МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ
Кафедра комп'ютерних наук

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему: **«РОЗРОБКА ПРОГРАМНОГО СОФТУ ДЛЯ ОПТИМІЗАЦІЇ ПК НА ОПЕРАЦІЙНІЙ СИСТЕМІ WINDOWS 10/11»**

Виконав: студент 4 курсу, групи 6.1220
спеціальності 122 Комп'ютерні науки
(шифр і назва спеціальності)
освітньої програми Комп'ютерні науки
(назва освітньої програми)
В.З. Зубко
(ініціали та прізвище)

Керівник доцент кафедри комп'ютерних наук, доцент, д.е.н. Полуектова Н.Р.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Рецензент завідувач кафедри фундаментальної та прикладної математики, професор, д.т.н. Гребенюк С.М.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя – 2024

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет математичний

Кафедра комп'ютерних наук

Рівень вищої освіти бакалавр

Спеціальність 122 Комп'ютерні науки
(шифр і назва)

Освітня програма Комп'ютерні науки

ЗАТВЕРДЖУЮ
Завідувач кафедри комп'ютерних наук,
д.т.н., доцент

_____ Шило Г.М.
(підпис)

" 25 " грудня 2023 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

Зубку Владиславу Заурійовичу
(прізвище, ім'я та по-батькові)

1. Тема роботи Розробка програмного софту для оптимізації ПК на операційній системі Windows 10/11

керівник роботи Полуектова Наталія Робертівна, д.е.н., доцент
(прізвище, ім'я та по-батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від « 21 » грудня 2023 року № 2180-с

2. Строк подання студентом роботи 05.06.2024

3. Вихідні дані до роботи 1. Постановка задачі.
2. Перелік літератури.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Постановка задачі.
2. Основні теоретичні відомості.
3. Розробка програмної системи для оптимізації роботи ПК

5. Презентація

6. Консультанти розділів роботи

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

7. Дата видачі завдання 25.12.2023

К А Л Е Н Д А Р Н И Й П Л А Н

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану роботи.	19.01.2024	
2.	Збір вихідних даних.	29.01.2024	
3.	Обробка методичних та теоретичних	12.04.2024	
	джерел.		
4.	Розробка першого та другого розділу.	17.04.2024	
5.	Розробка третього розділу.	27.05.2024	
6.	Оформлення та нормоконтроль	08.06.2024	
	кваліфікаційної роботи бакалавра.		
7.	Захист кваліфікаційної роботи.	22.06.2024	

Студент _____ В.З. Зубко
(підпис) (ініціали та прізвище)

Керівник роботи _____ Н.Р. Полуектова
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ О.Г.Спиця
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Кваліфікаційна робота бакалавра «Розробка програмного софту для оптимізації ПК на операційній системі Windows 10/11»: 44 с., 23 рис., 9 джерел, 4 додатки.

ОПЕРАЦІЙНА СИСТЕМА WINDOWS 10/11, ПРОДУКТИВНІСТЬ ПК, ПРОГРАМНА РОЗРОБКА, СЕРВІСНЕ ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, ТЕСТУВАННЯ, ТЕХНОЛОГІЇ ПРОГРАМУВАННЯ, ФАЙЛОВА СИСТЕМА, PYTHON, PYQT.

Об’єкт дослідження – продуктивність роботи ПК на операційній системі Windows 10/11.

Предмет дослідження – програмні продукти для підвищення продуктивності роботи ПК на операційній системі Windows 10/11.

Мета роботи: Розробити власне програмне забезпечення для оптимізації роботи ПК на базі Windows 10/11, власний інструмент, який зможе ефективно підвищувати швидкість, стабільність та продуктивність комп’ютерів, що працюють під управлінням цих операційних систем.

Для виконання мети реалізовані наступні завдання:

– аналіз існуючих програмних рішень для оптимізації ПК на Windows 10/11, їх можливостей та обмежень;
– вивчення технологій, алгоритмів та підходів, що застосовуються в таких програмах;
– розробка власного програмного забезпечення для оптимізації ПК на Windows 10/11;
– тестування розробленого програмного продукту.

SUMMARY

Bacelor’s Qualifying Theses « Development of software for PC optimization on the Windows 10/11 operating system»: 44 pages, 23 figures, 9 references, 4 supplements.

WINDOWS 10/11 OPERATING SYSTEM, PC PERFORMANCE, SOFTWARE DEVELOPMENT, SERVICE SOFTWARE, TESTING, PROGRAMMING TECHNOLOGIES, FILE SYSTEM, PYTHON, PYQT.

Object of study – PC performance on the Windows 10/11 operating system.

The subject of the study – software products to improve the performance of PCs running on the Windows 10/11 operating system.

Objective: To develop our own software for optimizing the performance of Windows 10/11-based PCs, a proprietary tool that can effectively increase the speed, stability and performance of computers running these operating systems.

To achieve this goal, the following tasks were implemented:

– analysis of existing software solutions for optimizing Windows 10/11 PCs, their capabilities and limitations;
– study of technologies, algorithms and approaches used in such programs;
– development of own software for PC optimization on Windows 10/11;
– testing of the developed software product.

ЗМІСТ

ВСТУП

Сучасні персональні комп’ютери, незважаючи на постійне вдосконалення апаратного забезпечення, все ще стикаються з проблемами продуктивності та швидкодії, особливо при роботі з ресурсоємними додатками та великими обсягами даних. Це пов’язано як з особливостями програмного забезпечення, так і з накопиченням різноманітного "сміття" в системі з плином часу. Кожен користувач ПК на базі Windows 10/11 знайомий з ситуацією, коли комп’ютер починає працювати повільніше чи починаються технічні перебої під час виконання звичайних операцій. Таке зниження продуктивності значно знижує ефективність роботи, викликає незручності та призводить до втрати часу. Тому розробка ефективних інструментів оптимізації та очищення операційної системи Windows 10/11 є вкрай актуальною задачею.

Мета роботи полягає у створенні власного програмного забезпечення, яке дозволить користувачам ПК на базі Windows 10/11 підвищувати швидкодію, стабільність та ефективність роботи їх комп’ютерів. Для досягнення цієї мети необхідно провести аналіз існуючих програмних рішень, дослідити технології та алгоритми, що застосовуються для оптимізації комп’ютерних систем, а потім розробити та протестувати власну програму для підвищення продуктивності ПК.

Актуальність роботи полягає у розробці власного програмного рішення для оптимізації ПК на операційних системах Windows 10/11, яке використовує сучасні підходи та технології для підвищення ефективності комп’ютерів. Існуючі програми для оптимізації, хоч і мають певні можливості, все ще мають ряд обмежень та недоліків. Тому створення нового, більш досконалого програмного забезпечення, здатного комплексно вирішувати завдання з підвищення швидкодії, стабільності та продуктивності ПК, є актуальною науковою задачею.

Практична значимість роботи визначається можливістю використання розробленого програмного забезпечення широким колом користувачів ПК на базі Windows 10/11 для підвищення швидкодії та продуктивності їх комп’ютерів. Запропонований інструмент дозволить оптимізувати роботу операційної системи, звільнити системні ресурси, покращити стабільність та надійність функціонування комп’ютерів. Це, в свою чергу, підвищить ефективність роботи користувачів та дозволить заощадити їхній час і зусилля.

1 АНАЛІЗ СУЧАСНИХ ПРОГРАМНИХ РІШЕНЬ ДЛЯ ОПТИМІЗАЦІЇ ПК НА БАЗІ WINDOWS 10/11

1.1 Виявлення основних можливостей та обмежень існуючих оптимізаційних інструментів

На ринку існує значна кількість програмних рішень, що пропонують функції оптимізації та підвищення продуктивності комп'ютерів на базі Windows 10/11. Серед найбільш популярних та відомих програм можна виділити:

CCleaner – одна з найпоширеніших утиліт для очищення системи від непотрібних файлів, тимчасових файлів, історії браузера тощо (див. рис.1.1). Також має інструменти для управління автозавантаженням, реєстром Windows та видалення застарілих програм.

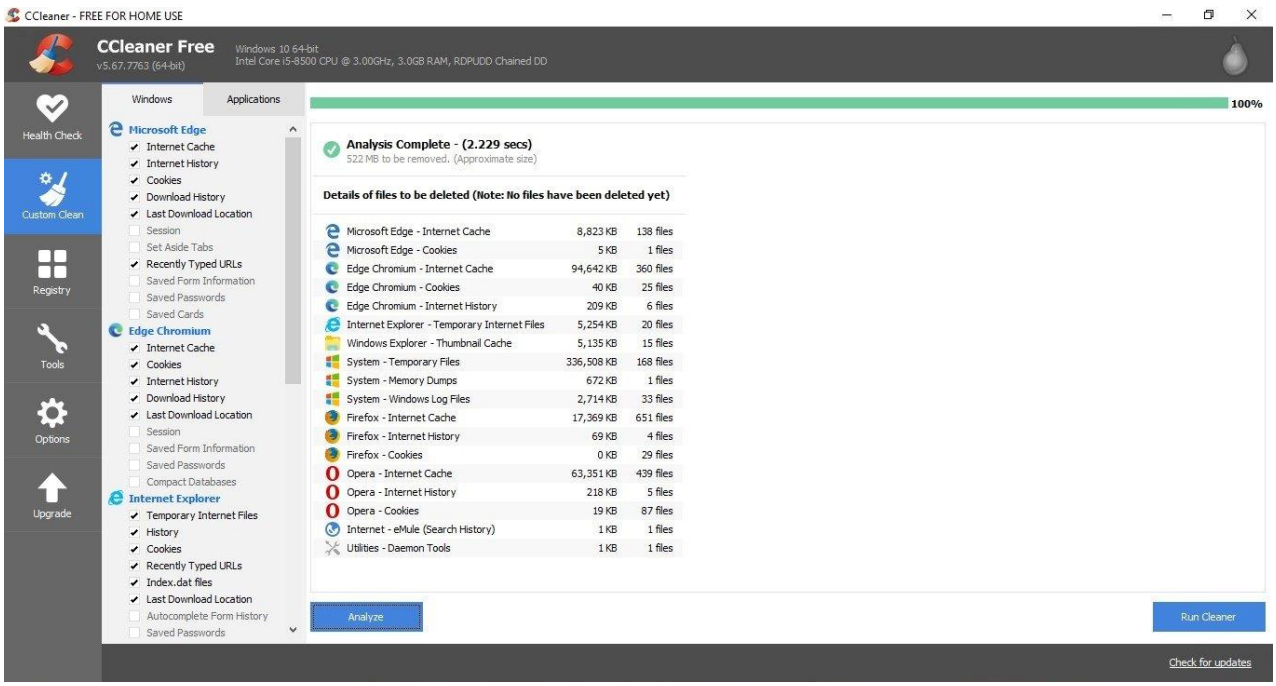

Рисунок 1.1 – Головний екран програми CCleaner

Advanced SystemCare – комплексне рішення від IObit, що включає в себе різноманітні інструменти для прискорення роботи ПК, оптимізації реєстру, очищення диску, видалення шкідливих програм та захисту конфіденційності (див. рис.1.2).

