**Форма № 17**

Міністерство освіти і науки України

**Національний університет «Запорізька політехніка»**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Кафедра програмних засобів\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(найменування кафедри)

**КУРСОВИЙ ПРОЄКТ**

**(РОБОТА)**

з дисципліни «Операційні системи»

(назва дисципліни)

на тему: «Застосунок аналізу системних ресурсів»

Студента 3 курсу КНТ-122 групи

спеціальності 121 Інженерія   
програмного забезпечення

освітня програма (спеціалізація) інженерія програмного забезпечення

\_\_\_\_\_\_\_\_\_\_\_\_Онишенко О. А.\_\_\_\_\_\_\_

(прізвище та ініціали)

(прізвище та ініціали)

Керівник Степаненко О. О.

(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Національна шкала \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кількість балів: \_\_\_\_\_\_\_\_\_\_Оцінка: ECTS \_\_\_\_\_

Члени комісії \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_   Степаненко О. О.

(підпис) (прізвище та ініціали)

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

(підпис) (прізвище та ініціали)

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

(підпис) (прізвище та ініціали

2024 рік

**Форма № 25**

Міністерство освіти і науки України

**Національний університет «Запорізька політехніка»**

( повне найменування закладу вищої освіти)

Інститут, факультет ІІРЕ, ФКНТ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кафедра програмних засобів\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ступінь вищої освіти      бакалавр**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Спеціальність 121 Інженерія програмного забезпечення**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(код і найменування)

Освітня програма (спеціалізація) Інженерія програмного забезпечення**\_\_\_\_\_\_\_**

(назва освітньої програми (спеціалізації)

**ЗАТВЕРДЖУЮ**

**Завідувач кафедри ПЗ, д.т.н, проф.**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_С.О. Субботін**

“\_\_\_\_” \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_року

З А В Д А Н Н Я

**НА КУРСОВИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА**

  Онишенко О. А

(прізвище, ім’я, по батькові)

1. Тема проєкту (роботи) Застосунок аналізу системних ресурсів

керівник проєкту (роботи)\_Степаненко Олександр Олексійович ,

( прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від

2. Строк подання студентом проєкту (роботи)\_03 грудня 2024 року\_\_\_\_\_\_\_\_\_\_

3. Вихідні дані до проєкту (роботи) створити застосунок аналізу системних ресурсів

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області; 2. Аналіз програмних засобів; 3. Основні рішення з реалізації компонентів системи; 4. Керівництво програміста; 5. Керівництво користувача; 6. Додатки.  
5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)

6. Консультанти розділів проєкту (роботи)

|  |  |  |  |
| --- | --- | --- | --- |
| Розділ | Прізвище, ініціали та посада  консультанта | Підпис, дата | |
| завдання видав | прийняв  виконане завдання |
| 1-5 Основна частина | Степаненко О. О. |  |  |
|  |  |  |  |
|  |  |  |  |

7. Дата видачі завдання\_\_16 жовтня 2024 р.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**КАЛЕНДАРНИЙ ПЛАН**

|  |  |  |  |
| --- | --- | --- | --- |
| №  з/п | Назва етапів курсового  проєкту (роботи) | Строк виконання етапів проєкту  ( роботи ) | Примітка |
| 1. | Аналіз індивідуального завдання. | 1 тиждень |  |
| 2. | Аналіз програмних засобів, що будуть використовуватись в роботі. | 2 тиждень |  |
| 3. | Аналіз структур даних, що необхідно використати в курсової роботі. | 3 тиждень |  |
| 4. | Затвердження завдання | 4 тиждень |  |
| 5. | Вивчення можливостей програмної реалізації структур даних та інтерфейсу користувача. | 5-9 тиждень |  |
| 6. | Аналіз вимог до апаратних засобів | 9 тиждень |  |
| 7. | Розробка програмного забезпечення | 9-13 тиждень |  |
| 8. | Проміжний контроль | 10 тиждень | Розділи 1-2 ПЗ |
| 9. | Оформлення, відповідних пунктів пояснювальної записки. | 10-14 тиждень | Розділи 1-5 ПЗ |
| 10. | Захист курсової роботи. | 15 тиждень |  |
|  |  |  |  |
|  |  |  |  |

