Міністерство освіти і науки України

Національний університет

«Львівська політехніка»

Кафедра електронних обчислювальних машин

**КУРСОВИЙ ПРОЄКТ**

з дисципліни “Системне програмне забезпечення”

на тему: “Розробка утиліти для моніторингу системи. розробка програми, що дозволяє моніторити систему на комп'ютері з операційною системою windows та отримувати інформацію про поточний стан системи”

Студента 3-го курсу групи КІ-306

123 «Комп’ютерна інженерія»

Ярмола Ю. Ю.

Керівник

Олексів М. В.

Національна шкала: \_\_\_\_\_\_\_

Кількість балів: \_\_\_\_\_\_\_

Оцінка ECTS: \_\_\_\_\_\_\_

Члени комісії: \_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали)

\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали)

\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали)

Львів 2024

# Анотація

У даній курсовій роботі представлено розробку утиліти для моніторингу системи на комп'ютерах з операційною системою Windows. Основною метою цієї утиліти є надання користувачам детальної інформації про поточний стан системи, що включає використання центрального процесора (CPU), оперативної пам'яті (RAM), стан жорстких дисків, мережеву активність та інші ключові параметри.

У роботі розглянуто основні методи та інструменти системного програмування, що використовуються для отримання та обробки системної інформації на платформі Windows. Описано процес розробки утиліти, включаючи вибір мови програмування, структуру програми, реалізацію основних функцій та інтерфейс користувача.

Розроблена утиліта має зручний графічний інтерфейс, що дозволяє користувачам легко відстежувати та аналізувати стан системи в режимі реального часу. Особлива увага приділена ефективності роботи утиліти та мінімізації її впливу на продуктивність системи.

Результати тестування показують, що утиліта успішно виконує поставлені завдання та може бути корисним інструментом для адміністраторів систем, розробників програмного забезпечення та інших користувачів, які потребують надійної інформації про стан системи.

# Зміст

[Анотація 2](#_Toc166710244)

[Зміст 3](#_Toc166710245)

[Вступ 4](#_Toc166710246)

[Аналітика 5](#_Toc166710247)

[Проектування 7](#_Toc166710248)

[Огляд 7](#_Toc166710249)

[Цілі програми 7](#_Toc166710250)

[Аудиторія 7](#_Toc166710251)

[Спосіб запуску 7](#_Toc166710252)

[Функціональність 8](#_Toc166710253)

[User Story 1: Моніторинг використання CPU 10](#_Toc166710254)

[User Story 2: Моніторинг використання пам'яті 11](#_Toc166710255)

[User Story 3: Моніторинг дискового простору 11](#_Toc166710256)

[User Story 4: Моніторинг мережевого трафіку 12](#_Toc166710257)

[User Story 5: Перевірка стану системи в реальному часі 12](#_Toc166710258)

[User Story 6: Моніторинг системних подій та журналів 13](#_Toc166710259)

[User Story 7: Моніторинг температури та вентиляції 13](#_Toc166710260)

[User Story 8: Системні інформаційні дані 13](#_Toc166710261)

[User Story 9: Налаштування сповіщень 14](#_Toc166710262)

[User Story 10: Ефективність роботи 14](#_Toc166710263)

[Реалізація 16](#_Toc166710264)

[Дизайн 16](#_Toc166710265)

[Високорівневий опис реалізації програмного рішення згідно розроблених вимог 16](#_Toc166710266)

[Структура програмного рішення; архітектуру класів, їх призначення і взаємодію, опис функцій 17](#_Toc166710267)

[Архітектура класів та функцій моніторингу, їх призначення і взаємодія 17](#_Toc166710268)

[Опис функцій інтерфейсу для взаємодії з користувачами 20](#_Toc166710269)

[Експорт 21](#_Toc166710270)

[Тестування 22](#_Toc166710271)

[Висновки 29](#_Toc166710272)

[список використаних джерел 30](#_Toc166710273)

[додатки 31](#_Toc166710274)

# Вступ

Сучасні комп'ютерні системи є складними комплексами апаратних і програмних компонентів, які працюють у тісній взаємодії. Забезпечення стабільної та ефективної роботи таких систем вимагає постійного моніторингу їхнього стану. Особливо це актуально для серверів, робочих станцій та інших критичних систем, де будь-яка несправність або зниження продуктивності може призвести до значних втрат даних або ресурсів.

Моніторинг системи дозволяє вчасно виявляти та усувати проблеми, аналізувати навантаження на апаратні ресурси, а також оптимізувати роботу програмного забезпечення. У зв'язку з цим існує потреба у надійних та зручних інструментах, які б надавали користувачам детальну інформацію про поточний стан системи.

Дана курсова робота присвячена розробці утиліти для моніторингу системи на комп'ютерах з операційною системою Windows. Утиліта надає користувачам можливість отримувати дані про використання центрального процесора, оперативної пам'яті, жорстких дисків, мережеву активність та інші ключові параметри системи в режимі реального часу.

Метою роботи є створення ефективного інструменту для моніторингу системи, який буде корисним як для звичайних користувачів, так і для системних адміністраторів та розробників програмного забезпечення. У процесі розробки утиліти будуть використані сучасні методи та технології системного програмування, що дозволить забезпечити високу продуктивність та надійність роботи програми.

У роботі розглядаються теоретичні аспекти системного моніторингу, методи збору та аналізу системної інформації, а також практичні питання реалізації програмного забезпечення для моніторингу системи. Результати даного дослідження можуть бути використані для подальшого розвитку інструментів моніторингу та оптимізації роботи комп'ютерних систем.

# Аналітика

Моніторинг системи – це процес спостереження за станом та продуктивністю комп'ютерної системи з метою забезпечення її стабільної та ефективної роботи. Основні показники, які зазвичай моніторяться, включають використання центрального процесора (CPU), оперативної пам'яті (RAM), жорстких дисків, мережеву активність та інші ключові параметри. Розробка утиліти для моніторингу системи на платформі Windows потребує застосування специфічних методів і технологій, що дозволяють отримувати та обробляти відповідну інформацію в реальному часі.

**Огляд методів**

* **Використання системних API**

1. Одним із основних методів отримання інформації про стан системи є використання системних API (Application Programming Interface), наданих операційною системою Windows. Це дозволяє програмі доступатися до низькорівневих даних та отримувати детальну інформацію про стан різних компонентів системи.
2. Performance Counters API: Дозволяє збирати дані про продуктивність різних компонентів системи, включаючи CPU, пам'ять, диски та мережеву активність.
3. Windows Management Instrumentation (WMI): Надає розширені можливості для управління та моніторингу системи, дозволяючи виконувати запити на отримання інформації про апаратне забезпечення та програмні компоненти.
4. QueryPerformanceCounter: Використовується для високоточних вимірювань часу, що корисно при аналізі продуктивності додатків.

* **Використання бібліотек та фреймворків**

Існують готові бібліотеки та фреймворки, що спрощують процес розробки утиліт для моніторингу системи. Вони надають високорівневі інтерфейси для доступу до системної інформації, що знижує складність розробки.

1. .NET System.Diagnostics: Набір класів у середовищі .NET, які дозволяють отримувати інформацію про процеси, продуктивність системи та інші аспекти.
2. Open Hardware Monitor: Безкоштовне програмне забезпечення з відкритим вихідним кодом, яке дозволяє моніторити температури, напруги, швидкість вентиляторів та завантаження процесора.
3. Performance Data Helper (PDH): API для доступу до даних про продуктивність Windows, дозволяє легко збирати та інтерпретувати дані.

