

Міністерство освіти і науки України
Криворізький національний університет
Кафедра моделювання та програмного забезпечення

КВАЛІФІКАЦІЙНА РОБОТА
на здобуття ступеня вищої освіти бакалавра
зі спеціальності 121 – Інженерія програмного забезпечення

На тему: Розробка програмного засобу для керування телеметрією та конфіденційністю в ОС сімейства Windows

Засвідчую, що в цій
кваліфікаційній роботі немає
запозичень із праць інших
авторів без відповідних
посилань.

Студент гр. ІПЗ–22–2

_____ / А. С. Трегубенко /

Керівник кваліфікаційної
роботи

_____ / Д. В. Швець /

В.о. завідувача кафедри

_____ / А. М. Стрюк /

Кривий Ріг

2026

Криворізький національний університет

Факультет: Інформаційних технологій

Кафедра: Моделювання та програмного забезпечення

Ступінь вищої освіти: бакалавр

Спеціальність: 121 – Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ:

В.о. зав. кафедри

_____ А. М. Стрюк

«____» _____ 2026 р.

ЗАВДАННЯ

на кваліфікаційну роботу

студенту групи ІПЗ–22–2 Трегубенко Андрію Сергійовичу

1. Тема: «Розробка програмного засобу для керування телеметрією та конфіденційністю в ОС сімейства Windows» затверджена наказом по КНУ № 285с від «28» травня 2026 р.
2. Термін подання студентом закінченої роботи: «31» травня 2026 р.
3. Вихідні дані по роботі: створений програмний продукт має забезпечити підвищення конфіденційності користувача під час використання ОС Windows.
4. Зміст пояснювальної записки (перелік питань, що їх треба розробити): провести аналіз існуючих на ринку аналогічних програмних продуктів, обґрунтувати функціонал розроблюваної системи, створити програмне забезпечення, здійснити тестування розробленого додатку.
5. Перелік ілюстративного матеріалу: функціональна схема, блок–схема алгоритму, зображення екранних форм додатку.

Календарний план:

№	Найменування етапів кваліфікаційної роботи	Термін виконання етапів роботи
1	Розгляд літературних джерел та пошук інтернет-ресурсів з заданої тематики	08.01.26 – 24.01.26
2	Аналіз існуючих методів вирішення проблеми	25.01.26 – 07.02.26
3	Формулювання актуальності роботи і постановка завдань	08.02.26 – 18.02.26
4	Оформлення матеріалів першого розділу роботи	18.02.26 – 02.03.26
5	Створення функціональної системи та алгоритму додатку	03.03.26 – 15.03.26
6	Оформлення матеріалів другого розділу роботи	16.04.26 – 09.04.26
7	Розробка баз даних, інтерфейсу програмного забезпечення, програмних модулів	10.04.26 – 01.05.26
8	Оформлення додатків	02.05.26 – 07.05.26
9	Тестування розробленої програми	08.05.26 – 15.05.26
10	Оформлення пояснювальної записки	16.05.26 – 30.05.26

Дата видачі завдання: «07» січня 2026 р.

Студент: _____ / А. С. Трегубенко /

Керівник роботи: _____ / Д. В. Швець /

РЕФЕРАТ

ТЕЛЕМЕТРИЯ, КОНФІДЕНЦІЙНІСТЬ, ОПЕРАЦІЙНА СИСТЕМА,
БЕЗПЕКА, СИСТЕМНИЙ РЕЄСТР, ПРОФІЛІ ЗАХИСТУ.

Пояснювальна записка: 61 с., 11 рис., 1 дод., 16 джерел.

Об'єкт дослідження – процес керування налаштуваннями приватності та оптимізації роботи в сучасних операційних системах.

Предмет дослідження – методи та програмні інструменти контролю телеметрії і системних служб в ОС сімейства Windows.

Мета роботи – створення програмного засобу для аналізу, налаштування та керування параметрами збору діагностичної інформації.

Розроблений програмний засіб надає можливість взаємодіяти з диспетчером служб та системним реєстром для припинення небажаної мережевої активності. Для полегшення роботи створено підсистему готових профілів, яка дозволяє застосовувати комплексні налаштування безпеки за одну дію. З метою забезпечення надійності впроваджено функцію автоматичного генерування текстових документів із резервними копіями конфігурації, що гарантує збереження цілісності операційної системи перед внесенням глобальних змін.

Отримані результати підтверджують ефективність запропонованого підходу. Створений інструмент є готовим до практичного використання і дозволяє захищати свою персональну інформацію, не маючи глибоких навичок системного адміністрування.

ABSTRACT

**TELEMETRY, PRIVACY, OPERATING SYSTEM, SECURITY, SYSTEM
REGISTRY, PROTECTION PROFILES.**

Explanatory note: 61 pages, 11 figures, 1 appendix, 16 sources.

Object of study: the process of managing privacy settings and optimizing performance in modern operating systems.

Subject of study: methods and software tools for controlling telemetry and system services in the Windows family of operating systems.

Purpose of the work: development of a software tool for analyzing, configuring, and managing diagnostic data collection parameters.

The developed software tool provides the ability to interact with the service manager and the system registry to terminate unwanted network activity. To facilitate the workflow, a subsystem of ready-made profiles has been created, which allows applying comprehensive security settings in a single action. To ensure reliability, a function for automatically generating text documents with configuration backups has been implemented, guaranteeing the preservation of the operating system's integrity before making global changes.

The obtained results confirm the effectiveness of the proposed approach. The created tool is ready for practical use and allows users to protect their personal information without having deep system administration skills.

ЗМІСТ

ВСТУП.....	7
1 АНАЛІЗ МЕХАНІЗМІВ ТЕЛЕМЕТРІЇ ТА КОНФІДЕНЦІЙНОСТІ В ОС WINDOWS.....	9
1.1. Поняття телеметрії в сучасних операційних системах	9
1.2. Механізми збору діагностичних даних у Windows	10
1.3. Основні механізми забезпечення конфіденційності в Windows.....	12
1.4. Аналіз існуючих програмних засобів керування конфіденційністю Windows	14
1.5 Висновки по проведеному аналізу	18
2 ПРОЄКТУВАННЯ ПРОГРАМНОГО ЗАСОБУ КЕРУВАННЯ ТЕЛЕМЕТРІЄЮ ТА КОНФІДЕНЦІЙНІСТЮ	20
2.1 Опис вимог до засобу керування телеметрією та конфіденційністю.....	20
2.2 Обґрунтування вибору засобів розробки	21
2.3 Проєктування функціональних модулів	23
2.3.1 Модуль аналізу телеметрії.....	23
2.3.2 Модуль керування конфіденційністю	24
2.3.3 Модуль профілів конфіденційності.....	25
2.3.4 Модуль резервного копіювання	25
3 РОЗРОБКА ПРОГРАМНОГО ЗАСОБУ ДЛЯ КЕРУВАННЯ ТЕЛЕМЕТРІЄЮ ТА КОНФІДЕНЦІЙНІСТЮ В ОС СІМЕЙСТВА WINDOWS	30
3.1 Функціональна схема	30
3.2 Алгоритм роботи	32
3.3 Демонстрація розробленої програми	34
ВИСНОВКИ	38
ПЕРЕЛІК ПОСИЛАНЬ	40
Додаток А - Програмний код.....	42

ВСТУП

Комп'ютерні технології сьогодні інтегровані в усі сфери життя людини, а операційні системи виступають базовим програмним забезпеченням, що дозволяє користувачеві взаємодіяти з апаратними компонентами та глобальною мережею. Найпопулярнішою на ринку настільних рішень залишається ОС сімейства Windows. Разом із розширенням функціоналу та переходом до концепції «операційна система як послуга», у нових версіях цієї системи значно зросли обсяги автоматичного збору даних [1]. Хоча розробники зазначають, що така інформація збирається виключно для діагностики помилок та покращення роботи системи, агресивна політика щодо телеметрії викликає обґрунтоване занепокоєння стосовно захисту особистих даних.

Актуальність теми дослідження зумовлена необхідністю забезпечення конфіденційності користувача під час щоденної роботи за комп'ютером. Стандартні засоби налаштування приватності в Windows часто є недостатніми, а самі параметри розпорошені по різних розділах системи [2], що робить їхнє комплексне адміністрування незручним. Більше того, велика кількість фонових служб, які здійснюють моніторинг активності, не можуть бути деактивовані через звичайний графічний інтерфейс системи без прямого втручання в реєстр або групові політики. Тому розробка спеціалізованого програмного засобу, який би надавав зручний доступ до прихованих параметрів і дозволяв обмежувати небажану мережеву активність операційної системи, є своєчасним і важливим завданням.

Метою кваліфікаційної роботи є розробка програмного засобу для аналізу, налаштування та керування параметрами телеметрії і конфіденційності в операційних системах сімейства Windows.

Для досягнення поставленої мети було сформульовано такі завдання:

а) провести аналіз поточного стану проблеми збору діагностичних даних в ОС Windows та дослідити можливі ризики для кінцевого користувача.

б) розглянути існуючі програмні аналоги на ринку та виділити їхні переваги й недоліки для формування переліку вимог до власного додатку.

в) спроектувати архітектуру майбутнього програмного рішення та розробити інтуїтивно зрозумілий графічний інтерфейс.

г) програмно реалізувати необхідний функціонал для роботи з системним реєстром, службами та планувальником завдань.

д) провести тестування розробленого програмного засобу.

Об'єкт дослідження – процес керування налаштуваннями конфіденційності та оптимізації роботи в сучасних операційних системах.

Предмет дослідження – методи, алгоритми та програмні засоби контролю телеметрії і системних служб в ОС сімейства Windows.

Результатом роботи стане готовий до використання програмний додаток, який дозволить звичайним користувачам та системним адміністраторам централізовано налаштовувати рівень приватності. Це знизить ймовірність витоку персональної інформації та дозволить зменшити споживання ресурсів ПК за рахунок зупинки процесів збору даних.

1 АНАЛІЗ МЕХАНІЗМІВ ТЕЛЕМЕТРІЇ ТА КОНФІДЕНЦІЙНОСТІ В ОС WINDOWS

1.1. Поняття телеметрії в сучасних операційних системах

Телеметрія в контексті комп'ютерних технологій являє собою автоматизований процес збору даних з віддалених пристроїв та їх подальшої передачі розробнику для аналізу.

В операційних системах, зокрема сімейства Windows, цей механізм інтегровано на глибокому рівні архітектури. Основне призначення телеметричних служб полягає у забезпеченні безперебійної роботи програмного забезпечення, виявленні програмних помилок до того, як вони набудуть масового характеру, та загальному покращенні користувацького досвіду. Компанія-розробник використовує зібрані масиви інформації, щоб розуміти, на якому апаратному забезпеченні працює їхній продукт та які його функції є найбільш затребуваними серед аудиторії.

Залежно від поточних налаштувань, операційна система збирає різні види діагностичних даних. Зазвичай їх поділяють на обов'язкові (або базові) та необов'язкові, які часто називають повними. До базового набору входить технічна інформація, необхідна для підтримки безпеки та стабільності пристрою: характеристики центрального процесора, обсяг встановленої оперативної пам'яті, версія самої операційної системи та звіти про критичні збої в роботі драйверів.

Натомість повний пакет діагностики охоплює набагато більші обсяги інформації. Він може включати детальну статистику використання конкретних прикладних програм [3], фрагменти оперативної пам'яті під час зависань системи, що потенційно містять особисті файли, а також історію перегляду вебсторінок у разі використання вбудованих системних засобів для доступу до інтернету.

Роль телеметрії у підтримці операційної системи є вагомою. Завдяки безперервному потоку інформації від мільйонів активних пристроїв, інженери можуть оперативно розробляти оновлення безпеки та виправляти знайдені вразливості. Крім того, діагностичні звіти допомагають оптимізувати процес розповсюдження масштабних оновлень системи [4], оскільки механізми доставки аналізують сумісність. Якщо система сповіщає, що певний драйвер відеокарти спричиняє конфлікт на специфічній конфігурації обладнання, оновлення для аналогічних комп'ютерів тимчасово блокується, що дозволяє уникнути серйозних несправностей у користувачів.

Незважаючи на очевидні переваги для підтримки працездатності комп'ютера, такий масштабний збір даних створює суттєві ризики для приватності. Користувач рідко має змогу зрозуміти, який саме обсяг інформації надсилається на віддалені сервери, адже межа між системною діагностикою та збором персональної аналітики є досить розмитою. Сама операційна система не надає зручних і зрозумілих інструментів для гнучкого керування цими потоками інформації, обмежуючись переважно перемикачами в налаштуваннях, які не зупиняють фонові служби повністю. Саме тому виникає потреба в розробці спеціалізованого програмного засобу, який би дозволив аналізувати та обмежувати небажану активність системних компонентів, надаючи користувачеві контроль над власним середовищем без шкоди для стабільності роботи комп'ютера.

1.2. Механізми збору діагностичних даних у Windows

Операційні системи Windows використовують цілу низку вбудованих інструментів для реалізації завдань зі збору діагностичної інформації. Ці механізми працюють у фоновому режимі та мають високі системні привілеї, що ускладнює їхнє контролювання звичайними засобами налаштування. Щоб зрозуміти, як саме розроблюваний додаток буде обмежувати надсилання особистої інформації, необхідно детально проаналізувати базові компоненти системи, які відповідають за цю активність.

Головним каналом передачі діагностики є служба Connected User Experiences and Telemetry (відома також як DiagTrack). Вона відповідає за маршрутизацію, шифрування та відправлення зібраних даних безпосередньо на сервери розробника. Навіть при спробах зупинити її через стандартні вікна параметрів, ця служба часто відновлює свою роботу самостійно, особливо після встановлення накопичувальних оновлень. Це створює потребу в глибшому втручанні з боку створюваного програмного засобу, зокрема через модифікацію системного реєстру.

Вагомим елементом мережевої інфраструктури збору даних є служба dmwappushservice. Хоча її початкове призначення пов'язане з обробкою специфічних мережевих повідомлень WAP-маршрутизації, у сучасних версіях операційної системи вона активно співпрацює з телеметричними вузлами [5]. Згідно з технічними оглядами, ця служба допомагає координувати передачу діагностичних звітів та може використовуватися як резервний канал зв'язку для телеметрії, що робить її обов'язковою ціллю для деактивації в додатку, що розробляється.