**Студент \_\_\_\_\_\_\_\_\_ \_\_\_** Онищенко О. А.**\_\_\_\_**

( підпис ) (прізвище та ініціали)

**Керівник проєкту (роботи) \_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_** Степаненко О. О.**\_\_\_\_**

( підпис ) (прізвище та ініціали)

реферат

Проєкт «Застосунок аналізу системних ресурсів» є комплексним застосунком, розробленим з використанням мови програмування Python. Застосунок призначений для полегшення процесу отримання актуальних та оновлювальних даних про поточні ресурси системи користувача у цифровому вигляді. Це зручна платформа, яка дозволяє користувачам переглядати інформацію про систему, її поточні характеристики та властивосі, а також показники роботи системи на конкретний момент часу. Застосунок також дозволяє отримати візуальне представлення зміни показників системи, що може полегшити сприйняття користувачем відповідних параметрів системи при аналізі системних ресурсів. Проєкт має на меті спростити процес витягання та взаємодії із ресурсами системи користувача за допомогою візуальних елементів інтерфейсу.

Система, ресурси системи, Python, курсова робота, курсовий проєкт, Windows застосунок, застосунок аналізу системних ресурсів.

зміст

[реферат 4](#_Toc184389310)

[зміст 5](#_Toc184389311)

[перелік умовних позначень, символів, одиниць, скорочень і термінів 6](#_Toc184389312)

[вступ 7](#_Toc184389313)

[1 Огляд і аналіз тематики 8](#_Toc184389314)

[1.1 Аналіз витягання системних ресурсів, як основи предметної області 8](#_Toc184389315)

[1.2 Огляд існуючих методів вирішення завдання 8](#_Toc184389316)

[1.2.1 Передмова 8](#_Toc184389317)

[1.2.2 Система «Cpu-Z» 8](#_Toc184389318)

[1.2.2 Система «Open Hardware Monitor» 11](#_Toc184389319)

[1.3 Висновки з першого розділу та постановка задачі 14](#_Toc184389320)

[2 Аналіз технічного завдання 15](#_Toc184389321)

[3 Вибір та обґрунтування структури системи 16](#_Toc184389322)

[4 Програмна реалізація 18](#_Toc184389323)

[5 Керівництво оператора 19](#_Toc184389324)

[6 Керівництво програміста 20](#_Toc184389325)

[Висновки 21](#_Toc184389326)

[ПЕРЕЛІК ПОСИЛАНЬ 22](#_Toc184389327)

[Додаток А – Технічне завдання 23](#_Toc184389328)

[Додаток Б – Текст програми 29](#_Toc184389329)

[А1 – run.py 29](#_Toc184389330)

[Додаток В – Методика та релузльтати тестування програми 34](#_Toc184389331)

перелік умовних позначень, символів, одиниць, скорочень і термінів

ОС – операційна система,

Python – мова програмування

NiceGUI – бібліотека Python для розробки графічних інтерфейсів

psutil – бібліотека Python для доступу до ресурсів системи

pywin32 або win32api – бібліотека Python для доступу до ресурсів системи

вступ

Застосунок аналізу системних ресурсів є цифровою платформою надання коректних поточних даних про характеристики користувацької системи та поточних показників конкретних компонентів. Оскільки наявність комп’ютера є корисною можливістю, аналіз ресурсів системи для системних адміністраторів або користувачів, які дбають про систему самостійно може бути корисним інструментом. Такий застосунок може полегшити отримання та опрацювання інформації про поточні характеристики системи та її показники на конкретний момент часу.

1 Огляд і аналіз тематики

Системні ресурси є частиною системи. Вони відображають характеристики системи, її конкретних компонентів, а також надають інформацію про поточні показники компонентів системи.

1.1 Аналіз витягання системних ресурсів, як основи предметної області

Витягання системних ресурсів є основною підзадачею аналізу ресурсів системи користувача. Для витягання ресурсів із системи розробники операційних систем зазвичай надають системні функції та методи для роботи з компонентами системи. Оскільки застосунок розробляється під платформу Windows, програма використовує системні функції та методи, зокрема WMI, для доступу до ресурсів системи, для витягання інформації про компоненти та аналіз їх поточних характеристик.

