розміру програмних проектів

yn – значення трудомісткості програмних проектів

delta – Δ довіри

yMinusDelta – мінімум інтервалу довіри

yPlusDelta – максимум інтервалу довіри

deltaPrognozirovaniya – Δ інтервалу прогнозу

yMinusDeltaPrognozirovaniya – мінімум інтервалу передбачення

yPlusDeltaPrognozirovaniya – мінімум інтервалу передбачення

Y – зворотнє перетворення yOproximated

YMinusDelta – мінімум інтервалу довіри після зворотнього перетворення

YPlusDelta – максимум інтервалу довіри після зворотнього перетворення

YMinusDeltaPrognozirovaniya – мінімум інтервалу передбачення після зворотнього перетворення

YPlusDeltaPrognozirovaniya – максимум інтервалу передбачення після зворотнього перетворення

a –коефіцієнт регресії

b – вільний член регресії

rSquare – величина R^2

MMRE – величина MMRE

PRED – величина PRED(25)

Методи класу:

Normalizer() – консруктор класу

countOproximation() – метод знаходження обчислених y

countDeltaDoveria() – метод знаходження Δ довіри

countIntervalDoveria() – метод знаходження інтервалу довіри

countDeltaPrognozirovaniya() – метод знаходження Δ передбачення

countIntervalPrognozirovaniya() – метод знаходження прогнозного інтервалу

countRSquare() – метод знаходження R^2

countMMRE() – метод знаходження MMRE

countPRED() – метод знаходження PRED(25)

getY() – метод знаходження y після зворотнього перетворення

getYMinusDelta() – метод знаходження мінімум інтервалу довіри після зворотнього перетворення

getYPlusDelta() – метод знаходження максимуму інтервалу довіри після зворотнього перетворення

getYMinusDeltaPrognozirovaniya() – метод знаходження мінімум інтервалу передбачення після зворотнього перетворення

getYPlusDeltaPrognozirovaniya() – метод знаходження максимуму інтервалу інтервалу передбачення після зворотнього перетворення

getRSquare() – метод знаходження R^2

getMMRE() – метод знаходження MMRE

Клас fileReader

Служить для імпорту даних з файлу.

Атрибути класу:

x – зберігає id проекту

y – зберігає ім’я програми

path – призначений для зберігання шляху до файлу з даними

Методи класу:

fileReader() – конструктор класу

getLines() – метод зчитування значень розміру

getTrudoemkost() –метод зчитування значень трудомісткості

read() – метод зчитування інформації з файлу

3.2.3 Динамічна модель ПЗ для оцінювання трудомісткості розробки логістичних web-сервісів на Python

На основі моделі сценаріїв варіантів використання і статичної моделі програми була створена динамічна модель ПЗ, яка представлена діаграмою послідовності (рис.3.7). На діаграмі послідовності зображена взаємодія користувача з системою у часі. Діаграма послідовності використовується для представлення екземплярів об'єктів та повідомлень, якими обмінюються екземпляри в межах варіанту використання.

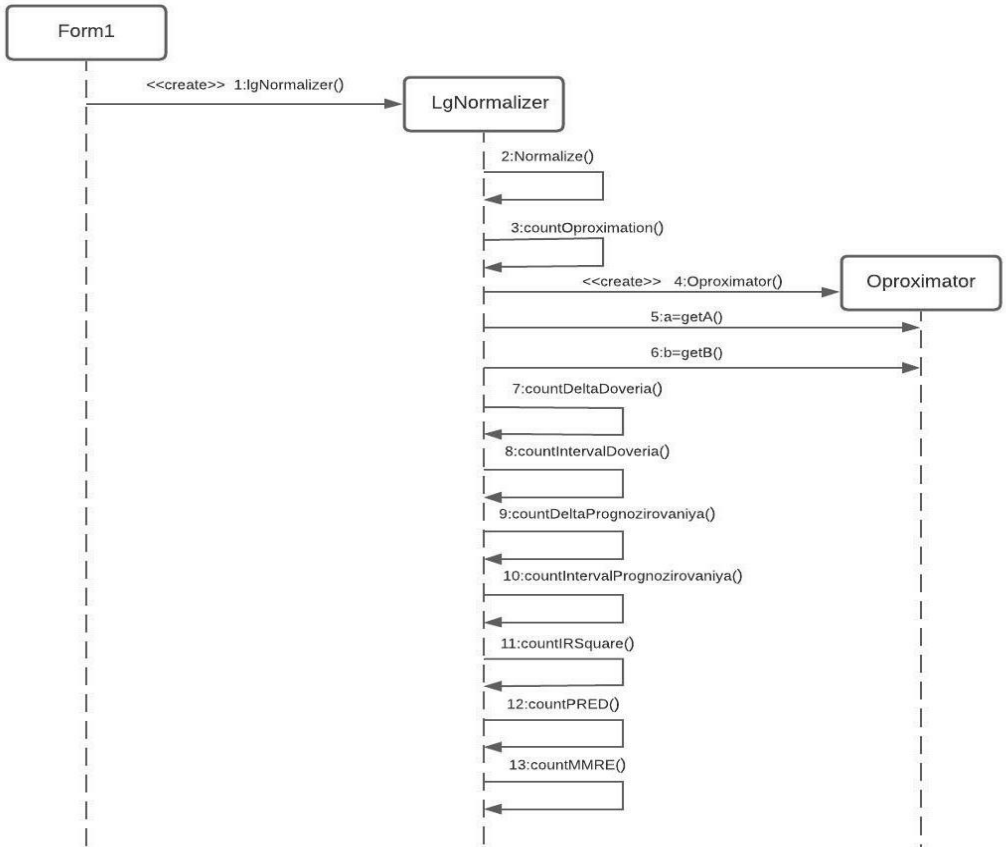

Рисунок 3.7 – Динамічна модель програмного забезпечення

З допомогою діаграми послідовності можна представити поведінку об'єктів для одного варіанту використання. З допомогою діаграми послідовності зручно представляти взаємодію об'єктів, але не точне визначення їх поведінки.

3.3 Робочий проект програмного забезпечення для оцінювання трудомісткості розробки логістичних web-сервісів на Python

На етапі робочого проектування перетворюються у форму, що доступна користувачеві, всі рішення, які були прийняті на попередніх етапах. Також на цьому етапі відбувається вибір засобів розробки ПЗ, розробляється діаграма компонентів, проводиться реалізація, тестування та випробування програмного забезпечення.

Докладна розробка алгоритмів складових комплексу має бути повністю закінчена перед початком їх реалізації на мові програмування.

3.3.1 Обґрунтування вибору мови та засобів розробки ПЗ для оцінювання трудомісткості розробки логістичних web-сервісів на Python

Програмне забезпечення проектувалося під розробку на ООП. Найбільш поширеними об’єктно-орієнтованими мовами на даному етапі є Java та C#.

На даний момент вважається, що Java не розвивається.

C# – об'єктно- і компонентно-орієнтована мова програмування. У C# Є такі мовні конструкції, які підтримують таку концепцію роботи. Саме тому C# використовують для роботи з програмними компонентами. З моменту створення мова C# збагатилася функціями для підтримки нових робочих навантажень та сучасними рекомендаціями щодо розробки ПЗ. C# – це об'єктно-орієнтована мова програмування.

Розглянемо декілька важливих функцій C#:

– Збирання сміття (автоматично звільняє пам'ять, зайняту недоступними об'єктами, що не використовуються).

– Типи, які допускають значення null (забезпечують захист від змінних, що мають посилаються на виділені об'єкти).

– Обробка винятків (структурує та розширює підхід до знаходження помилок та відновлення після них).

– Лямбда-вирази (для підтримки прийомів функціонального програмування).

– LINQ (створює спільний шаблон роботи з інформацією з різних джерел).

– Підтримка мов для асинхронних операцій (забезпечує синтаксис для створення розподілених систем).

– В C# діє єдина система типів (всі типи C#, включаючи типи-примітиви, такі як int та double, успадковують від одного кореневого типу object). Всі типи використовують загальний набір операцій, а значення будь-якого типу можна зберігати, передавати та обробляти подібним чином.

– C# підтримує визначені користувачами типи посилань і типи значень.

– C# дозволяє динамічно виділяти об'єкти та зберігати спрощені структури у стеку.

– C# підтримує універсальні методи та типи, що забезпечують підвищену безпеку типів та продуктивність.

– C# надає літератори (дозволяють розробникам класів колекцій визначати варіанти поведінки для клієнтського коду).

У C# приділяється велика увага керуванню версіями для забезпечення сумісності програм та бібліотек у разі їх зміни. Питання управління версіями істотно вплинули такі аспекти розробки C#, як роздільні модифікатори virtual і override, правила дозволу навантаження методів і підтримка явного оголошення членів інтерфейсу.

3.3.2 Діаграма компонентів програмного забезпечення для оцінювання трудомісткості розробки логістичних web-сервісів на Python

Діаграма компонентів призначена для візуалізації загальної організації структури вихідного коду програми.

З допомогою діаграми компонентів здійснюється перехід від логічного уявлення розроблюваного програмного забезпечення до фізичного у вигляді програмних компонентів. Причому одна частина програмних компонентів може існувати тільки на стадії компіляції, а інша – на стадії виконання. Діаграма компонентів може містити компоненти, класи, інтерфейси та зв‘язки між ними. Компонент представляє фізичну складову розроблюваної системи.

Діаграма компонентів програмного забезпечення для оцінювання трудомісткості розробки логістичних web-сервісів на Python представлена на рисунку 3.8

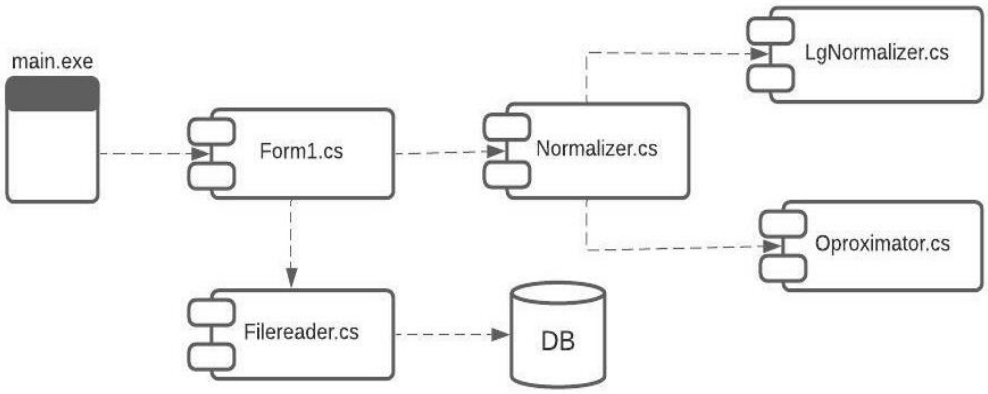

Рисунок 3.8 – Діаграма компонентів програмного забезпечення для оцінювання трудомісткості розробки логістичних web-сервісів на Python

Компоненти, які є файлами з вихідним кодом класів, бібліотеками, модулями тощо повинні мати узгоджений набір інтерфейсів.