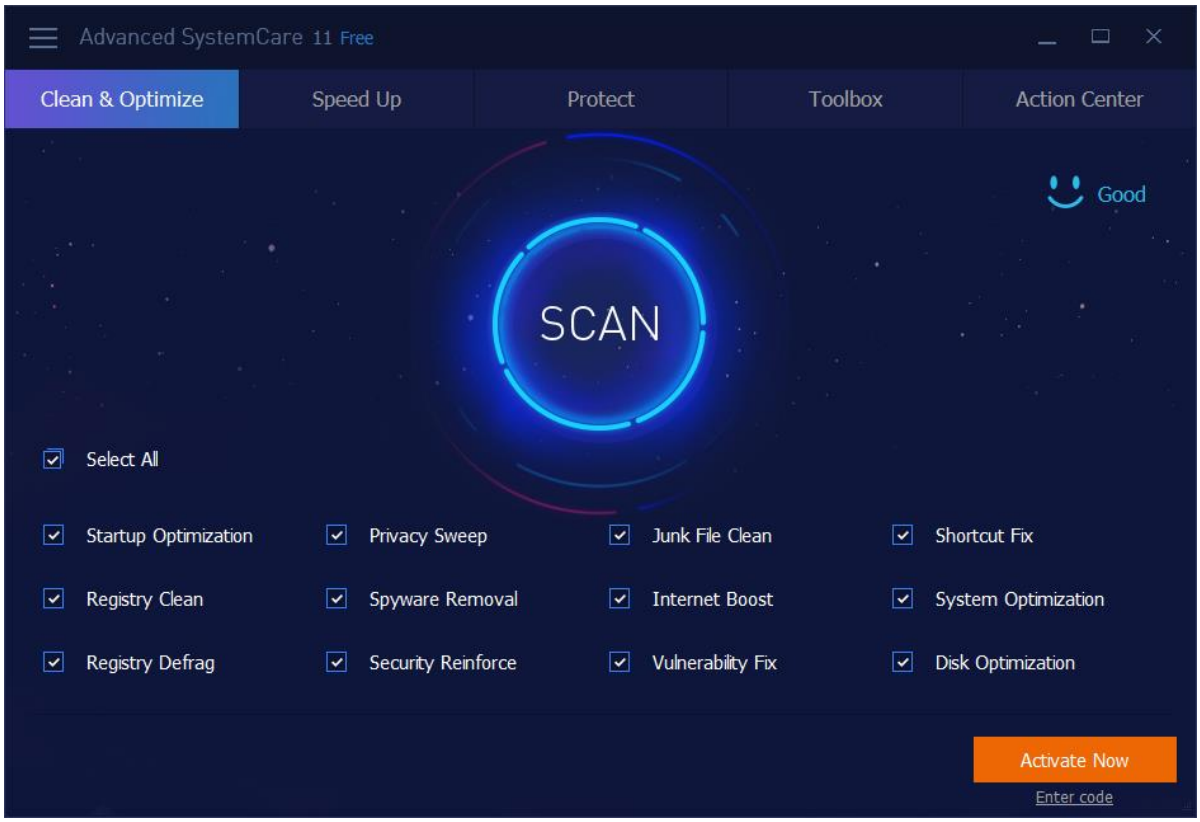

Рисунок 1.2 – Головний екран програми Advanced SystemCare

Wise Care 365 – потужна програма для оптимізації Windows, що пропонує функції дефрагментації дисків, оптимізації реєстру, управління автозавантаженням, очищення непотрібних файлів та налаштування продуктивності системи (див. рис.1.3)

Перераховані програми мають значну функціональність та широкий спектр можливостей для оптимізації комп’ютерних систем на базі Windows 10/11. Проте, кожна з них має свої переваги, недоліки та особливості, які необхідно детально розглянути.

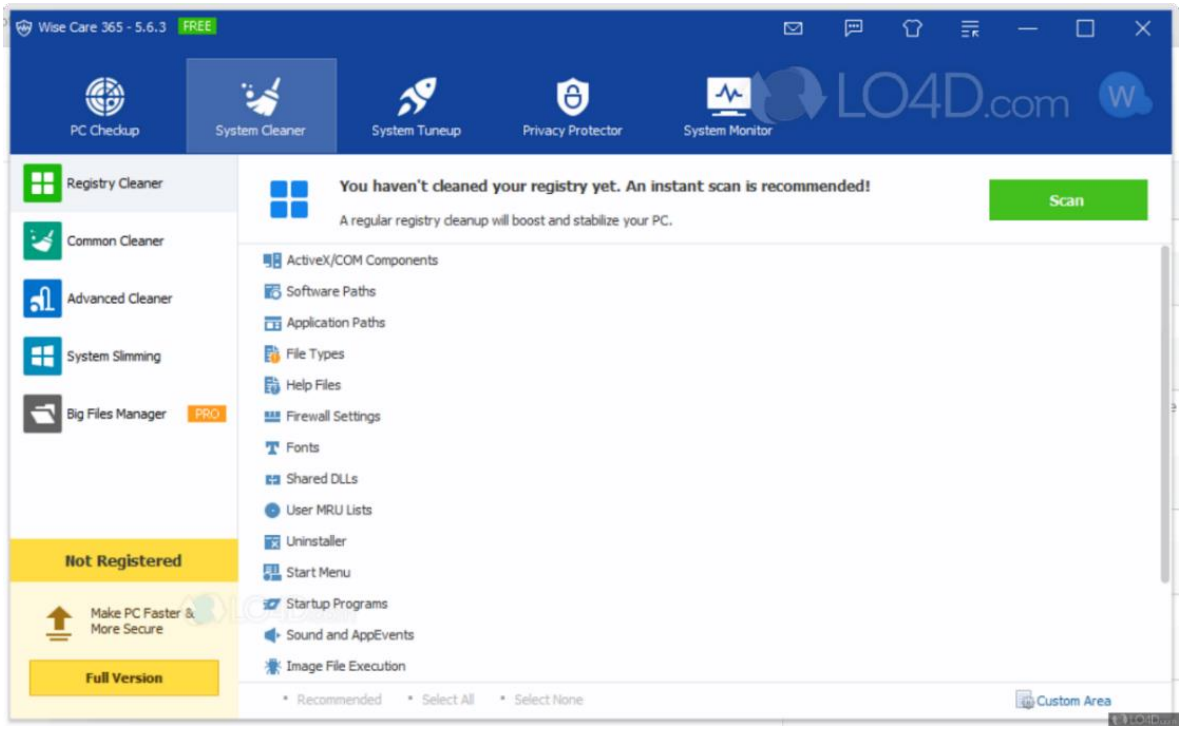

Рисунок 1.3 – Головний екран програми Wise Care 365

1.2 Виявлення основних можливостей та обмежень існуючих оптимізаційних інструментів

Існуючі програми для оптимізації комп’ютерних систем на базі Windows 10/11 мають широкий спектр можливостей, проте також і певні обмеження. Розглянемо основні з них:

Можливості:

– очищення жорстких дисків від непотрібних файлів, тимчасових файлів, історії браузера тощо для звільнення місця та прискорення роботи;
– оптимізація системного реєстру шляхом видалення застарілих або пошкоджених записів для підвищення стабільності системи;
– управління автозавантаженням програм та сервісів для запобігання надмірного навантаження на систему під час запуску;

– налаштування параметрів продуктивності операційної системи, таких як розмір файлу підкачки, використання оперативної пам’яті тощо;
– дефрагментація жорстких дисків для оптимізації доступу до файлів та прискорення читання/запису даних;
– видалення шкідливих програм, рекламного програмного забезпечення та інших потенційно небажаних додатків;
– інструменти для аналізу та усунення системних помилок, конфліктів драйверів та інших проблем.

Обмеження:

– багато програм мають обмежену функціональність, зосереджуючись лише на певних аспектах оптимізації (очищення, оптимізація реєстру тощо);
– деякі інструменти можуть бути надмірно агресивними у видаленні системних файлів чи записів реєстру, що може призвести до нестабільності системи;
– відсутність комплексного підходу до оптимізації різних компонентів операційної системи та програмного забезпечення;
– недостатня гнучкість у налаштуванні та персоналізації процесу оптимізації відповідно до індивідуальних потреб користувача;
– обмежені можливості аналізу та виявлення ключових факторів, що знижують продуктивність системи;
– деякі програми можуть бути ресурсоємними самі по собі, навантажуючи систему під час виконання оптимізації;
– відсутність регулярних оновлень та підтримки для адаптації до змін в операційних системах Windows.

Тож, існуючі інструменти для оптимізації ПК на базі Windows пропонують корисну функціональність, але також мають певні недоліки та обмеження. Для створення ефективного програмного рішення необхідно

врахувати ці фактори та розробити комплексний підхід до оптимізації з широкими можливостями налаштування та аналізу.

1.3 Дослідження технологій, алгоритмів і підходів, що застосовуються в програмах для оптимізації ПК

Програми для оптимізації комп'ютерних систем на базі Windows використовують широкий спектр технологій, алгоритмів та підходів для досягнення своїх цілей. Розглянемо основні з них:

Управління системними ресурсами

- моніторинг та оптимізація використання оперативної пам'яті, регулювання розміру файлу підкачки;
- завершення непотрібних процесів та служб для звільнення ресурсів;
- налаштування параметрів енергоспоживання та продуктивності процесора.

Очищення жорстких дисків

- алгоритми сканування та видалення тимчасових файлів, файлів cookies, історії браузера;
- визначення та видалення дублікатів файлів;
- очищення кошика для повної деінсталяції програм.

Оптимізація реєстру Windows

- алгоритми пошуку та усунення пошкоджених, застарілих або зайвих записів в реєстрі;
- стиснення та дефрагментація реєстру для підвищення продуктивності.

Управління автозавантаженням

- сканування та видалення непотрібних програм з автозавантаження;
- відстеження та оптимізація служб, що запускаються разом із системою.

Дефрагментація дисків

– алгоритми дефрагментації для оптимізації розташування файлів на диску;
– технології паралельної дефрагментації для прискорення процесу.

Інструменти аналізу та діагностики

– сканування на наявність шкідливих програм, вірусів та рекламного ПЗ;
– аналіз системних помилок, конфліктів драйверів та інших проблем;
– збір та аналіз даних про продуктивність системи.

Технології захисту конфіденційності

– очищення історії браузера, кешу, автозаповнювачів;
– шифрування конфіденційних даних;
– видалення слідів активності користувача.

2 РОЗРОБКА ВЛАСНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ОПТИМІЗАЦІЇ ПК НА БАЗІ WINDOWS 10/11

2.1 Вибір технологій та середовища розробки

Для розробки власного програмного забезпечення для оптимізації ПК на базі Windows 10/11 буде використовуватися мова програмування Python та середовище розробки Visual Studio Code (VS Code).

Python – це сучасна мова програмування високого рівня з простим та зрозумілим синтаксисом, що дозволяє швидко створювати різноманітні програмні рішення. Для реалізації функцій оптимізації та взаємодії з операційною системою Windows 10/11 будуть використовуватися відповідні бібліотеки та модулі Python.

Visual Studio Code – це безкоштовний редактор вихідного коду з відкритим кодом, розроблений Microsoft. Незважаючи на свою легковагість, VS Code пропонує потужні можливості для розробки на різних мовах програмування, включаючи Python. Він підтримує інтелектуальне автодоповнення коду, налагодження, інтеграцію з системами контролю версій та багато інших корисних функцій.

Для роботи з Python у VS Code буде використовуватися розширення Python, що забезпечує повну підтримку цієї мови в середовищі розробки. Це розширення включає в себе інтелектуальний автодоповнювач коду, перевірку синтаксису в реальному часі, функції рефакторингу, налагодження та багато іншого.

Для створення графічного інтерфейсу користувача (GUI) буде використовуватися бібліотека PyQt або Tkinter, які дозволяють швидко розробляти переносні між платформами графічні додатки з сучасним інтерфейсом.

Використання Visual Studio Code як середовища розробки забезпечить зручний та гнучкий процес написання коду, налагодження та тестування програмного забезпечення для оптимізації ПК на базі Windows 10/11 мовою Python. Завдяки підтримці великої кількості корисних розширень та інтеграції з різними інструментами, VS Code є потужним та водночас простим у використанні середовищем для реалізації цього проекту.

2.2 Проектування архітектури та функціональності програмного продукту

На етапі проектування архітектури програмного забезпечення для оптимізації ПК на базі Windows 10/11 необхідно ретельно продумати його структуру, модулі та функціональні можливості. Правильне проектування забезпечить гнучкість, масштабованість та зручність подальшої розробки і супроводу програми.