**Існуючі розробки**

* **Системні** **утиліти** **Windows**

1. Task Manager: Вбудований інструмент Windows, який надає базову інформацію про використання CPU, пам'яті, диска та мережі. Task Manager також дозволяє управляти запущеними процесами та службами.
2. Resource Monitor: Більш розширений інструмент, який дозволяє детально аналізувати використання ресурсів системи, включаючи активність дисків, мережі та пам'яті.

* **Комерційні та відкриті утиліти**

1. HWMonitor: Популярна утиліта для моніторингу температури та напруги компонентів комп'ютера. Вона підтримує широкий спектр апаратного забезпечення та надає детальну інформацію про його стан.
2. AIDA64: Потужний інструмент для діагностики та моніторингу системи, який пропонує широкі можливості для тестування та отримання інформації про всі аспекти комп'ютера.
3. Open Hardware Monitor: Як вже зазначалося, це безкоштовне програмне забезпечення з відкритим вихідним кодом, яке дозволяє моніторити широкий спектр показників апаратного забезпечення.

Висновки

Моніторинг системи є важливим аспектом забезпечення стабільної та ефективної роботи комп'ютерів. Існує широкий спектр методів та інструментів для отримання інформації про стан системи на платформі Windows. Використання системних API, таких як Performance Counters, WMI, та готових бібліотек, дозволяє розробникам створювати потужні утиліти для моніторингу.

Розробка власної утиліти для моніторингу системи на базі цих методів та інструментів дозволяє отримати гнучкий та зручний інструмент для спостереження за станом комп'ютерної системи, що може бути корисним як для звичайних користувачів, так і для професійних адміністраторів та розробників.

# Проектування

## Огляд

Програма " MoniSy " розроблена для моніторингу та аналізу операційної системи Windows. Її основна мета - забезпечити користувачам детальну інформацію про стан їхньої системи в реальному часі. Програма має широкий спектр функцій, включаючи моніторинг використання ресурсів, аналіз поточних процесів та системних подій, а також сповіщення про перевищення заданих порогових значень.

## Цілі програми

1. Надати користувачам інструмент для моніторингу та аналізу різних аспектів роботи операційної системи Windows.
2. Забезпечити можливість вчасного виявлення проблем з ресурсами системи.
3. Забезпечити зручний та налаштовуваний інтерфейс для відслідковування ключових показників продуктивності та роботи системи.

## Аудиторія

Програма адресована широкому колу користувачів операційної системи Windows, включаючи:

- Досвідчених користувачів, які прагнуть отримати детальну інформацію про роботу своєї системи для оптимізації її продуктивності.

- Системних адміністраторів, які відповідають за моніторинг та підтримку деякої кількості комп'ютерів або серверів під управлінням Windows.

- Користувачів, які стикаються з проблемами продуктивності або виявляють неефективне використання ресурсів системи.

- Користувачів, які бажають отримати дані про систему охолодження та навантаження ресурсів.

## Спосіб запуску

Програма " MoniSy " буде поставлятися у вигляді виконуваного файлу (.exe), який можна буде завантажити та встановити на комп'ютер з операційною системою Windows, або його портативну версію. Крім того, можливе поширення через інтернет-магазини або пряме завантаження з веб-сайту розробника.

## Функціональність

**Моніторинг використання CPU:**

Функція моніторингу використання CPU розрахована для користувачів, що хочуть отримати інформацію про завантаження ядер системи в зрозумілому та інформативному вигляді.

Вимоги:

* + Програма має забезпечувати збір та відображення інформації про використання процесора в реальному часі для ефективного контролю над ресурсами.
  + Збирати стан завантаження кожного з ядер процесора в реальному часі.
  + Відображати стан завантаження кожного з ядер процесора з мінімальною затримкою.
  + Надати можливість перегляду інформації про завантаження окремого ядра процесора.
  + Забезпечити можливість перегляду середнього завантаження всіх ядер процесора.
  + Відображення стану має бути у формі графіку.
  + Забезпечити можливість отримання детальної інформації по завантаженню процесора за допомогою миші.

**Відслідковування використання пам'яті:**

Дана функція має забезпечувати відслідковування використання оперативної пам’яті, як для всієї системи загалом, так і для окремих процесів.

Вимоги:

* + Програма має показувати статистику використання оперативної та віртуальної пам'яті для контролю над ресурсами.
  + Відображення використання оперативної та віртуальної пам'яті в реальному часі.
  + Перегляд загального обсягу використаної та доступної пам'яті.
  + Моніторинг розподілу пам'яті між процесами та задачами.
  + Надання сповіщень у випадку перевищення певних порогових значень використання пам'яті.

**Моніторинг дискового простору:**

Моніторинг дискового простору потрібен, щоб користувач міг отримати інформацію про вільне місце на комп’ютерній системі, та як можна покращити її використання

Вимоги:

* + Програма має відображати інформацію про залишковий дисковий простір на різних дисках для управління місцем на диску.
  + Відображення вільного та загального обсягу дискового простору для кожного диска.
  + Надання сповіщень у випадку недостатнього вільного дискового простору.

**Моніторинг мережевого трафіку:**

Щоб забезпечити відслідковування стабільності та швидкості інтернету, користувач має отримувати дані про використання мережі системою та процесам.

Вимоги:

* + Програма має відстежувати швидкість передачі та отримання даних через мережу для контролю над мережевими ресурсами.
  + Відображення швидкості передачі та отримання даних у реальному часі.
  + Моніторинг використання мережевого трафіку за окремими програмами або процесами.

**Перевірка стану системи в реальному часі:**

Дана функція буде об’єднувати, доповнювати та відображати загальні поточні показники системи, для зручного перегляду всіх показників.

Вимоги:

* + Програма має аналізувати поточні процеси та служби з відображенням їхньої активності для виявлення проблем.
  + Відображення списку поточних процесів та служб з їхньою активністю.
  + Моніторинг використання системних ресурсів (процесор, пам'ять, диск) кожним процесом.
  + Надання сповіщень у випадку аномальної активності або проблемних станів процесів.

**Системні події та журнали:**

Для досвідчених користувачів програма має надати доступ до системних журналів, зручного їх перегляду та надання сповіщень для налаштувань які обрав користувач.

Вимоги:

* + Програма має відображати системні події та журнали для виявлення можливих проблем.
  + Відображення системних подій та журналів у зручній для аналізу формі.
  + Фільтрація подій за типом (попередження, помилки, інформація).
  + Надання сповіщень у випадку виявлення критичних помилок або аномалій.

**Моніторинг температури та вентиляції:**

Програма має забезпечити користувача зручним інтерфейсом для перегляду температури ядер процесора, відеокарти та усіх температурних датчиків що доступні у системі. При необхідності сповіщати користувача про перевищення заданих температур, а також контролювання швидкості кулерів, якщо це дозволено.

Вимоги:

* + Програма має зчитувати температуру компонентів та швидкість обертання вентиляторів для контролю над температурним режимом.
  + Відображення температури процесора, відеокарти та інших компонентів системи.
  + Моніторинг швидкості обертання вентиляторів для кожного компонента.
  + Надання сповіщень у випадку підвищення температури до критичного рівня або недостатньої швидкості обертання вентиляторів.

**Системні інформаційні дані:**

Користувач матиме змогу швидко отримати інформацію про систему та конфігурацію комп’ютера.