Окрім безпосередньо системних служб, велику кількість інформації збирає Customer Experience Improvement Program (програма покращення якості програмного забезпечення). Цей компонент збирає статистику щодо використання конкретних додатків, частоти їхніх запусків та загальної продуктивності комп'ютера. Хоча формально користувач може відмовитися від участі в цій ініціативі, багато пов'язаних із нею завдань продовжують функціонувати в системі. Вони періодично пробуджують пристрій та формують звіти, що призводить до зайвого споживання ресурсів процесора та оперативної пам'яті.

Також заслуговує на увагу механізм Microsoft Compatibility Appraiser. Це заплановане системне завдання регулярно сканує файлову систему, встановлені програми та драйвери, щоб визначити готовність комп'ютера до отримання нових версій операційної системи. Цей процес часто стає причиною тривалого та високого навантаження на накопичувач даних.

Сканування сумісності генерує великі масиви телеметрії про апаратне забезпечення, які згодом відправляються через згадану раніше службу DiagTrack.

Розуміння архітектури та принципів роботи цих механізмів є необхідним етапом для проектування програмного рішення. Оскільки всі розглянуті компоненти реалізовані як системні служби або періодичні завдання, майбутній додаток повинен вміти взаємодіяти з відповідними програмними інтерфейсами ОС для їхнього надійного вимкнення та запобігання подальшому самостійному перезапуску.

1.3. Основні механізми забезпечення конфіденційності в Windows

Окрім інтегрованих системних служб збору діагностики, операційні системи сімейства Windows містять низку користувацьких налаштувань, які безпосередньо впливають на рівень приватності. Хоча розробники надають певні інструменти для контролю над цими параметрами [6], за замовчуванням більшість із них активована, що сприяє максимальному збору інформації для екосистеми сервісів. Аналіз цих механізмів є необхідним етапом для подальшого проектування розроблюваного додатку.

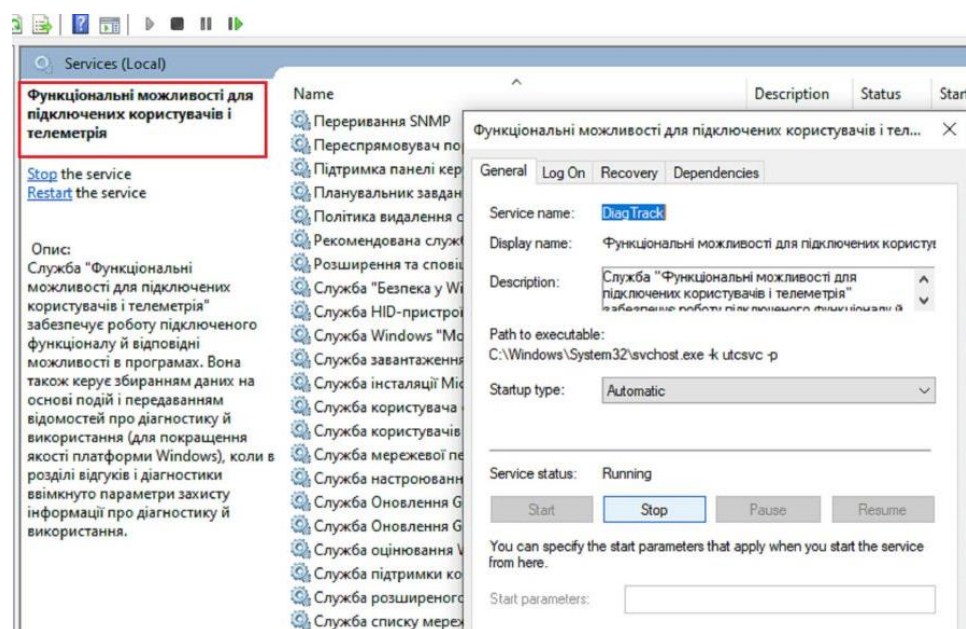


Рисунок 1.1 – Ручне відключення служби, що відповідає збір телеметрії

Одним із найбільш суперечливих елементів є рекламний ідентифікатор. Це унікальний номер, який присвоюється кожному обліковому запису пристрою для відстеження активності користувача в різних програмах. Зібрані дані використовуються для формування профілю інтересів і подальшого показу персоналізованої реклами. Вимкнення цього ідентифікатора через стандартне меню параметрів лише припиняє персоналізацію оголошень, проте для повного блокування створення такого профілю на рівні системи необхідно вносити зміни до системного реєстру, що й буде автоматизовано у створюваному програмному засобі.

Вагомим аспектом захисту особистих даних є контроль над геолокацією. Операційна система та встановлені застосунки можуть визначати фізичне місцезнаходження комп'ютера або ноутбука, використовуючи дані про підключення до мереж та інші доступні датчики [7]. Система також зберігає історію переміщень протягом певного часу. Щоб гарантувати, що стороннє програмне забезпечення не зможе таємно отримувати координати, має виконуватися примусове блокування служб визначення місцезнаходження на глобальному рівні.

Збір інформації також охоплює функцію збереження історії активності. Цей механізм фіксує перелік відкритих файлів, відвіданих вебсторінок та запущених програм, щоб дозволити користувачеві продовжити роботу з того ж місця на іншому пристрої. Незважаючи на загальну зручність, ця функція постійно синхронізує локальні дані з віддаленими хмарними серверами розробника ОС. Деактивація історії активності є кроком для зменшення обсягів особистої інформації, що залишають межі локального накопичувача.

На основі зібраної інформації система часто генерує персоналізовані рекомендації. Це поради, пропозиції щодо встановлення додаткових програм та підказки на екрані блокування або в меню системи. Для формування таких пропозицій аналізується текст, що вводиться з клавіатури, та загальний стиль використання комп'ютера. Вимкнення цих рекомендацій дозволяє не лише

обмежити надсилання статистики, але й позбутися зайвих елементів інтерфейсу, що відвертають увагу.

Окрему загрозу становлять параметри, що регулюють доступ до камери та мікрофона. Оскільки сучасне зловмисне програмне забезпечення може намагатися отримати прихований контроль над аудіо- та відеопристроями, керування цими дозволами має бути максимально надійним. Замість того, щоб перевіряти налаштування кожної окремої програми у вікнах операційної системи, можна реалізувати можливість заборони доступу до цих апаратних компонентів шляхом модифікації групових політик, гарантуючи захист від небажаного прослуховування або спостереження.

Отже, розглянуті механізми вимагають від користувача значних витрат часу на їхнє ручне адміністрування. Зведення цих розрізнених параметрів у єдиний зрозумілий інтерфейс стане одним із головних завдань при подальшій розробці програмного рішення.

1.4. Аналіз існуючих програмних засобів керування конфіденційністю Windows

Для формування обґрунтованих вимог до розроблюваного додатку необхідно проаналізувати наявні на ринку рішення. За останні роки незалежні розробники створили чимало утиліт для керування системними параметрами приватності, і порівняння їхніх можливостей допомагає краще зрозуміти, які підходи є найбільш вдалим для кінцевого користувача.

Одним із найпоширеніших на сьогодні рішень є утиліта O&O ShutUp10++ [8]. Ця програма здобула значну популярність завдяки тому, що пропонує великий набір перемикачів для деактивації телеметрії, згрупованих за логічними категоріями, такими як безпека, захист локальних даних та обмеження мережових запитів. Важливою перевагою цього засобу є його портативний формат: програма не потребує встановлення на накопичувач, що дозволяє запускати її безпосередньо зі змінного носія для швидкого обслуговування кількох пристроїв.

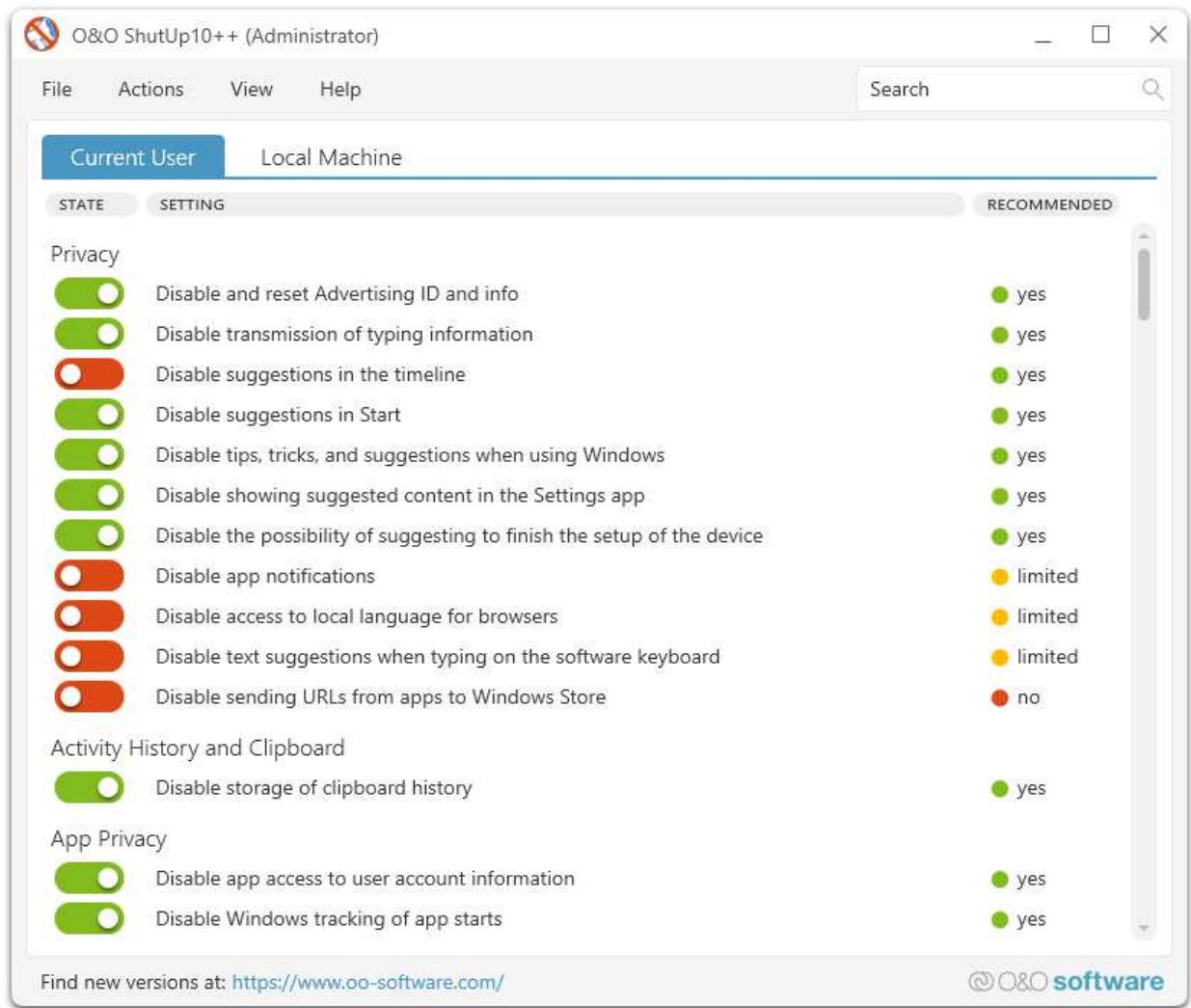


Рисунок 1.2 – Програма O&O ShutUp10++

Її сильною стороною є продумана вбудована система рекомендацій. Вона використовує інтуїтивне маркування кольоровими позначками, де зелений колір означає безпеку застосування, жовтий попереджає про можливі незручності в повсякденній роботі, а червоний вказує, що зміна відповідних параметрів з високою ймовірністю призведе до нестабільної роботи комп'ютера або порушить функціонування центру оновлень.

Проте ця утиліта має суттєвий недолік – закритий вихідний код. Оскільки для виконання поставлених завдань подібне програмне забезпечення обов'язково вимагає прав адміністратора, відсутність відкритого коду унеможливорює незалежну експертну перевірку. Користувач не може достеменно дізнатися, які саме модифікації вносяться до системного реєстру,

політик чи конфігурації фонових процесів під час її роботи. Такий підхід змушує покладатися виключно на репутацію компанії-розробника, що закономірно знижує загальний рівень довіри до інструменту серед фахівців з інформаційної безпеки та досвідчених користувачів.

Іншим популярним засобом є WPD (Windows Privacy Dashboard) [9]. Цей інструмент вирізняється сучасним візуальним оформленням, де всі налаштування згруповані за логічними категоріями, що значно полегшує навігацію. Окрім базового блокування передачі даних, WPD надає можливість видаляти системні програми, які неможливо деінсталиювати звичайним способом. Основним недоліком WPD є нестача текстових пояснень: недосвідчений користувач може деактивувати важливу системну службу, зовсім не розуміючи реальних наслідків своєї дії.