2.2.1 Архітектура програмного забезпечення

Розроблене програмне забезпечення базується на архітектурі клієнт-серверної моделі з використанням бібліотеки PyQt для створення графічного інтерфейсу користувача. Основною концепцією є розділення логіки програми на окремі віджети (widgets), кожен з яких відповідає за певний функціональний аспект:

- DiskScanningThread – окремий потік для сканування дисків та пошуку тимчасових файлів. Він використовує сигнали PyQt для повідомлення про знайдені файли та загальний розмір;
- DiskScanningWidget – віджет для візуалізації процесу сканування та очищення дисків. Він взаємодіє з DiskScanningThread та відображає список знайдених файлів і вивід операцій;

– BrowserCleaningWidget – віджет для очищення кешу браузерів, таких як Google Chrome, Microsoft Edge та Mozilla Firefox;
– RecycleBinCleaningWidget – віджет для очищення кошика операційної системи;
– AutostartManagementWidget – віджет для управління автозавантаженням програм під час запуску системи;
– MainWindow – головне вікно програми, що поєднує всі віджети в єдиний інтерфейс за допомогою бокового меню та стекового макету (QStackedWidget).

2.2.2 Функціональність програмного продукту

Програмне забезпечення матиме такі основні функціональні можливості:

Сканування та очищення дисків

– пошук і видалення тимчасових файлів у різних каталогах системи;
– відображення списку знайдених файлів з можливістю їх видалення;
– вивід інформації про загальний розмір звільненого дискового простору.

Очищення браузерів

– очищення кешу для браузерів Google Chrome, Microsoft Edge та Mozilla Firefox.

Очищення кошика

– повне видалення вмісту кошика операційної системи

Управління автозавантаженням

– відображення списку програм, що запускаються під час завантаження системи;
– можливість додавання та видалення програм з автозавантаження.

2.3 Реалізація програмного коду

У розробленому програмному забезпеченні для оптимізації комп’ютерних систем на базі Windows реалізовано кілька основних функціональних модулів, які використовують різні бібліотеки, алгоритми та підходи. Розглянемо детально кожен з них.

2.3.1 Модуль сканування та очищення дисків (DiskScanningWidget)

Цей модуль реалізований у вигляді класу DiskScanningWidget, який успадковується від QWidget. Він містить графічний інтерфейс користувача з кнопками для запуску сканування та очищення дисків, список для відображення знайдених тимчасових файлів та текстове поле для виведення повідомлень.

Для сканування дисків використовується окремий потік DiskScanningThread, який успадковується від QThread. Цей потік виконує рекурсивний обхід заздалегідь визначених системних директорій, таких як тимчасові папки, кеш браузерів тощо. При знаходженні тимчасового файлу потік випромінює сигнал file_found, передаючи шлях до файлу та його розмір. Ці дані додаються до списку в GUI за допомогою сигналу та слоту.

Для видалення тимчасових файлів використовується функція clean_disks. Вона проходить через список знайдених файлів і намагається видалити кожен файл за допомогою функції os.remove. Повідомлення про успішне чи невдале видалення виводяться в текстове поле.

Повний код модуля представлений у додатку А.

2.3.2 Модуль очищення браузерів (BrowserCleaningWidget)

Цей модуль реалізований у вигляді класу BrowserCleaningWidget і також успадковується від QWidget. Він містить графічний інтерфейс з вспливаючим

списком для вибору браузера, кнопку для очищення та текстове поле для виводу повідомлень.

Функція clean_browser викликається при натисканні кнопки очищення. Вона визначає шлях до кеш-папки вибраного браузера (Google Chrome, Microsoft Edge або Mozilla Firefox) та намагається видалити її вміст за допомогою функції shutil.rmtree. Повідомлення про успішне чи невдале очищення виводяться в текстове поле.

Повний код модуля представлений у додатку Б.

2.3.3 Модуль очищення кошика (RecycleBinCleaningWidget)

Цей модуль реалізований у вигляді класу RecycleBinCleaningWidget і успадковується від QWidget. Він містить кнопку для очищення кошика та текстове поле для виводу повідомлень.

Функція clean_recycle_bin викликається при натисканні кнопки очищення. Вона використовує функцію windll.shell32.SHEmptyRecycleBinW з бібліотеки ctypes для виклику системної функції очищення кошика в Windows. Повідомлення про успішне чи невдале очищення виводяться в текстове поле.

Повний код модуля представлений у додатку В.

2.3.4 Модуль управління автозавантаженням (AutostartManagementWidget)

Цей модуль реалізований у вигляді класу AutostartManagementWidget і успадковується від QWidget. Він містить список для відображення програм, що запускаються при старті системи, кнопки для додавання та видалення програм з автозавантаження, а також текстове поле для виводу повідомлень.

Функція populate_autostart_list сканує системний каталог автозавантаження та заповнює список програмами, що там міститься.

Функція add_program відкриває діалогове вікно для вибору виконавчого файлу (.exe) та копіює вибраний файл до каталогу автозавантаження.

Функція remove_program видаляє вибрані в списку програми з каталогу автозавантаження.

Всі ці модулі об’єднані в головному вікні програми MainWindow, яке використовує бібліотеку PyQt5 для створення графічного інтерфейсу з бічною панеллю для переходу між модулями.

Розроблене програмне забезпечення демонструє використання різних бібліотек та підходів, таких як багатопотокове програмування, робота з файловою системою, взаємодія з системними функціями Windows та створення графічних інтерфейсів за допомогою PyQt5.

Повний код модуля представлений у додатку Г.

2.4 Тестування програми

Під час розробки програми для оптимізації комп’ютерних систем на базі Windows було проведено ряд тестувань для перевірки правильності роботи реалізованих функцій та модулів. Розглянемо основні етапи тестування та результати, отримані на кожному з них.

2.4.1 Модульне тестування

Тестування модуля DiskScanningWidget

Створив тестові тимчасові файли в різних системних директоріях (див. рис. 2.1):

– C:\Users\vlad1\AppData\Local\Temp;

– C:\Windows\Temp.

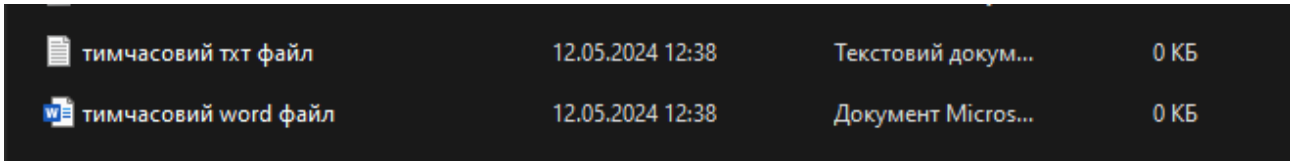

Рисунок 2.1 – Створені тимчасові файли

Зробивши сканування можемо помітити, що знайшло створені тимчасові файли, після їх знаходження можемо зробити очищення (див. рис.2.2).

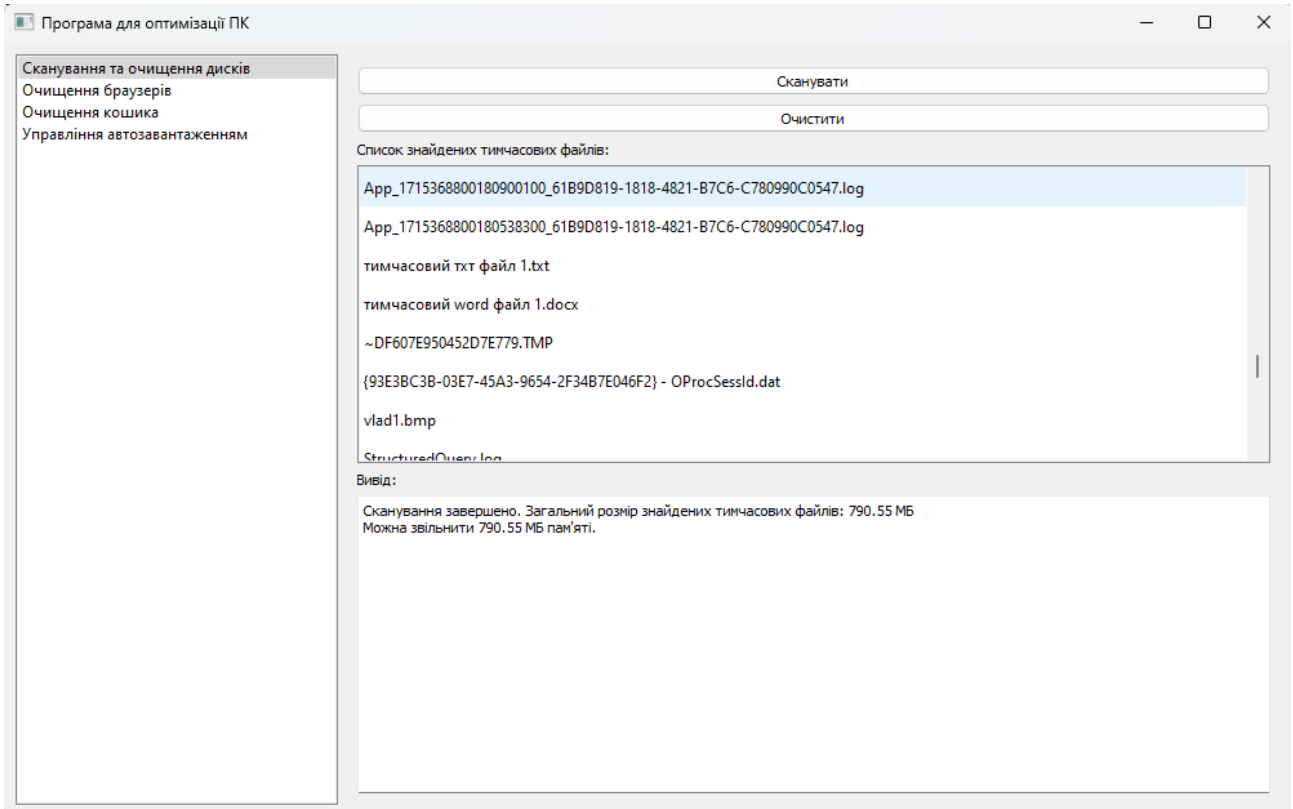

Рисунок 2.2 – Сканування тимчасових файлів

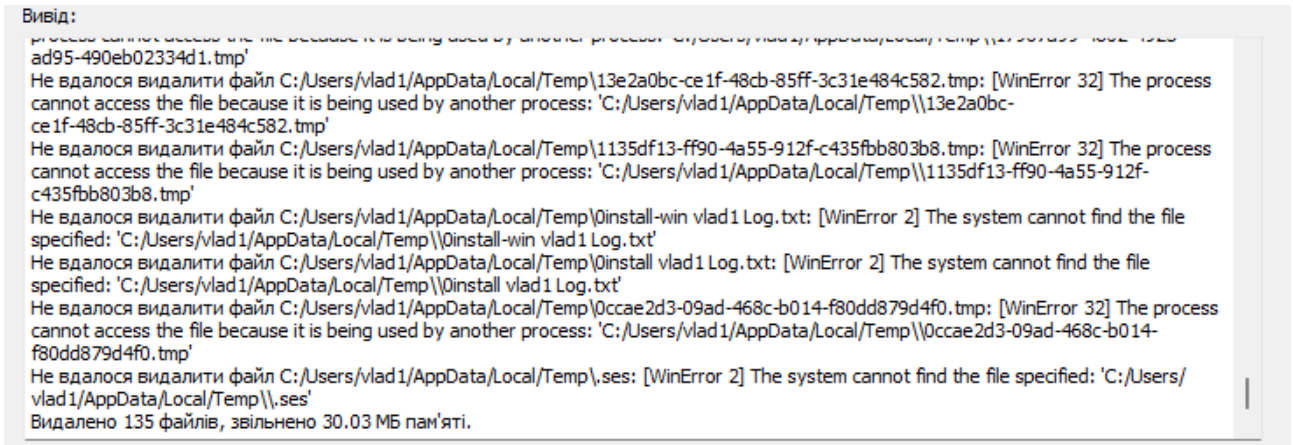

Рисунок 2.3 – Вивід після очищення тимчасових файлів

У виводі можна помітити, що не вдалося видалити деякі файли (див. рис.2.3), це значить, що ці файли зараз використовуються, або обмежені правами, і через це не можна їх видалити. Також бачимо скільки файлів було видалено, і скільки звільнилося пам’яті.

Перевіряємо системні директорії у яких були створені тимчасові файли (див. рис.2.4).