Вимоги:

* + Програма має виводити основну інформацію про систему для забезпечення зручного аналізу.
  + Відображення інформації про операційну систему, версію ядра, тип процесора, обсяг оперативної та віртуальної пам'яті.
  + Показ інформації про встановлені пристрої, драйвери та мережеві параметри.

**Налаштування сповіщень:**

Користувач має можливість встановлювати порогові значення для отримання сповіщень у разі їх зниження чи підвищення.

Вимоги:

* + Програма має можливість встановлення порогових значень для різних параметрів та отримання сповіщень при їх перевищенні для оперативного реагування на проблеми.
  + Налаштування порогових значень для використання CPU, пам'яті, дискового простору, мережевого трафіку, температури та інших параметрів.
  + Надання сповіщень у випадку перевищення встановлених порогів через повідомлення у системі

**Ефективність роботи:**

Програма аналізуватиме та надаватиме короткий чи розгорнутий звіт щодо загального стану системи та рекомендації щодо його покращення.

Вимоги:

* + Програма має забезпечувати збір та аналіз відомостей щодо завантаження системи для виявлення можливих проблем або неефективності використання ресурсів.
  + Збір інформації про завантаження CPU, використання пам'яті, дискового простору та мережевого трафіку.
  + Аналіз використання ресурсів для виявлення періодів найвищого навантаження та можливих проблем.
  + Надання рекомендацій щодо оптимізації використання ресурсів та підвищення ефективності роботи системи.

## User Story 1: Моніторинг використання CPU

**Оцінка часу виконання:** 4 дні

Я, як користувач, хочу мати можливість переглядати використання CPU моєї системи, щоб слідкувати за її продуктивністю та виявляти можливі проблеми.

**Користь для користувача:**

* Забезпечення можливості моніторингу та аналізу використання CPU для забезпечення оптимальної продуктивності та вчасного виявлення можливих проблем.

**Критерії готовності:**

* Функції для збору інформації про використання CPU реалізовані.
* Створений інтерфейс для відображення цієї інформації.
* Пройдено тестування та відлагодження коду.
* Оновлена документація з описом нового функціоналу.

## User Story 2: Моніторинг використання пам'яті

**Оцінка часу виконання:** 4 дні

Я, як користувач, хочу бачити статистику використання оперативної та віртуальної пам'яті моєї системи, щоб контролювати ресурси та виявляти проблеми.

**Користь для користувача:**

* Надання можливості контролювати використання оперативної та віртуальної пам'яті для оптимізації ресурсів та вчасного виявлення проблем.

**Критерії готовності:**

* Реалізовані функції для збору інформації про використання пам'яті.
* Інтерфейс для відображення цих даних створений.
* Проведено тестування та відлагодження коду.
* Оновлена документація з описом нового функціоналу.

## User Story 3: Моніторинг дискового простору

**Оцінка часу виконання:** 2 дні

Я, як користувач, хочу бачити інформацію про залишковий дисковий простір на моїй системі, щоб контролювати використання місця та уникати проблем зі зберіганням.

**Користь для користувача:**

* Забезпечення можливості моніторингу залишкового дискового простору для попередження проблем зі зберіганням та оптимізації використання місця.

**Критерії готовності:**

* Розроблені функції для отримання даних про диск.
* Інтерфейс для відображення цих даних реалізований.
* Проведено тестування та відлагодження коду.
* Оновлена документація з описом нового функціоналу.

## User Story 4: Моніторинг мережевого трафіку

**Оцінка часу виконання:** 5 дні

Я, як користувач, хочу бачити швидкість передачі та отримання даних через мережу, щоб контролювати мережеві ресурси та виявляти проблеми зі зв'язком.

**Користь для користувача:**

* Надання можливості моніторингу мережевого трафіку для контролю ресурсів та вчасного виявлення проблем зі зв'язком.

**Критерії готовності:**

* Написані функції для збору інформації про мережевий трафік.
* Інтерфейс для відображення цих даних створений.
* Проведено тестування та відлагодження коду.
* Оновлена документація з описом нового функціоналу.

## User Story 5: Перевірка стану системи в реальному часі

**Оцінка часу виконання:** 5 дні

Я, як користувач, хочу бачити списки поточних процесів та служб з їхньою активністю та моніторити використання системних ресурсів, щоб виявляти проблеми та аномалії.

**Користь для користувача:**

* Забезпечення можливості моніторингу активності процесів та служб для вчасного виявлення проблем та аномалій у використанні системних ресурсів.

**Критерії готовності:**

* Реалізовані функції для отримання інформації про поточні процеси та служби.
* Інтерфейс для відображення цих даних розроблений.
* Проведено тестування та відлагодження коду.
* Оновлена документація з описом нового функціоналу.

## User Story 6: Моніторинг системних подій та журналів

**Оцінка часу виконання:** 2 дні

Я, як користувач, хочу бачити системні події та журнали для виявлення можливих проблем та аномалій у роботі системи.

**Користь для користувача:**

* Надання можливості моніторингу системних подій та журналів для вчасного виявлення проблем та аномалій у роботі системи.

**Критерії готовності:**

* Функції для отримання інформації про системні події та журнали реалізовані.
* Створений інтерфейс для відображення цих даних.
* Пройдено тестування та відлагодження коду.
* Оновлена документація з описом нового функціоналу.

## User Story 7: Моніторинг температури та вентиляції

**Оцінка часу виконання:** 6 днів

Я, як користувач, хочу бачити інформацію про температуру компонентів та швидкість обертання вентиляторів для контролю над температурним режимом моєї системи.

**Користь для користувача:**

* Забезпечення можливості моніторингу температури та швидкості обертання вентиляторів для попередження перегріву та забезпечення оптимального температурного режиму системи.

**Критерії готовності:**

* Функції для отримання інформації про температуру та швидкість обертання вентиляторів реалізовані.
* Інтерфейс для відображення цих даних створений.
* Пройдено тестування та відлагодження коду.
* Оновлена документація з описом нового функціоналу.

## User Story 8: Системні інформаційні дані

**Оцінка часу виконання:** 4 дні

Я, як користувач, хочу бачити основну інформацію про мою систему, таку як операційна система, версія ядра, тип процесора, обсяг оперативної та віртуальної пам'яті.

**Користь для користувача:**

* Надання можливості перегляду основної інформації про систему для швидкого доступу до важливих даних про конфігурацію системи.

**Критерії готовності:**

* Функції для отримання системної інформації реалізовані.
* Створений інтерфейс для відображення цих даних.
* Проведено тестування та відлагодження коду.
* Оновлена документація з описом нового функціоналу

## User Story 9: Налаштування сповіщень

**Оцінка часу виконання:** 4 дні

Я, як користувач, хочу мати можливість налаштувати сповіщення для різних параметрів системи, щоб оперативно реагувати на проблеми та аномалії.

**Користь для користувача:**

* Забезпечення можливості налаштування сповіщень для оперативного реагування на проблеми та аномалії в роботі системи.

**Критерії готовності:**

* Реалізовані функції для налаштування сповіщень.
* Додана можливість встановлення порогових значень для різних параметрів системи.
* Пройдено тестування та відлагодження коду.
* Оновлена документація з описом нового функціоналу.

## User Story 10: Ефективність роботи

**Оцінка часу виконання:** 7 днів

Я, як користувач, хочу бачити зведену інформацію про ефективність роботи моєї системи, щоб отримувати рекомендації щодо оптимізації використання ресурсів та підвищення її продуктивності.