3.3.3 Кодування та тестування програмного забезпечення для оцінювання трудомісткості розробки логістичних web-сервісів на Python

Програма реалізована на мові програмування C#. Код програми наведений в додатку D.

Тестування ПЗ використовується для перевірки якості програмного продукту. Тестування включає: розробку тестів, проведення тестування і аналіз отриманих результатів.

Частіше за все використовуються дві методики тестування: функціональна та структурна. Функціональне тестування – це тестування, яке проводиться для перевірки відповідності програмної системи або програмного модуля специфікованим функціональним вимогам.

Структурне тестування – це тестування, яке аналізує внутрішню логіку програми.

Розроблене ПЗ протестуємо одним із методів функціонального тестування, а саме, – розбиття на класи еквівалентності.

При розробці ПЗ було виконано тестування всіх функцій. В роботі приводиться тестування функцій «Імпорт даних» та «Оцінювання трудомісткості» методом «чорної скриньки».

В таблиці 3.11 представлені класи еквівалентності вхідних даних для функції «Оцінювання трудомісткості».

Таблиця 3.11 – Класи еквівалентності вхідних даних для функції «Оцінювання трудомісткості»

Вхідні умови	Вірні класи еквівалентності	Невірні класи еквівалентності
Кількість строк коду	1 Ціле число>0	2 Пустий рядок 3 Не число 4 Число<0

Таблиця 3.12 містить тести по вірним класам еквівалентності для функції «Оцінювання трудомісткості»

Таблиця 3.12 – Тести по вірним класам еквівалентності для функції «Оцінювання трудомісткості»

№ п/п	№ класу еквівалентності	Введені дані	Отриманий результат
1	1	100	Програма виконала оцінювання

Таким чином, за результатами тестування по вірним класам еквівалентності для функції «Оцінювання трудомісткості» програма працює правильно і без збоїв.

Таблиця 3.13 містить тести по невірним класам еквівалентності для функції «Оцінювання трудомісткості».

Таблиця 3.13 – Тести по невірним класам еквівалентності для функції «Оцінювання трудомісткості»

№ п/п	№ класу еквівалентності	Введені дані	Отриманий результат
1	2		Повідомлення про помилку
2	3	вв	Повідомлення про помилку
3	4	-3	Повідомлення про помилку

Отже, за результатами тестування невірних класів еквівалентності для функції «Оцінювання трудомісткості» програма працює правильно.

Розглянемо тестування функції «Імпорт даних».

Таблиця 3.14 – Класи еквівалентності для функції «Імпорт даних»

Вхідні умови	Вірні класи еквівалентності	Невірні класи еквівалентності
Імпортовані дані	1 Формат правильний	2 Інформація в файлі відсутня 3 Формат неправильний

Таблиця 3.15 містить тести по вірним класам еквівалентності для функції «Імпорт даних».

Таблиця 3.15 – Тести по вірним класам еквівалентності для функції «Імпорт даних»

№ п/п	№ класу еквівалентності	Введені дані	Отриманий результат
1	1	150 500	Інформація по новому проекту була записана до бази даних

Отже, за результатами тестування вірних класів еквівалентності для функції «Імпорт даних» програма працює правильно і без збоїв.

Таблиця 3.16 містить тести по невірним класам еквівалентності для функції «Імпорт даних»

Таблиця 3.16 – Тести по невірним класам еквівалентності для функції «Імпорт даних»

№ п/п	№ класу еквівалентності	Введені дані	Отриманий результат
1	2		Повідомлення про помилку
2	3	Вв аа	Повідомлення про помилку

Отже, за результатами тестування невірних класів еквівалентності «Імпорт даних» програма працює правильно.

3.3.4 Випробування програмного забезпечення для оцінювання трудомісткості розробки логістичних web-сервісів на Python

Програма та методика випробувань програмного забезпечення наведена в Додатку Г. Згідно програми і методики випробувань провели випробування програмного забезпечення для оцінювання трудомісткості розробки логістичних web-сервісів на Python.

– Випробування функції введення зібраних даних після передобробки

- Натиснули кнопку «Імпорт даних».
- В вікні, що відкрилося, обрали шлях до файлу.
- Відбулося відкриття файлу та з’явилася назви файлу поблизу кнопки «Імпорт даних».
- Натиснули кнопку «Обрати файл».

Очікуваний результат: після натискання кнопки «Обрати файл» файл додався.

– Випробування функції побудови моделі та її аналізу

- Прочитали дані з файлу.
- Натиснули на кнопку «Побудова та аналіз».

Очікуваний результат (рис. 3.9): після натискання кнопки «Побудова та аналіз» на головній формі зявилися результати розрахунку коефіцієнта детермінації R^2, середньої величини відносної похибки $MMRE$ та рівня прогнозування $PRED(0{,}25)$.

– Випробування функції «Прогнозування даних».

- Прочитали дані з файлу.
- Натиснули на кнопку «Побудова та аналіз».
- Натиснули на кнопку «Розрахунок».
- В поле «Строки коду» ввели розмір коду для прогнозування трудомісткості.
- Натиснули на кнопку «Оцінити».

Очікуваний результат (рис. 3.10): після натискання кнопки «Оцінити» на головній формі з’явилися результати прогнозу трудомісткості та відповідні довірчий інтервал та інтервал передбачення.

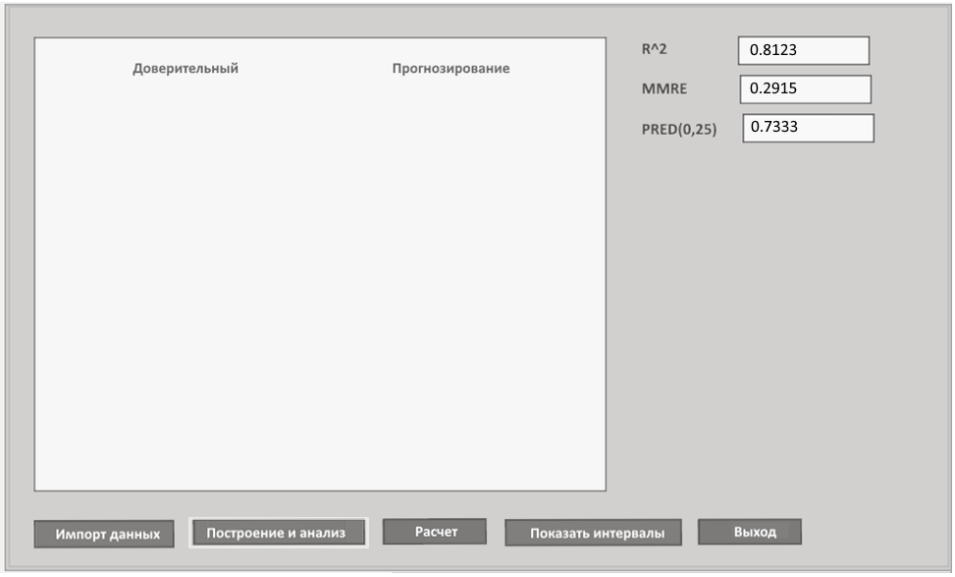

Рисунок 3.9 – Результати випробування функції побудови моделі та її аналізу

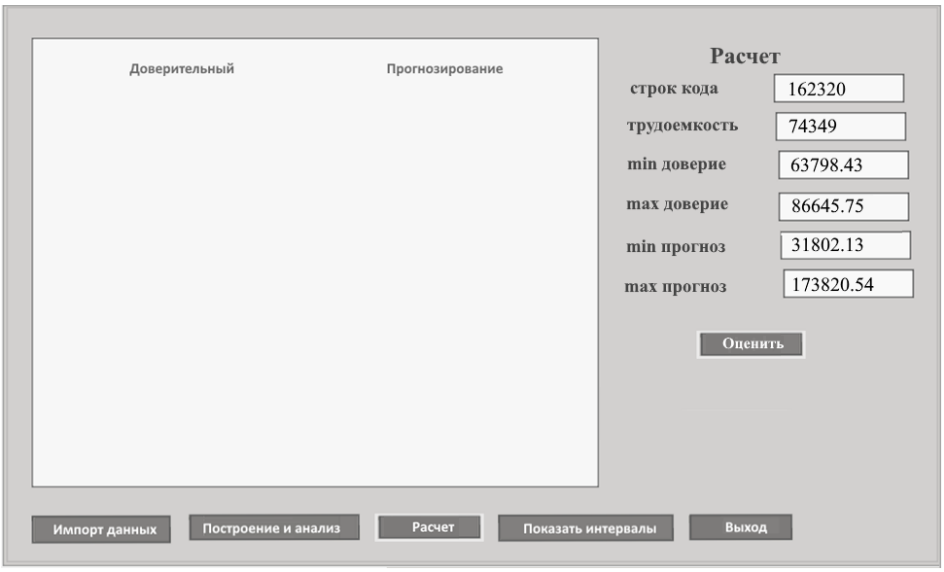

Рисунок 3.10 – Результати прогнозування

Дані випробування показали, що розроблене програмне забезпечення функціонує нормально.

4 РОЗРАХУНОК ЕКОНОМІЧНОЇ ЕФЕКТИВНОСТІ ВІД РОЗРОБКИ ТА ВПРОВАДЖЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ОЦІНЮВАННЯ ТРУДОМІСТКОСТІ РОЗРОБКИ ЛОГІСТИЧНИХ WEB-СЕРВІСІВ НА PYTHON

4.1 Вступ

Дана робота присвячена розробці програми для оцінювання трудомісткості розробки логістичних web-сервісів на Python. Її можна використовувати для прогнозування трудомісткості в комп’ютерних фірмах.

Ефективність програмних 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.

4.2 Розрахунок витрат на створення й експлуатацію програми для оцінювання трудомісткості розробки логістичних web-сервісів на Python

Витрати на розробку системи складаються з витрат на:

– заробітну платню розробника;

– амортизацію ЕОМ, на якій виконується розробка;
– експлуатацію цієї ЕОМ;
– засоби розробки;
– матеріали і комплектуючі.

Виходячи з того, що основна заробітна платня розробників програмного забезпечення складає 10000 грн./міс., вартість сучасної ПЕОМ складає 12000 грн. і вартість кіловат-години електроенергії складає 1,68 грн., розрахуємо вартість розробки системи.

Розрахунок заробітної плати:

$$З_{зп}=ЗП_{роз} \cdot T_{роз} \quad (4.1)$$

де $ЗП_{роз}$ – зарплатня розробника за місяць; $T_{роз}$ – тривалість розробки (дослідження, створення, налагодження і впровадження).

Для розробки даної системи необхідно 4 місяці (за експертною оцінкою часу на розробку аналогічних систем). Кількість розробників – 1 людина.