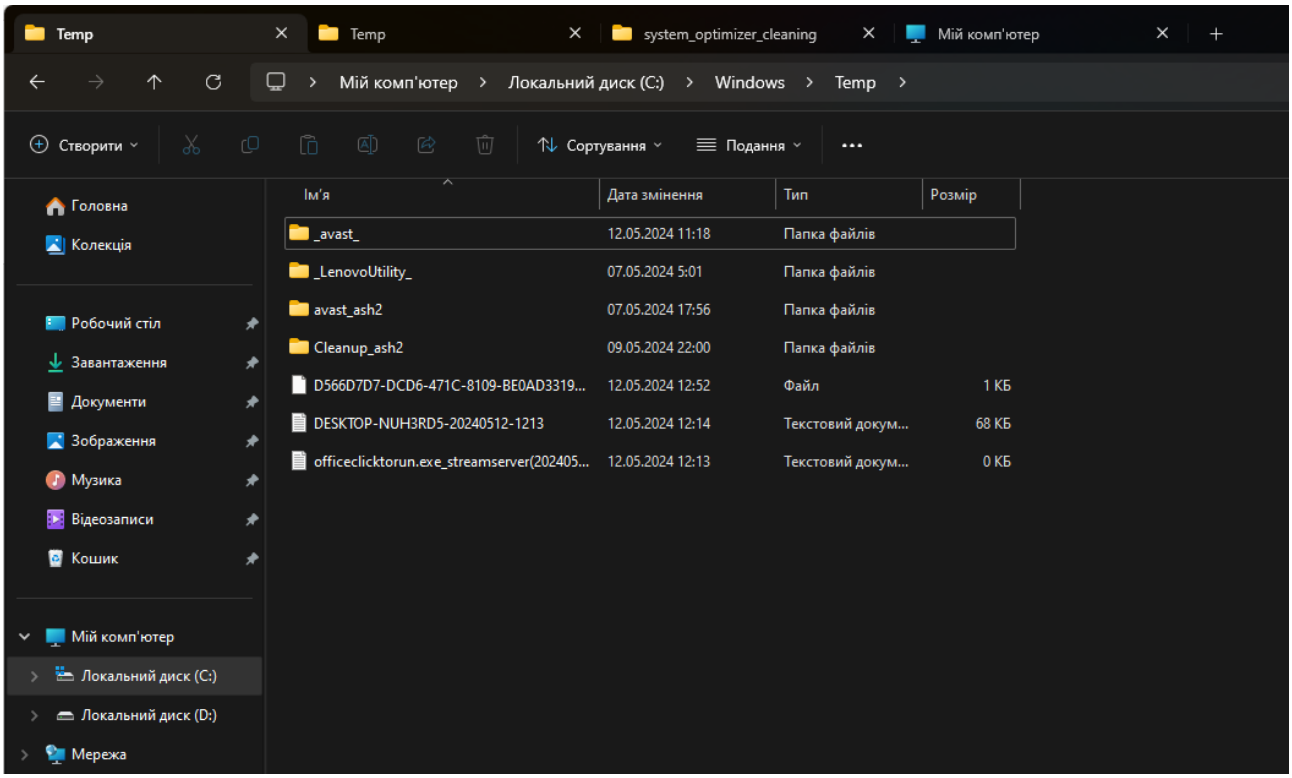

Рисунок 2.4 – C:\Windows\Temp

Продивившись директорії, впевнилися, що тимчасові файли і справді видалилися (див. рис.2.5).

Тестування модуля BrowserCleaningWidget

На вибір є 3 браузери, в яких можна видалити кеш (див. рис.2.6).

Так частіше використовується Google Chrome можна протестувати очистку кешу на цьому браузері.

Як відомо кеш браузеру Google Chrome знаходиться за цим шляхом C:\Users\vlad1\AppData\Local\Google\Chrome\User_Data\Default, тому переходимо по ньому і перевіряємо чи є кеш.

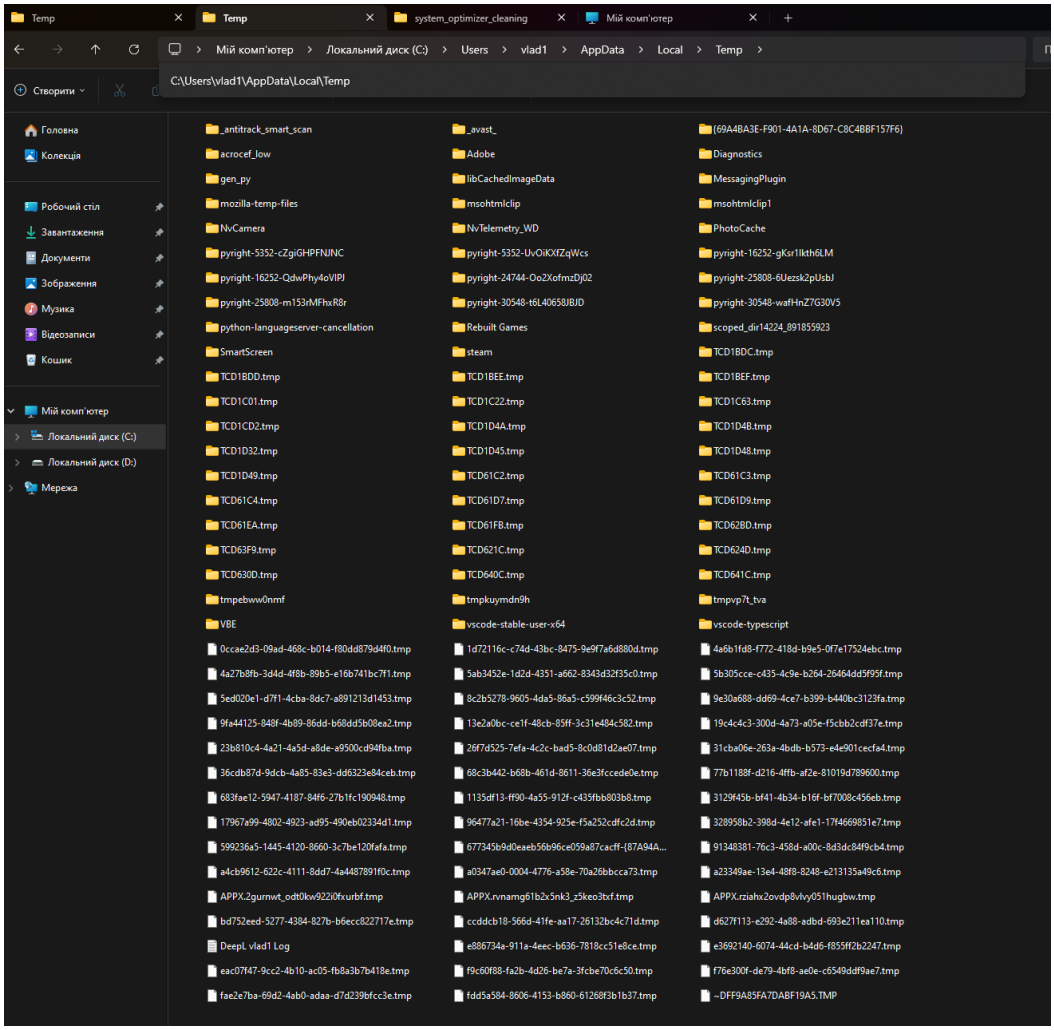

Рисунок 2.5 – C:\Users\vlad1\AppData\Local\Temp

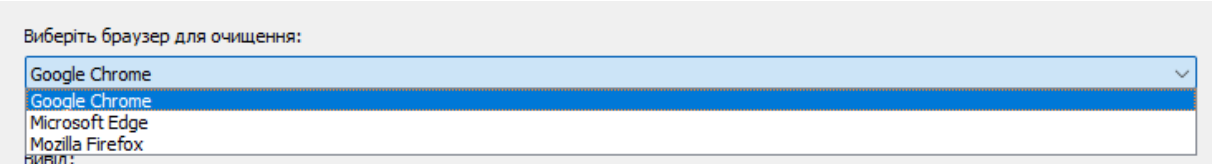

Рисунок 2.6 – Вибір браузера для очищення кешу

Бачимо, що кеш є, тому ми можемо провести очищення (див. рис. 2.7–2.8).

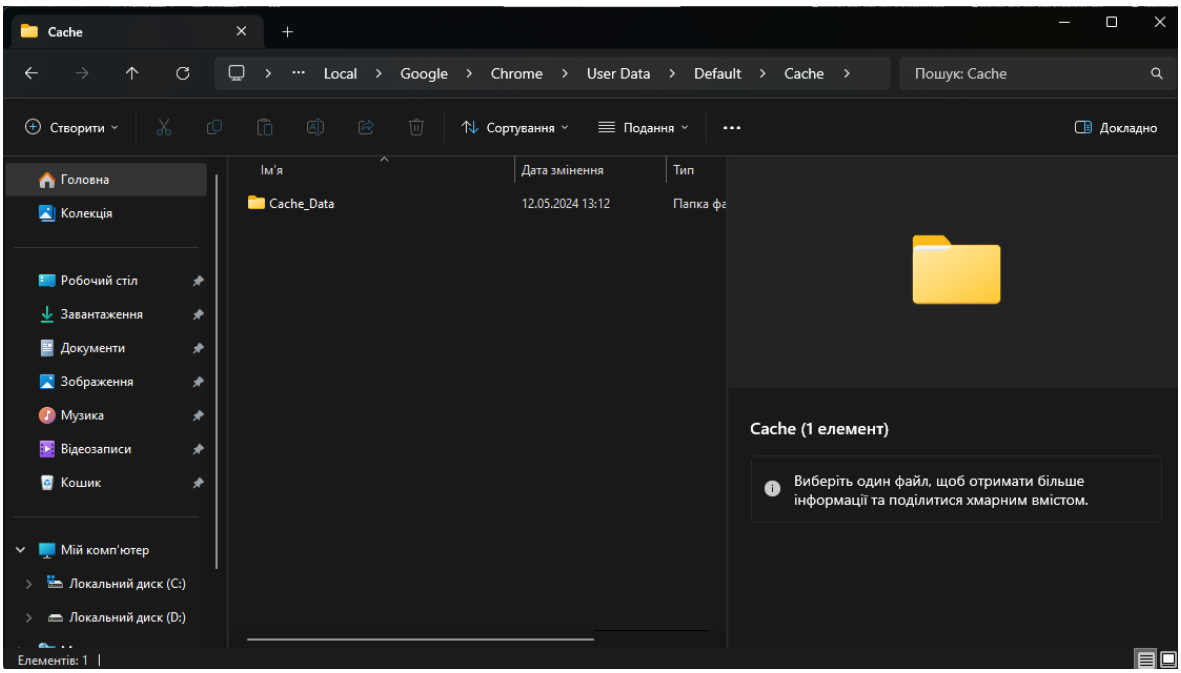

Рисунок 2.7 – Кеш Google Chrome

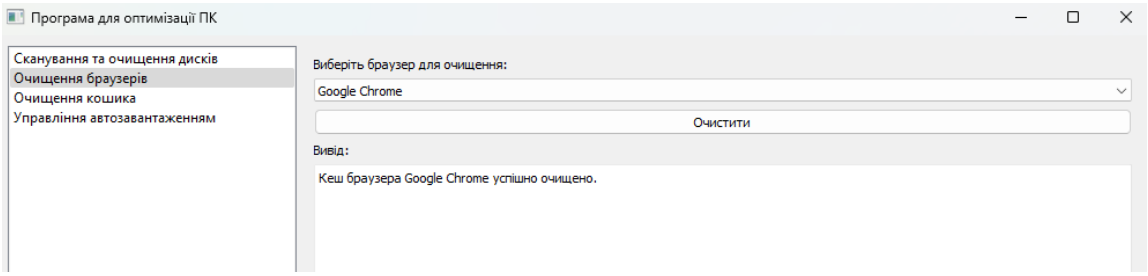

Рисунок 2.8 – Вивід результату очищення кешу

Папка з кешем видалилися успішно (див. рис. 2.9).