**Користь для користувача:**

* Надання можливості перегляду зведеної інформації про ефективність роботи системи та отримання рекомендацій для підвищення продуктивності та оптимізації ресурсів.

**Критерії готовності:**

* Написані функції для збору і аналізу інформації про використання ресурсів системи.
* Розроблений інтерфейс для відображення зведеної інформації та надання рекомендацій.
* Пройдено тестування та відлагодження коду.
* Оновлена документація з описом нового функціоналу.

# Реалізація

Для реалізації проектованого рішення я використовував модуль flet для побудови графічного інтерфейсу користувача. Для отримання даних системи я використовував поєднання системних та встановлених модулів мови Python, а саме platform – для отримання інформації про встановлену операційну систему.

GPUtil та psutil для отримання інформації про використання ресурсів таких як пам’ять, процесор, та інші. WinTmp використовується для отримання температури ядер процесора. win32evtlog виконував роль збирача інформації про системні логи.

Використання саме цих бібліотек та модулів, обумовлене уже готовими рішеннями для ефективного отримання інформації та легкого її об’єднання. Використання їх разом дозволило отримати практично всю інформацію про hardware та software до яких має доступ комп’ютер.

## Дизайн

Для зручного інтуїтивного інтерфейсу було використано можливості бібліотеки Flet яка надає стандартні значки, підлаштування під тему користувача та розміри екрану. Також для відображення та тимчасового збереження інформації, було прийняте рішення використовувати графіки у реальному часі з буфером в 1 хв. Це дозволить отримати користувачеві інформацію про зміну завантаження та температури протягом цього часу.

## Високорівневий опис реалізації програмного рішення згідно розроблених вимог

1. **Архітектура програми:**

* Мови програмування: Програма реалізована з використанням Python та C++ для оптимальної продуктивності та швидкодії.
* Модульна структура: Кожен аспект моніторингу та аналізу ресурсів відокремлений у власні компоненти для полегшення розширення та керування.

1. **Моніторинг та аналіз ресурсів:**

* Використання системних викликів та бібліотек: Для збору даних про використання CPU, пам'яті, дискового простору та мережевого трафіку у реальному часі.
* Оптимізована структура даних: Зберігання даних відбувається з урахуванням оптимальної організації для забезпечення ефективного доступу та аналізу.

1. **Інтерфейс користувача:**

* Графічний інтерфейс: Розроблений з використанням Python Flet для швидкого та сучасного інтерфейсу на базі фреймворку Flutter
* Налаштування інтерфейсу: Користувач може налаштовувати відображення показників.

1. **Системні дані та журнали:**

* Використання системних API Windows: Для отримання доступу до системних подій та журналів, що дозволяє програмі відслідковувати системні зміни та помилки.

1. **Підтримка та поширення:**

* Підтримка операційної системи Windows: Програма підтримує версії Windows починаючи з Windows 7 для обох архітектур, 32-х та 64-бітних.

## Структура програмного рішення; архітектуру класів, їх призначення і взаємодію, опис функцій

Програмне рішення для моніторингу системи включає кілька класів, кожен з яких відповідає за збір та обробку певного типу даних. Основний клас SystemMonitor забезпечує інтеграцію та відображення даних у консолі, використовуючи інформацію, зібрану іншими класами.

## Архітектура класів та функцій моніторингу, їх призначення і взаємодія

**1. Клас SystemMonitor**

**Призначення**: Клас, що відповідає за загальний моніторинг системи через консоль, координуючи збір інформації від інших класів.

**Функції**:

* start\_monitoring(): Починає моніторинг системи, викликаючи update\_system\_info() в циклі з інтервалом в 1 секунду.
* stop\_monitoring(): Зупиняє моніторинг системи.
* update\_system\_info(): Оновлює інформацію про систему, збираючи дані про використання CPU, пам'яті, диска та швидкість вентиляторів, і відображає їх у консолі.

**Взаємодія**: Використовує методи з класів CPUUsageMonitor, MemoryUsageMonitor, DiskSpaceMonitor, NetworkTrafficMonitor, Temperature, SystemInformation.

**2. Клас CPUUsageMonitor**

**Призначення**: Збір та надання інформації про стан центрального процесора (CPU).

**Функції**:

* get\_cpu\_count(): Повертає кількість логічних процесорів.
* get\_cpu\_frequency(): Повертає поточну частоту процесора.
* get\_cpu\_utilization(): Повертає поточний рівень використання процесора.

**Взаємодія**: Використовується класом SystemMonitor для отримання даних про CPU.

**3. Клас MemoryUsageMonitor**

**Призначення**: Збір та надання інформації про використання оперативної пам'яті (RAM).

**Функції**:

* get\_available\_memory(): Повертає обсяг доступної пам'яті.
* get\_total\_memory(): Повертає загальний обсяг пам'яті.
* get\_used\_memory(): Повертає обсяг використаної пам'яті.
* get\_memory\_usage\_percentage(): Повертає відсоток використаної пам'яті.

Взаємодія: Використовується класом SystemMonitor для отримання даних про пам'ять.

**4. Клас DiskSpaceMonitor**

**Призначення**: Збір та надання інформації про використання дискового простору.

**Функції**:

* get\_all\_disks(): Повертає список всіх дискових розділів.
* get\_disk\_usage(disk): Повертає інформацію про використання дискового простору для заданого розділу.
* get\_all\_disk\_usage(): Повертає інформацію про використання дискового простору для всіх розділів.

**Взаємодія**: Використовується класом SystemMonitor для отримання даних про дисковий простір.

**5. Клас NetworkTrafficMonitor**

**Призначення**: Збір та надання інформації про мережевий трафік.

**Функції**:

* get\_network\_usage(): Повертає інформацію про мережевий трафік для всіх мережевих інтерфейсів.

**Взаємодія**: Використовується класом SystemMonitor для отримання даних про мережеву активність.

**6. Клас Temperature**

**Призначення**: Збір та надання інформації про температуру компонентів системи.

**Функції**:

* get\_cpu\_temperature(): Повертає температуру центрального процесора.
* get\_gpu\_temperature(): Повертає температуру графічного процесора.

**Взаємодія**: Використовується класом SystemMonitor для отримання даних про температуру.

**7. Клас SystemInformation**

**Призначення**: Збір та надання базової інформації про систему та її компоненти.

**Функції**:

* get\_basic\_system\_info(): Повертає базову інформацію про систему, включаючи платформу, архітектуру, процесор, оперативну пам'ять та графічні процесори.
* get\_event\_logs(total\_logs, logtype): Повертає журнали подій системи для заданого типу логів.

**Взаємодія**: Використовується класом SystemMonitor для отримання базової інформації про систему та журнали подій.

**8. Клас Processes**

**Призначення**: Збір та надання інформації про запущені процеси.

**Функції**:

* get\_processes(): Повертає список запущених процесів, включаючи їх ідентифікатори, назви, використання CPU та пам'яті.

**Взаємодія**: Використовується класом SystemMonitor для отримання даних про активні процеси.

## Опис функцій інтерфейсу для взаємодії з користувачами

**Клас SystemMonitor**

* **Призначення**: Клас для моніторингу системи через консоль.
* **Функції**:
  + start\_monitoring(): Починає моніторинг системи.
  + stop\_monitoring(): Зупиняє моніторинг системи.
  + update\_system\_info(): Оновлює інформацію про систему.