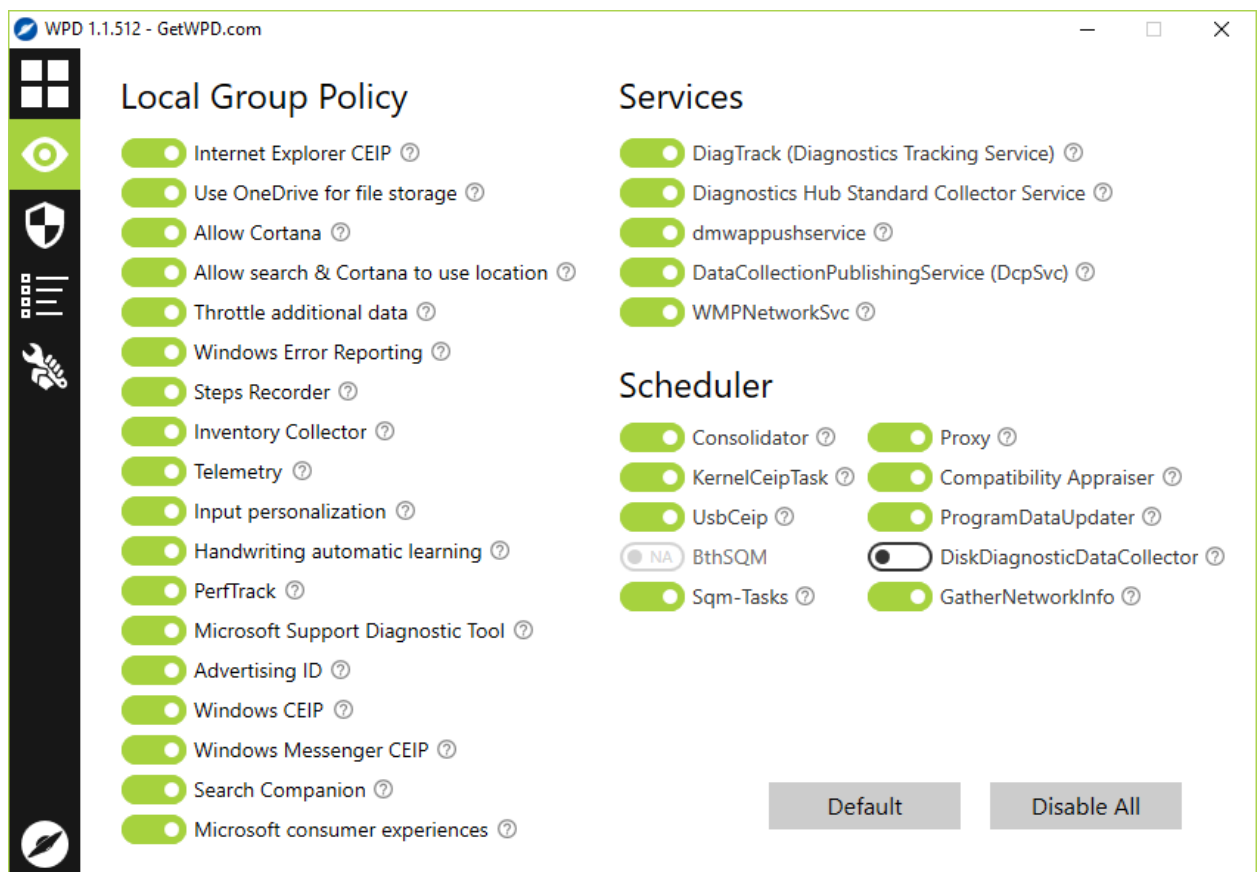


Рисунок 1.3 – Програма WPD (Windows Privacy Dashboard)

На відміну від згаданих програм, інструмент Privatezilla розповсюджується з відкритим вихідним кодом [10], що дозволяє будь-якому фахівцю перевірити безпечність її алгоритмів. Програма автоматизує застосування готових скриптів для вимкнення небажаного функціоналу. Однак графічна оболонка цього засобу реалізована у вигляді суцільного текстового списку без зручного розподілу на розділи. Такий підхід робить програму візуально перевантаженою та складною для сприйняття.

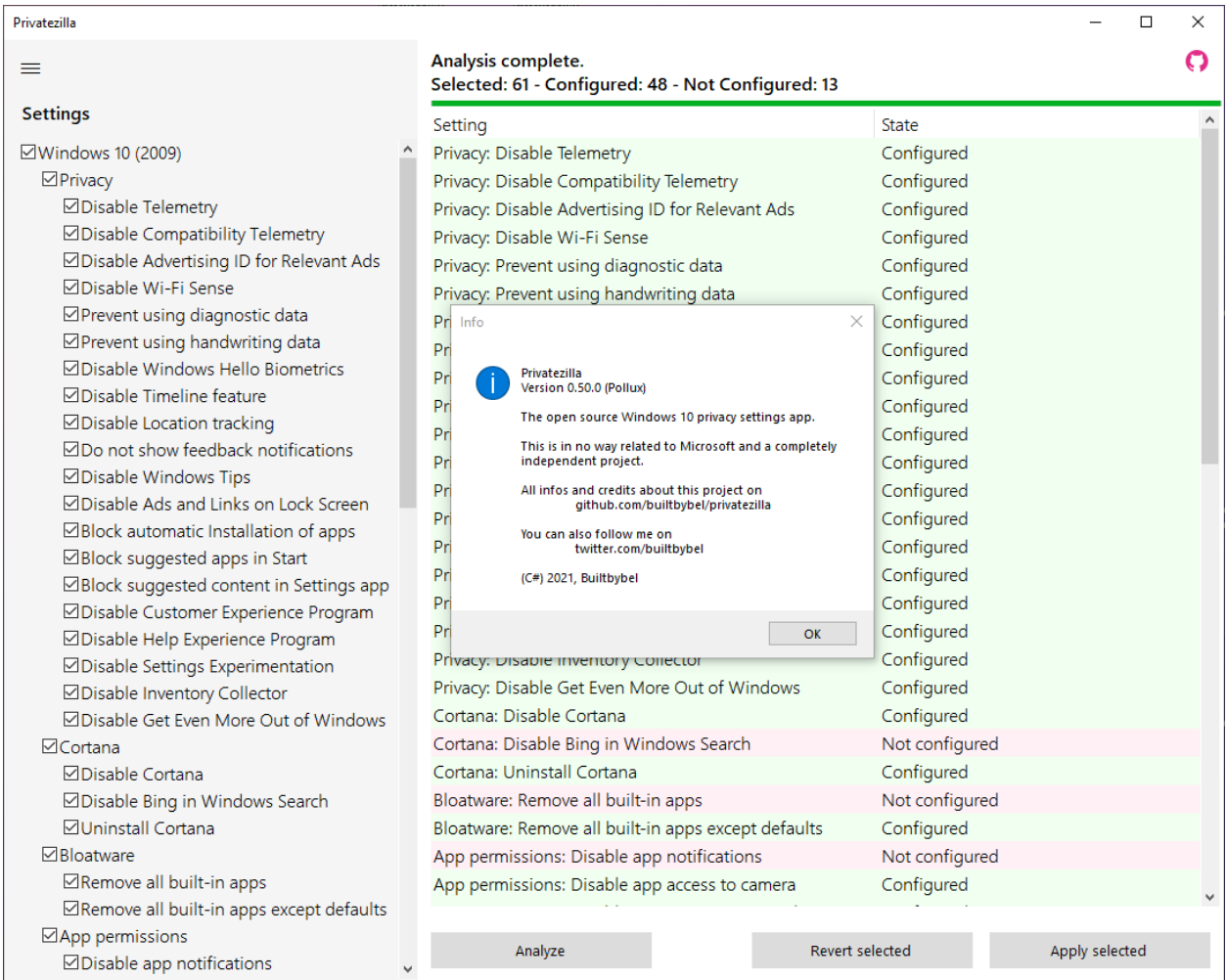


Рисунок 1.4 – Програма Privatezilla

Програма DoNotSpy11 також містить великий перелік опцій для налаштування приватності. Важливою перевагою цього засобу є те, що перед початком застосування змін [11] користувачеві обов'язково пропонується

створити точку відновлення системи. Це гарантує можливість повернути комп'ютер до робочого стану в разі непередбачених помилок.

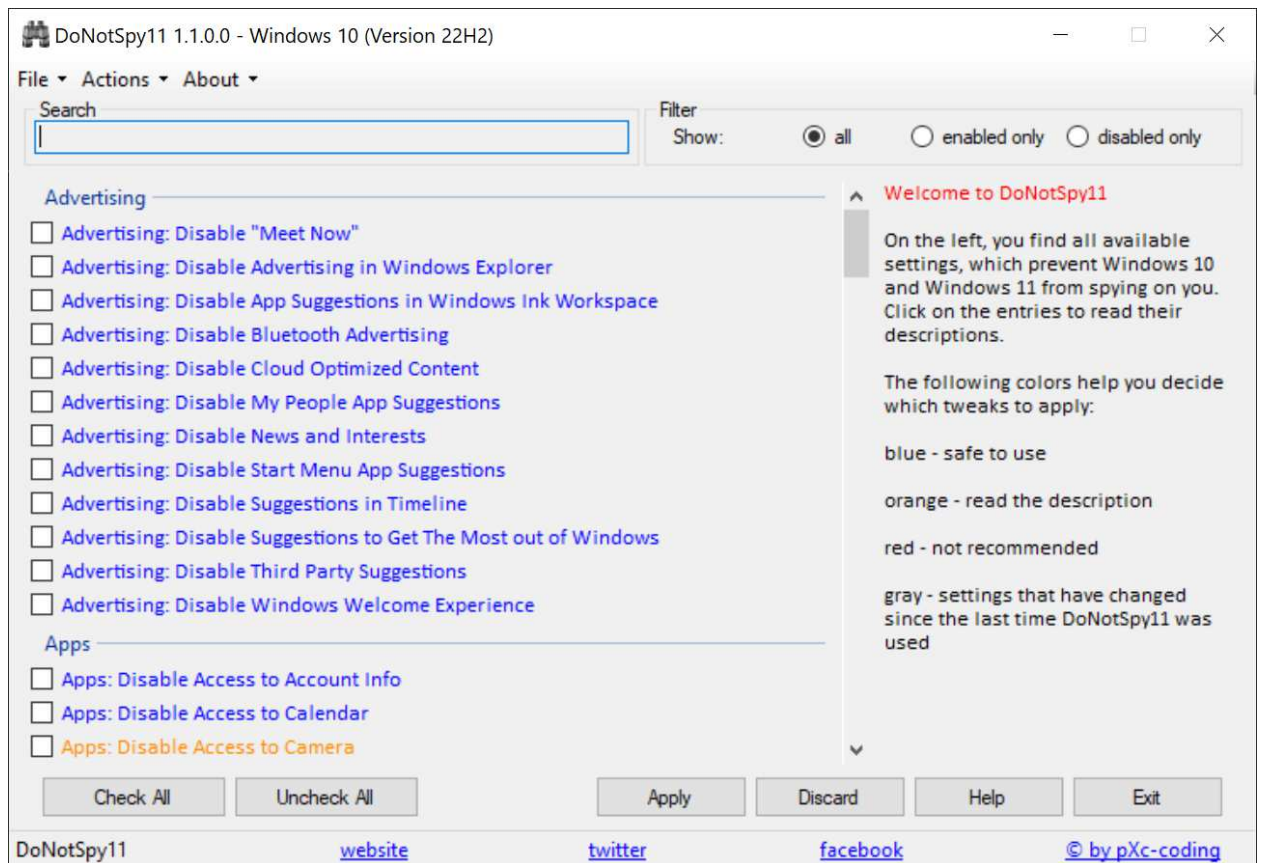


Рисунок 1.5 – Програма DoNotSpy11

Водночас інтерфейс DoNotSpy11 виглядає застарілим, а історично безкоштовна версія цієї програми містила рекламні модулі, що негативно вплинуло на ставлення аудиторії.

1.5 Висновки по проведеному аналізу

Підсумовуючи результати аналізу, проведеного в першому розділі роботи, можна зробити обґрунтований висновок про те, що операційні системи сімейства Windows мають інтегровані механізми збору діагностичних даних [12]. Водночас штатні засоби налаштування приватності часто є розпорошеними по різних меню або взагалі не дозволяють повністю зупинити певні телеметричні процеси. Саме тому створення спеціалізованого

програмного забезпечення для централізованого керування цими параметрами є актуальним завданням.

Аналіз існуючих на ринку утиліт показав, що оптимальний програмний засіб має гармонійно поєднувати прозорість своєї роботи, інтуїтивно зрозуміле візуальне оформлення та надійні захисні механізми. Оскільки багато програм з цього сегменту або приховують суть виконуваних дій за закритим кодом, або перевантажують користувача незрозумілими аббревіатурами, архітектура нашого рішення має бути дружньою до користувача. Проектуючи власний додаток, доцільно запозичити та розвинути ідею створення розгорнутих текстових підказок для кожного налаштування. Такий підхід допоможе людині усвідомлювати, за що відповідає конкретна системна служба чи параметр реєстру, та ухвалювати виважене рішення перед внесенням змін до конфігурації комп'ютера.

Впровадження детальних інформативних описів, у комбінації з функцією обов'язкового попереднього створення точки відновлення чи резервного копіювання стану системи, дозволить створити безпечний інструмент. Він буде орієнтований на звичайних користувачів, які прагнуть ефективно захистити свої особисті дані та зменшити фонове споживання ресурсів апаратної частини без необхідності глибокого занурення в ручне системне адміністрування. Визначені на цьому етапі вимоги та підходи слугуватимуть базою для проектування інтерфейсу та написання програмного коду в наступних розділах дипломної роботи.

2 ПРОЄКТУВАННЯ ПРОГРАМНОГО ЗАСОБУ КЕРУВАННЯ ТЕЛЕМЕТРІЄЮ ТА КОНФІДЕНЦІЙНІСТЮ

2.1 Опис вимог до засобу керування телеметрією та конфіденційністю

Після аналізу предметної області та існуючих програмних аналогів наступним логічним кроком є складання детального переліку вимог до розроблюваного додатку. Цей етап дозволяє окреслити основні функції, які очікує отримати майбутній споживач, а також визначити технічні межі, в яких працюватиме система. Відповідно до стандартів інженерії програмного забезпечення, ці параметри традиційно поділяються на три основні категорії: користувацькі, системні та вимоги до графічної оболонки.

Основне призначення програмного засобу полягає у наданні зручного та безпечного інструментарію для керування телеметрією. Користувач повинен мати можливість переглядати поточний статус фонових служб збору діагностики (наприклад, служби відстеження діагностики та маршрутизації повідомлень) і керувати їхнім запуском. Також необхідний прямий доступ до системних параметрів приватності, щоб людина могла вмикати чи вимикати рекламний ідентифікатор, служби геолокації, історію активності та доступ сторонніх програм до камери й мікрофона.

Оскільки ручне налаштування десятків параметрів потребує багато часу та специфічних знань, важливою вимогою є наявність попередньо налаштованих профілів безпеки. Програма повинна пропонувати вибір між стандартним, підвищеним та максимальним рівнями конфіденційності, застосовуючи необхідний набір змін за одну дію. Для запобігання втраті даних додаток має створювати резервні копії конфігурації перед внесенням будь-яких модифікацій у систему, щоб за потреби користувач міг повернути попередні налаштування. Крім того, інструмент повинен самостійно

розраховувати загальний індекс приватності на основі поточних параметрів та формувати звіти з рекомендаціями щодо покращення захисту.