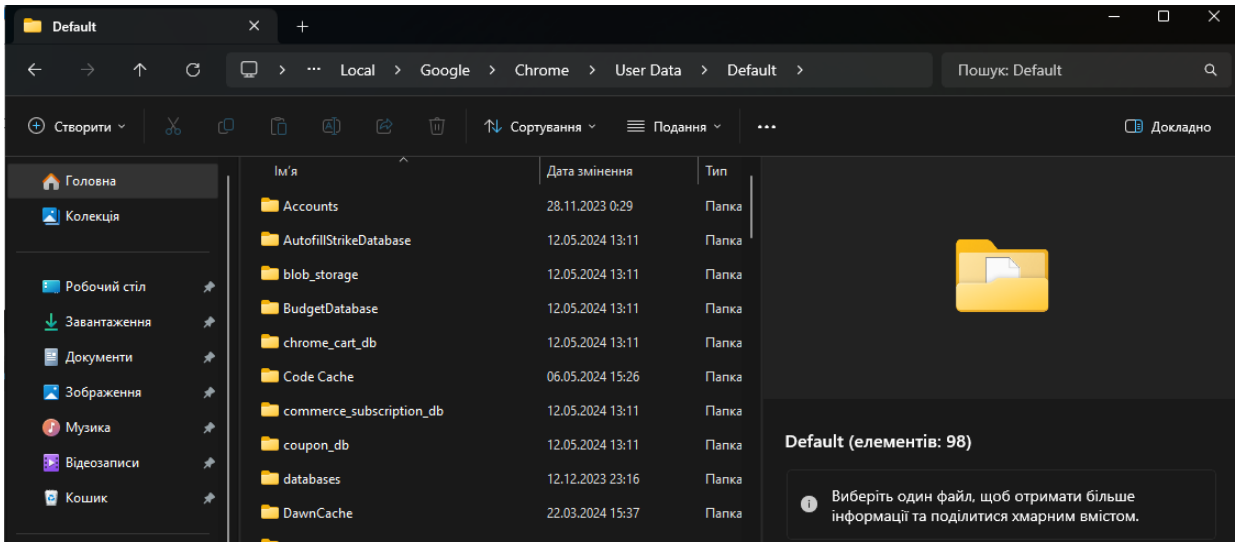

Рисунок 2.9 – Перевірка видалення кешу

Тестування модуля RecycleBinCleaningWidget

Кошик заповнив сміттям, щоб перевірити його очищення (див. рис.2.10).

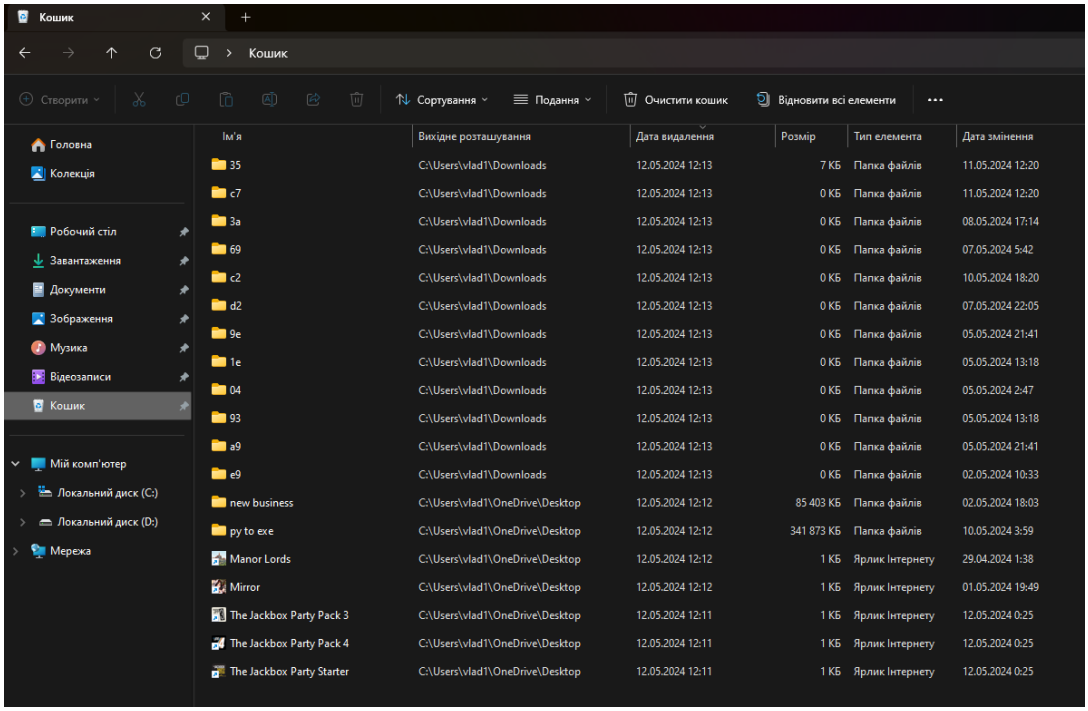

Рисунок 2.10 – Кошик

При натисканні на кнопку “Очистити кошик”, видає запит, який запитує, чи впевнений я, що хочу остаточно очистити кошик (див. рис. 2.11–2.13).

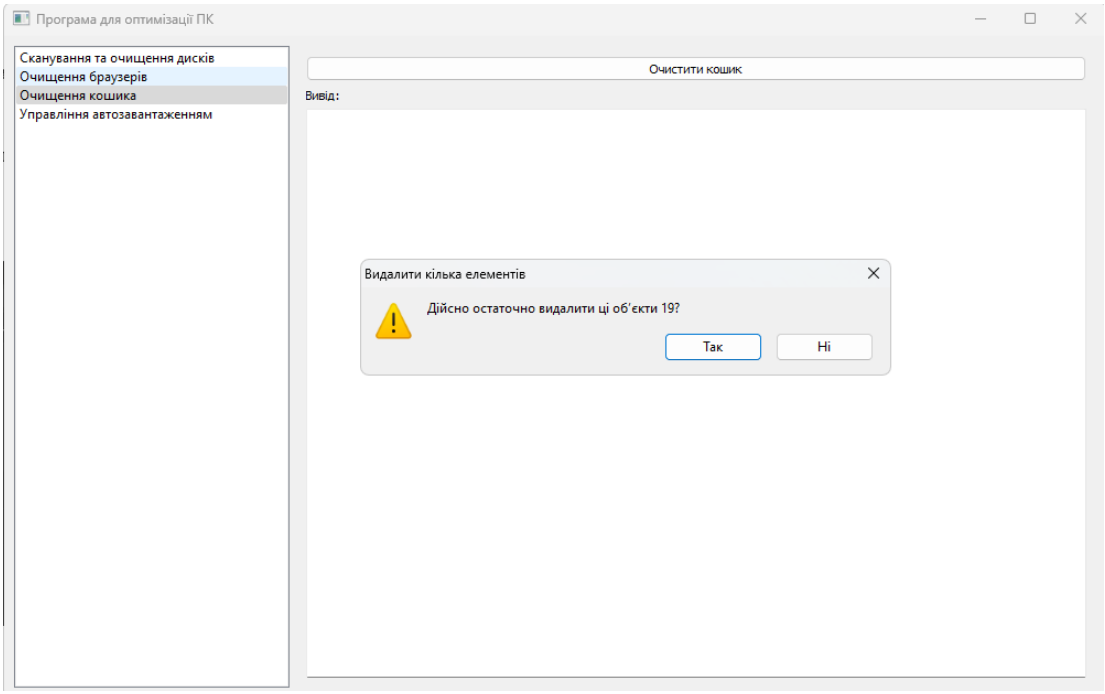

Рисунок 2.11 – Підтвердження очистки кошика

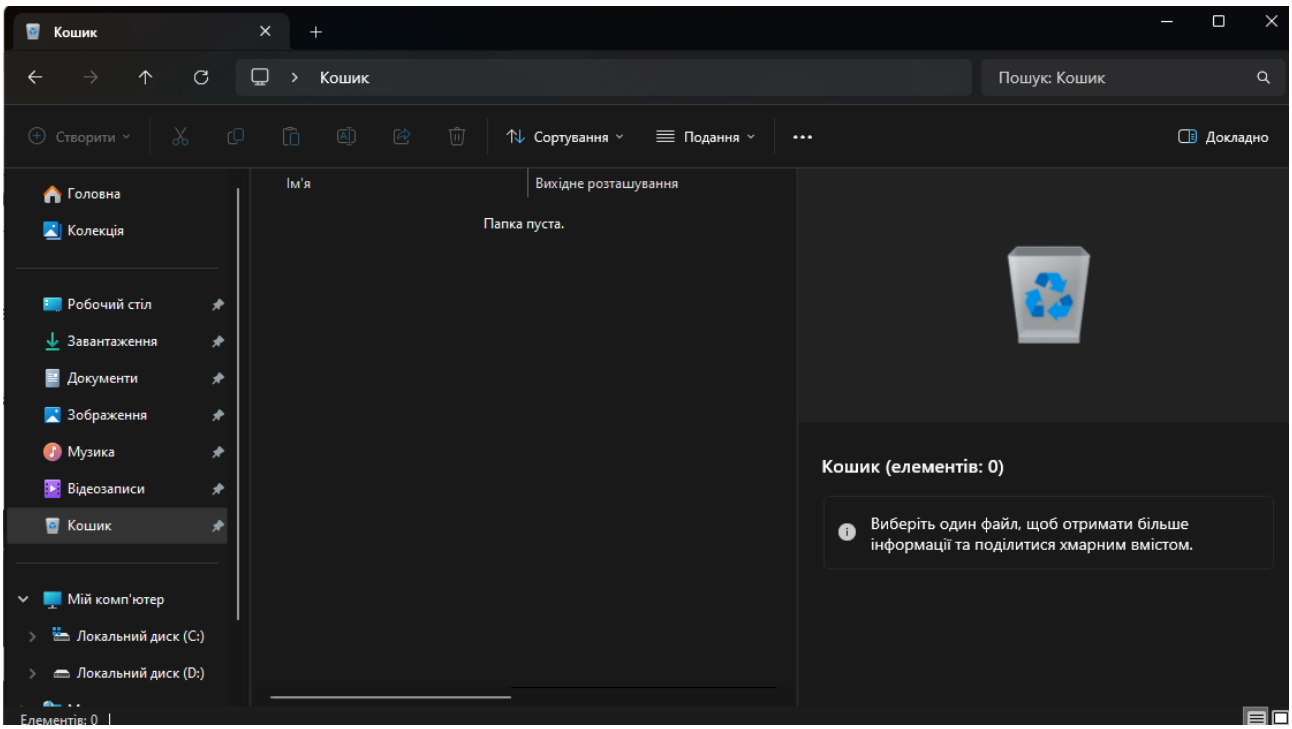

Рисунок 2.12 – Кошик очистився

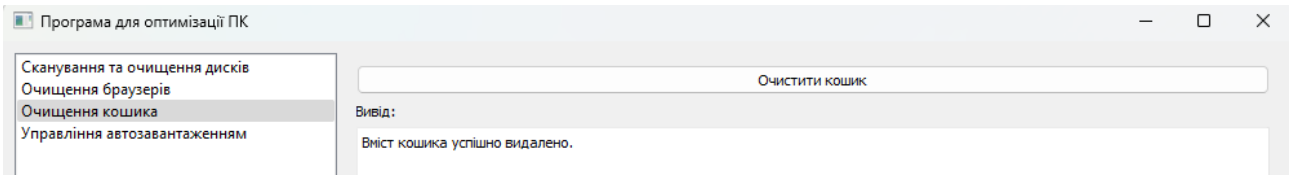

Рисунок 2.13 – Вивід в програмі успішного очищення

Тестування модуля RecycleBinCleaningWidget

Спробуємо додати якусь програму у автозавантаження, додам Wise Memory Optimizer, це програма, яка очищає оперативну пам’ять (див. рис.2.14).

Бачимо, в списку програма з’явилася (див. рис.2.15), перевіримо директорію куди додаються програми для автозавантаження C:\Users\vlad1\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup (див. рис.2.16).