1.2 Огляд існуючих методів вирішення завдання

1.2.1 Передмова

Оскільки задача аналізу системних ресурсів є необхідною для користувачів які обслуговують системи власноруч, розробники з різних країн зробили подібні системи для витягання та аналізу ресурсів користувацьких систем.

1.2.2 Система «Cpu-Z»

CPU-Z є безкоштовною програмою для Windows, яка збирає інформацію про основні компоненти системи: процесор, материнська плата, пам'ять. CPU-Z повністю підтримується на Windows 11.

Переваги:

* Детальна та точна інформація.

Недоліки:

* Відсутність графічних елементів.

Робота програми наведена нижче.

Зображення, що містить текст, знімок екрана, програмне забезпечення, Комп’ютерна піктограма

Автоматично згенерований опис

Рисунок 1.1 – Робота програми «Cpu-Z»

Зображення, що містить текст, знімок екрана, число, Шрифт

Автоматично згенерований опис

Рисунок 1.2 – Робота програми «Cpu-Z»

Зображення, що містить текст, знімок екрана, монітор, число

Автоматично згенерований опис

Рисунок 1.3 – Робота програми «Cpu-Z»

1.2.2 Система «Open Hardware Monitor»

Open Hardware Monitor є безкоштовним програмним забезпеченням з відкритим вихідним кодом, яке відстежує температурні датчики, швидкість обертання вентиляторів, напругу, навантаження і тактову частоту комп'ютера.

Переваги:

* Детальна інформація;
* Дані чітко структуровано;
* Наявність графічних елементів.

Робота програми наведена нижче.

Зображення, що містить текст, знімок екрана, програмне забезпечення, число

Автоматично згенерований опис

Рисунок 1.4 – Робота програми «Open Hardware Monitor»

Зображення, що містить текст, знімок екрана, монітор, число

Автоматично згенерований опис

Рисунок 1.5 – Робота програми «Open Hardware Monitor»

Зображення, що містить текст, знімок екрана, монітор, програмне забезпечення

Автоматично згенерований опис

Рисунок 1.6 – Робота програми «Open Hardware Monitor»

1.3 Висновки з першого розділу та постановка задачі

Існуючі програмні засоби наводять детальну актуальну та точну інформацію про систему користувача, але не всі мають графічні елементи.

2 Аналіз технічного завдання

аіваі

3 Вибір та обґрунтування структури системи

Амінь Теоретична модель програми тут

Система має бути виконана у веб-браузері. Сторінка має бути умовно розділена на такі теоретичні секції:

* Системні дані;
* Інформація про процесор;
* Інформація про пам’ять;
* Інформація про накопичувачі;
* Інформація про мережу.

У розділі Системні дані користувач має бачити основну інформація про систему, а саме:

* Тип системи;
* Ім’я авторизованого користувача;
* Номер випуску системи;
* Повна версія системи;
* Тип процесора;
* Дата та час запуску системи.

Розділ інформації про процесор має розділятися на дві умовні секції:

* Теоретична інформація;
* Показники характеристик на поточний момент часу.

Усі теоретичні дані системи користувача мають бути виведенні з використанням таблиць даних, в яких перша колонка міститиме назву характеристики, а друга міститиме значення характеристики.

Теоретична інформація про процесор користувача має включати такі дані:

* Повна назва процесора;
* Назва платформи процесора;
* Кількість ядер процесора;
* Кількість потоків процесора.

Характеристики процесор мають бути розділені на дві оновлювальні підсекції:

* Інформація про частоту процесора
* Інформація про використовуваність процесора

Інформація про частоту процесора має бути представлена у вигляді таблиці та включати такі характеристики:

* Мінімальна частота
* Максимальна частота
* Поточна частота

Такий набір характеристик зумовлен тим, що частота роботи процесора може змінюватися в ході роботи системи. Для відображення таких змін програма має передбачати виведення як максимальної, так і мінімальної частот для полегшення процесу відлагодження системи в разі наявності несправностей.

Підсекція інформації про частоту процесора також має містити графічний елемент – кругову діаграму, що показуватиме поточну частоту процесора порівняно з мінімальною та максимальною. Така кругова діаграма має мати початкове значення як мінімальна частота процесора, поточне значення як поточна частота процесора, і максимальне значення як максимальна тактова частота процесора у Мегагерцах.