Оскільки керування системними службами та редагування параметрів реєстру належить до операцій з підвищеним ризиком, обов'язковою умовою функціонування є запуск додатку з правами адміністратора. Без відповідних привілеїв операційна система просто заблокує спроби втручання в конфігурацію телеметрії. Також програма вимагає дозволу на читання та запис файлів у локальних директоріях для збереження звітів та текстових файлів резервних копій у форматі JSON.

Оскільки розроблюваний додаток орієнтований на широку аудиторію, його графічна оболонка має бути інтуїтивно зрозумілою та не перевантаженою зайвими технічними подробицями. Для логічної організації робочого простору доцільно використати систему вкладок, розподіливши інструменти за категоріями: телеметрія, конфіденційність, профілі, резервні копії та звіти.

Щоб полегшити візуальне сприйняття інформації, інтерфейс повинен використовувати кольорову індикацію. Наприклад, розрахований індекс конфіденційності має відображатися червоним кольором при низькому рівні захисту (від 0 до 33 балів), жовтим – при середньому (від 34 до 66 балів) та зеленим – у разі високого рівня безпеки. Під час виконання тривалих операцій з реєстром програма повинна зберігати працездатність та не блокувати головне вікно, інформуючи користувача про перебіг процесу. Будь-які критичні дії, зокрема застосування глобальних профілів або відновлення налаштувань із резервної копії, мають супроводжуватися додатковими діалоговими вікнами підтвердження для запобігання випадковим змінам.

2.2 Обґрунтування вибору засобів розробки

Успішна реалізація будь-якого програмного продукту багато в чому залежить від правильного підбору технологічного стеку. Оскільки створюваний додаток повинен взаємодіяти з низькорівневими компонентами

операційної системи, засоби розробки мають забезпечувати стабільний доступ до системних бібліотек, реєстру та інструментів адміністрування.

Для написання логіки роботи було обрано об'єктно-орієнтовану мову C#. Цей інструмент відрізняється високою надійністю та має вбудовані механізми управління пам'яттю [13], що дозволяє уникнути багатьох типових помилок під час роботи зі складними структурами даних. Важливою перевагою C# є її глибока інтеграція з операційною системою Windows. Завдяки цьому мова надає зручні можливості для звернення до системних функцій через технологію виклику платформозалежного коду. Це важливо для поставленого завдання, адже керування деякими параметрами приватності потребує прямої взаємодії із закритими ділянками системи. Крім того, багата стандартна бібліотека класів дозволяє легко реалізувати роботу з текстовими файлами конфігурацій і звітами без необхідності підключати велику кількість сторонніх модулів.

Як базове середовище виконання було обрано платформу .NET Framework. Таке рішення обґрунтоване тим, що ця платформа постачається разом з операційною системою Windows 10 за замовчуванням [14]. Це означає, що кінцевому користувачеві не доведеться завантажувати та встановлювати додаткові компоненти для запуску розробленого програмного засобу, що значно покращує загальне враження від використання. Платформа містить спеціалізовані простори імен, такі як System.ServiceProcess та Microsoft.Win32, які ідеально підходять для роботи з фоновими службами збору діагностики та гілками реєстру. Обрання перевіреної часом платформи гарантує, що додаток працюватиме передбачувано на комп'ютерах з різним апаратним забезпеченням.

Для побудови графічної оболонки використано технологію Windows Forms. Ця технологія залишається оптимальним вибором для створення системних утиліт. Вона дозволяє швидко генерувати вікна, списки та перемикачі за допомогою зручного візуального редактора [15]. Оскільки головна мета додатку – надати користувачеві контроль над налаштуваннями

телеметрії та профілями безпеки, використання потужних графічних систем зі складною анімацією було б недоцільним і лише збільшило б споживання оперативної пам'яті. Модель програмування, заснована на подіях, дуже добре підходить для реагування на дії користувача, наприклад, при натисканні на кнопку застосування профілю або ініціалізації відновлення конфігурації з резервної копії. Під час виконання тривалих операцій програма інформуватиме про статус виконання через відповідні елементи вікна.

Написання коду, компонування графічного інтерфейсу та тестування програми здійснювалося в інтегрованому середовищі розробки Visual Studio. Це середовище пропонує розробнику великий набір інструментів, що полегшують виконання великої кількості рутинних процесів [16]. Вбудований інструмент для пошуку помилок дозволяє покроково перевіряти правильність зчитування та запису даних у реєстр, що критично важливо під час тестування функцій зміни рівня приватності. Також Visual Studio має зручний диспетчер пакетів, який спрощує підключення бібліотек для роботи з форматом JSON, що планується використовувати для збереження резервних копій.

2.3 Проєктування функціональних модулів

Після вибору технологічної бази наступним етапом є проєктування внутрішньої структури програмного рішення. Для ефективного виконання поставлених завдань архітектура розроблюваного додатка поділена на окремі логічні частини. Це дозволяє спростити процес написання коду та організувати взаємодію між різними елементами графічного інтерфейсу.

2.3.1 Модуль аналізу телеметрії

Однією з найважливіших складових системи є модуль аналізу телеметрії. Його призначення полягає у моніторингу та зміні стану системних служб, які відповідають за збір і відправку діагностичної інформації. Програмна логіка цього модуля взаємодіє з диспетчером служб операційної

системи, отримуючи актуальні дані про службу відстеження діагностики (DiagTrack) та службу маршрутизації повідомлень (dmwappushservice).

Для кожної служби на екран виводиться її поточний стан (працює або зупинено), а також тип запуску, який може бути автоматичним, ручним або повністю відключеним. Користувач отримує можливість не лише переривати роботу цих компонентів, але й змінювати конфігурацію їхнього старту, щоб запобігти відновленню збору даних після перезавантаження комп'ютера. Також у цьому модулі закладено базові структури для подальшої взаємодії з системним планувальником завдань. При кожному оновленні списку або після внесення змін до стану служб, програмний засіб ініціює перерахунок загального індексу приватності, передаючи нові дані до відповідної підсистеми оцінювання.

2.3.2 Модуль керування конфіденційністю

Модуль керування конфіденційністю відповідає за надання зручного доступу до параметрів, які зазвичай приховані або розпорошені в глибинах меню операційної системи. Цей функціональний блок зосереджений на взаємодії із системним реєстром, оскільки саме там зберігаються конфігурації дозволів.

Підсистема розпізнає та відображає сім основних показників: рекламний ідентифікатор, службу геолокації, історію активності, персоналізовані поради, налаштування рукописного введення, а також доступ сторонніх застосунків до мікрофона та камери. Навпроти кожного елемента в списку виводиться його поточний статус (увімкнено чи відключено), категорія, зрозумілий текстовий опис та шлях до відповідного запису в реєстрі. Такий підхід робить процес прозорим, дозволяючи розуміти, який саме параметр модифікується під час роботи.

Управління налаштуваннями відбувається шляхом програмного запису нових значень у визначені гілки реєстру. Під час натискання кнопок зміни стану певного параметра, модуль звертається до системної бази даних та

перепише інформацію. Для зручності передбачена функція ручного оновлення списку параметрів, яка повторно зчитує всі значення, гарантуючи, що відображена на екрані інформація повністю відповідає дійсному стану комп'ютера.

2.3.3 Модуль профілів конфіденційності

Для того, щоб спростити взаємодію з розроблюваним додатком, було передбачено окремий модуль профілів конфіденційності. Ручне налаштування кожного окремого параметра вимагає від користувача певного рівня знань та витрат часу. Тому агрегація налаштувань у готові шаблони є поширеною практикою.

Програма пропонує на вибір три системні профілі. Стандартний профіль зберігає базові налаштування ОС, при яких усі служби телеметрії та відстеження залишаються увімкненими, що відповідає низькому рівню приватності. Підвищений рівень зупиняє роботу служби відстеження діагностики та блокує ідентифікатор реклами разом із визначенням геолокації. Максимальний профіль повністю вимикає всі доступні для редагування параметри відстеження, а також службу маршрутизації повідомлень.

Графічний інтерфейс цього модуля виводить опис вибраного шаблону, загальну кількість параметрів, що підлягають зміні, та прогнозований рівень захисту, спираючись на загальноприйняті принципи проєктування безпечних системних утиліт. Застосування всіх змін відбувається при натисканні кнопки підтвердження. Оскільки така операція редагує системний реєстр та стан служб, модуль запитує додаткове підтвердження дії від користувача, щоб уникнути помилкових рішень.

2.3.4 Модуль резервного копіювання

Внесення будь-яких масштабних змін до конфігурації операційної системи завжди пов'язане з ризиком виникнення помилок у роботі системних компонентів, тому модуль резервного копіювання виступає необхідним

інструментом для безпечної роботи. Надійне збереження попередніх станів системи дозволяє легко повернути комп'ютер до робочого вигляду в разі непередбачених ситуацій.

Програмна логіка цього модуля зчитує поточні стани служб і параметрів приватності та зберігає їх у виділену локальну директорію «Backups». Дані формуються у вигляді структурованих текстових файлів.

Вибір формату JSON для цих потреб зумовлений його зручністю для серіалізації об'єктів і повною підтримкою з боку стандартних бібліотек обраної платформи розробки. Кожній створеній копії автоматично присвоюється ім'я на основі поточної дати та часу генерації. Інтерфейс модуля містить перелік усіх збережених файлів, показуючи їхній обсяг та дату створення.

Користувач має змогу ініціювати повне відновлення налаштувань із вибраного файлу після відповідного діалогового вікна або ж видалити застарілі копії для звільнення місця на накопичувачі.

2.3.5. Модуль формування звітів

Для надання користувачеві зворотного зв'язку про поточний стан операційної системи розроблено модуль формування звітів.

Цей модуль збирає дані від інших підсистем додатка: інформацію про активні служби телеметрії, статуси показників реєстру та загальний розрахований індекс конфіденційності.

Уся ця інформація об'єднується в єдиний документ, до якого також додаються автоматично згенеровані рекомендації щодо подальшого налаштування операційної системи та покращення загального рівня безпеки особистих даних.

Аналогічно до системи резервних копій, користувач отримує змогу переглядати список створених звітів, читати їхній вміст та видаляти непотрібні документи через окрему вкладку графічного вікна.

2.4 Проєктування структури резервних копій у форматі JSON

Під час розробки механізму збереження попередніх конфігурацій постало питання вибору оптимального формату для обміну та зберігання даних. Серед доступних варіантів, таких як XML, INI або спеціалізовані бінарні формати, перевагу було надано формату JSON. Такий вибір пояснюється його високою читабельністю як для програмних аналізаторів, так і для людини, що суттєво спрощує процес налагодження логіки на етапах тестування. Вбудована підтримка серіалізації в обраній платформі розробки дозволяє перетворювати об'єкти з пам'яті комп'ютера безпосередньо у текстовий вигляд без необхідності написання додаткових алгоритмів синтаксичного аналізу.

Проєктування структури файлу відіграє вагомую роль у забезпеченні надійності подальшого відновлення. Кожен згенерований документ резервної копії являє собою єдиний об'єкт, який логічно розділений на кілька інформаційних блоків. Верхній рівень структури містить базові метадані, зокрема точну дату та час створення копії, а також загальний опис стану системи на момент запису.

Основна частина файлу складається з двох масивів даних. Перший масив зберігає конфігурацію телеметричних служб, фіксуючи системну назву кожної служби, її поточний статус та налаштований тип запуску. Другий масив відповідає за показники конфіденційності, записуючи точний шлях до гілки реєстру та відповідне числове значення для кожного параметра. Такий підхід забезпечує повне охоплення всіх елементів конфігурації, які користувач міг змінити під час роботи з додатком.

Процедура повернення операційної системи до попереднього стану також потребує детального проєктування. Коли людина обирає певну резервну копію зі списку в графічному вікні, розроблюваний інструмент зчитує відповідний текстовий файл і перетворює його назад у програмні класи. Оскільки процес відновлення передбачає масове перезаписування системного реєстру та зміну параметрів фонових процесів, перед початком застосування

конфігурації на екран обов'язково виводиться діалогове вікно з проханням підтвердити цю операцію. Після отримання згоди від користувача, внутрішні механізми програми послідовно обходять зчитані масиви даних, передаючи потрібні команди диспетчеру служб та компонентам управління реєстром. Після успішного завершення всіх операцій відбувається автоматичний перерахунок загального показника безпеки особистих даних, щоб графічна оболонка відображала актуальну інформацію.

2.5 Проектування інтерфейсу користувача

Графічна оболонка є єдиним каналом взаємодії між людиною та внутрішньою логікою програми, тому її правильне проектування безпосередньо впливає на зручність використання готового продукту. Враховуючи специфіку налаштування операційної системи, структура головної форми розроблюваного додатка має бути максимально простою та не перевантаженою зайвими візуальними ефектами. За основу взято класичну віконну архітектуру, де весь доступний простір розподілено на логічні зони. Верхня частина вікна зазвичай відводиться під інформаційні повідомлення та відображення загального показника конфіденційності, тоді як основна робоча область використовується для розміщення безпосередньо функціональних елементів.

Для уникнення нагромадження інформації на одному екрані навігацію побудовано на основі системи вкладок. Такий підхід у розробці настільних рішень дозволяє логічно згрупувати інструменти за їхнім цільовим призначенням та приховати від користувача ті налаштування, які йому наразі не потрібні. Відповідно до визначених раніше модулів, у програмі передбачено п'ять основних розділів: «Телеметрія», «Конфіденційність», «Профілі», «Резервні копії» та «Звіти». Перехід між цими вкладками відбувається без затримок, що дає змогу швидко знаходити потрібний інструмент. Наприклад, на вкладці телеметрії відображається перелік фонових служб, тоді як розділ профілів містить лише три глобальні перемикачі

конфігурацій та розгорнуті текстові описи до них, що суттєво полегшує сприйняття порівняно зі звичайними довгими списками параметрів.

Для відображення поточного стану системних служб та параметрів реєстру використовуються табличні списки, де кожен рядок містить назву параметра, опис і його поточний статус. Взаємодія з цими даними здійснюється за допомогою стандартизованих кнопок активації та деактивації. Щоб підвищити рівень інформативності та полегшити розуміння поточного стану комп'ютера, розроблено систему візуальної індикації, що використовує кольорові позначки. Це інформує людину про загальний рівень безпеки без необхідності читати додаткові звіти. Усі елементи керування, натискання яких призводить до масштабної зміни системних файлів або відновлення з резервних копій, обладнані додатковими вікнами підтвердження дій для запобігання випадковому порушенню працездатності системи.