З цього випливає, що загальна сума витрат на заробітну платню складе:

$$З_{зп} = ЗП_{роз} \cdot T_{роз} = 10000 \cdot 4 = 40000 \text{ (грн.)}$$

Витрати на амортизацію ЕОМ, на якій виконується розробка, розраховується за формулою:

$$З_{амор} = Ц_{варт} \cdot A \cdot T_{роз} \quad (4.2)$$

де $Ц_{варт} = 12000$ грн. – балансова вартість ЕОМ; A = 60% - амортизація за рік; термін служби ЕОМ – 5 років; $T_{роз}$ = 0,33 року – час, необхідний для розробки системи.

$$З_{амор} = 12000 \cdot 0{,}6 \cdot 0{,}33 = 2376 \text{ (грн.)}$$

Витрати на експлуатацію ЕОМ, на якій виконується розробка, полягають в оплаті споживаної нею електричної енергії і розраховуються по формулі:

$$З_{амор} = P_{ЕОМ} \cdot T_{розр} \cdot N_{рд} \cdot N_{рч} \cdot E_{ел} \quad (4.3)$$

де $P_{ЕОМ}$ = 0,4 квт./год. – потужність ЕОМ; $T_{розр}$ = 4 місяці – тривалість розробки; $N_{рд}$ = 22 дні – число робочих днів у місяці; $N_{рч}$ = 8 годин – число годин у робочому дні; $E_{ел}$ = 1,68 грн. – вартість 1 кВт/год електроенергії:

$$З_{амор} = 0{,}4 \cdot 4 \cdot 22 \cdot 8 \cdot 1{,}68 = 102{,}73 \text{ (грн.)}$$

Витрати на матеріали і комплектуючі вироби враховують витрати на папір для друкувальних пристроїв, на картридж і тонер для принтера, а також на непередбачені витрати. Розрахунок приведений у таблиці 4.1 (ціни взяті відповідно до прайс-листів та договірних цін на даний вид матеріалів).

Таблиця 4.1 – Розрахунок витрат на матеріали і комплектуючі вироби

№ з/п	Найменування виробів	Вартість, грн.
1	Папір для принтера	65,00
2	Картридж та тонер для принтера	250,00
3	Література (доступ до Internet)	75,00
4	Непередбачені витрати	60,00
Разом		450,00

Загальний кошторис витрат на створення системи з урахуванням вищевказаного представлений у таблиці 4.2.

Таблиця 4.2 – Загальний кошторис витрат на створення системи

№ з/п	Найменування витрат	Вартість, грн.
1	Заробітна платня основна	40 000,00
2	Заробітна плата додаткова (20% від п.1)	8 000,00
3	Відрахування на соціальне страхування (38% від пп. 1 та 2)	18 240,00
4	Витрати на амортизацію ЕОМ	2376,00
5	Витрати на експлуатацію ЕОМ	102,73
6	Витрати на матеріали і комплектуючі вироби	450,00
7	Адміністративні витрати (50% від основної заробітної платні)	20 000,00
Разом на створення системи		89168,73

До витрат на впровадження системи можна віднести витрати на придбання технічного забезпечення, вартість програмного забезпечення, вартість навчання кадрів, витрати на монтаж и настроювання мережі.

Оскільки параметри технічних засобів, які вже є, відповідають вимогам, то їх вартість при розрахунку витрат враховувати не будемо.

Виходячи з вимог до програмного забезпечення, а також проаналізувавши цінову політику, можемо прийняти наступне (таблиця 4.3).

Таблиця 4.3 – Перелік програмного забезпечення, необхідного для впровадження системи

№	Найменування ПО	Кількість	Вартість, USD	Вартість, грн.
1	MS Windows 7 Professional	1	328	2689,00
2	Microsoft Office	безкоштовно		
Разом		2689,00		

Витрати на програмне забезпечення склали 2689 грн.

Згідно з досвідом створення аналогічних програм, приймаємо, що вартість підготовки кадрів дорівнює 500,00 грн. Витрати на супроводження програмного забезпечення дорівнюватимуть 1% від вартості програми и становлять 268,90 грн.

Інвестиційні витрати на впровадження системи з урахуванням вартості навчання кадрів і витрати на супровід представлені в таблиці 4.4.

Таблиця 4.4 – Інвестиційні витрати на впровадження системи

№	Найменування інвестицій	Вартість, грн.
1	Вартість устаткування	0,00
2	Вартість програмного забезпечення	2689,00
3	Вартість підготовки кадрів	500,00
4	Супроводження системи	268,90
Разом на впровадження системи		3457,90

Разом загальна сума витрат на створення і впровадження системи складає 92 626,63 грн.

4.3 Розрахунок економічної ефективності розробки і впровадження ПЗ оцінювання трудомісткості розробки логістичних web-сервісів на Python

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

Економічна ефективність впровадження підсистеми оцінювання трудомісткості розбирання списку товарів в залежності від кількості ключів очікується за рахунок вивільнення робочого часу працівників та підвищення продуктивності праці. Крім того, не піддається прямій грошовій оцінці зменшення кількості помилкових та необережних рішень, підвищення оперативності керування, поліпшення організації роботи та своєчасне отримання необхідної інформації тощо.

Використання даної програми дозволяє вивільнити 0,5 робочого часу. Оскільки з системою працює один робітник, то система умовно вивільнить 0,5 робітника.

Визначимо пряму економічну ефективність, ґрунтуючись на тому, що впровадження системи вивільнить 0,5 робітника.

Зарплата 0,5 робітника у рік складає $\Delta C = 10000 \cdot 12 \cdot 0,5 = 60000$ (грн.). Річний економічний ефект розраховується за формулою 4.4:

$$E_{рік} = \Delta C - Ц_{супр} - E_n \cdot k, \quad (4.4)$$

де $Ц_{супр}$ – вартість супроводження системи; $E_п$ – нормативний коефіцієнт економічної ефективності капітальних вкладень у галузь – для обчислювальної техніки приймається = 0,5; k – додаткові капітальні

вкладення з урахуванням витрат на проектування, створення і функціонування системи.

$E_{рік} = 60000 – 268{,}90 – 0{,}5 \cdot 92626{,}63 = 13417{,}79$ (грн.)

Строк окупності системи розраховується за формулою 4.5:

$$I = \frac{k}{\Delta C – Ц_{супр}}. \quad (4.5)$$

Отже

$$I \approx 1{,}55 \text{ (року).}$$

Тобто, система окупиться через 1,55 року, що приблизно дорівнює 19 місяцям.

4.4 Висновки за розділом 4

У цьому розділі були здійснені розрахунки витрат на створення й експлуатацію підсистеми оцінювання трудомісткості розробки логістичних web-сервісів на Python, а також розрахунки економічної ефективності та строку окупності даного ПЗ. Виходячи з розрахунків, впровадження такого ПЗ окупить себе за 19 місяців. Отже, можна зробити висновок, що дана розробка програмного забезпечення економічно вигідна та обґрунтована.

5 ОХОРОНА ПРАЦІ

5.1 Вступ

Охорона праці – система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних і лікувально-профілактичних заходів та засобів, спрямованих на збереження здоров'я і працездатності людини в процесі роботи.

Науково-технічний прогрес вніс серйозні зміни в умови виробничої діяльності працівників розумової праці. Їх робота стала інтенсивніше, напруженою, такий, що вимагає значних витрат розумової, емоційної і фізичної енергії. Це зажадало комплексного рішення проблем ергономіки, гігієни та організації роботи, регламентації режимів праці та відпочинку.

Дана система включає [23]:

- аналіз шкідливих факторів і небезпек, які діють на людину;
- розробка заходів по усуненню несприятливих дій на чоловіка і створенню нормальних умов роботи.

5.2 Аналіз небезпечних і шкідливих факторів у відділі при роботі з комп’ютерами

При роботі на персональному комп'ютері співробітники відділу стикаються з дією таких небезпечних фізичних і психологічних чинників як - підвищена температура зовнішнього середовища, недостатня освітленість робочої зони, відсутність або нестача природного світла, електричний струм, статична електрика, розумове перенапруження, перенапруження очей, монотонність праці, емоційні перевантаження. Розглянемо чинники докладніше.

Велику небезпеку становлять дисплеї з електронно-променевими трубками (ЕЛТ) - джерела шкідливих випромінювань, небезпечно впливає на здоров'я операторів.

При роботі монітора виникають два типи випромінювань: електростатичне і електромагнітне. Навколо електростатичного зарядженого монітора підвищується концентрація пилу. Електромагнітне випромінювання створюється магнітними катушками, які знаходяться біля цокольній частині ЕЛТ. Результати медичних досліджень показують, що пил, який електризується, може викликати опік шкіри, а електромагнітне випромінювання може призводити до зниження загальної продуктивності оператора.

Крім цього важливим моментом є показники частоти вертикальної і горизонтальної розгортки ЕЛТ. Прийняті стандарти на монітори не дозволяють використовувати ЕЛТ і відеоадаптери з частотою вертикальної розгортки менше 75 Гц.

Мікрокліматичні параметри впливають на самопочуття і здоров'я людини, а також на надійність роботи засобів обчислювальної техніки. Для комфортної роботи приймається температура 230 С і 180 С в теплу і холодну пору року відповідно, при відносній вологості 55%. [ГОСТ 12.1.005-88].

Також існують інші небезпечні шкідливі фактори, такі як:

Небезпека ураження електричним струмом.

Все обладнання ЕОМ, як будь-яка електроустановка, представляють для людини велику потенційну небезпеку, оскільки в процесі експлуатації або проведення профілактичних робіт людина може торкнутися струмовідних частин.

Експериментальні дослідження показали, що людина відчуває подразнюючу дію змінного струму промислової частоти силою 0,6-1,5 мА і постійного струму 5-7 мА. Ці струми не становлять серйозної небезпеки для діяльності організму людини, оскільки при такій силі струму можливо самостійне звільнення осіб від контакту з струмоведучими частинами. Для

змінного струму промислової частоти сила не відпускає струму знаходиться в межах 6-20 мА. Постійний струм не викликає не відпускає ефекту, але призводить до сильних больових відчуттів, сила такого струму 15-80 мА і більше. (ГОСТ 12.1.038-82)

Недостатня освітленість робочого місця. Правильно спроектоване і виконане висвітлення в приміщенні, де працюють оператори, забезпечує високу працездатність, надає позитивний психологічний вплив на тих, хто працює, сприяє підвищенню продуктивності роботи.

Рекомендована освітленість для роботи для роботи з екраном дисплея становить 400-750 лк. Рекомендовані яскравості в полі зору операторів повинні лежати в межах 1:5-1:10.

Можливість виникнення пожежі.

Приміщення, в якому розміщені ПЕОМ за категоріями пожежної небезпеки відноситься до категорії "В". Зазвичай в ньому знаходиться велика кількість можливих джерел спалаху: кабельні лінії, що використовуються для живлення ПЕОМ від мережі змінного струму напругою 220 В, електронно-променева трубка монітора, яка є вибухонебезпечною без додаткового захисту, устаткування, меблі з горючих матеріалів, папір.

5.3 Розрахунок системи штучного освітлення приміщення відділу

До сучасного виробничого освітлення пред'являються високі вимоги як гігієнічного, так і техніко-економічного характеру. Правильно спроектоване і виконане висвітлення забезпечує високий рівень працездатності, надає позитивне психологічне дію на тих, хто працює, сприяє підвищенню продуктивності роботи.