**MemoryUsageMonitor**

* **Призначення**: Клас для моніторингу використання пам'яті.
* **Функції**:
  + get\_memory\_usage\_percentage(): Повертає відсоток використаної пам'яті.
  + get\_used\_memory(): Повертає кількість використаної пам'яті в байтах.
  + get\_total\_memory(): Повертає загальну кількість пам'яті в байтах.

**CPUUsageMonitor**

* **Призначення**: Клас для моніторингу використання процесора.
* **Функції**:
  + get\_cpu\_utilization(): Повертає відсоток використання процесора.
  + get\_cpu\_frequency(): Повертає поточну частоту процесора.

**DiskSpaceMonitor**

* **Призначення**: Клас для моніторингу використання дискового простору.
* **Функції**:
  + get\_all\_disk\_usage(): Повертає інформацію про використання дискового простору для всіх дисків.

**Temperature**

* **Призначення**: Клас для моніторингу температури процесора і графічного процесора.
* **Функції**:
  + get\_cpu\_temperature(): Повертає поточну температуру процесора.
  + get\_gpu\_temperature(): Повертає поточну температуру графічного процесора.

**Processes**

* **Призначення**: Клас для моніторингу запущених процесів.
* **Функції**:
  + get\_processes(): Повертає список запущених процесів.

**NetworkTrafficMonitor**

* **Призначення**: Клас для моніторингу мережевого трафіку.
* **Функції**:
  + get\_network\_usage(): Повертає інформацію про використання мережевого трафіку.

**SystemInformation**

* **Призначення**: Клас для отримання загальної інформації про систему.
* **Функції**:
  + get\_basic\_system\_info(): Повертає базову інформацію про систему.
  + get\_event\_logs(total\_logs, logtype): Повертає журнали подій системи.

**Взаємодія між класами:**

1. SystemMonitor взаємодіє з усіма класами для збору та оновлення інформації про стан системи.
2. MemoryUsageMonitor, CPUUsageMonitor, DiskSpaceMonitor, Temperature, Processes, NetworkTrafficMonitor, SystemInformation надають інформацію, яка використовується SystemMonitor для відображення даних в інтерфейсі.

**Опис функцій:**

**Функції, пов'язані з моніторингом:**

* start\_monitoring(): Запускає моніторинг, ініціалізуючи процеси збору даних.
* stop\_monitoring(): Зупиняє процеси збору даних.
* update\_system\_info(): Оновлює поточну інформацію про систему, викликаючи відповідні методи з класів, що відповідають за збір даних (наприклад, get\_memory\_usage\_percentage(), get\_cpu\_utilization() тощо).

**Функції інтерфейсу (написані на основі бібліотеки Flet):**

* main(page: ft.Page): Головна функція для ініціалізації та рендерингу інтерфейсу.
* change\_chart(chart, value): Оновлює дані графіку.
* render\_usage(e): Відображає інформацію про використання ресурсів (CPU, RAM).
* render\_temp(e): Відображає інформацію про температуру CPU та GPU.
* render\_system\_info(e): Відображає основну інформацію про систему.
* render\_processes(e): Відображає інформацію про запущені процеси.
* render\_network(e): Відображає інформацію про мережевий трафік.
* render\_logs(e): Відображає журнали подій системи.
* change\_body(e): Змінює контент головної області інтерфейсу на основі вибраної секції.

Ця структура забезпечує чітке розділення обов'язків між класами, що дозволяє легко розширювати функціональність та підтримувати код.

## Експорт

Для зручності користувача, проект було експортовано за допомого утиліти python2exe в виконавчий .exe файл який можна запустити без попереднього встановлення. Усі бібліотеки та модулі використовуються всередині файлу. Для отримання доступу до температури процесора та захищених файлів, режим запуску за замовчуванням поставлений як «Запуск від імені адміністратора».

# Тестування

Так як системні дані отримуються через графічний інтерфейс у реальному часі, написання типових Unit тестів є невиправданим. Для тестування функцій програми я використовував порівняння з уже готовими рішеннями такими як фірмова утиліта «Gigabyte Control Center 3.0» «Налаштування» та «Диспетчер задач».

* 1. Порівняння даних про систему та пристрій

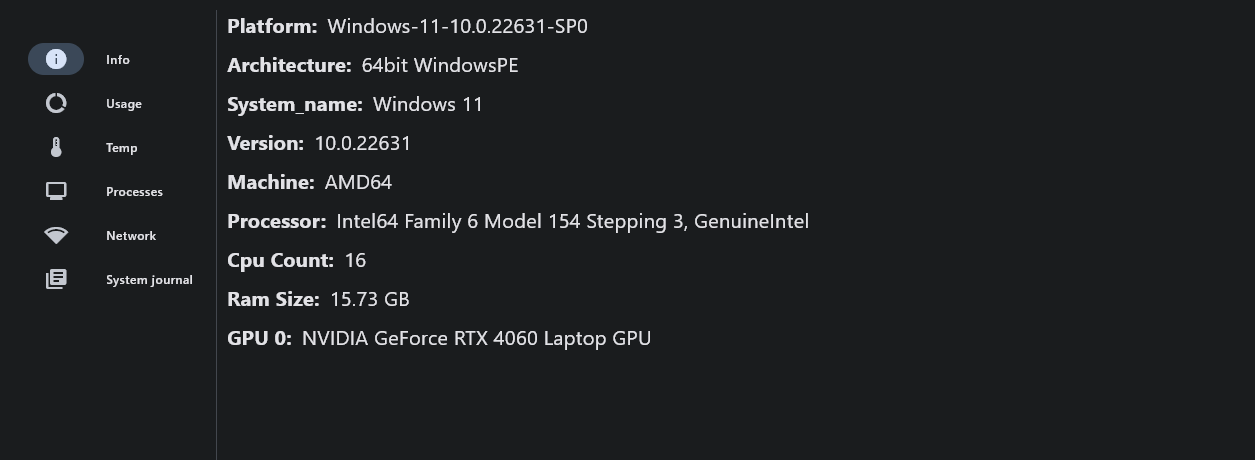


Рис х.1 – Специфікації системи у робочому проєкті

Специфікації пристрою в налаштуваннях:

Device name Yurii

Processor 12th Gen Intel(R) Core(TM) i5-12500H 2.50 GHz

Installed RAM 16.0 GB (15.7 GB usable)

Device ID 41A9F811-FF45-4C1E-9622-8DFBA282F328

Product ID 00330-80000-00000-AA704

System type 64-bit operating system, x64-based processor

Pen and touch No pen or touch input is available for this display

Специфікації Windows в налаштуваннях:

Edition Windows 11 Pro

Version 23H2

Installed on ‎2024-‎02-‎22

OS build 22631.3593

Experience Windows Feature Experience Pack 1000.22700.1003.0

* 1. Порівняння інформації про використання ресурсів

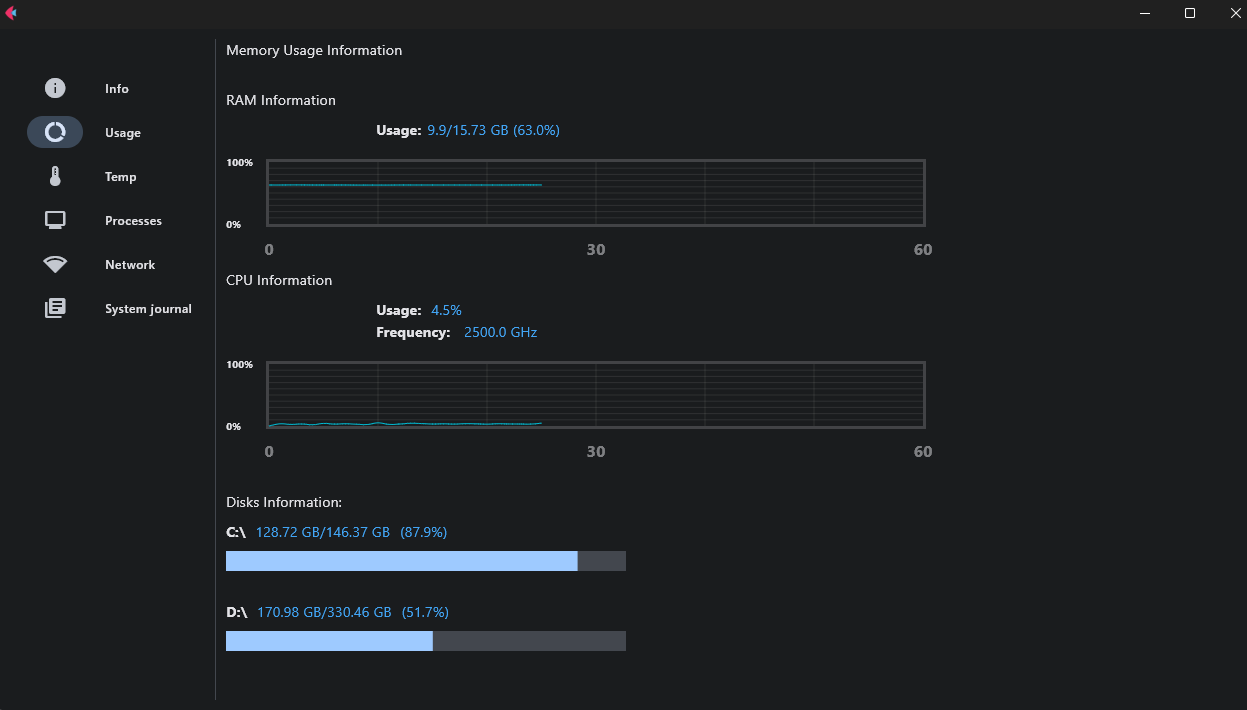


Рис х.2 – Використання ресурсів системи у робочому проєкті

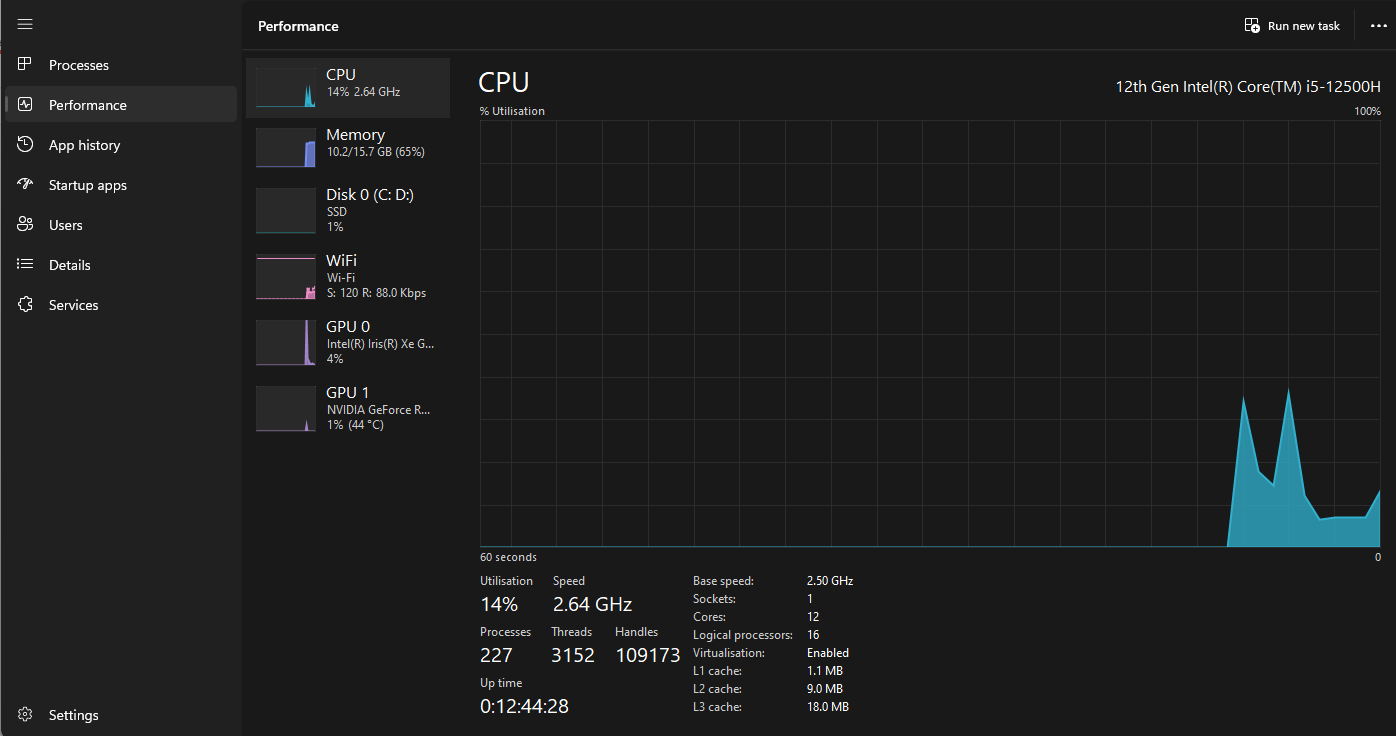


Рис х.3 – Використання ресурсів системи у «Диспетчері задач»