3 РОЗРОБКА ПРОГРАМНОГО ЗАСОБУ ДЛЯ КЕРУВАННЯ ТЕЛЕМЕТРІЄЮ ТА КОНФІДЕНЦІЙНІСТЮ В ОС СІМЕЙСТВА WINDOWS

3.1 Функціональна схема

Для забезпечення надійності та можливості подальшого масштабування, архітектуру розроблюваного додатка побудовано на основі багаторівневої моделі. Відповідно до спроектованої структури, загальну схему програмного рішення можна поділити на шість взаємопов'язаних блоків.

Першим рівнем є шар користувацького інтерфейсу. Він представлений головним вікном програми, робочий простір якого розділено на п'ять вкладок. Головне завдання цього рівня полягає у відображенні інформації та передачі команд від користувача до внутрішніх механізмів без прямого втручання в системні процеси. Усі звернення з графічної оболонки спрямовуються виключно до сервісного шару.

Сервісний шар є базою програми, де зосереджена вся логіка управління та обробки даних. Він містить низку спеціалізованих компонентів:

а) Сервіс телеметрії відповідає за безпосереднє керування двома службами: відстеженням діагностики (DiagTrack) та маршрутизацією повідомлень (dmwappushservice).

б) Сервіс конфіденційності контролює стан семи основних системних параметрів конфіденційності.

в) Сервіс профілів оперує трьома попередньо налаштованими конфігураціями захисту (стандартною, підвищеною та максимальною).

г) Сервіси резервного копіювання та генерації звітів забезпечують збереження поточного стану системи та формування документації.

д) Окремий сервіс відповідає за розрахунок загального індексу безпеки даних після кожної зміни налаштувань.

Для стандартизації обміну інформацією між сервісами та інтерфейсом використовується модельний шар. Він складається з класів-контейнерів, які описують структуру окремих елементів: телеметрії, параметрів конфіденційності, профілів, звітів та резервних копій. Використання таких моделей дозволяє впорядкувати дані перед їхнім виведенням на екран або записом у файл.

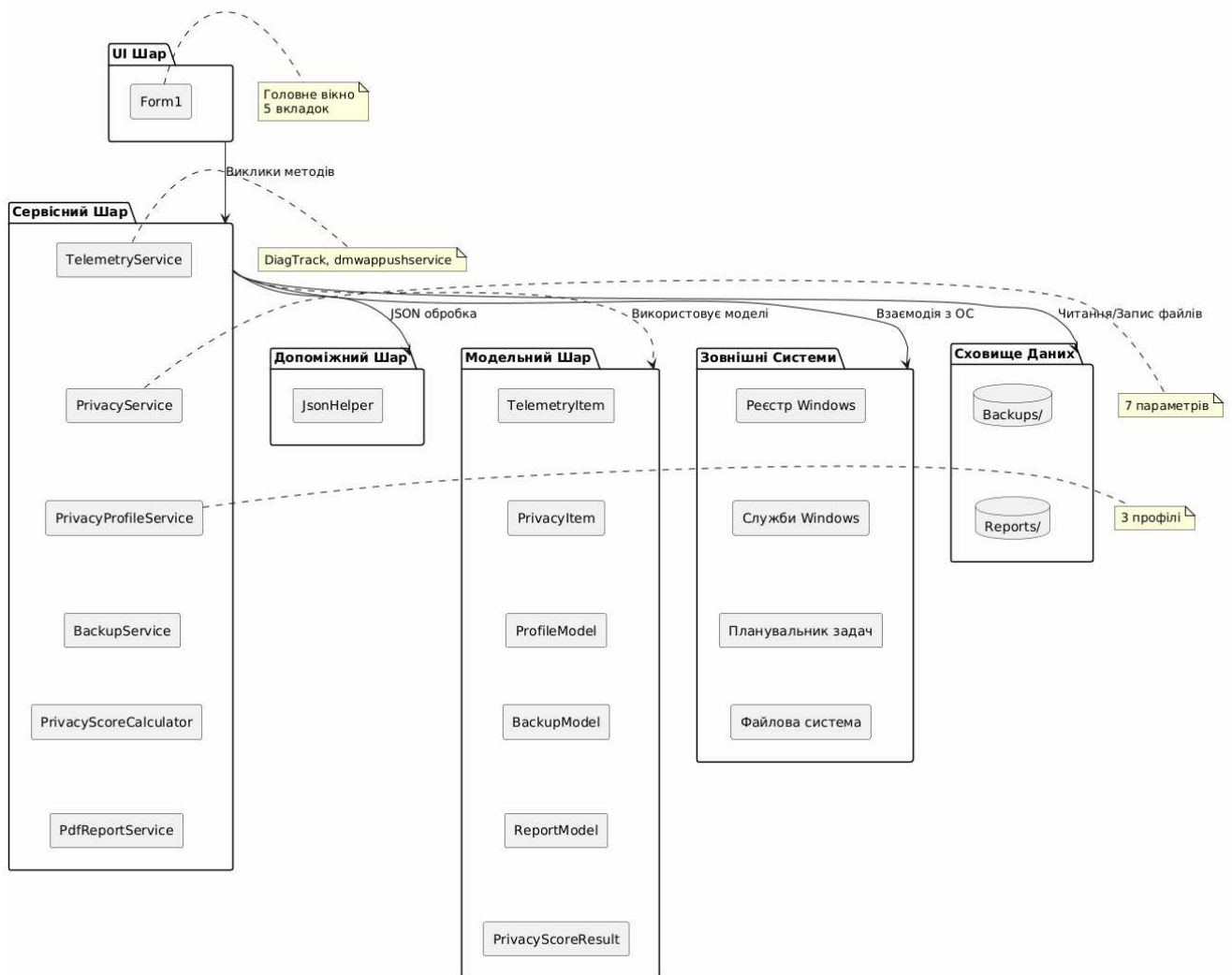


Рисунок 3.1 – Функціональна схема програмного засобу для керування телеметрією та конфіденційністю в ОС сімейства Windows

Допоміжний шар містить інструменти для виконання рутинних операцій. Він представлений компонентом для обробки формату JSON, який серіалізує об'єкти з пам'яті комп'ютера у текстовий вигляд та десеріалізує їх

назад. Сервісний шар звертається до цього інструменту під час формування та читання конфігураційних файлів.

Останні два блоки схеми описують зовнішнє середовище, з яким взаємодіє додаток. Блок зовнішніх систем охоплює компоненти самої ОС: системний реєстр, диспетчер служб, планувальник завдань і базову файлову систему. Блок сховища даних представляє локальні директорії на накопичувачі комп'ютера («Backups» та «Reports»), призначені для постійного збереження згенерованих документів і файлів конфігурації.

Така схема гарантує безпечну ізоляцію компонентів: графічна оболонка ніколи не намагається редагувати реєстр напряму, а сервіси не залежать від візуального оформлення вікон. Це робить програмний код більш стабільним і спрощує його подальший супровід.

3.2 Алгоритм роботи

Загальний алгоритм роботи розроблюваного додатка починається з етапу ініціалізації та перевірки середовища виконання. Оскільки керування системними параметрами вимагає підвищених прав доступу, першим кроком після запуску виконуваного файлу є запит до операційної системи щодо статусу адміністратора. Якщо програма запущена зі звичайними правами, алгоритм виводить на екран повідомлення та перериває виконання, щоб запобігти помилкам доступу під час спроб редагування реєстру. У разі успішної перевірки прав керування передається модулю графічної оболонки.

На наступному етапі відбувається збір актуальної інформації про стан комп'ютера. Програмний засіб послідовно звертається до диспетчера служб для отримання статусів телеметрії та зчитує значення дозволів із визначених гілок реєстру. Зібрані дані завантажуються у внутрішні структури пам'яті та використовуються для розрахунку початкового індексу конфіденційності. Після завершення математичних обчислень алгоритм формує списки елементів керування у відповідних вкладках головного вікна та забарвлює індикатор загального стану в потрібний колір.

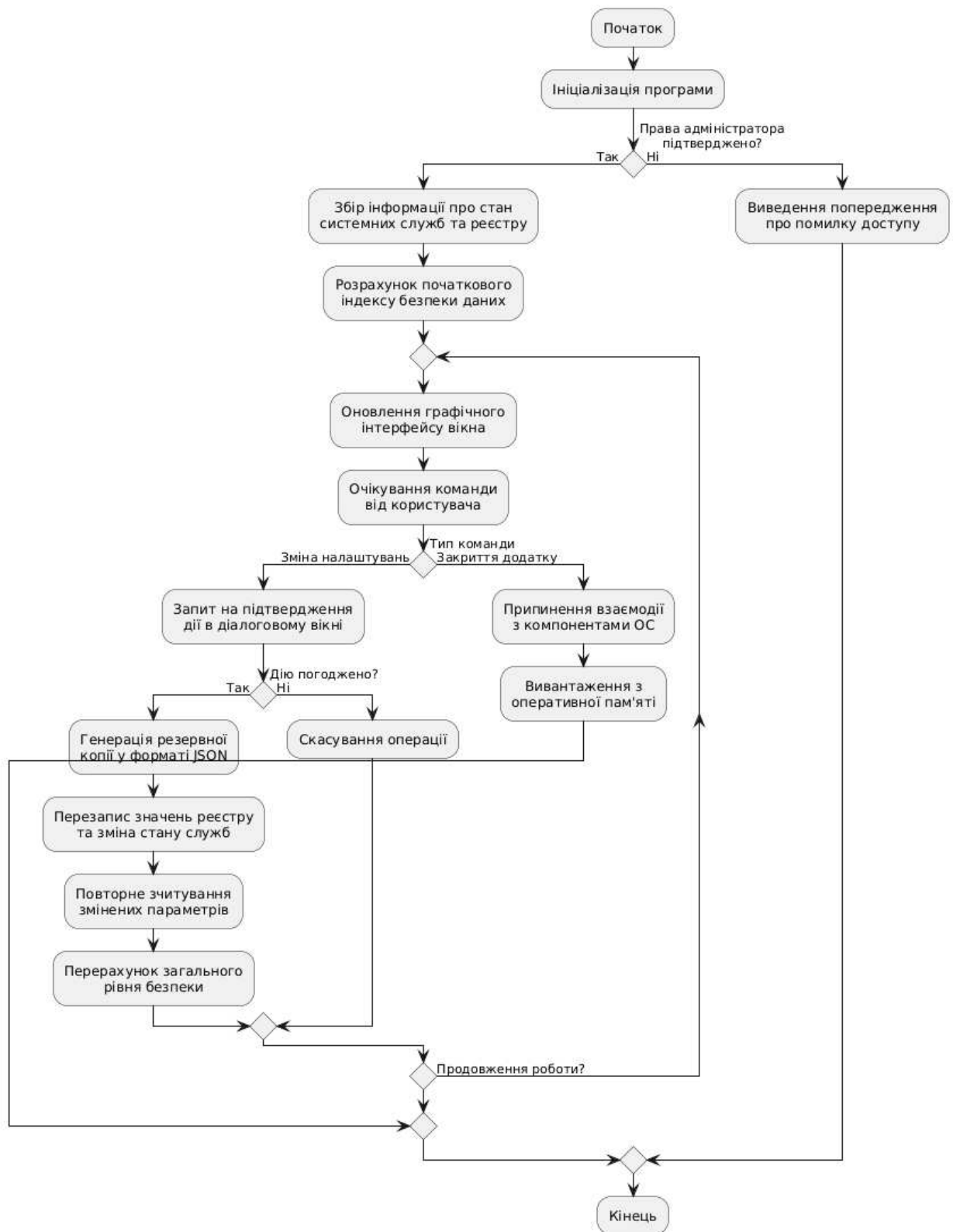


Рисунок 3.2 – Блок-схема алгоритма програмного засобу для керування телеметрією та конфіденційністю в ОС сімейства Windows

Коли користувач ініціює зміну певного параметра або обирає один із глобальних профілів захисту, алгоритм переходить до активної фази

застосування конфігурації. Перед внесенням будь-яких масштабних модифікацій система призупиняє виконання алгоритму і показує діалогове вікно для підтвердження дії. Після отримання згоди від людини логіка програми генерує текстовий файл із резервною копією поточних налаштувань і зберігає його в локальній директорії. Тільки після успішного запису файлу алгоритм починає послідовно перезаписувати значення в реєстрі та надсилати команди на зупинку або запуск фонових процесів збору діагностики.

Кінцевим кроком циклу обробки команд є оновлення інформації на екрані. Програма зчитує змінені параметри з операційної системи, щоб підтвердити їхнє успішне застосування, та перераховує загальний рівень безпеки. Завдяки цьому користувач одразу бачить результати своїх дій у вигляді оновлених статусів.

Процес роботи додатка завершується стандартним закриттям вікна, під час якого програма просто припиняє взаємодію з системними компонентами та вивантажується з оперативної пам'яті комп'ютера.

3.3 Демонстрація розробленої програми

Головне вікно розробленого додатка зустрічає користувача панеллю з 5 основними вкладками. Використання такої навігаційної архітектури дозволяє зручно організувати робочий простір без візуального перевантаження інформацією. У верхній частині екрана постійно відображається загальний індекс безпеки даних. Цей показник автоматично перераховується після будь-яких втручань у системну конфігурацію, інформуючи людину про поточний стан комп'ютера.

У розділі «Телеметрія» реалізовано інструментарій для моніторингу фонових процесів зі збору діагностичної інформації. У зручній таблиці виводяться назви служб, їхній поточний стан роботи та тип запуску. Ця інформація зчитується безпосередньо з системного диспетчера. За допомогою кнопок керування користувач має змогу зупинити передачу інформації та відключити автозапуск непотрібних компонентів.