До систем освітлення висувають такі вимоги [23]:

- відповідність рівня освітленості робочих місць характеру виконуваної зорової роботи;

- досить рівномірний розподіл яскравості на робочих поверхнях і в навколишньому просторі;

- постійність освітленості за часом;

- оптимальна спрямованість випромінюваного освітлювальними приладами світлового потоку;

- довговічність, економічність, електро- та пожежобезпечність, зручність і простота експлуатації.

У тих випадках, коли одного природного освітлення в приміщенні недостатньо, вводять спільне освітлення. При цьому додаткове штучне освітлення застосовують не тільки в темний, але і в світлий час доби.

По конструктивному виконанню штучне освітлення може бути загальним і місцевим. При загальному освітленні всі робочі місця отримують освітлення від загальної освітлювальної установки. Комбіноване освітлення разом із загальним включає місцеве освітлення, яке зосереджує світловий потік безпосередньо на робочих місцях. Застосування лише місцевого освітлення неприпустимо, оскільки виникає необхідність частої адаптації зору, створюються глибокі і різкі тіні і інші несприятливі фактори.

Для штучного освітлення приміщень використовують люмінесцентні лампи, в яких висока світлова віддача і тривалий термін служби, разом з тим необхідно враховувати недоліки: висока пульсація світлового потоку, необхідність застосування спеціальної пускорегулювальної апаратури, складність їх утилізації через наявність у лампах парів ртуті. Сьогодні на ринку систем штучного освітлення з'явилися нові схеми люмінесцентних ламп з підвищеною частотою мерехтіння. Проте широке просування даного сектора утруднено у зв'язку з необізнаністю про даному виді товару та необхідності заміни старого обладнання, яке тягне підвищені витрати. Освітлення нової системи починають застосовувати при капітальному ремонті приміщень і при повній заміні старого обладнання.

Норми освітленості побудовані на основі класифікації зорових робіт за певними кількісними ознаками. Головною ознакою, що визначає розряд

роботи, є розмір розріджених деталей. У свою чергу розряди ділять на чотири підрозділи в залежності від світлості фону і контрасту між деталями і фоном.

На стадії проектування основою, завданням світлотехнічних розрахунків, є визначення необхідної площі світлоприймачів при природному освітленні і потрібної потужності освітлювальної установки - при штучному. Особливістю розрахунку освітленості від світильників з люмінесцентними лампами є, як правило, свідомо відомі їх тип і потужність.

Розрахунок освітленості робочого місця зводиться до вибору системи освітлення, визначення необхідної кількості освітлювачів, їх типу і розміщення.

Вхідні дані: довжина А = 4 м, ширина В = 5 м, висота Н = 3 м. Висота робочої поверхні hp = 1 м. Мінімальна освітленість лампи розжарювання за нормами Emin = 100 лк. Коефіцієнт відображення стелі Sn = 70%, стін Sc = 50%, робочої поверхні Sp = 10%. Напруга в мережі 220 В. Кз = 1,3 і Z = 1,15.

Розрахунок:

Відстань від стелі до робочої поверхні

Але = Н - hp = 3-1 = 2 м;

Відстань від стелі до світильника

hc = 0,2 • Але = 0,4 м;

Висота підвісу світильника над поверхнею, яка висвітлюється

h = Але - hc = 2-0,4 = 1,6 м;

Висота підвісу світильника над підлогою

Нп = h + hp = 1,6 +1 = 2,6 м;

Для досягнення рівномірності освітлення приймаємо ставлення

L / h = 1,5.

Тоді відстань між центром світильників

L = 1,5 • h = 2,4 м;

Необхідна кількість світильників

N = S/L2 = 20/5, 76 = 3,47 світло.;

Приймаються 4 світло. (2 ряди по 2 світло.)

Індекс приміщення

$i = (A \cdot B) / (h \cdot (A + B)) = (20) / 1,6\ (4 + 5) = 1,59 \approx 2;$

За таблицею коефіцієнтів використання світлового потоку при i = 0.6, Sn = 30%, Sc = 10%, Sp = 10% для світильника коефіцієнт використання світлового потоку $\eta = 0,7$.

Світловий потік однієї лампи

$Фр = (Emin \cdot S \cdot Кз \cdot Z) / (N \cdot \eta) = (100 \cdot 20 \cdot 1,3 \cdot 1,15) / (4 \cdot 0,7) = 1067,9$ лм

З таблиці "Параметри ламп розжарювання загального призначення з розрахунковими напругами 130 і 220" вибираємо лампу Б215-255-100 потужністю 100 Вт, що має світловий потік 1100 лм, найбільш близький до розрахункового.

Таким чином фактична освітленість Еф буде рівняти

$Еф = Emin \cdot (Фл / Фр) = 100 \cdot (1100 / 1067,9) = 103,0$ лк;

Загальна потужність

$Рзаг = Рл \cdot N = 100 \cdot 4 = 400$ Вт = 0,4 кВт.

5.4 Розробка заходів по зменшенню дії небезпечних і шкідливих факторів

При розробці заходів, які забезпечують охорону праці та БЖ, враховують всі шкідливі і небезпечні фактори, зменшуючи кожен з них до допустимих значень. Тим не менше, тільки дотримання правил і норм технічної безпеки та охорони праці забезпечить гарну робочу обстановку і запобіжить нещасні випадки на робочому місці [23].

При придбанні моніторів, слід вибирати монітори узгоджених з рекомендаціями щодо зменшення електричних і магнітних полів, а також що підтримують вертикальну частоту розгортки 75 Гц і більше.

Вимоги до мікроклімату в робочому приміщенні.

Для підтримки необхідних мікрокліматичних параметрів необхідно встановлювати кондиціоноване обладнання. Для зменшення кількості пилу встановлюють періодичність вологого прибирання в приміщенні.

Вимоги до електробезпеки.

Велике значення для запобігання електротравматизму має правильна організація обслуговування діючих електроустановок, проведення ремонтних та профілактичних робіт, що здійснюється за допомогою наступних заходів: допуск до роботи, нагляд під час роботи, виробництво відключень під час ремонту, вивішування попереджувальних плакатів і знаків безпеки, перевірка відсутності напруги, накладення заземлення.

Для забезпечення електробезпеки обслуговуючого персоналу передбачених заземлювальні пристрої, до яких підключені всі металеві частини робочого обладнання. У зв'язку з необхідністю розміщення проводів електроживлення обладнання без зайвих витрат на переобладнання приміщення, доцільно влаштовувати підлога з підпіллям.

Для зниження величин виникаючих статичних зарядів в обчислювальних центрах застосовують покриття технологічне з одношарового антистатичного лінолеуму марки ASN. Можна застосовувати загальне і місцеве зволоження повітря. Одним з нових методів зменшення статичної напруги в приміщенні є нейтралізація електрики іонізованим газом.

Вимоги до освітленості.

В дисплейних залах, звичайно, застосовують одностороннє природне бічне освітлення. З метою уникнення прямого сонячного світла використовують приміщення з вікнами з північної, північно-східній або північно-західною орієнтацією. Монітори розташовують подалі від вікон і так, щоб вікна були збоку. Якщо екран монітора розташований до вікна, необхідні спеціальні екранують пристрої.

Для штучного освітлення приміщень слід використовувати люмінесцентні лампи, оскільки в них висока світлова віддача (до 75 лм / Вт і більше), тривалий термін служби (до 10000 годин), мала яскравість поверхні, яка світиться, близький до природного спектральний склад випромінюваного світла, який забезпечує хороше перенесення кольорів. Найбільш прийнятними для дисплейних приміщень є люмінесцентні лампи ЛБ (білого світла) і ЛТБ (білого-тепло-білого світла) потужністю 40, 80 Вт

Для виключення засвічена екранів дисплеїв прямими світловими потоками світильники загального освітлення розташовують збоку від робочого місця, паралельно лінії зору оператора і стіні з вікнами. Таке розміщення світильників дозволяє проводити їх послідовне включення залежно від величини природної освітленості і виключає роздратування очей смугами світла і тіні, що чергуються, виникають при поперечному розташуванні світильників.

Вимоги до пожежної безпеки.

Для запобігання виникнення пожежі необхідно передбачити заходи пожежної профілактики: дотримання протипожежних вимог під час проектування і експлуатації систем вентиляції згідно зі СНиП 1.01.02-84; дотримання умов пожежної безпеки електроустановок згідно вимог; наявність засобів сповіщення:

- Пожежні сповіщувачі (ЛІП-1, ІП-105 2/1 і т.д.);
- Установки пожежогасінні (АУП);
- Інструкції по заходам протипожежної безпеки, план евакуації людей і технічних засобів.

Для поліпшення умов пожежної безпеки в приміщенні має бути встановлений підлога з негорючих матеріалів, технологічно знімний; папір зберігається в металевій шафі; в наявності два вогнегасники типу OU-5, а також два димових датчика; в машинному залі систематично проводиться прибирання та вентилювання приміщення.

Загальні вимоги з техніки безпеки

Кожен користувач ЕОМ зобов'язаний:

- Дотримуватися правил внутрішнього трудового розпорядку;
- Не вносити на територію підприємства і не розпивати спиртні напої, палити лише у відведеному для цієї мети місці.

На території підприємства необхідно виконувати наступні вимоги:

Рухатися тільки по тротуарах, у разі пересування по проїжджій частині дороги потрібно йти по лівій стороні назустріч рухомого транспорту, виконуючи заходи обережності.

При вимушеній зупинці на середині дороги не кидатися в сторони, а стояти на одному місці, щоб дати можливість водієві об'їхати Вас з тієї чи іншої сторони.

Обходити на безпечній відстані місця, де ведуться висотні роботи.

До основних небезпек, які можуть призвести до травм при роботі оператора ЕОМ, відносять небезпеку ураження електричним струмом та іонізуючим випромінюванням.

Необхідно стежити за тим, щоб підлога в машинному залі була рівна і неслизький, все люки, кабелю, проводу повинні бути закриті або захищені.

Суворо дотримуватися правил пожежної безпеки на робочому місці. У разі загоряння відключити електроживлення і зателефонувати в пожежну частину, пояснивши, що і де горить.

Вимоги до безпеки перед початком роботи

Прийнявши ЕОМ від змінника, необхідно перевірити, чи добре прибране робоче місце, ознайомитися з неполадками, які були в попередній зміні, в роботі ЕОМ і з прийнятими заходами по їх усуненню. Також перед початком роботи необхідно:

- Отримати завдання у керівника роботи, а також інструктаж з техніки безпеки.

- Підготувати необхідні інструменти, прилади. Перевірити їх справність, якщо необхідно, отримати захисні пристосування, запчастини, радіодеталі, матеріали.

- Перевірити надійність кабелів в місцях їх підключення до джерел живлення.

- Перевірити відсутність замикання між земляними ланцюгами та ланцюгами живлення напруги.

- Перевірити наявність, справність і відповідність по струму запобіжників в блоках ремонтованої ЕОМ або пристроїв.

Вимоги до безпеки під час роботи:

- Не дозволяється залишати ЕОМ, що знаходиться під напругою, без спостереження.