Ще можемо перевірити за допомогою Диспетчера завдань в якому показуються програми, які завантажуються при запуску ПК (див. рис.2.17).

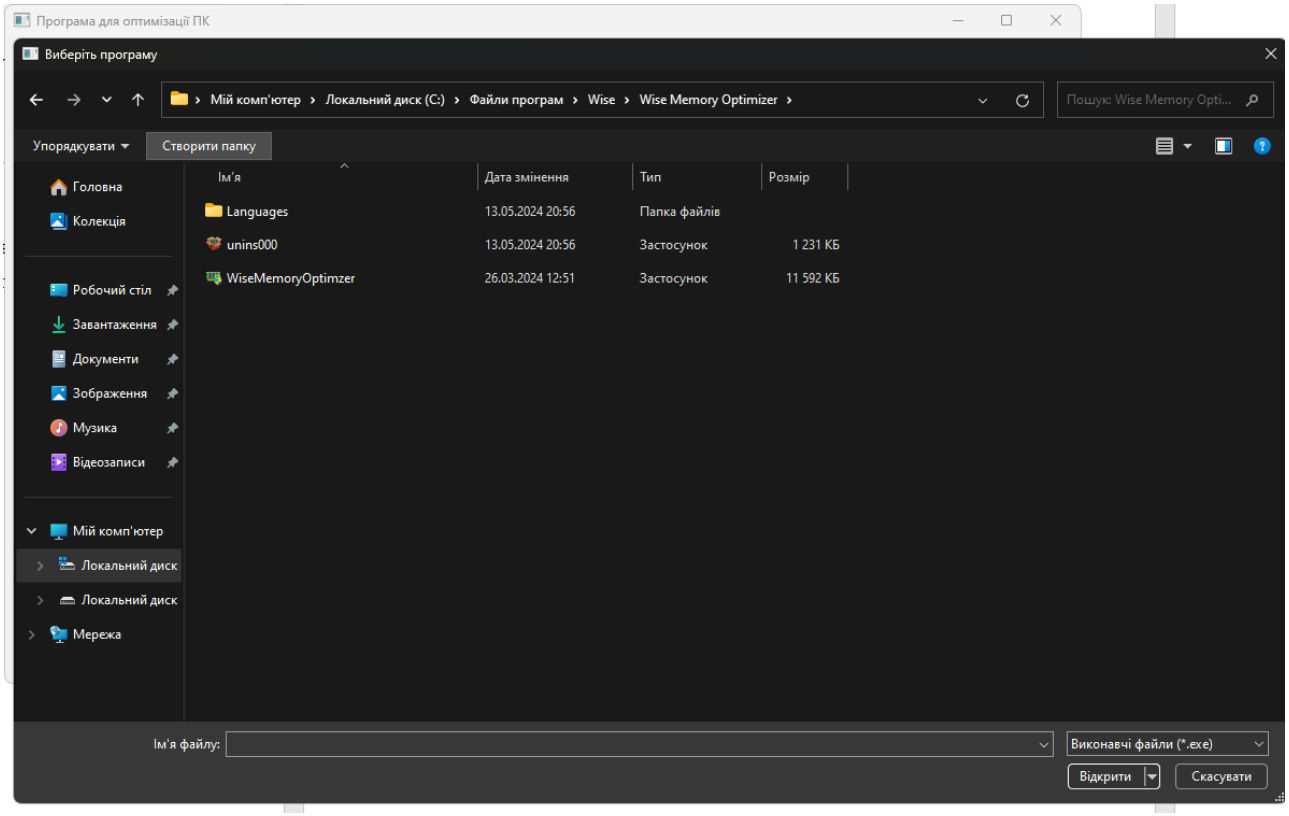

Рисунок 2.14 – Додавання програми у автозавантаження

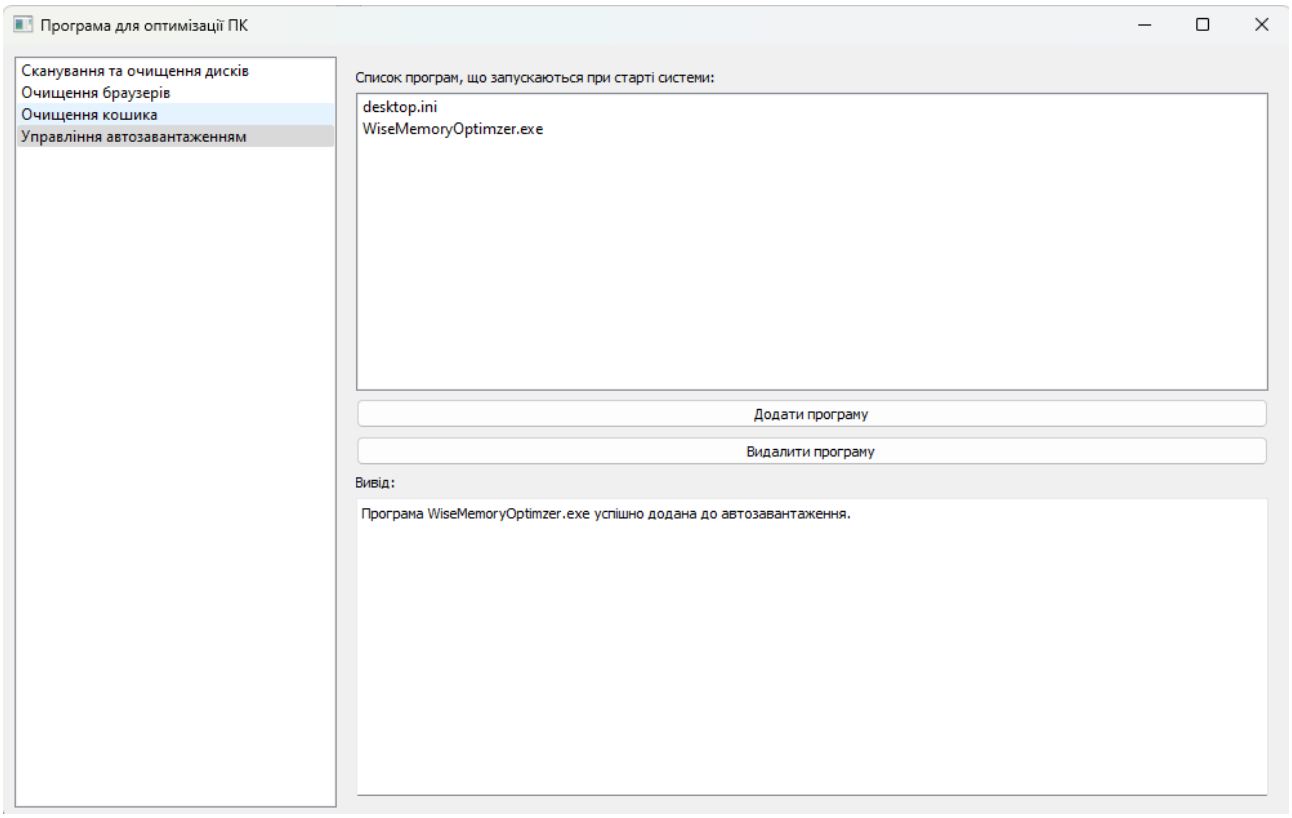

Рисунок 2.15 – Інтерфейс Управління автозавантаженням

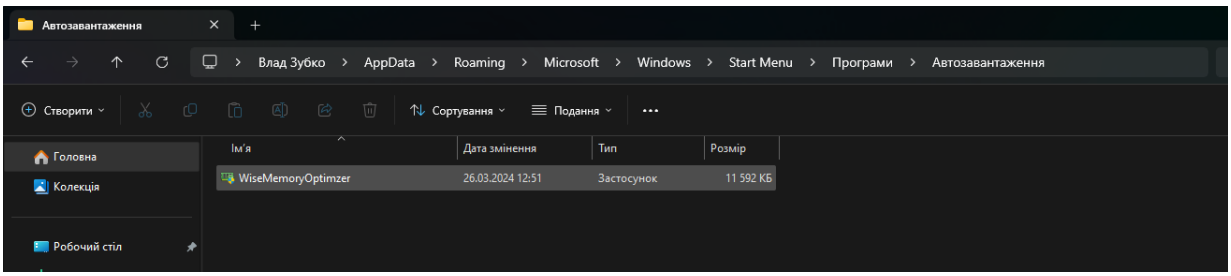

Рисунок 2.16 – Папка з програмами для автозавантаження

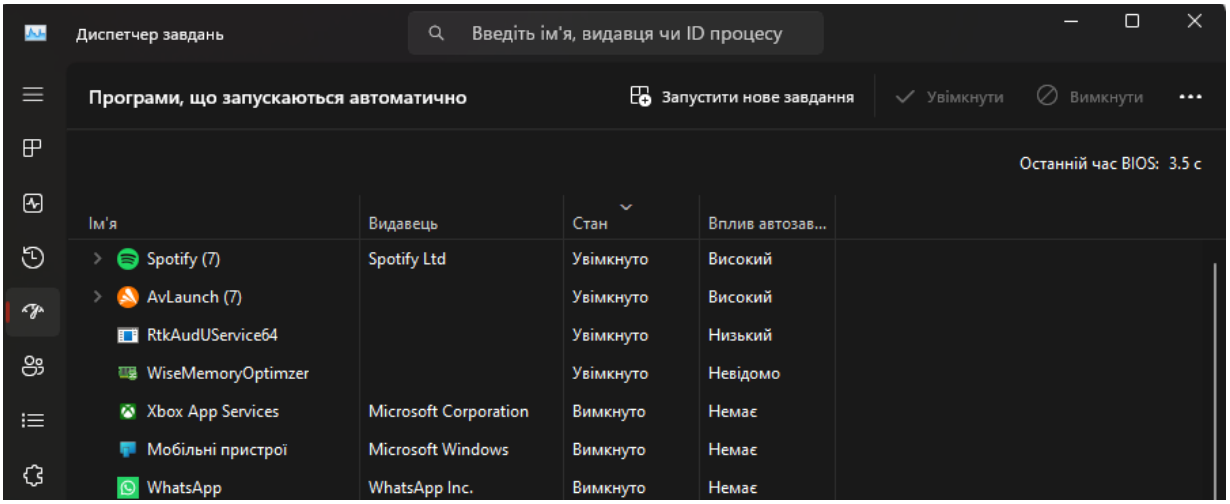

Рисунок 2.17 – Диспетчер завдань

Тепер спробуємо видалити програму з автозавантаження (див. рис. 2.18–2.19).