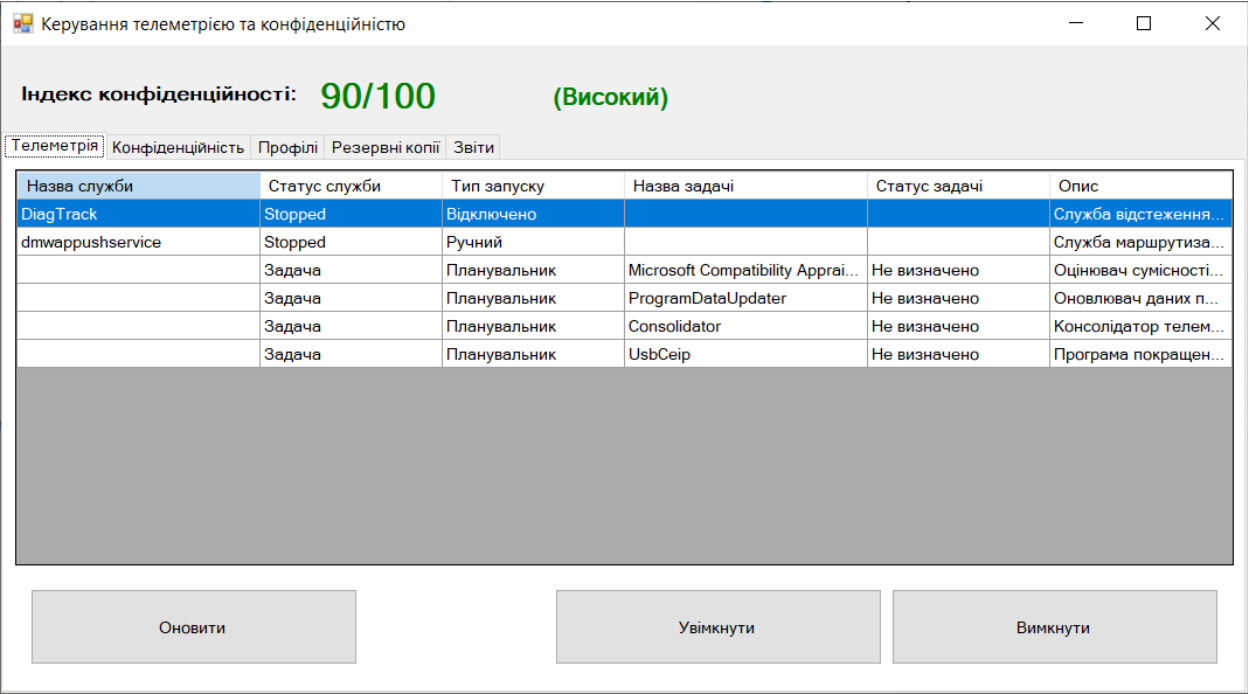


Рисунок 3.3 – Телеметрія

Наступна вкладка, яка відповідає за конфіденційність, надає доступ до семи основних параметрів системи, серед яких доступ до мікрофона, камери та служба геолокації.

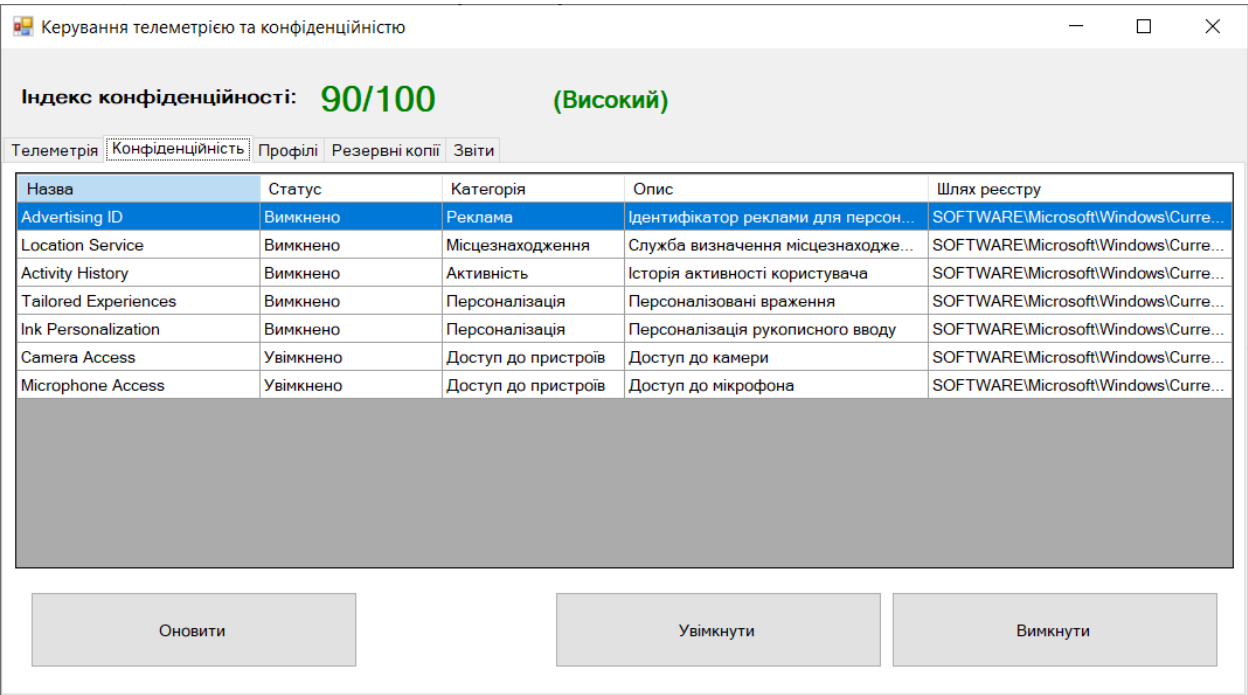


Рисунок 3.4 – Конфіденційність

Кожен пункт списку супроводжується детальним описом та вказує шлях до відповідного запису у системному реєстрі. Завдяки цьому процес налаштування стає набагато прозорішим. Для збереження працездатності вікна під час внесення великої кількості змін до реєстру, усі операції запису та читання виконуються окремим паралельним потоком. Це гарантує, що програма не зависне під час збереження налаштувань.

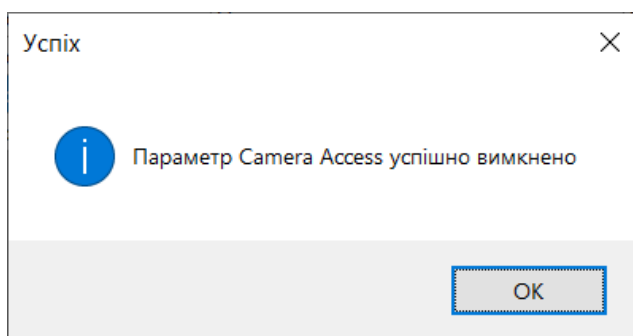


Рисунок 3.5 – Зміна стану параметра

Вкладка «Профілі» пропонує три готові конфігурації захисту. Застосування певного профілю автоматизує рутинну роботу, змінюючи десятки параметрів за одне натискання. Щоб попередити випадкове пошкодження операційної системи, перед ініціалізацією цієї операції на екран виводиться діалогове вікно, де користувач має свідомо підтвердити намір застосувати глобальні зміни.

Розділи для роботи з резервними копіями та звітами мають схожу структуру: вони відображають списки збережених файлів у відповідних локальних директоріях. Інструмент для створення резервних копій формує структуровані текстові документи, що дозволяє згодом надійно повернути комп'ютер до попереднього стану. Зі свого боку, підсистема звітів генерує підсумкові файли з аналітикою та рекомендаціями, що допомагає краще розуміти слабкі місця в захисті персональної інформації на поточному пристрої та вживати необхідних заходів.

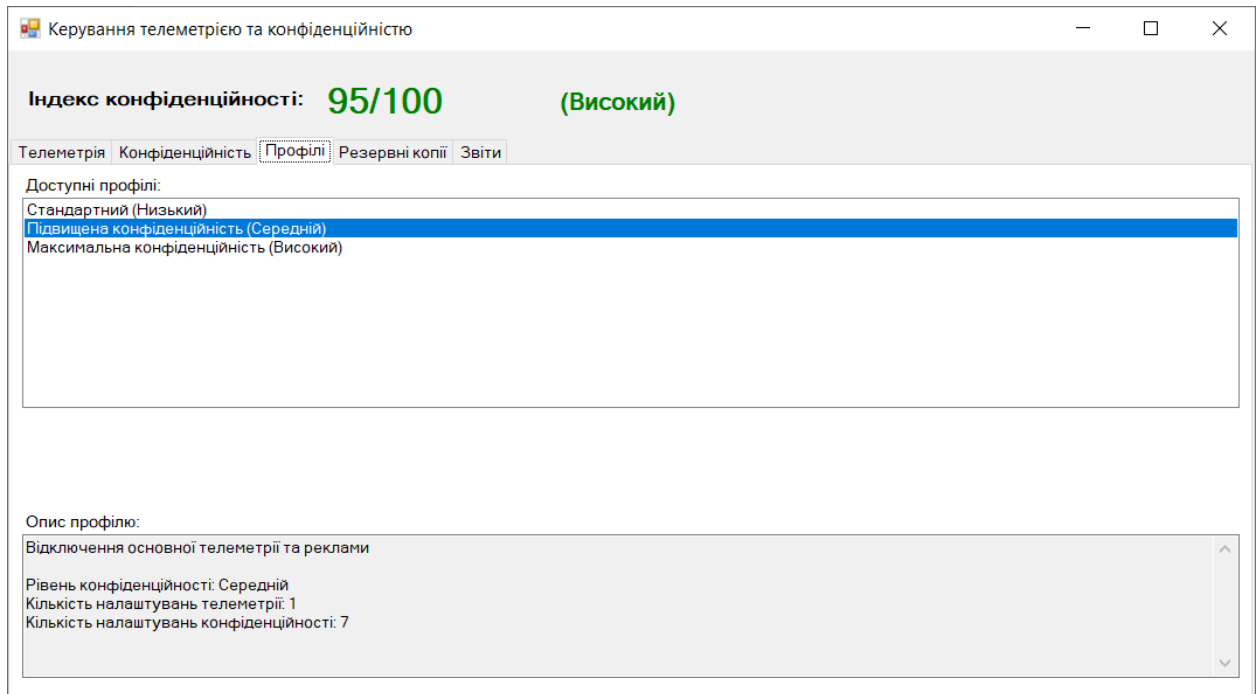


Рисунок 3.6 – Профілі

Таким чином, розроблений програмний засіб для керування телеметрією та конфіденційністю в ОС сімейства Windows повною мірою відповідає поставленим при проектуванні вимогам.

ВИСНОВКИ

В процесі виконання кваліфікаційної роботи було успішно вирішено актуальне завдання розробки спеціалізованого програмного засобу для керування параметрами телеметрії та конфіденційності в операційних системах сімейства Windows. Проведений детальний аналіз механізмів збору діагностичних даних показав, що вбудовані інструменти операційної системи є розпорошеними по різних меню і не дозволяють повністю зупинити фонові процеси передачі інформації, що створює постійні загрози для приватності.

Дослідження існуючих програмних аналогів виявило додаткові проблеми: більшість популярних утиліт або має закритий вихідний код, що унеможлиблює незалежну перевірку їхніх дій, або пропонують надто складний візуальний інтерфейс. Спираючись на ці дані, було сформовано вимоги до власного додатка, серед яких головними стали прозорість операцій, зрозуміла графічна оболонка та наявність захисних алгоритмів.

Відповідно до сформованих вимог було спроектовано та реалізовано логічну архітектуру рішення за допомогою об'єктно-орієнтованої мови C# та платформи .NET Framework. Створений програмний засіб напряду взаємодіє з диспетчером служб і системним реєстром, дозволяючи користувачеві зупиняти роботу служби відстеження діагностики та маршрутизації повідомлень, а також контролювати доступ сторонніх програм до мікрофона, камери та геолокації.

Для спрощення взаємодії реалізовано підсистему профілів безпеки, завдяки якій людина може застосувати комплекс глобальних налаштувань за одну дію, обравши стандартний, підвищений або максимальний рівень захисту. Впровадження алгоритму розрахунку загального показника конфіденційності дає змогу наочно оцінювати поточний стан комп'ютера. Крім того, з метою запобігання пошкодженню системних файлів розроблено модуль резервного копіювання, який автоматично генерує структуровані

конфігураційні документи у форматі JSON перед внесенням будь-яких масштабних модифікацій.

Отримані результати підтверджують досягнення мети, поставленої на початку роботи. Розроблений додаток є повністю готовим до практичного використання і може стати надійним помічником для користувачів, які прагнуть захистити свої особисті дані без необхідності заглиблюватися у складні процеси системного адміністрування.