- Не дозволяється замінювати знімні елементи і проводити переміщення і перемонтування на включеній ЕОМ.

- Не дозволяється користуватися несправною апаратурою та інструментами.

- Не дозволяється поєднувати і роз'єднувати розетки і вилки роз'ємів, які знаходяться під напругою.

- Не дозволяється знімати кришки та щити, які закривають доступ до струмоведучих частин, при включенні обладнання.

- Не дозволяється змінювати запобіжники під напругою.

- При заміні запобіжників в блоках і пристроях, суворо керуватися маркування по струму.

- Не дозволяється користуватися паяльником з незаземленим корпусом.

- При ремонті знімних блоків системи електроживлення їх корпус необхідно заземлити.

- Всі операції, пов'язані з установкою переносних приладів і вимірами, повинні виключати дотик струмоведучих частин.

- При ремонті системи електроживлення необхідно вивішувати плакати, НЕ ВКЛЮЧАТИ! ПРАЦЮЮТЬ ЛЮДИ або НЕ ВКЛЮЧАТИ! РОБОТА НА ЛІНІЇ
- При проведенні робіт необхідно вимагати присутності другої людини, допущеного до роботи з електричними установкам і з напругою до 1000 вольт.

Вимоги до безпеки по закінченню робіт:

- Вимкніть електроживлення від ЕОМ і обладнання, яке забезпечує роботу ЕОМ.
- Приведіть у порядок робоче місце.
- Повідомте керівника робіт (начальника зміни, начальника машини) про закінчення роботи та її результати.
- Зробіть запис в "Журналі зауважень про роботу ЕОМ".

6 ОХОРОНА НАВКОЛИШНЬОГО СЕРЕДОВИЩА

6.1 Вступ до проблеми

Охорона навколишнього середовища здійснюється різними, у тому числі й правовими, засобами. При цьому в правових формах захищаються переважно всі компоненти, які утворюють природне середовище.

Сучасними головними нормативно-правовими актами що регулюють основи організації охорони навколишнього природного середовища, є Закони України «Про охорону навколишнього природного середовища» від 25 червня 1991 р., «Про охорону атмосферного повітря» від 16 жовтня 1992 р., «Про природно-заповідний фонд України» від 16 червня 1992 р., «Про тваринний світ» від 3 березня 1993 р., «Про карантин рослин» від 30 червня 1993 р та інші.

До того ж деякі відносини у сфері використання і охорони навколишнього природного середовища врегульовані кодексами (земельним, водним, лісовим, про надра), а також Законами України «Про плату за землю» від 3 липня 1992 р., «Про ветеринарну медицину» від 25 червня 1992 р. Важливе значення у вирішенні цього питання має затверджений Постановою Верховної Ради «Порядок обмеження, тимчасової заборони (зупинення) чи припинення діяльності підприємств, установ, організацій і об'єктів у разі порушення ними законодавства про охорону навколишнього природного середовища».

Різновидами права природокористування є: право землекористування,право водокористування, право лісокористування, право користуватися надрами, право користуватися тваринним світом, право користування природно-заповідним фондом. Право природокористування це процес раціонального використання людиною природних ресурсів для задоволення різних потреб та інтересів.

Найважливішими принципами природокористування є його цільовий характер, плановість і тривалість, ліцензування, врахування надзвичайного значення у житті суспільства тощо. При цьому виділяють такі групи природокористування, як право загального і спеціального використання землі, вод, лісів, надр, тваринного світу та інших природних ресурсів. Суб'єктами права загального користування природними ресурсами можуть бути, згідно з Законом України «Про охорону навколишнього природного середовища», усі громадяни для задоволення найрізноманітніших потреб і інтересів. Воно здійснюється громадянами безкоштовно і безліцензійно, тобто для цього не потрібен відповідний дозвіл уповноважених органів і осіб. Загальним є, наприклад, використання парків, скверів, водойм, лісів, збір дикорослих ягід,грибів, горіхів і т. ін.

Право загального природокористування закріплене у ст. 13 Конституції України: «Кожний громадянин має право користуватися природними об'єктами права власності народу відповідно до закону». Похідним від загального природокористування є спеціальне використання природних ресурсів. На відміну від першого, це використання конкретних природних ресурсів, що здійснюється громадянами, підприємствами, установами і організаціями у випадках, коли відповідна, визначена у законодавстві частина природних ресурсів передається їм для використання. Як правило, така передача має вартість і визначена в часі. Надання природних ресурсів відбувається на основі спеціальних дозволів державних актів на право постійного користування.

Крім прав суб'єктів, як природокористувачів, сучасною юридичною наукою сформовані й інтенсивно розвиваються екологічні права і обов'язки. Так, у Конституції України записано, що «кожен має право на безпечне для життя і здоров'я довкілля та на відшкодування завданої порушенням цього права шкоди. Кожному гарантується право вільного доступу до інформації про стан довкілля, про якість харчових продуктів і предметів побуту, а також право на її поширення». Аналогічні формулювання є й у Законі України «Про

охорону навколишнього природного середовища», бо це право одне з основних прав людини. Цьому праву відповідає обов'язок держави забезпечувати здійснення санітарно-гігієнічних заходів, спрямованих на поліпшення та оздоровлення навколишнього природного середовища. Усі екологічні права громадян захищаються і відновлюються у судовому порядку.

Поряд з правами Закон України «Про охорону навколишнього природного середовища» передбачає і певні обов'язки громадян. Так, незалежно від того, є громадяни природокористувачами, чи ні, вони зобов'язані берегти природу, раціонально використовувати її запаси, не завдавати шкоди. Крім того, Закон України «Про охорону навколишнього природного середовища» покладає на громадян і підприємства, установи й організації, як суб'єктів спеціального використання природних ресурсів, спеціальні обов'язки. Так, плата за спеціальне природокористування встановлюється на основі нормативів плати і лімітів використання природних ресурсів. Ці нормативи визначаються з урахуванням розповсюдження природних ресурсів, їх якості, можливості використання, місцезнаходження, можливості переробки і зберігання відходів. До того ж суб'єкти спеціального природокористування зобов'язані сплачувати певні кошти за забруднення навколишнього природного середовища, що встановлюються за викиди у атмосферу забруднюючих речовин; скидання забруднюючих речовин на поверхню води, у територіальні і морські води, а також під землю.

Контроль у сфері природовикористання і охорони навколишнього природного середовища здійснюється шляхом перевірки, нагляду, обстеження, інвентаризації та експертиз. Він може здійснюватись як уповноваженими державними органами, так і громадськими формуваннями. Державний контроль покладається на Ради народних депутатів, державні адміністрації та Міністерство охорони навколишнього природного середовища і його органи на місцях.

До основних пріоритетів охорони довкілля та раціонального використання природних ресурсів належать:

– гарантування екологічної безпеки ядерних об'єктів і радіаційного захисту населення та довкілля, зведення до мінімуму шкідливого впливу наслідків аварії на Чорнобильській АЕС;

– поліпшення екологічного стану басейнів рік України та якості питної води;

– стабілізація та поліпшення екологічного стану в містах і промислових центрах Донецько-Придніпровського регіону;

– будівництво нових та реконструкція діючих потужностей комунальних очисних каналізаційних споруд;

– запобігання забрудненню Чорного та Азовського морів і поліпшення їх екологічного стану;

– формування збалансованої системи природокористування та адекватна структурна перебудова виробничого потенціалу економіки, екологізація технологій у промисловості, енергетиці, будівництві, сільському господарстві, на транспорті;

– збереження біологічного та ландшафтного різноманіття, заповідна справа.

Для досягнення цього передбачається вирішення таких завдань:

– зменшення до мінімуму рівня радіаційного забруднення;

– захист повітряного басейну від забруднення, насамперед у великих містах і промислових центрах;

– захист і збереження земельних ресурсів від забруднення, виснаження і нераціонального використання;

– збереження і розширення територій з природним станом ландшафту, посилення природоохоронної діяльності на заповідних і рекреаційних територіях;

– підвищення стійкості та екологічних функцій лісів;

– знешкодження, утилізація та захоронення промислових та побутових відходів;

– запобігання забрудненню морських і внутрішніх вод, зменшення та припинення скиду забруднених стічних вод у водні об'єкти, захист підземних вод від забруднення;

– збереження та відродження малих річок, здійснення управління водними ресурсами на основі басейнового принципу;

– завершення створення державної системи моніторингу навколишнього природного середовища;

– створення системи прогнозування, запобігання та оперативних дій у разі надзвичайних ситуацій природного і природно-техногенного походження;

– забезпечення екологічного супроводу процесу конверсії військово-промислового комплексу;

– здійснення заходів щодо екологічного контролю за діяльністю Збройних Сил України;

– розробка механізмів реалізації схем природокористування;

– впровадження дійових економічних складових впливу на систему природокористування;

– створення системи екологічної освіти, виховання та інформування.

6.2 Забруднення навколишнього середовища підприємством з розробки програмного забезпечення

Основними чинниками забруднення навколишнього середовища фірмою є лише використання автотранспорту. Бо підприємство окрім забруднення повітряного середовища не забруднює ні водне, ні ґрунтове.

В сучасних умовах автомобільний транспорт стає найбільш значним джерелом забруднення атмосферного повітря, особливо великих міст.

Відомо, що атмосферне повітря міст постійно забруднюється і за всіма параметрами докорінно відрізняється від повноцінного природного повітря.

Міські поселення характеризуються найвищими рівнями антропогенних навантажень на навколишнє середовище, в результаті чого воно деформується, набуває якісно нових рис, аж до зміни мікрокліматичних факторів і фізико-хімічних властивостей середовища, зокрема повітряного басейну.

Метеорологічні спостереження свідчать, що температура повітря в межах міських територій у середньому на декілька градусів вища, ніж у приміських районах. Над містами, особливо великими, частіше випадають атмосферні опади, бувають густі тумани, а також смоги – густі тумани, змішані з димом, кіптявою та вихлопними газами. Прозорість атмосфери в містах менша, ніж за її межами і в сільській місцевості. Тумани, а також запиленість повітря помітно зменшують проникнення до земної поверхні сонячних променів.

Однак за рахунок автомобільних викидів якість атмосферного повітря в містах погіршилась.

Прогресуючому забрудненню атмосфери в містах сприяє висока питома вага автомобілів, адже зростання кількості автотранспортних засобів супроводжується збільшенням об’ємів забруднюючих речовин із вихлопних труб. Останнім часом у міському повітрі виріс об’єм оксидів вуглецю,вуглеводнів, оксидів азоту, сажі. Але найбільшу небезпеку, окрім оксидів азоту, складають сірчані та свинцеві сполуки. Їх вміст у міському повітрі значною мірою виріс. Місто не пристосоване до такої кількості автотранспорту. Довжина пробігу без зупинок між світлофорами становить лише 400–600 м., внаслідок цього середня швидкість руху вдень у центрі міста і на великих автошляхах знижується до 12–20 км/год, а це збільшує витрати палива в 3–4 рази.

Відповідно збільшуються й викиди забруднюючих речовин. Автотранспорт також призводить до специфічних форм забруднення повітря.