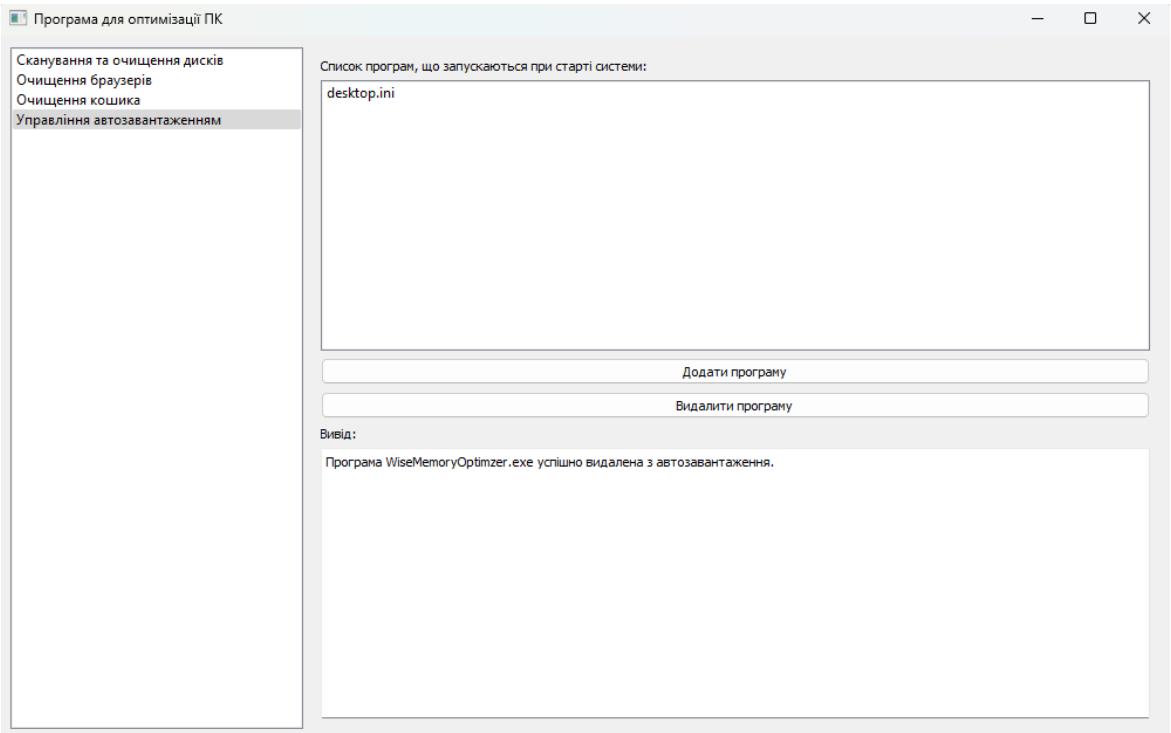

Рисунок 2.18 – Видалення програми з Управління автозавантаженням

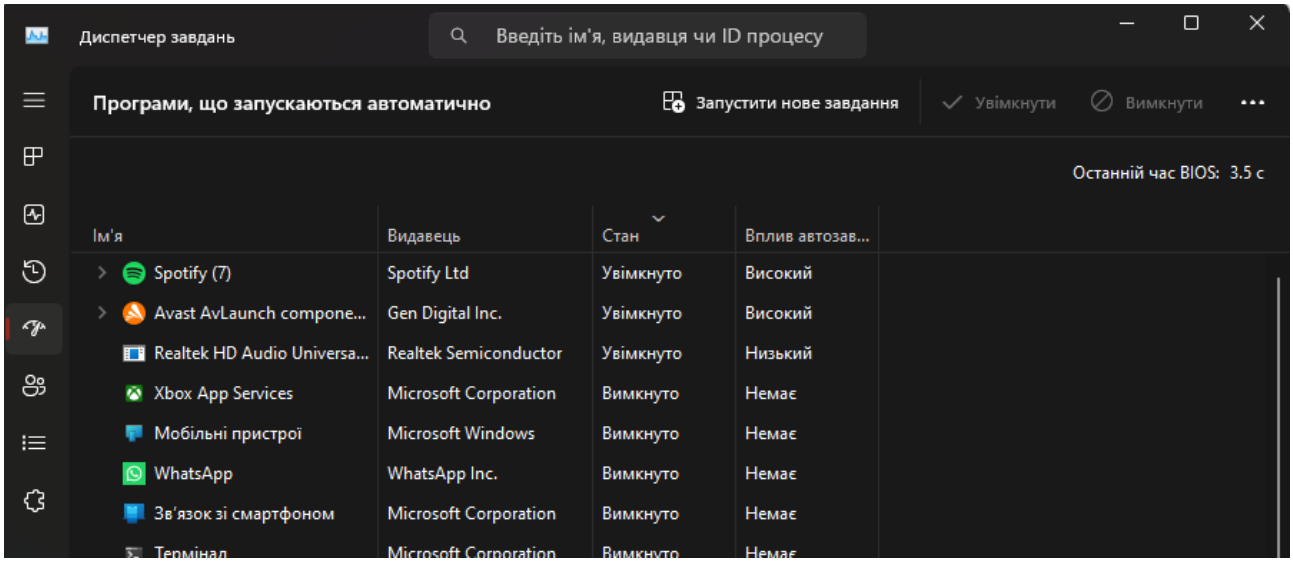

Рисунок 2.19 – Диспетчер завдань без програми

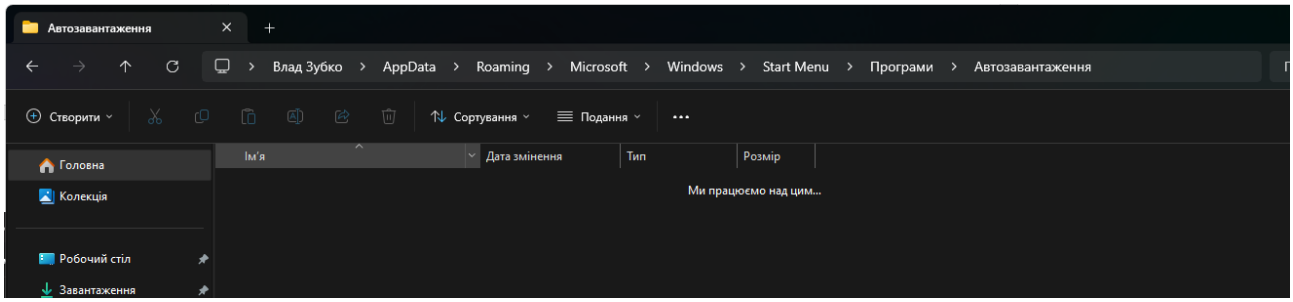

Рисунок 2.20 – Папка без програми для автозавантаження

3 ПЕРСПЕКТИВИ ПОДАЛЬШОГО РОЗВИТКУ

3.1 Можливості для вдосконалення та розширення функціональності

Незважаючи на досягнуті результати, розроблене програмне забезпечення для оптимізації комп'ютерних систем має значний потенціал для вдосконалення та розширення функціональності. Розглянемо основні напрямки можливих покращень:

Реалізація модуля оптимізації реєстру Windows

- розробка алгоритмів пошуку та усунення пошкоджених, застарілих або зайвих записів у реєстрі;
- імплементація функцій стиснення та дефрагментації реєстру для підвищення продуктивності системи.

Додавання модуля управління системними ресурсами

- моніторинг та оптимізація використання оперативної пам'яті;
- регулювання розміру файлу підкачки;
- завершення непотрібних процесів та служб для звільнення ресурсів;
- налаштування параметрів енергоспоживання та продуктивності процесора.

Розширення функціональності очищення жорстких дисків

- реалізація алгоритмів визначення та видалення дублікатів файлів;
- додавання можливості повної деінсталяції програм шляхом очищення відповідних каталогів та записів у реєстрі.

Вдосконалення модуля очищення браузерів

- додавання підтримки очищення кешу для інших популярних браузерів, таких як Opera тощо;
- реалізація функції видалення файлів cookie, історії переглядів та інших даних браузерів.

Покращення графічного інтерфейсу користувача

– оптимізація розміщення елементів та загального дизайну для кращої зручності використання;
– додавання можливості перегляду статистики та звітів про виконані операції оптимізації;
– інтеграція системи налаштувань та персоналізації програми.

ВИСНОВКИ

У рамках кваліфікаційної роботи було розроблено програмне забезпечення для оптимізації комп’ютерних систем на базі Windows. Хоча не всі заплановані функції були повністю реалізовані, було виконано значний обсяг роботи.

Проведено дослідження технологій, алгоритмів та підходів, що використовуються в програмах для оптимізації ПК. Зокрема, розглянуто питання управління системними ресурсами, очищення жорстких дисків та управління автозавантаженням.

Реалізовано модуль сканування та очищення дисків (DiskScanningWidget). Цей модуль використовує багатопотоковий підхід для сканування системних директорій у пошуках тимчасових файлів. Знайдені файли відображаються у списку, і користувач може вибірково видаляти їх.

Розроблено модуль очищення браузерів (BrowserCleaningWidget). Реалізовано функцію для очищення кешу браузера Google Chrome шляхом видалення відповідної кеш-директорії.

Створено модуль очищення кошика (RecycleBinCleaningWidget), який дозволяє користувачеві повністю очистити кошик операційної системи Windows.

Імплементовано модуль управління автозавантаженням (AutostartManagementWidget). Цей модуль надає можливість переглядати список програм, що запускаються під час старту системи, додавати нові програми до автозавантаження та видаляти непотрібні.

Всі ці модулі були інтегровані в єдине головне вікно програми (MainWindow) з використанням бібліотеки PyQt5 для створення графічного інтерфейсу користувача.

Проведено модульне тестування реалізованих функцій та модулів. Під час тестування були виявлені та виправлені деякі помилки, забезпечивши коректну роботу розроблених компонентів.

Хоча не всі заплановані технології та підходи були реалізовані, наприклад, оптимізація реєстру Windows та управління системними ресурсами, розроблене програмне забезпечення демонструє використання різноманітних алгоритмів, бібліотек та підходів для досягнення оптимізації комп’ютерних систем у певних аспектах.

Загалом, виконана робота є вагомим внеском у розробку інструментів для оптимізації та підвищення продуктивності комп’ютерних систем на базі Windows.

ПЕРЕЛІК ПОСИЛАНЬ

1. 20 tips and tricks to increase PC performance on Windows 10. Windowscentral. URL : https://www.windowscentral.com/tips-tricks-increase-pc-performance-windows-10/ (дата звернення: 12.04.2024).

2. Tips to improve PC performance in Windows. Microsoft. URL : https://support.microsoft.com/en-us/windows/tips-to-improve-pc-performance-in-windows-b3b3ef5b-5953-fb6a-2528-4bbed82fba96/ (дата звернення: 12.04.2024).

3. 8 Ways to Free Up Disk Space on Windows. Howtogeek. URL : https://www.howtogeek.com/125923/free-up-disk-space-on-windows/ (дата звернення: 14.04.2024).

4. How to launch apps automatically during startup on Windows 10. Windowscentral. URL : https://www.windowscentral.com/how-launch-apps-automatically-during-startup-windows-10/ (дата звернення: 15.04.2024).

5. How to Empty Recycle Bin in Windows 10? Minitool. URL : https://www.minitool.com/news/how-to-empty-recycle-bin.html (дата звернення: 15.04.2024).

6. Why Is It A Good Idea To Clear Your Browser’s Cache? Robots. URL : https://robots.net/tech/why-is-it-a-good-idea-to-clear-your-browsers-cache/#google_vignette/ (дата звернення: 12.04.2024).

7. The Benefits of Clearing Cache on Your Browser Regularly. Consumersearch. URL : https://www.consumersearch.com/technology/benefits-clearing-cache-browser-regularly/ (дата звернення: 17.04.2024).

8. Azam Seyedi. Computer Memory and Data Storage: exploration. Norway: 2024. 84p. (дата звернення: 07.04.2024).

9. Sukanta Nayak. Fundamentals of Optimization Techniques with Algorithms: Academic Press. India: 2021. 305p. (дата звернення: 12.04.2024).