* 1. Температура процесора та відеокарти



Рис х.4 – Температури системи у робочому проєкті

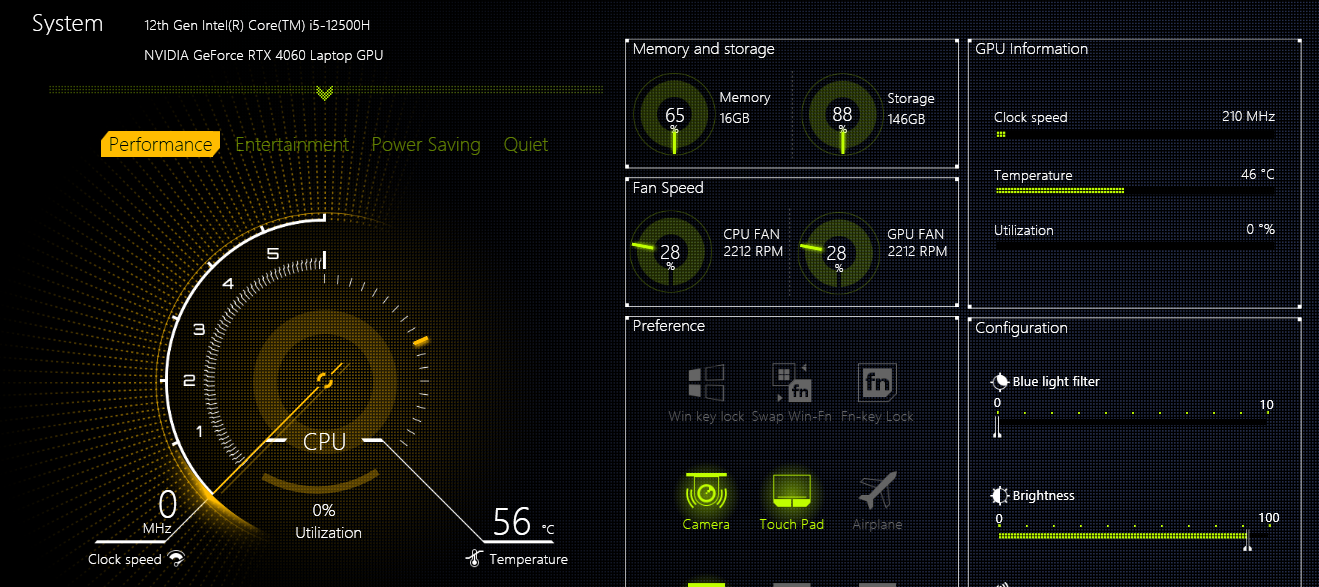


Рис х.5 – Температури системи у фірмовій утиліті ноутбуку

* 1. Активні процеси

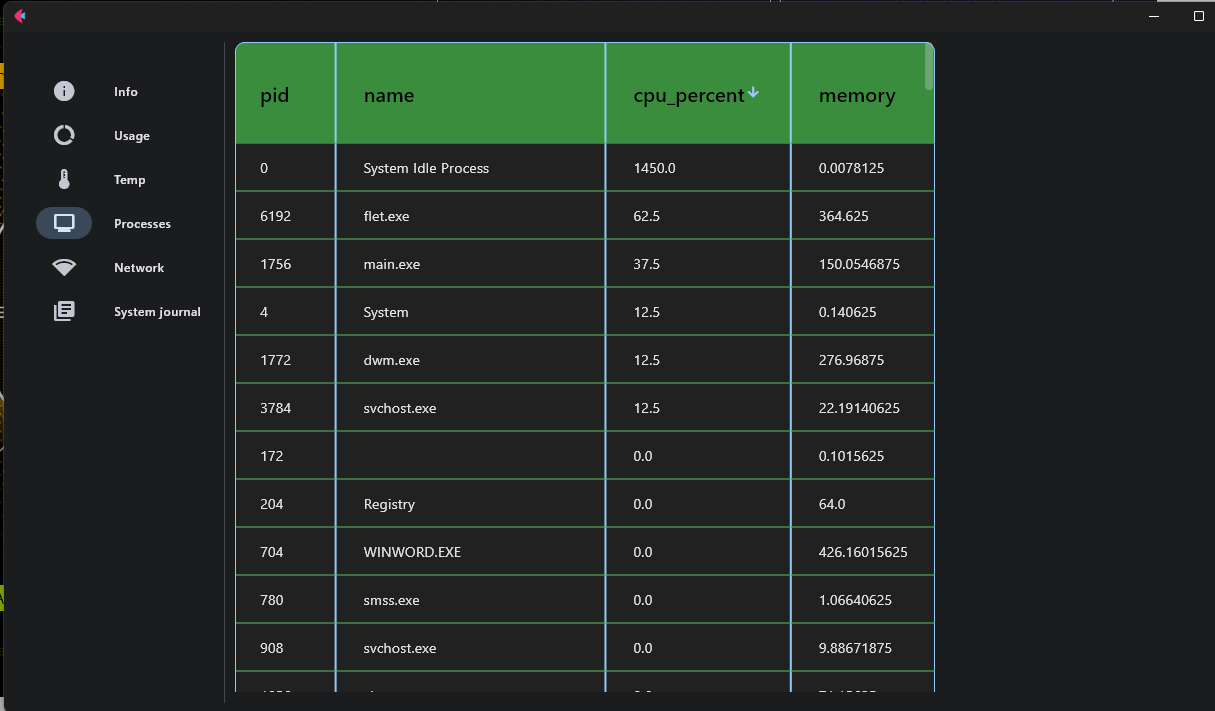


Рис х.6 – Активні процеси у робочому проєкті

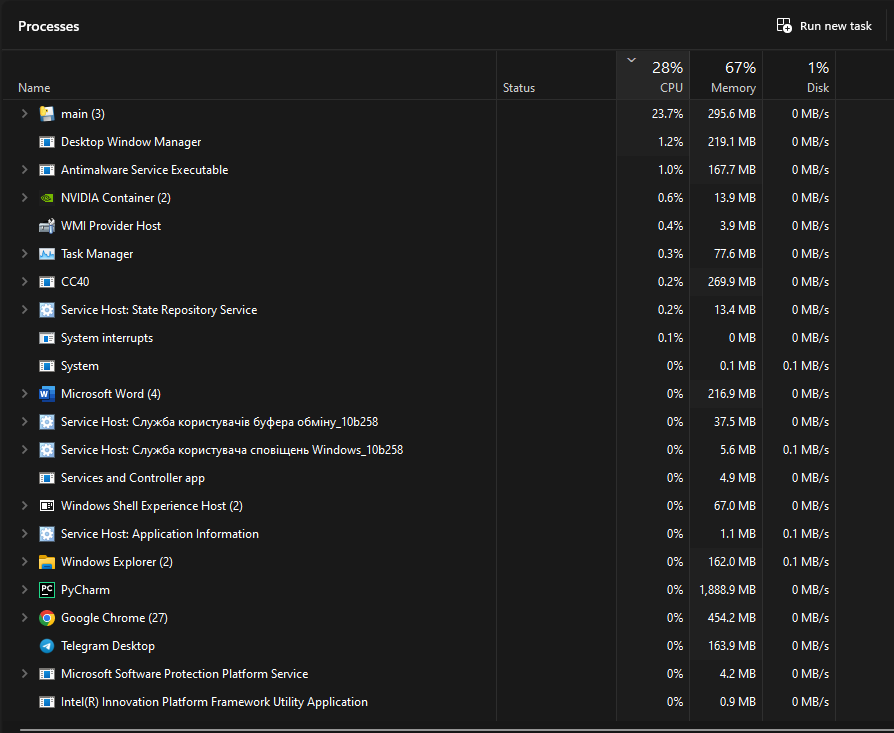


Рис х.7 – Активні процеси у «Диспетчері задач»