При русі стираються шини і тисячі тонн гуми у вигляді пилу потрапляє в повітря.

Автомобільний транспорт не тільки забруднює повітря продуктами згорання палива, він сприяє зростаючому надходженню свинцю в навколишнє середовище. В Україні поки ще використовують бензин із вмістом свинцю 0,36 г/л, тоді як в Англії, Німеччині та США – 0,013–0,15.

6.3 Розробка заходів щодо зменшення забруднення

Природоохоронною є будь-яка діяльність, спрямована на збереження якості навколишнього середовища на рівні, що забезпечує стійкість біосфери. До неї відноситься як великомасштабна, здійснювана на загальнодержавному рівні діяльність по збереженню еталонних зразків недоторканої природи і збереженню розмаїтості видів на Землі, організації наукових досліджень, підготовці фахівців-екологів і вихованню населення, так і діяльність окремих підприємств по очищенню від шкідливих речовин стічних вод і відхідних газів, зниженню норм використання природних ресурсів і т.д. Така діяльність здійснюється в основному інженерними методами.

Існують два основних напрямки природоохоронної діяльності підприємств. Перший – очищення шкідливих викидів. Цей шлях «у чистому вигляді» малоефективний, тому що з його допомогою далеко не завжди вдається цілком припинити надходження шкідливих речовин у біосферу. До того ж скорочення рівня забруднення одного компонента навколишнього середовища веде до посилення рівня забруднення іншого.

І, наприклад, встановлення вологих фільтрів при газоочищенні дозволяє скоротити забруднення повітря, але веде до ще більшого забруднення води. Уловлені з відхідних газів і зливальних вод речовини часто отруюють значні земельні площі.

Використання очисних споруд, навіть найефективніших, різко скорочує рівень забруднення навколишнього середовища, однак не вирішує цієї

проблеми цілком, оскільки в процесі функціонування цих установок теж виробляються відходи, хоча й у меншому обсязі, але, як правило, з підвищеною концентрацією шкідливих речовин. Нарешті, робота більшої частини очисних споруджень вимагає значних енергетичних витрат, що, у свою чергу, теж небезпечно для навколишнього середовища.

Крім того, забруднювачі, на знешкодження яких йдуть величезні засоби, являють собою речовини, на які уже витрачена праця і які за рідкісним винятком можна було б використовувати в народному господарстві.

Для досягнення високих еколого-економічних результатів необхідно процес очищення шкідливих викидів сполучити з процесом утилізації уловлених речовин, що уможливить об'єднання першого напрямку з другим.

Другий напрямок – усунення самих причин забруднення, що вимагає розробки маловідходних, а в перспективі і безвідхідних технологій виробництва, які дозволяли б комплексно використовувати вихідну сировину й утилізувати максимум шкідливих для біосфери речовин.

Однак далеко не для усіх виробництв знайдені прийнятні техніко-економічні рішення по різкому скороченню кількості відходів, що утворюються,і їх утилізації, тому на даний час доводиться працювати в обох зазначених напрямках.

Піклуючись про удосконалювання інженерної охорони навколишньої природного середовища, треба пам'ятати, що ніякі очисні спорудження і безвідхідні технології не зможуть відновити стійкість біосфери, якщо будуть перевищені припустимі (граничні) значення скорочення природних, не перетворених людиною природних систем, у чому виявляється чинність закону незамінності біосфери.

Таким порогом може виявитися використання більш 1% енергетики біосфери і глибоке перетворення більш 10% природних територій (правила одного і десяти відсотків). Тому технічні досягнення не знімають необхідності рішення проблем зміни пріоритетів суспільного розвитку,

стабілізації народонаселення, створення достатнього числа заповідних територій і інших, розглянутих раніше.

Основними напрямками робіт в галузі захисту атмосфери від забруднення викидами автотранспорту є:

а) створення і розширення виробництва автомобілів з високоекономічними і малотоксичними двигунами, у тому числі подальша дизелізація автомобілів;

б) розвиток робіт зі створення і впровадження ефективних систем нейтралізації відпрацьованих газів;

в) зниження токсичності моторних палив;

г) розвиток робіт з раціональної організації руху автотранспорту в містах, удосконаленню дорожнього будівництва з метою забезпечення невпинного руху на автомагістралях.

ВИСНОВКИ

Мета, яка ставилася на початку виконання кваліфікаційної роботи, була досягнута. Тобто була підвищена достовірність оцінювання трудомісткості розробки логістичних web-сервісів на Python за рахунок удосконалення нелінійної регресійної моделі.

Для досягнення поставленої мети були виконані наступні завдання:

- проведений аналіз існуючих методів та моделей оцінювання трудомісткості розробки програмного забезпечення;
- удосконалена математична модель для оцінювання трудомісткості розробки логістичних web-сервісів на Python ;
- розроблене програмне забезпечення для реалізації удосконаленої моделі.

Програмне забезпечення, яке розроблене в ході виконання кваліфікаційної роботи, дозволить автоматизувати процес оцінювання трудомісткості розробки логістичних web-сервісів на Python та допоможе зменшити час, необхідний на ці підрахунки.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Голованова, М.А. Оценка трудоемкости работ на ранних стадиях создания программного обеспечения [Текст] / М.А. Голованова, Є.В. Надін // Системи обробки інформації. – Харків, 2014. – № 8 (124). – С.151-156.

2.Сидоров Н.А. Модели, методы и средства оценки стоимости программного обеспечения / Н.А. Сидоров, Д.В. Баценко, Ю.Н. Василенко, Ю.В. Щебетин // Пробл. програмув. – 2006. – 2-3 [спец. вип.]. – С. 290-298.

3. Ваганова Е. В. Оценка стоимости разработки программного продукта: Обзор / Е. В.Ваганова, А. А. Земцов, С. Л. Миньков // Проблемы учета и финансов. – Томск:ТГУ, 2016. – № 1(21). – С. 58–62.

4. Boehm, B.W. Software engineering economics [Текст] / B.W. Boehm. – Prentice-Hall, 1981. – 320 p.

5. Boehm, B.W. The COCOMO 2.0 Software Cost Estimation Model / B.W. Boehm. – American Programmer, 2000. – 586 p.

6. Глазова, М.А., Моделирование стоимости разработки проектов в ИТ-компаниях : Дис. канд. экон. наук / М.А. Глазова. – М., 2008. – 205 с.

7. Кульдин, С.П. Генетический подход к проблеме оценки сроков и трудоемкости разработки программного обеспечения с заданными требованиями к качеству [Текст] / С.П. Кульдин // Прикладная информатика. – 2010. – №5(29). – С.30-42.

8. Coates, J. Technological Forecasting and Social Change [Текст] / J.Coates. – Elsevier Science Inc, 1999. – 235 p.

9. Johnson, Kim. Software cost estimation – Metrics and models [Текст] / Johnson Kim. – University of Calgary, 2001. – 115 p.

10. Макконнелл, С. Сколько стоит программный проект [Текст] / С. Макконнелл. – СПб.: Питер, 2007. – 297 с.

11. Заславський Д.А. Переваги та недоліки використання python для web-розробок / Д.А. Заславський, Л.О. Латанська, І.В. Устенко // «Free and Open Source Software»: Матеріали XIII-ої Міжнародної науково-практичної

конференції (16-18 листопада 2021 р.). – Харків: Харківський національний університет будівництва та архітектури, 2021. – с.16-17.

12. Создание web-сервисов на Python [Электронный ресурс]. – Режим доступа к ресурсу: https://www.my-mooc.com/ru/mooc/sozdanie-web-servisov-na-python/

13. TIOBE Index for November 2021 [Электронный ресурс]. – Режим доступа к ресурсу: https://www.tiobe.com/tiobe-index//

14. Использование Python в веб-разработке: плюсы и минусы [Электронный ресурс]. – Режим доступа к ресурсу: https://pythonist.ru/ispolzovanie-python-v-veb-razrabotke-plyusy-i-minusy/

15. Albrecht A.J., Gaffney J.E. Software function, source lines of codes, and development effort prediction: a software science validation. – IEEE Trans Software Eng. – 1983. – 4 p.

16. Malathi, S. et al (2012). Analysis of size metrics and effort performance criterion in software cost estimation. Indian Journal of Computer Science and Engineering, 3 (1). – P.24-31.

17. Гмурман, В.Е. (2010). Теория вероятностей и математическая статистика. Учеб. пособие. М.: Высшее образование.

18. Прикладна економетрика : навч. посіб. : у двох частинах. Частина 1 / Л. С. Гур'янова, Т. С. Клебанова, С. В. Прокопович та ін. – Харків: ХНЕУ ім. С. Кузнеця, 2016. – 235 с.

19. Регресійний аналіз [Електронний ресурс]. – Режим доступу: https://pidruchniki.com/17280924/ekonomika/regresiyniy_analiz – Назва з екрану

20. Samprit. Handbook of Regression Analysis [Text] / Samprit Chatterjee, Jeffrey S. Somonoff. Wiley, 2012. – 240 p.

21 Расстояние Махаланобиса [Електронний ресурс]. – Режим доступу: http://statistica.ru/theory/rasstoyanie-makhalonobisa/ – Назва з екрану

22. Приходько С.Б. Методичні вказівки до виконання лабораторних робіт з дисципліни "Обробка експериментальних даних на ЕОМ" [Текст] / С.Б. Приходько. – Миколаїв: НУК , 2005. – 52 с.

23.Купчик М.П. Основи охорони праці [Текст]/ М.П.Купчик, М.П.Гандзюк, І.Ф.Степанець. – К.: Основа, 2000. – 416 с.

ДОДАТОК А

Технічне завдання на розробку програмного забезпечення для оцінювання трудомісткості розробки логістичних web-сервісів на Python

Вступ

Назва програми: «Tryd».

Область застосування:

Програма призначена для оцінювання трудомісткості розробки логістичних web-сервісів на Python .

1 Підстави для розробки

Підставою для розробки програмного продукту є завдання на кваліфікаційну роботу «Нелінійна регресійна модель для оцінювання трудомісткості розробки логістичних web-сервісів на Python та розробка програми для її реалізації».

2 Призначення програми

2.1 Функціональне призначення

Даний програмний продукт призначений для оцінювання трудомісткості розробки логістичних web-сервісів на Python.

2.2 Експлуатаційне призначення

Програма може використовуватися для оцінювання трудомісткості проектів в компаніях та командах програмістів, що займаються розробкою логістичних web-сервісів на Python.

3 Вимоги до програми

3.1 Вимоги до функціональних характеристик

3.1.1 Вимоги до складу функцій, що виконуються

Програма повинна забезпечувати можливість виконання наведених нижче функцій:

1) Введення зібраних даних після перед обробки;

2) Нормалізація зібраних даних;

3) Розробка лінійної моделі для отриманих гаусівських даних;

4) Знаходження довірчих інтервалів для лінійної моделі;

5) Знаходження інтервалів прогнозування для лінійної моделі;

6) Побудова нелінійної моделі для вихідних даних, її довірчих інтервалів та інтервалів прогнозування;

7) Перевірити якість побудованої моделі;

8) Відображення інтервалів довіри та прогнозування;

9) Вилучення з екрану інтервалів довіри та прогнозування;

10) Прогнозування даних.