ДОДАТОК А

DiskScanningWidget та DiskScanningThread

```
class DiskScanningThread(QThread):
    file_found = pyqtSignal(str, int)  # Сигнал для повідомлення про знайдений тимчасовий файл та його розмір
    error_occurred = pyqtSignal(str)  # Сигнал для повідомлення про помилку
    scan_finished = pyqtSignal()  # Сигнал про завершення сканування

    def __init__(self):
        super().__init__()
        self.temp_dirs = [
            QDir.tempPath(),
            os.path.join(os.environ.get("TEMP", ""), ""),
            os.path.join(os.environ.get("TMP", ""), ""),
            os.path.join(os.path.expanduser("~\\AppData\\Local\\Temp"), ""),
            os.path.expanduser("~\\AppData\\Local\\Microsoft\\Windows\\INetCache"),
            os.path.join(os.environ.get("WINDIR", "C:\\Windows"), "Temp"),
            os.path.join(os.environ.get("LOCALAPPDATA", ""), "Temp"),
            os.path.join(os.environ.get("SYSTEMROOT", "C:\\Windows"), "Prefetch"),
            os.path.join(os.environ.get("SYSTEMROOT", "C:\\Windows"), "SoftwareDistribution", "Download"),
            os.path.join(os.environ.get("SYSTEMROOT", "C:\\Windows"), "System32", "LogFiles"),
            os.path.join(os.environ.get("SYSTEMROOT", "C:\\Windows"), "System32", "Wbem", "Logs"),
            os.path.join(os.environ.get("SYSTEMROOT", "C:\\Windows"), "Logs"),
            os.path.expanduser("~\\AppData\\Local\\Microsoft\\Windows\\Temporary Internet Files"),
            os.path.expanduser("~\\AppData\\Local\\Microsoft\\Windows\\Explorer"),
            os.path.expanduser("~\\AppData\\Local\\Microsoft\\Windows\\Burn\\Burn"),
        ]
        self.total_size = 0  # Загальний розмір знайдених тимчасових файлів

    def run(self):
        try:
            for temp_dir in self.temp_dirs:
                for root, dirs, files in os.walk(temp_dir):
                    for file in files:
                        file_path = os.path.join(root, file)
                        file_size = os.path.getsize(file_path)
                        self.total_size += file_size
                        self.file_found.emit(file_path, file_size)

        except Exception as e:
            self.error_occurred.emit(str(e))
        finally:
```

```
            self.scan_finished.emit()

# Віджет для сканування та очищення дисків від тимчасових файлів
class DiskScanningWidget(QWidget):
    def __init__(self):
        super().__init__()
        self.setWindowTitle("Сканування та очищення дисків")
        self.scan_thread = None  # Змінна для зберігання потоку сканування

        self.scan_button = QPushButton("Сканувати")
        self.clean_button = QPushButton("Очистити")
        self.files_list_widget = QListWidget()
        self.output_text_edit = QTextEdit()
        self.output_text_edit.setReadOnly(True)

        layout = QVBoxLayout(self)
        layout.addWidget(self.scan_button)
        layout.addWidget(self.clean_button)
        layout.addWidget(QLabel("Список знайдених тимчасових файлів:"))
        layout.addWidget(self.files_list_widget)
        layout.addWidget(QLabel("Вивід:"))
        layout.addWidget(self.output_text_edit)

        self.scan_button.clicked.connect(self.scan_disks)
        self.clean_button.clicked.connect(self.clean_disks)
        self.files_list_widget.setSelectionMode(QListWidget.ExtendedSelection)
        self.total_size_to_delete = 0  # Скидання лічильника розміру для видалення

    def scan_disks(self):
        """
        Сканування дисків для пошуку тимчасових файлів.
        """
        # Зупинка попереднього потоку сканування, якщо він існує
        if self.scan_thread and self.scan_thread.isRunning():
            self.scan_thread.terminate()
            self.scan_thread.wait()

        self.output_text_edit.clear()
        self.files_list_widget.clear()

        self.scan_thread = DiskScanningThread()
        self.scan_thread.file_found.connect(self.add_file_to_list)
        self.scan_thread.error_occurred.connect(self.output_text_edit.append)
        self.scan_thread.scan_finished.connect(self.scan_finished)
        self.scan_thread.start()

    def add_file_to_list(self, file_path, file_size):
        item = QListWidgetItem(os.path.basename(file_path))
        item.setSizeHint(QSize(0, 30))
        item.setData(Qt.UserRole, file_path)
        item.setData(Qt.UserRole + 1, file_size)
        self.files_list_widget.insertItem(0, item)  # Додати елемент на початок списку
```

```
            self.total_size_to_delete += file_size

        def format_file_size(self, size):
            units = ["Б", "КБ", "МБ", "ГБ", "ТБ"]
            for unit in units:
                if size < 1024:
                    return f"{size:.2f} {unit}"
                size /= 1024
            return f"{size:.2f} ЕБ"

        def scan_finished(self):
            total_size = self.scan_thread.total_size
            self.output_text_edit.append(
                f"Сканування завершено. Загальний розмір знайдених тимчасових файлів: {self.format_file_size(total_size)}"
            )
            self.output_text_edit.append(
                f"Можна звільнити {self.format_file_size(self.total_size_to_delete)} пам’яті."
            )

        def clean_disks(self):
            """
            Очищення дисків від тимчасових файлів.
            """
            self.output_text_edit.clear()

            deleted_files_count = 0
            deleted_size = 0

            for index in range(self.files_list_widget.count()):
                item = self.files_list_widget.item(index)
                file_path = item.data(Qt.UserRole)
                file_size = item.data(Qt.UserRole + 1)
                try:
                    os.remove(file_path)
                    deleted_files_count += 1
                    deleted_size += file_size
                    self.output_text_edit.append(f"Видалено файл: {file_path}")
                except Exception as e:
                    self.output_text_edit.append(f"Не вдалося видалити файл {file_path}: {e}")

            self.files_list_widget.clear()
            self.total_size_to_delete = 0
            self.output_text_edit.append(f"Видалено {deleted_files_count} файлів, звільнено {self.format_file_size(deleted_size)} пам’яті.")
```

ДОДАТОК Б

BrowserCleaningWidget

```
class BrowserCleaningWidget(QWidget):
    def __init__(self):
        super().__init__()
        self.setWindowTitle("Очищення браузерів")

        self.browsers_combo_box = QComboBox()
        self.clean_button = QPushButton("Очистити")
        self.output_text_edit = QTextEdit()
        self.output_text_edit.setReadOnly(True)

        layout = QVBoxLayout(self)
        layout.addWidget(QLabel("Виберіть браузер для очищення:"))
        layout.addWidget(self.browsers_combo_box)
        layout.addWidget(self.clean_button)
        layout.addWidget(QLabel("Вивід:"))
        layout.addWidget(self.output_text_edit)

        self.populate_browsers_combo_box()
        self.clean_button.clicked.connect(self.clean_browser)

    def populate_browsers_combo_box(self):
        """
        Додавання списку доступних браузерів у комбо-бокс.
        """
        browsers = ["Google Chrome", "Microsoft Edge", "Mozilla Firefox"]
        self.browsers_combo_box.addItems(browsers)

    def clean_browser(self):
        """
        Очищення кешу вибраного браузера.
        """
        browser = self.browsers_combo_box.currentText()
        if not browser:
            self.output_text_edit.append("Будь ласка, виберіть браузер для очищення.")
            return

        browser_cache_path = None
        if browser == "Google Chrome":
            browser_cache_path = os.path.expanduser("~/AppData/Local/Google/Chrome/User Data/Default/Cache")
        elif browser == "Microsoft Edge":
            browser_cache_path = os.path.expanduser("~/AppData/Local/Microsoft/Edge/User Data/Default/Cache")
        elif browser == "Mozilla Firefox":
```

```
                firefox_profiles_path = os.path.join(os.path.expanduser("~"), "AppData", "Local", "Mozilla", "Firefox", "Profiles")
                profiles = os.listdir(firefox_profiles_path)
                if profiles:
                    browser_cache_path = os.path.join(firefox_profiles_path, profiles[0], "cache2")
                else:
                    self.output_text_edit.append("Не вдалося знайти профіль Mozilla Firefox.")
                    return

            if browser_cache_path:
                if os.path.exists(browser_cache_path):
                    try:
                        shutil.rmtree(browser_cache_path)
                        self.output_text_edit.append(f"Кеш браузера {browser} успішно очищено.")
                    except Exception as e:
                        self.output_text_edit.append(f"Не вдалося очистити кеш браузера {browser}: {e}")
                else:
                    self.output_text_edit.append(f"Кеш браузера {browser} відсутній.")
            else:
                self.output_text_edit.append(f"Не вдалося знайти кеш браузера {browser}.")
```

ДОДАТОК В

RecycleBinCleaningWidget

```
class RecycleBinCleaningWidget(QWidget):
    def __init__(self):
        super().__init__()
        self.setWindowTitle("Очищення кошика")

        self.clean_button = QPushButton("Очистити кошик")
        self.output_text_edit = QTextEdit()
        self.output_text_edit.setReadOnly(True)

        layout = QVBoxLayout(self)
        layout.addWidget(self.clean_button)
        layout.addWidget(QLabel("Вивід:"))
        layout.addWidget(self.output_text_edit)

        self.clean_button.clicked.connect(self.clean_recycle_bin)

    def clean_recycle_bin(self):
        """
        Очищення вмісту кошика.
        """
        try:
            windll.shell32.SHEmptyRecycleBinW(None, None, 0x0004) # SHCNE_UPDATERECYCLEBINDATA
            self.output_text_edit.append("Вміст кошика успішно видалено.")
        except Exception as e:
            self.output_text_edit.append(f"Не вдалося видалити вміст кошика: {e}")
```

ДОДАТОК Г

AutostartManagementWidget

```
class AutostartManagementWidget(QWidget):
    def __init__(self):
        super().__init__()
        self.setWindowTitle("Управління автозавантаженням")

        self.autostart_list_widget = QListWidget()
        self.add_button = QPushButton("Додати програму")
        self.remove_button = QPushButton("Видалити програму")
        self.output_text_edit = QTextEdit()
        self.output_text_edit.setReadOnly(True)

        layout = QVBoxLayout(self)
        layout.addWidget(QLabel("Список програм, що запускаються при старті системи:"))
        layout.addWidget(self.autostart_list_widget)
        layout.addWidget(self.add_button)
        layout.addWidget(self.remove_button)
        layout.addWidget(QLabel("Вивід:"))
        layout.addWidget(self.output_text_edit)

        self.add_button.clicked.connect(self.add_program)
        self.remove_button.clicked.connect(self.remove_program)
        self.populate_autostart_list()

    def populate_autostart_list(self):
        """
        Заповнення списку автозавантаження поточними програмами.
        """
        self.autostart_list_widget.clear()  # Очищення існуючих елементів

        autostart_path = os.path.expanduser("~\\AppData\\Roaming\\Microsoft\\Windows\\Start Menu\\Programs\\Startup")
        for filename in os.listdir(autostart_path):
            filepath = os.path.join(autostart_path, filename)
            item = QListWidgetItem(filename)
            item.setData(Qt.UserRole, filepath)
```

```
            self.autostart_list_widget.addItem(item)

        def add_program(self):
            """
            Додавання програми до автозавантаження.
            """
            file_path, _ = QFileDialog.getOpenFileName(self, "Виберіть програму", "", "Виконавчі файли (*.exe)")
            if file_path:
                autostart_path = os.path.expanduser("~\\AppData\\Roaming\\Microsoft\\Windows\\Start Menu\\Programs\\Startup")
                shutil.copy2(file_path, autostart_path)
                self.output_text_edit.clear()  # Очищення виводу перед додаванням повідомлення
                self.output_text_edit.append(f"Програма {os.path.basename(file_path)} успішно додана до автозавантаження.")
                self.populate_autostart_list()

        def remove_program(self):
            """
            Видалення програми з автозавантаження.
            """
            selected_items = self.autostart_list_widget.selectedItems()
            if selected_items:
                for item in selected_items:
                    file_path = item.data(Qt.UserRole)
                    os.remove(file_path)
                    self.output_text_edit.clear()  # Очищення виводу перед додаванням повідомлення
                    self.output_text_edit.append(f"Програма {os.path.basename(file_path)} успішно видалена з автозавантаження.")
                self.populate_autostart_list()
            else:
                self.output_text_edit.append("Будь ласка, виберіть програму для видалення.")
```

Декларація

академічної доброчесності

здобувача ступеня вищої освіти ЗНУ

Я, Зубко Владислав Заурійович

студент __4__ курсу, ___денної____ форми навчання, математичного факультету, спеціальності __122 комп'ютерні науки_____, адреса електронної пошти ___panbobr18real@gmail.com_____ ,

– підтверджую, що написана мною кваліфікаційна робота бакалавра на тему «___Розробка програмного софту для оптимізації ПК на операційній системі Windows 10/11_____» відповідає вимогам академічної доброчесності та не містить порушень, що визначені у ст. 42 Закону України «Про освіту», зі змістом яких ознайомлений/ознайомлена;

– заявляю, що надана мною для перевірки електронна версія роботи є ідентичною її друкованій версії;

– згоден/згодна на перевірку моєї роботи на відповідність критеріям академічної доброчесності у будь-який спосіб, у тому числі за допомогою інтернет-системи, а також на архівування моєї роботи в базі даних цієї системи.

Студент

_____	_____	_____
(дата)	(підпис)	(прізвище, ініціали)

Науковий керівник

_____	_____	_____
(дата)	(підпис)	(прізвище, ініціали)