ПЕРЕЛІК ПОСИЛАНЬ

1. Microsoft. Configure Windows diagnostic data in your organization. Microsoft Learn. URL: <https://learn.microsoft.com/en-us/windows/privacy/configure-windows-diagnostic-data-in-your-organization>
2. Бурячок В.Л., Толюпа С.В., Аносов А.О., Козачок В.А., Лукова-Чуйко Н.В. Системний аналіз та прийняття рішень в інформаційній безпеці: підручник. / В.Л. Бурячок, С.В. Толюпа, А.О. Аносов, В.А. Козачок, Н.В. Лукова-Чуйко / – К.:ДУТ, 2015. – 345 с.
3. Bott E. Windows telemetry secrets: Where, why, and how Microsoft collects your data. ZDNet. 2021. URL: <https://www.zdnet.com/article/windows-10-telemetry-secrets/>
4. Смірнов О.А., Коноплицька-Слободенюк О.К., Смірнов С.А., Буравченко К.О., Смірнова Т.В., Книшук А.В. Вступ до кібербезпеки: навч. посіб. – Кропивницький: ЦНТУ, 2022. – 967 с.
5. Богуш В.М., Довидьков О.А., Кривуца В.Г. Теоретичні основи захищених інформаційних технологій. Навч. посібник. – К.: ДУІКТ, 2010. – 454 с.
6. Microsoft. General privacy settings in Windows. Microsoft Support. URL: <https://support.microsoft.com/en-us/windows/general-privacy-settings-in-windows-7c7f6a09-cebd-5589-c376-7f505e5bf65a>
7. Microsoft. Windows location service and privacy. Microsoft Learn. URL: <https://support.microsoft.com/en-us/windows/windows-location-service-and-privacy-3a8eee0a-5b0b-dc07-eede-2a5ca1c49088>
8. O&O Software. O&O ShutUp10++: Free antispy tool for Windows 10 and 11. O&O Software. URL: <https://www.oo-software.com/en/shutup10>
9. WPD Team. WPD: The real privacy dashboard for Windows. WPD. URL: <https://wpd.app/>
10. Privatezilla. URL: <https://github.com/builtbybel/privatezilla>
11. DoNotSpy11. URL: <https://pxc-coding.com/donotspy11/>

12. Configure Windows diagnostic data in your organization. URL: <https://learn.microsoft.com/en-us/windows/privacy/configure-windows-diagnostic-data-in-your-organization>
13. Microsoft. C# docs. Microsoft Learn. URL: <https://learn.microsoft.com/en-us/dotnet/csharp/>
14. Microsoft. .NET Framework overview. Microsoft Learn. URL: <https://learn.microsoft.com/en-us/dotnet/framework/get-started/overview>
15. Microsoft. Windows Forms documentation. Microsoft Learn. URL: <https://learn.microsoft.com/en-us/dotnet/desktop/winforms/>
16. Microsoft. Visual Studio documentation. Microsoft Learn. URL: <https://learn.microsoft.com/en-us/visualstudio/>

Додаток А - Програмний код

```
using System;
using System.Collections.Generic;
using System.Windows.Forms;
using TelemetryPrivacyManager.Services;
using TelemetryPrivacyManager.Models;

namespace TelemetryPrivacyManager
{
    // Головна форма програми для керування телеметрією та
    // конфіденційністю
    public partial class Form1 : Form
    {
        // Сервіс для роботи з телеметрією
        private TelemetryService telemetryService;
        // Сервіс для роботи з конфіденційністю
        private PrivacyService privacyService;
        // Сервіс для роботи з профілями конфіденційності
        private PrivacyProfileService profileService;
        // Список профілів конфіденційності
        private List<ProfileModel> profiles;
        // Сервіс для роботи з резервними копіями
        private BackupService backupService;
        // Сервіс для розрахунку індексу конфіденційності
        private PrivacyScoreCalculator privacyScoreCalculator;

        // Конструктор головної форми
        public Form1()
        {
            // Ініціалізація компонентів форми
            InitializeComponent();

            // Ініціалізація сервісу телеметрії
            telemetryService = new TelemetryService();

            // Ініціалізація сервісу конфіденційності
            privacyService = new PrivacyService();

            // Ініціалізація сервісу профілів
            profileService = new PrivacyProfileService();
            profiles = new List<ProfileModel>();

            // Ініціалізація сервісу резервних копій
            backupService = new BackupService();

            // Ініціалізація сервісу розрахунку індексу
            // конфіденційності
            privacyScoreCalculator = new
            PrivacyScoreCalculator();
        }
    }
}
```

```

        // Підключення обробника події завантаження форми
        this.Load += Form1_Load;

        // Підключення обробників подій кнопок
        btnRefreshTelemetry.Click +=
BtnRefreshTelemetry_Click;
        btnEnableTelemetry.Click +=
BtnEnableTelemetry_Click;
        btnDisableTelemetry.Click +=
BtnDisableTelemetry_Click;

        // Підключення обробників подій кнопок
конфіденційності
        btnRefreshPrivacy.Click += BtnRefreshPrivacy_Click;
        btnEnablePrivacy.Click += BtnEnablePrivacy_Click;
        btnDisablePrivacy.Click += BtnDisablePrivacy_Click;

        // Підключення обробників подій профілів
        listBoxProfiles.SelectedIndexChanged +=
ListBoxProfiles_SelectedIndexChanged;
        btnApplyProfile.Click += BtnApplyProfile_Click;

        // Підключення обробників подій резервних копій
        btnCreateBackup.Click += BtnCreateBackup_Click;
        btnRestoreBackup.Click += BtnRestoreBackup_Click;
        btnDeleteBackup.Click += BtnDeleteBackup_Click;
    }

    // Обробник події завантаження форми
    private void Form1_Load(object sender, EventArgs e)
    {
        // Завантаження даних телеметрії при відкритті форми
        LoadTelemetryData();

        // Завантаження даних конфіденційності при відкритті
форми
        LoadPrivacyData();

        // Завантаження профілів при відкритті форми
        LoadProfiles();

        // Завантаження резервних копій при відкритті форми
        LoadBackups();

        // Розрахунок індексу конфіденційності при відкритті
форми
        CalculatePrivacyScore();
    }

    // Завантаження даних телеметрії в таблицю
    private void LoadTelemetryData()

```

```

    {
        try
        {
            // Отримання списку елементів телеметрії
            var telemetryItems =
telemetryService.GetTelemetryServices();

            // Налаштування стовпців таблиці
            SetupTelemetryColumns();

            // Очищення таблиці перед заповненням
            dataGridViewTelemetry.Rows.Clear();

            // Заповнення таблиці даними
            foreach (var item in telemetryItems)
            {
                // Додавання нового рядка в таблицю
                dataGridViewTelemetry.Rows.Add(
                    item.ServiceName,
                    item.ServiceStatus,
                    item.StartType,
                    item.TaskName,
                    item.TaskStatus,
                    item.Description
                );
            }
        }
        catch (Exception ex)
        {
            // Відображення помилки користувачу
            MessageBox.Show(
                "Помилка при завантаженні даних телеметрії:
" + ex.Message,
                "Помилка",
                MessageBoxButtons.OK,
                MessageBoxIcon.Error
            );
        }
    }

    // Налаштування стовпців таблиці телеметрії
    private void SetupTelemetryColumns()
    {
        // Очищення наявних стовпців
        dataGridViewTelemetry.Columns.Clear();

        // Додавання стовпців
        dataGridViewTelemetry.Columns.Add("ServiceName",
"Назва служби");
        dataGridViewTelemetry.Columns.Add("ServiceStatus",
"Статус служби");
    }
}

```



```

        dataGridViewTelemetry.Columns.Add("StartType", "Тип
запуску");
        dataGridViewTelemetry.Columns.Add("TaskName", "Назва
задачі");
        dataGridViewTelemetry.Columns.Add("TaskStatus",
"Статус задачі");
        dataGridViewTelemetry.Columns.Add("Description",
"Опис");

        // Налаштування ширини стовпців

dataGridViewTelemetry.Columns["ServiceName"].FillWeight = 20;

dataGridViewTelemetry.Columns["ServiceStatus"].FillWeight = 15;

dataGridViewTelemetry.Columns["StartType"].FillWeight = 15;
        dataGridViewTelemetry.Columns["TaskName"].FillWeight
= 20;

dataGridViewTelemetry.Columns["TaskStatus"].FillWeight = 15;

dataGridViewTelemetry.Columns["Description"].FillWeight = 15;
    }

    // Завантаження даних конфіденційності в таблицю
    private void LoadPrivacyData()
    {
        try
        {
            // Отримання списку елементів конфіденційності
            var privacyItems =
privacyService.GetPrivacySettings();

            // Налаштування стовпців таблиці
            SetupPrivacyColumns();

            // Очищення таблиці перед заповненням
            dataGridViewPrivacy.Rows.Clear();

            // Заповнення таблиці даними
            foreach (var item in privacyItems)
            {
                // Додавання нового рядка в таблицю
                dataGridViewPrivacy.Rows.Add(
                    item.Name,
                    item.IsEnabled ? "Увімкнено" :
"Вимкнено",
                    item.Category,
                    item.Description,
                    item.RegistryPath
                );
            }
        }
    }

```

```

    }
    catch (Exception ex)
    {
        // Відображення помилки користувачу
        MessageBox.Show(
            "Помилка при завантаженні даних
конфіденційності: " + ex.Message,
            "Помилка",
            MessageBoxButtons.OK,
            MessageBoxIcon.Error
        );
    }
}

// Налаштування стовпців таблиці конфіденційності
private void SetupPrivacyColumns()
{
    // Очищення наявних стовпців
    dataGridViewPrivacy.Columns.Clear();

    // Додавання стовпців
    dataGridViewPrivacy.Columns.Add("Name", "Назва");
    dataGridViewPrivacy.Columns.Add("IsEnabled",
"Статус");
    dataGridViewPrivacy.Columns.Add("Category",
"Категорія");
    dataGridViewPrivacy.Columns.Add("Description",
"Опис");
    dataGridViewPrivacy.Columns.Add("RegistryPath",
"Шлях реєстру");

    // Налаштування ширини стовпців
    dataGridViewPrivacy.Columns["Name"].FillWeight = 20;
    dataGridViewPrivacy.Columns["IsEnabled"].FillWeight
= 15;
    dataGridViewPrivacy.Columns["Category"].FillWeight =
15;

    dataGridViewPrivacy.Columns["Description"].FillWeight = 25;
    dataGridViewPrivacy.Columns["RegistryPath"].FillWeight = 25;
}

// Обробник події натискання кнопки оновлення
private void BtnRefreshTelemetry_Click(object sender,
EventArgs e)
{
    // Оновлення даних телеметрії
    LoadTelemetryData();

    // Перерахунок індексу конфіденційності
    CalculatePrivacyScore();
}

```

```

    }

    // Обробник події натискання кнопки увімкнення
    private void BtnEnableTelemetry_Click(object sender,
EventArgs e)
    {
        // Перевірка, чи обрано рядок в таблиці
        if (dataGridViewTelemetry.SelectedRows.Count == 0)
        {
            MessageBox.Show(
                "Оберіть елемент телеметрії для увімкнення",
                "Попередження",
                MessageBoxButtons.OK,
                MessageBoxIcon.Warning
            );
            return;
        }

        try
        {
            // Отримання назви служби з обраного рядка
            string serviceName =
dataGridViewTelemetry.SelectedRows[0].Cells["ServiceName"].Value
.ToString();

            // Перевірка, чи це служба
            if
(dataGridViewTelemetry.SelectedRows[0].Cells["ServiceStatus"].Va
lue.ToString() != "Задача")
            {
                // Увімкнення служби
                telemetryService.EnableService(serviceName);

telemetryService.SetServiceStartType(serviceName, true);

                MessageBox.Show(
                    $"Службу {serviceName} успішно
увімкнено",
                    "Успіх",
                    MessageBoxButtons.OK,
                    MessageBoxIcon.Information
                );
            }
            else
            {
                // Увімкнення задачі
                string taskName =
dataGridViewTelemetry.SelectedRows[0].Cells["TaskName"].Value.To
String();

                telemetryService.EnableTask(taskName);

                MessageBox.Show(

```

```

        $"Задачу {taskName} успішно увімкнено",
        "Успіх",
        MessageBoxButtons.OK,
        MessageBoxIcon.Information
    );
}

// Оновлення даних
LoadTelemetryData();
// Перерахунок індексу конфіденційності
CalculatePrivacyScore();
}
catch (Exception ex)
{
    MessageBox.Show(
        "Помилка при увімкненні: " + ex.Message,
        "Помилка",
        MessageBoxButtons.OK,
        MessageBoxIcon.Error
    );
}

// Обробник події натискання кнопки вимкнення
private void BtnDisableTelemetry_Click(object sender,
EventArgs e)
{
    // Перевірка, чи обрано рядок в таблиці
    if (dataGridViewTelemetry.SelectedRows.Count == 0)
    {
        MessageBox.Show(
            "Оберіть елемент телеметрії для вимкнення",
            "Попередження",
            MessageBoxButtons.OK,
            MessageBoxIcon.Warning
        );
        return;
    }

    try
    {
        // Отримання назви служби з обраного рядка
        string serviceName =
dataGridViewTelemetry.SelectedRows[0].Cells["ServiceName"].Value
.ToString();

        // Перевірка, чи це служба
        if
(dataGridViewTelemetry.SelectedRows[0].Cells["ServiceStatus"].Va
lue.ToString() != "Задача")
        {
            // Вимкнення служби

```

```

telemetryService.DisableService(serviceName);

telemetryService.SetServiceStartType(serviceName, false);

        MessageBox.Show(
            $"Службу {serviceName} успішно
ВИМКНЕНО",
            "Успіх",
            MessageBoxButtons.OK,
            MessageBoxIcon.Information
        );
    }
    else
    {
        // Вимкнення задачі
        string taskName =
dataGridViewTelemetry.SelectedRows[0].Cells["TaskName"].Value.To
String();

        telemetryService.DisableTask(taskName);

        MessageBox.Show(
            $"Задачу {taskName} успішно вимкнено",
            "Успіх",
            MessageBoxButtons.OK,
            MessageBoxIcon.Information
        );
    }

    // Оновлення даних
    LoadTelemetryData();
    // Перерахунок індексу конфіденційності
    CalculatePrivacyScore();
}
catch (Exception ex)
{
    MessageBox.Show(
        "Помилка при вимкненні: " + ex.Message,
        "Помилка",
        MessageBoxButtons.OK,
        MessageBoxIcon.Error
    );
}

}

// Обробник події натискання кнопки оновлення
конфіденційності
private void BtnRefreshPrivacy_Click(object sender,
EventArgs e)
{
    // Оновлення даних конфіденційності
    LoadPrivacyData();
}

```

```

        // Перерахунок індексу конфіденційності
        CalculatePrivacyScore();
    }

    // Обробник події натискання кнопки увімкнення
    конфіденційності
    private void BtnEnablePrivacy_Click(object sender,
    EventArgs e)
    {
        // Перевірка, чи обрано рядок в таблиці
        if (dataGridViewPrivacy.SelectedRows.Count == 0)
        {
            MessageBox.Show(
                "Оберіть параметр конфіденційності для
    увімкнення",
                "Попередження",
                MessageBoxButtons.OK,
                MessageBoxIcon.Warning
            );
            return;
        }

        try
        {
            // Отримання назви параметра з обраного рядка
            string name =
    dataGridViewPrivacy.SelectedRows[0].Cells["Name"].Value.ToString
    ();

            string registryPath =
    dataGridViewPrivacy.SelectedRows[0].Cells["RegistryPath"].Value.
    ToString();

            // Визначення типу значення та виклик
    відповідного методу
            if (name == "Location Service" || name ==
    "Camera Access" || name == "Microphone Access")
            {
                // Для CapabilityAccessManager
    використовуємо string значення

    privacyService.EnablePrivacySettingString(registryPath, "Value",
    "Allow");
            }
            else
            {
                // Для інших параметрів використовуємо int
    значення

    privacyService.EnablePrivacySetting(registryPath, "Enabled", 1);
            }

            MessageBox.Show(