3.1.2 Вимоги до організації вхідних та вихідних даних

Вхідні дані мають бути організовані у текстових файлах. Внесення вхідних даних виконується шляхом вибору шляху до відповідного текстового файлу у спеціальному вікні.

Вихідні дані (довірчі інтервали, інтервали передбачення та метрики якості) повинні виводитися в відповідних місцях на створеній програмою формі.

3.2 Вимоги до надійності

3.2.1 Вимоги до забезпечення надійності функціонування програми

Надійне (стійке) функціонування програми має бути забезпечено надійним (стійким) функціонуванням операційної системи.

3.2.2 Відмова виконання програмного продукту через некоректні дії користувача.

Відмова програми можлива внаслідок некоректних дій користувача при взаємодії з операційною системою. Щоб уникнути виникнення відмов програми за вказаною вище причини необхідно забезпечити стабільне функціонування операційної системи.

3.4 Умови експлуатації

Кліматичні умови експлуатації програмного продукту відповідають умовам експлуатації апаратної частини персонального комп'ютера.

3.5 Вимоги до інформаційної та програмної сумісності

Системні програмні засоби (не вільно поширювані), використовувані програмою, повинні бути представлені ліцензійною версією ОС Windows.

3.6 Вимоги до захисту інформації та програм

Вимоги до захисту інформації та програм відсутні.

3.7 Вимоги до маркування та упаковки

3.7.1 Вимоги до маркування

Вимоги до маркування відсутні.

3.7.2 Вимоги до упаковки

Вимоги до упаковки відсутні.

3.8 Вимоги до транспортування та зберігання

Вимоги до транспортування програми відсутні.

4 Вимоги до програмної документації

Склад програмної документації:

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

5 Техніко-економічні показники

Економічна ефективність впровадження програмного забезпечення розрахована у розділі 4. Згідно з отриманими результатами впровадження даного ПЗ є доцільним, а термін його окупності становить приблизно 4 місяці.

6 Стадії та етапи розробки програмного забезпечення

Стадії та етапи розробки програми для оцінювання трудомісткості розробки логістичних web-сервісів на Python , зазначені в таблиці А.1.

Таблиця А.1 – Стадії та етапи розробки програмного забезпечення

Стадії розробки	Етапи робіт	Вміст робіт	Контрольні точки
1	2	3	4
Технічне завдання	Обумовлення необхідності розробки	Постановка задачі. Збір початкових матеріалів.	З 05.09.2021 до 08.09.2021
	Розробка та затвердження технічного завдання	Визначення вимог до ПЗ. Обговорення та затвердження технічного завдання	З 09.09.2021 до 20.10.2021
Ескізний проект	Розробка ескізного проекту	Попередня розробка структури вхідних та вихідних даних, уточнення методів рішення завдань, загальний опис алгоритму	З 21.10.2021 до 02.11.2021
	Затвердження ескізного проекту	Розробка пояснювальної записки. Узгодження і затвердження ескізного проекту	З 03.11.2021 до 10.11.2021

Продовження табл. А.1

1	2	3	4
Технічний проект	Розробка технічного проекту	Уточнення структури вхідних та вихідних даних, розробка алгоритму рішення задачі, розробка структури ПЗ	З 11.11.2021 до 19.11.2021
	Затвердження технічного проекту	Розробка пояснювальної записки. Узгодження і затвердження технічного проекту	З 20.11.2021 до 25.11.2021
Робочий проект	Розробка ПЗ	Програмування та налагодження програми	З 26.11.2021 до 01.12.2021
	Розробка програмної документації	Розробка програмної документації відповідно до вимог ГОСТ 19.101-77	З 02.12.2021 до 05.12.2021
	Випробування програми	Розробка, узгодження та затвердження програми і методики випробувань, коригування програми і програмної документації за результатами випробувань	З 06.12.2021 до 07.12.2021

7 Порядок контролю та приймання

Контроль здійснюється замовником на кожній стадії розробки ПЗ. Приймання здійснюється замовником за програмою та методикою випробувань. Випробування проводиться в зазначений термін.

Хід проведення приймально-здавальних випробувань документується за допомогою протоколу проведення випробувань. На підставі протоколу проведення випробувань виконувач сумісно з замовником підписують акт приймання-здачі програми в експлуатацію.

ДОДАТОК Б

Опис програми для оцінювання трудомісткості розробки логістичних web-сервісів на Python

1 Загальні відомості

1.1 Позначення і найменування програми

Найменування: Програмне забезпечення для оцінювання трудомісткості розробки логістичних web-сервісів на Python.

Позначення: «Tryd».

1.2 Програмне забезпечення, необхідне для функціонування програми

Програмне забезпечення «Tryd» є крос-платформним і основною необхідною вимогою для функціонування виробу є наявність .NET Framework (4.5 і вище версії).

1.3 Мови програмування

ПЗ «Tryd» реалізовано на мові високого рівня C#. Модулі, реалізовані на інших мовах програмування, не використовуються.

2 Функціональне призначення

2.1 Класи вирішуваних завдань і призначення програми

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

1) Введення зібраних даних після перед обробки;

2) Нормалізація зібраних даних;

3) Розробка лінійної моделі для отриманих гаусівських даних;

4) Знаходження довірчих інтервалів для лінійної моделі;

5) Знаходження інтервалів прогнозування для лінійної моделі;

6) Побудова нелінійної моделі для вихідних даних, її довірчих інтервалів та інтервалів прогнозування;

7) Перевірити якість побудованої моделі;

8) Відображення інтервалів довіри та прогнозування;

9) Вилучення з екрану інтервалів довіри та прогнозування;

10) Прогнозування даних.

2 Функціональні обмеження

Файли повинні бути формату EXE.

3 Використовувані технічні засоби

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

4 Виклик і завантаження

Запуск ПЗ «Tryd» в графічному режимі здійснюється двома способами:

1) за допомогою файлу запуску: Tryd.exe;

2) з використанням командного рядку: Tryd.exe.

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

Інструкція користувача програмного забезпечення для оцінювання трудомісткості розробки логістичних web-сервісів на Python

1. Для того, щоб запустити програму, потрібно запустити файл «Tryd.exe». При запуску програми з’являється початкова форма (рис. Д.1).

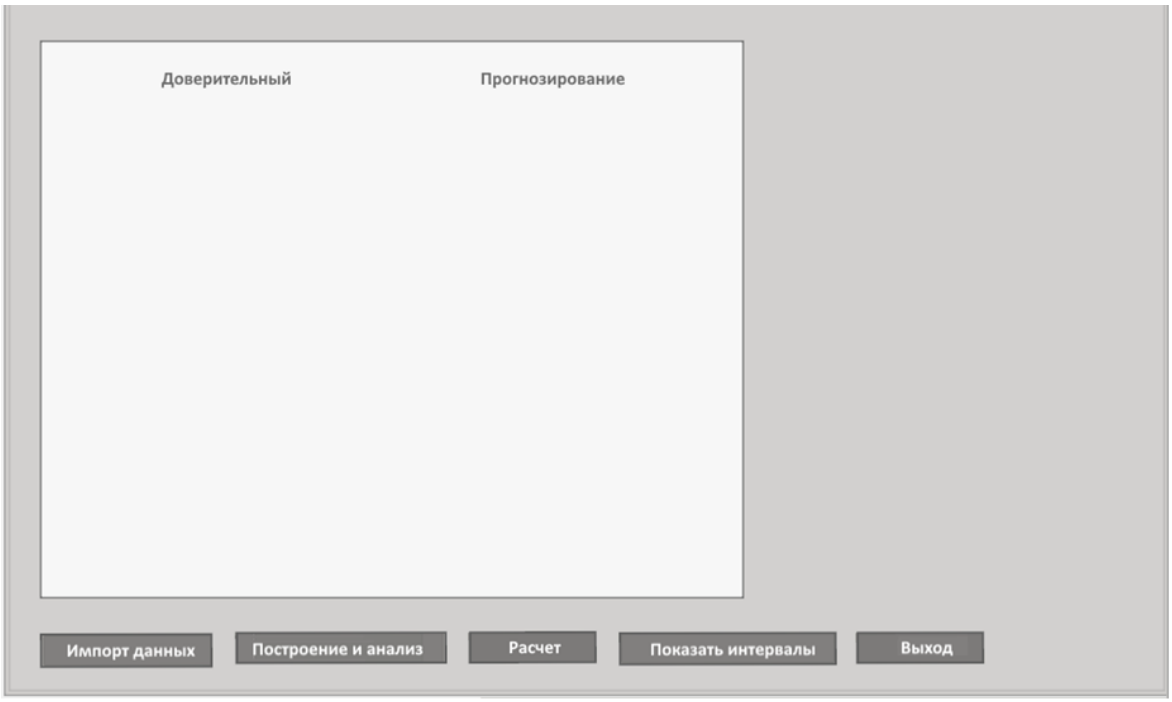

Рисунок Д.1 – Головна форма

2. Для внесення інформації про дані потрібно натиснути на кнопку «Імпорт даних» (рис. Д.2).

Рисунок Д.2 – Додавання даних

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

4. Після цього потрібно натиснути на кнопку «Побудова та аналіз» (рис. Д.3).

Рисунок Д.3 – Кнопка «Побудова та аналіз»

Після цього програма виконає побудову нелінійної регресійної моделі та на екрані зявляться характеристики якості побудованої моделі: R^2, MMRE, PRED(0,25) (рис. Д.4).

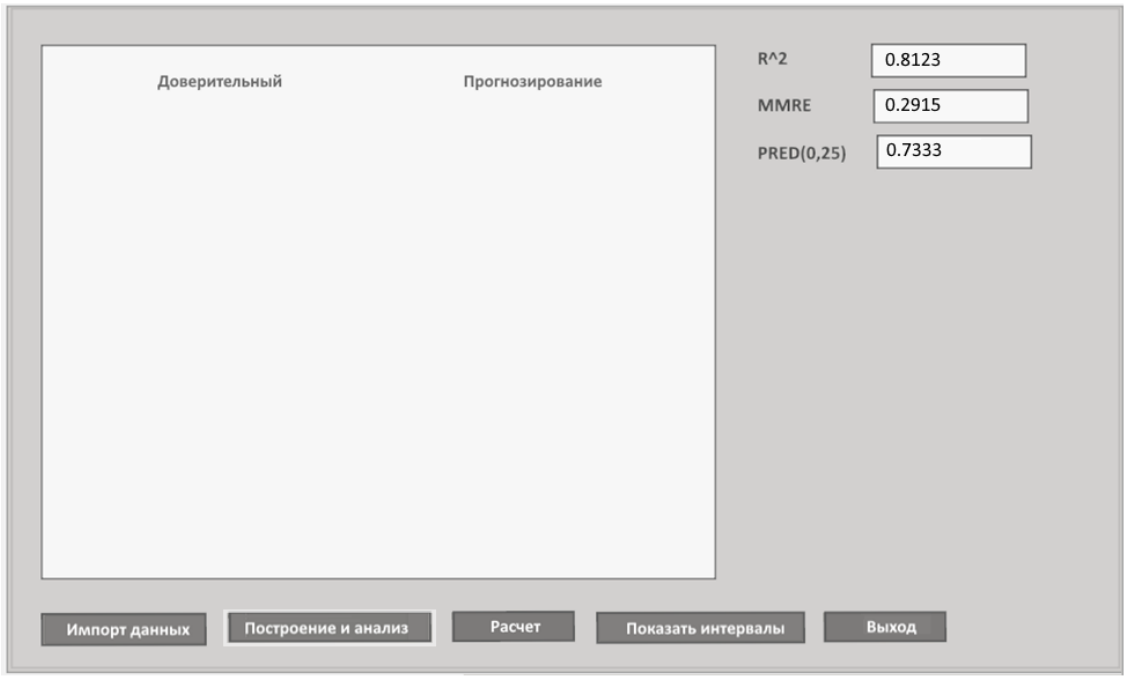

Рисунок Д.4 – Результат побудови моделі та її аналізу

5. Для відображення довірчих інтервалів та інтервалів передбачення необхідно натиснути на кнопку «Показати інтервали».