* 1. Інформація про мережу

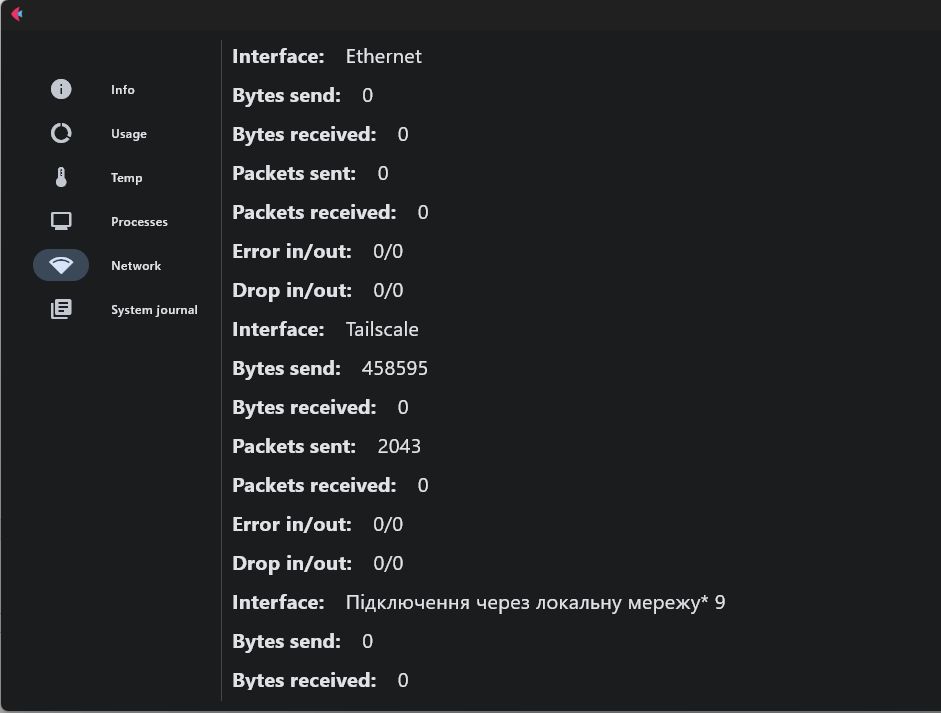


Рис х.8 – Інформація про мережу у робочому проєкті

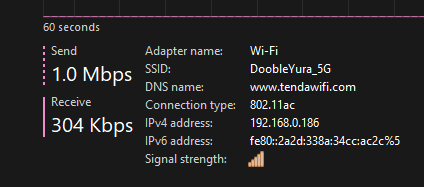


Рис х.9 – Інформація про мережу в «Диспетчері задач»

* 1. Системні журнали

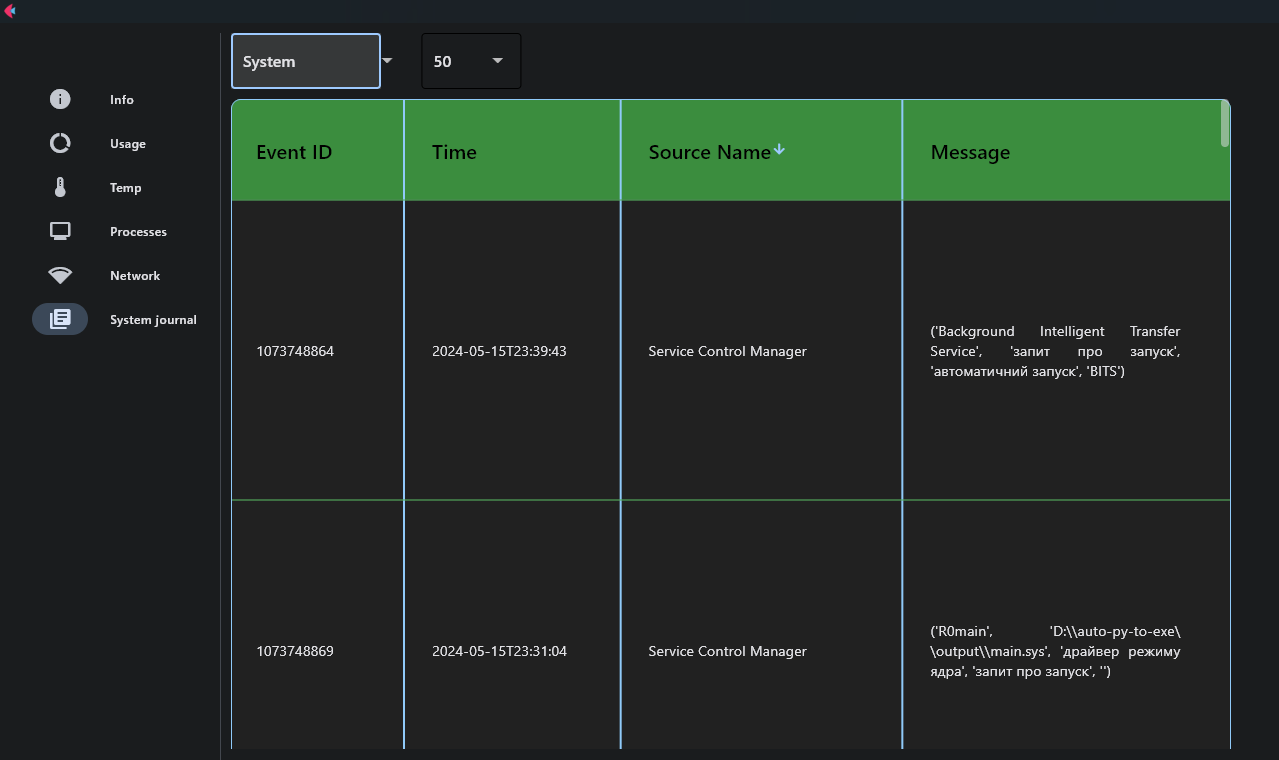


Рис х.10 – Інформація про системні записи у робочому проєкті

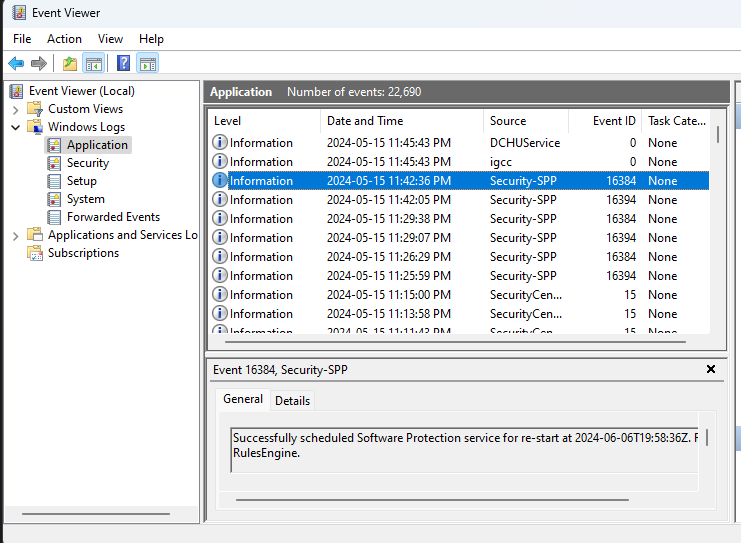


Рис х.11 – Інформація про системні журнали у Event Viewer

Отже провівши порівняння значень з впевненістю можна сказати що дана програма показує актуальні значення та записи в комп’ютері, що дозволяє зручно отримати інформацію про систему.

# Висновки

В цьому курсовому проекті було розроблено програму для моніторингу системи через консоль та графічний інтерфейс на основі мови програмування Python. Програма надає користувачеві можливість отримати різноманітну інформацію про систему, таку як використання пам'яті, температура процесора, інформація про процеси, мережевий трафік та журнали подій Windows.

У результаті розробки програми було досягнуто наступних цілей:

1. Реалізовано функціонал для моніторингу та відстеження використання системних ресурсів.

2. Забезпечено доступ до інформації про процеси, які запущені на комп'ютері, включаючи їх використання пам'яті та CPU.

3. Надано можливість аналізу журналів подій Windows для виявлення та вирішення проблем в системі.

4. Реалізовано зручний інтерфейс користувача для отримання необхідної інформації.

Отже, цей проект є важливим інструментом для моніторингу та аналізу роботи комп'ютерних систем, що може бути корисним як для звичайних користувачів, так і для адміністраторів мережі та системних адміністраторів. Додатково, програма має потенціал для розширення функціональності та вдосконалення шляхом додавання нових можливостей та оптимізації інтерфейсу користувача.

# список використаних джерел

# додатки