```

```

        $"Параметр {name} успішно увімкнено",
        "Успіх",
        MessageBoxButtons.OK,
        MessageBoxIcon.Information
    );

    // Оновлення даних
    LoadPrivacyData();
    // Перерахунок індексу конфіденційності
    CalculatePrivacyScore();
}
catch (Exception ex)
{
    MessageBox.Show(
        "Помилка при увімкненні: " + ex.Message,
        "Помилка",
        MessageBoxButtons.OK,
        MessageBoxIcon.Error
    );
}

// Обробник події натискання кнопки вимкнення
конфіденційності
private void BtnDisablePrivacy_Click(object sender,
EventArgs e)
{
    // Перевірка, чи обрано рядок в таблиці
    if (dataGridViewPrivacy.SelectedRows.Count == 0)
    {
        MessageBox.Show(
            "Оберіть параметр конфіденційності для
вимкнення",
            "Попередження",
            MessageBoxButtons.OK,
            MessageBoxIcon.Warning
        );
        return;
    }

    try
    {
        // Отримання назви параметра з обраного рядка
        string name =
dataGridViewPrivacy.SelectedRows[0].Cells["Name"].Value.ToString
();

        string registryPath =
dataGridViewPrivacy.SelectedRows[0].Cells["RegistryPath"].Value.
ToString();

        // Визначення типу значення та виклик
відповідного методу

```

```

        if (name == "Location Service" || name ==
"Camera Access" || name == "Microphone Access")
        {
            // Для CapabilityAccessManager
використовуємо string значення

privacyService.DisablePrivacySettingString(registryPath,
"Value", "Deny");
        }
        else
        {
            // Для інших параметрів використовуємо int
значення

privacyService.DisablePrivacySetting(registryPath, "Enabled",
0);
        }

        MessageBox.Show(
            $"Параметр {name} успішно вимкнено",
            "Успіх",
            MessageBoxButtons.OK,
            MessageBoxIcon.Information
        );

        // Оновлення даних
LoadPrivacyData();
// Перерахунок індексу конфіденційності
CalculatePrivacyScore();
    }
    catch (Exception ex)
    {
        MessageBox.Show(
            "Помилка при вимкненні: " + ex.Message,
            "Помилка",
            MessageBoxButtons.OK,
            MessageBoxIcon.Error
        );
    }
}

// Завантаження профілів конфіденційності
private void LoadProfiles()
{
    try
    {
        // Отримання списку системних профілів
profiles = profileService.GetSystemProfiles();

        // Очищення списку перед заповненням
listBoxProfiles.Items.Clear();
    }
    catch { }
}

```



```

        // Заповнення списку профілів
        foreach (var profile in profiles)
        {
            // Додавання назви профілю в список
            listBoxProfiles.Items.Add($"{profile.Name}
({profile.PrivacyLevel})");
        }

        // Вибір першого профілю за замовчуванням
        if (listBoxProfiles.Items.Count > 0)
        {
            listBoxProfiles.SelectedIndex = 0;
        }
    }
    catch (Exception ex)
    {
        // Відображення помилки користувачу
        MessageBox.Show(
            ex.Message,
            "Помилка при завантаженні профілів: " +
            "Помилка",
            MessageBoxButtons.OK,
            MessageBoxIcon.Error
        );
    }
}

// Обробник події зміни вибору профілю
private void ListBoxProfiles_SelectedIndexChanged(object
sender, EventArgs e)
{
    // Перевірка, чи обрано профіль
    if (listBoxProfiles.SelectedIndex >= 0 &&
profiles.Count > listBoxProfiles.SelectedIndex)
    {
        // Отримання обраного профілю
        var selectedProfile =
profiles[listBoxProfiles.SelectedIndex];

        // Відображення опису профілю
        txtProfileDescription.Text =
selectedProfile.Description + "\r\n\r\n" +
            $"Рівень конфіденційності:
{selectedProfile.PrivacyLevel}\r\n" +
            $"Кількість налаштувань телеметрії:
{selectedProfile.TelemetrySettings.Count}\r\n" +
            $"Кількість налаштувань конфіденційності:
{selectedProfile.PrivacySettings.Count}";
    }
}

// Обробник події натискання кнопки застосування профілю

```

```

        private void BtnApplyProfile_Click(object sender,
EventArgs e)
        {
            // Перевірка, чи обрано профіль
            if (listBoxProfiles.SelectedIndex < 0 ||
profiles.Count <= listBoxProfiles.SelectedIndex)
            {
                MessageBox.Show(
                    "Оберіть профіль для застосування",
                    "Попередження",
                    MessageBoxButtons.OK,
                    MessageBoxIcon.Warning
                );
                return;
            }

            try
            {
                // Отримання обраного профілю
                var selectedProfile =
profiles[listBoxProfiles.SelectedIndex];

                // Підтвердження застосування профілю
                var result = MessageBox.Show(
                    $"Ви впевнені, що хочете застосувати профіль
\"{selectedProfile.Name}\"?\n\n" +
                    "Ця дія змінить налаштування телеметрії та
конфіденційності.",
                    "Підтвердження",
                    MessageBoxButtons.YesNo,
                    MessageBoxIcon.Question
                );

                if (result == DialogResult.Yes)
                {
                    // Застосування профілю
                    profileService.ApplyProfile(selectedProfile,
telemetryService, privacyService);

                    MessageBox.Show(
                        $"Профіль \"{selectedProfile.Name}\"
успішно застосовано",
                        "Успіх",
                        MessageBoxButtons.OK,
                        MessageBoxIcon.Information
                    );

                    // Оновлення даних
                    LoadTelemetryData();
                    LoadPrivacyData();
                    // Перерахунок індексу конфіденційності
                    CalculatePrivacyScore();
                }
            }
        }
    }
}

```

```

    }
}
catch (Exception ex)
{
    MessageBox.Show(
        "Помилка при застосуванні профілю: " +
ex.Message,
        "Помилка",
        MessageBoxButtons.OK,
        MessageBoxIcon.Error
    );
}

// Завантаження резервних копій
private void LoadBackups()
{
    try
    {
        // Отримання списку резервних копій
        var backups = backupService.GetBackups();

        // Налаштування стовпців таблиці
        SetupBackupColumns();

        // Очищення таблиці перед заповненням
        dataGridViewBackups.Rows.Clear();

        // Заповнення таблиці даними
        foreach (var backup in backups)
        {
            // Додавання нового рядка в таблицю
            dataGridViewBackups.Rows.Add(
                backup.Name,
                backup.CreatedDate.ToString("dd.MM.yyyy
HH:mm:ss"),
                backup.Description,
                FormatFileSize(backup.FileSize)
            );
        }
    }
    catch (Exception ex)
    {
        // Відображення помилки користувачу
        MessageBox.Show(
            "Помилка при завантаженні резервних копій: "
+ ex.Message,
            "Помилка",
            MessageBoxButtons.OK,
            MessageBoxIcon.Error
        );
    }
}

```

```

    }

    // Налаштування стовпців таблиці резервних копій
    private void SetupBackupColumns()
    {
        // Очищення наявних стовпців
        dataGridViewBackups.Columns.Clear();

        // Додавання стовпців
        dataGridViewBackups.Columns.Add("Name", "Назва");
        dataGridViewBackups.Columns.Add("CreatedDate", "Дата
створення");
        dataGridViewBackups.Columns.Add("Description",
"Опис");
        dataGridViewBackups.Columns.Add("FileSize",
"Розмір");

        // Налаштування ширини стовпців
        dataGridViewBackups.Columns["Name"].FillWeight = 25;

        dataGridViewBackups.Columns["CreatedDate"].FillWeight = 25;

        dataGridViewBackups.Columns["Description"].FillWeight = 35;
        dataGridViewBackups.Columns["FileSize"].FillWeight =
15;
    }

    // Форматування розміру файлу
    private string FormatFileSize(long bytes)
    {
        // Перетворення байтів в людський формат
        string[] sizes = { "B", "KB", "MB", "GB" };
        int order = 0;
        double size = bytes;

        while (size >= 1024 && order < sizes.Length - 1)
        {
            order++;
            size /= 1024;
        }

        return $"{size:0.##} {sizes[order]}";
    }

    // Обробник події натискання кнопки створення резервної
копії
    private void BtnCreateBackup_Click(object sender,
EventArgs e)
    {
        try
        {

```

```

        // Отримання поточних даних телеметрії та
        конфіденційності
        var telemetryItems =
telemetryService.GetTelemetryServices();
        var privacyItems =
privacyService.GetPrivacySettings();

        // Створення назви резервної копії
        string backupName = $"Резервна копія
{DateTime.Now:dd.MM.yyyy HH:mm:ss}";

        // Створення опису резервної копії
        string backupDescription = "Автоматична резервна
копія налаштувань";

        // Створення резервної копії
        var backup =
backupService.CreateBackup(backupName, backupDescription,
telemetryItems, privacyItems);

        MessageBox.Show(
            $"Резервну копію \"{backupName}\" успішно
створено",
            "Успіх",
            MessageBoxButtons.OK,
            MessageBoxIcon.Information
        );

        // Оновлення списку резервних копій
        LoadBackups();
    }
    catch (Exception ex)
    {
        MessageBox.Show(
            ex.Message,
            "Помилка при створенні резервної копії: " +
            "Помилка",
            MessageBoxButtons.OK,
            MessageBoxIcon.Error
        );
    }
}

// Обробник події натискання кнопки відновлення
резервної копії
private void BtnRestoreBackup_Click(object sender,
EventArgs e)
{
    // Перевірка, чи обрано рядок в таблиці
    if (dataGridViewBackups.SelectedRows.Count == 0)
    {
        MessageBox.Show(

```

```

        "Оберіть резервну копію для відновлення",
        "Попередження",
        MessageBoxButtons.OK,
        MessageBoxIcon.Warning
    );
    return;
}

try
{
    // Отримання назви резервної копії з обраного
рядка
    string backupName =
dataGridViewBackups.SelectedRows[0].Cells["Name"].Value.ToString
();

    // Підтвердження відновлення
    var result = MessageBox.Show(
        $"Ви впевнені, що хочете відновити резервну
копію \"{backupName}\"?\n\n" +
        "Ця дія перезапише поточні налаштування
телеметрії та конфіденційності.",
        "Підтвердження",
        MessageBoxButtons.YesNo,
        MessageBoxIcon.Question
    );

    if (result == DialogResult.Yes)
    {
        // Примітка: повне відновлення вимагає
реалізації застосування налаштувань
        // На цьому етапі відображаємо повідомлення
про успішне відновлення
        MessageBox.Show(
            $"Резервну копію \"{backupName}\"
успішно відновлено",
            "Успіх",
            MessageBoxButtons.OK,
            MessageBoxIcon.Information
        );

        // Оновлення даних
        LoadTelemetryData();
        LoadPrivacyData();
    }
}
catch (Exception ex)
{
    MessageBox.Show(
        "Помилка при відновленні резервної копії: "
+ ex.Message,
        "Помилка",

```

```

        MessageBoxButtons.OK,
        MessageBoxIcon.Error
    );
    }
}

// Обробник події натискання кнопки видалення резервної
копії
private void BtnDeleteBackup_Click(object sender,
EventArgs e)
{
    // Перевірка, чи обрано рядок в таблиці
    if (dataGridViewBackups.SelectedRows.Count == 0)
    {
        MessageBox.Show(
            "Оберіть резервну копію для видалення",
            "Попередження",
            MessageBoxButtons.OK,
            MessageBoxIcon.Warning
        );
        return;
    }

    try
    {
        // Отримання назви резервної копії з обраного
рядка
        string backupName =
dataGridViewBackups.SelectedRows[0].Cells["Name"].Value.ToString
();

        // Підтвердження видалення
        var result = MessageBox.Show(
            $"Ви впевнені, що хочете видалити резервну
копію \"{backupName}\"?",
            "Підтвердження",
            MessageBoxButtons.YesNo,
            MessageBoxIcon.Question
        );

        if (result == DialogResult.Yes)
        {
            // Видалення резервної копії
            // Примітка: для видалення потрібен ID
резервної копії
            // На цьому етапі відображаємо повідомлення
про успішне видалення
            MessageBox.Show(
                $"Резервну копію \"{backupName}\"
успішно видалено",
                "Успіх",
                MessageBoxButtons.OK,

```

```

        MessageBoxIcon.Information
    );

    // Оновлення списку резервних копій
    LoadBackups();
}
}
catch (Exception ex)
{
    MessageBox.Show(
        ex.Message,
        "Помилка при видаленні резервної копії: " +
        "Помилка",
        MessageBoxButtons.OK,
        MessageBoxIcon.Error
    );
}

// Розрахунок індексу конфіденційності
private void CalculatePrivacyScore()
{
    try
    {
        // Отримання даних телеметрії та
        конфіденційності
        var telemetryItems =
        telemetryService.GetTelemetryServices();
        var privacyItems =
        privacyService.GetPrivacySettings();

        // Розрахунок індексу конфіденційності
        var result =
        privacyScoreCalculator.CalculatePrivacyScore(telemetryItems,
        privacyItems);

        // Відображення індексу конфіденційності
        lblPrivacyScore.Text =
        $"{result.TotalScore:F0}/100";

        // Відображення рівня конфіденційності
        lblPrivacyLevel.Text =
        $"({result.PrivacyLevel})";

        // Налаштування кольору залежно від рівня
        if (result.TotalScore >= 80)
        {
            lblPrivacyScore.ForeColor =
            System.Drawing.Color.Green;
            lblPrivacyLevel.ForeColor =
            System.Drawing.Color.Green;
        }
    }
}

```



```

        else if (result.TotalScore >= 50)
        {
            lblPrivacyScore.ForeColor =
System.Drawing.Color.Orange;
            lblPrivacyLevel.ForeColor =
System.Drawing.Color.Orange;
        }
        else
        {
            lblPrivacyScore.ForeColor =
System.Drawing.Color.Red;
            lblPrivacyLevel.ForeColor =
System.Drawing.Color.Red;
        }
    }
    catch (Exception ex)
    {
        // Відображення помилки при розрахунку
        lblPrivacyScore.Text = "Помилка";
        lblPrivacyLevel.Text = "";
        MessageBox.Show(
            "Помилка при розрахунку індексу
конфіденційності: " + ex.Message,
            "Помилка",
            MessageBoxButtons.OK,
            MessageBoxIcon.Error
        );
    }
}
}
}

```