6. Для прогнозування необхідно натиснути на кнопку «Розрахунок», ввести дані для прогнозування (кількість строк коду) та натиснути на кнопку «Оцінити» (рис. Д.5). На екрані зявиться прогнозне значення трудомісткості та відповідні довірчий інтервал та інтервал прогнозування.

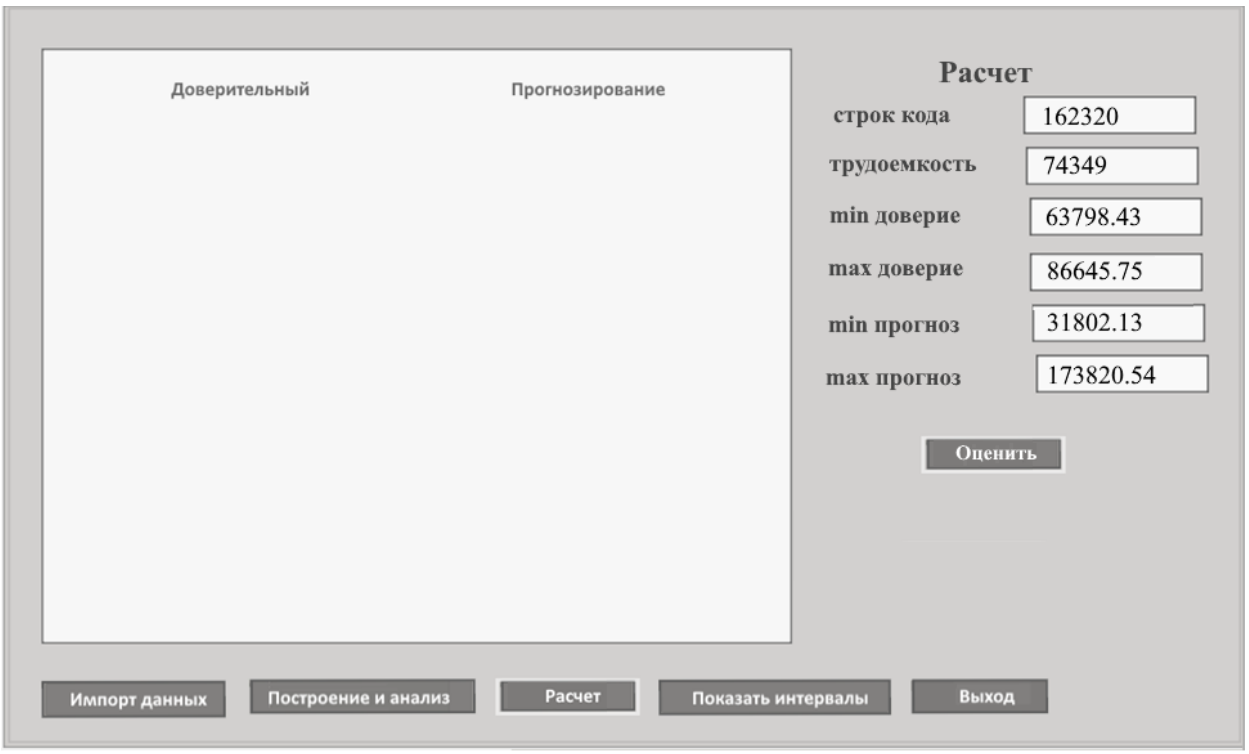

Рисунок Д.5 – Результат прогнозування

7. Для завершення роботи програми необхідно натиснути на кнопку «Вихід».

ДОДАТОК Г

Програма і методика випробувань програмного забезпечення для оцінювання трудомісткості розробки логістичних web-сервісів на Python

1 Об’єкт випробувань

Об’єктом випробувань є програмне забезпечення для оцінювання трудомісткості розробки логістичних web-сервісів на Python .

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:

1) Введення зібраних даних після перед обробки;

2) Нормалізація зібраних даних;

3) Розробка лінійної моделі для отриманих гаусівських даних;

4) Знаходження довірчих інтервалів для лінійної моделі;

5) Знаходження інтервалів прогнозування для лінійної моделі;

6) Побудова нелінійної моделі для вихідних даних, її довірчих інтервалів та інтервалів прогнозування;

7) Перевірити якість побудованої моделі;

8) Відображення інтервалів довіри та прогнозування;

9) Вилучення з екрану інтервалів довіри та прогнозування;

10) Прогнозування даних.

4 Вимоги до програмної документації

Склад програмної документації, що надається на випробування:

– технічне завдання;

– текст програми;

– опис програми;

– програма і методика випробувань;

– інструкція користувача.

5 Засоби і порядок випробувань

Випробування проводилось на платформі Windows 7 32bit.

Порядок випробувань:

– встановити і запустити програму;

– перевірити можливості додавання даних до програми;

– перевірити можливість продивитися довірчі інтервали та інтервали передбачення;

– перевірити можливість сховати інтервали;

– перевірити можливість розрахувати метрики якості;

– перевірити неможливість ввести файл з даними неправильного формату.

6 Методика випробувань

– Випробування функції введення зібраних даних після перед обробки

- Натиснути кнопку «Імпорт даних».
- В вікні, що відкрилося, обрати шлях до файлу.
- Повинно відбутися відкриття файлу та з’явитися назви файлу поблизу кнопки «Імпорт даних».
- Натиснути кнопку «Обрати файл».

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

– Випробування функції відображення інтервалів довіри та прогнозування

- Прочитати дані з файлу.
- Натиснути на кнопку «Побудова та аналіз».
- Натиснути на кнопку «Показати інтервали».
- Інтервали повинні з’явитися на екрані у вигляді таблиці.
- Повинен з’явитися текст «Сховати інтервали» замість «Показати інтервали».

Очікуваний результат: після натискання кнопки «Показати інтервали» на головну форму повинна додатися таблиця, яка містить ці інтервали.

– Випробування неможливості введення файлу з даними неправильного формату

- Ввести файл з даними неправильного формату.
- Повинно з’явитися повідомлення про неправильний формат даних.

Очікуваний результат: програма видає повідомлення про неправильний формат даних.

ДОДАТОК Д

Текст програми для оцінювання для оцінювання трудомісткості розробки логістичних web-сервісів на Python

FileReader.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.IO;
using System.Diagnostics;

namespace WindowsFormsApp1
{
   class FromFile
   {
      private double[] x;
      private double[] y;
      private string path;

      public void read()
      {
         using (var reader = new StreamReader(path))
         {
            string row;
            int i = 0;
```

```
var lineCount = File.ReadLines(path).Count();

x = new double[lineCount];
y = new double[lineCount];

while ((row = reader.ReadLine()) != null)
{
   try
   {
                        x[i] = Double.Parse(row.Split(new[] { ' ' },
                StringSplitOptions.RemoveEmptyEntries)[0]);

   }
   catch (Exception e)
   {
                        throw new Exception("Неправильный
                формат количества строк кода");
   }

   if (x[i] <= 0)
   {
      throw new Exception("Количество строк ода меньше
                или равно 0");
   }

   try
   {
                        y[i] = Double.Parse(row.Split(new[] { ' ' },
                StringSplitOptions.RemoveEmptyEntries)[1].Trim
                Start());
```

```
                    }
            catch (Exception e)
            {
                throw new Exception("Неправильный формат
                        трудоемкости");
            }

            if (y[i] <= 0)
            {
                        throw new Exception("Значение
                    трудоемкости меньше или равно 0");
            }

            i++;

        }

    }
}

public double[] GetLines()
{
    return x;
}

public double[] GetTrudoemkost()
{
    return y;
}
```

```
        public FromFile(string path)
        {
            this.path = path;
            this.read();
        }
    }
}
```

Program.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace WindowsFormsApp1
{
    static class Program
    {
        /// <summary>
        /// Главная точка входа для приложения.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new Graph());
        }
    }
```

```
}
```

Form1.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Diagnostics;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Windows.Forms.DataVisualization.Charting;

namespace WindowsFormsApp1
{
   public partial class Graph : Form
   {
      private double[] x;
      private double[] y;
      private bool isShown = false;
      private bool isFirst = true;
      private LgNormalizer lg;

private void showIntervals()
      {
         DataTable dt = new DataTable();

         dt.Columns.Add("№", typeof(int));
         dt.Columns.Add("Доверительный интервал lg", typeof(string));
```

```
    dt.Columns.Add("Интервал прогнозирования lg", typeof(string));
    for (int i = 0; i < x.Length; i++)
    {
        string lgDov = "[ " + Math.Round(lg.getYMinusDelta()[i]
                , 4).ToString() + "; " +
        Math.Round(lg.getYPlusDelta()[i], 4).ToString() + " ]";
                        string lgProg = "[ " +
                    Math.Round(lg.getYMinusDeltaPrognozirovaniya()[i],
                    4).ToString() + "; " +
                    Math.Round(lg.getYPlusDeltaPrognozirovaniya()[i],
                    4).ToString() + " ]";
                            dt.Rows.Add(new object[] { i + 1, lgDov, lgProg,
                    });
    }

    DataGridView grid = new DataGridView();
    grid.AllowUserToAddRows = false;
    grid.AllowUserToDeleteRows = false;
    grid.DataSource = dt;
    grid.Width = this.panel4.Width;
                        grid.AutoSizeColumnsMode =
            DataGridViewAutoSizeColumnsMode.DisplayedCells;
                        grid.DefaultCellStyle.Alignment =
                DataGridViewContentAlignment.MiddleCenter;

    this.panel4.Controls.Add(grid);
}

private void checkHide()
{
```

```
    this.panel4.Height = 0;

    int a = this.tableLayoutPanel1.Location.Y;

    this.tableLayoutPanel1.Height += 150;
    this.tableLayoutPanel1.Location = new Point(
        this.tableLayoutPanel1.Location.X,
        a - 150
    );
    isShown = false;
    this.button3.Text = "Показать интервалы";
}

private void button3_Click(object sender, EventArgs e)
{
    if (!checkShow())
    {
        checkHide();
    }
}
```