Інформація про використовуваність (Англійською usage або застосування) усіх потоків процесора має бути наведена у вигляді таблиці та включати таку інформацію:

* Для кожного потоку процесора процент використовуваності

Таблиця інформації про застосовуваність процесора має бути динамічно оновлюваною і таблиця має включати дані про всі доступні потоки процесора користувача. Дані таблиці також мають бути оновлюваними щосекунди, що є інтервалом оновлення інтерфейсу користувача.

4 Програмна реалізація

Вибір засобів та компоновка проєкту

5 Керівництво оператора

Інструкція з використання

6 Керівництво програміста

Інструкція з переробки чи модернізації

Таблиці з назвами функцій та аргументами

Висновки

Під час виконання курсового проєкту благодаттю Господа нашого Ісуса Христа було розроблено проєкт аналізу системних ресурсів мовою програмування Python із використанням зовнішніх пакетів NiceGUI, psutil, та win32api. Програма має візуальний графічний інтерфейс, оновлює дані щосекунди, має графічні елементи відображення змін параметрів системи та детальний опис інформації системи користувача.

ПЕРЕЛІК ПОСИЛАНЬ

1) How to Get Hardware and System Information in Python [Електронний ресурс]. – Режим доступу: <https://thepythoncode.com/article/get-hardware-system-information-python>

2) psutil documentation [Електронний ресурс]. – Режим доступу: [https://psutil.readthedocs.io/en/latest/?badge=latest#](https://psutil.readthedocs.io/en/latest/?badge=latest)

3) NiceGUI Documentation [Електронний ресурс]. – Режим доступу: <https://nicegui.io/documentation>

4) How to Make a Network Usage Monitor in Python [Електронний ресурс]. – Режим доступу: <https://thepythoncode.com/article/make-a-network-usage-monitor-in-python>

5) Is it possible to get in Python the CPU/core/processor id that the python program itself is using? - Quora [Електронний ресурс]. – Режим доступу: <https://www.quora.com/Is-it-possible-to-get-in-Python-the-CPU-core-processor-id-that-the-python-program-itself-is-using>

6) Python get cpu info [Електронний ресурс]. – Режим доступу: <https://www.programcreek.com/python/?CodeExample=get%20cpu%20info>

7) How to use the psutil.net\_if\_addrs function in psutil | Snyk [Електронний ресурс]. – Режим доступу: <https://snyk.io/advisor/python/psutil/functions/psutil.net_if_addrs>

Додаток А – Технічне завдання

**РОЗРОБКА ПРОГРАМИ, ЯКА РЕАЛІЗУЄ ОСНОВНІ ЗАДАЧІ АНАЛІЗУ СИСТЕМНИХ РЕСУРСІВ ПЕОМ**

**Мета розробки та її призначення**

Метою розробки програми є отримання необхідної системною інформації за запитом користувача. Особливістю розроблюваної програми є створення десктопного застосунку для швидкого отримання точної інформації про поточний статус системи та про роботу її окремих пристроїв.

**Змісти теоретичних посилок до розробки**

Системні ресурси описують доступні можливості Персональної електронно-обчислювальної машини (ПЕОМ). Для їх аналізу операційні системи дають доступ до функцій, через які можна отримати інформацію про поточний статус пристроїв.

Пристрої системи зазвичай включають дискові накопичувачі, процесори, оперативну пам’ять, графічні процесори, пристрої охолодження. Кожен з цих пристроїв має свої характеристики, інформація про які може бути потрібна користувачеві. Для таких ситуацій розроблюється програма цього проєкту. Вона дозволятиме користувачеві побачити інформацію про пристрої системи та їх поточні характеристики. Наприклад, програма має надавати доступ до назви процесора машини, назви графічного процесора, обсягу оперативної пам’яті, поточної швидкості роботи процесора, і так далі.

Проєкт розроблюється під операційну систему Windows, яка надає доступ до необхідної інформації через системний регістр, засоби Windows Management Instrumentation (WMI) або через засоби Windows API.

Кожен пристрій системи може мати різні характеристики. Наприклад, оперативна пам’ять має тактову частоту, виміряну у мегагерцах; процесор має кількість ядер, кількість потоків, поточну швидкість у мегагерцах; дисковий накопичувач має температуру у цельсіях або фаренгейтах, загальний простір виміряний і гігабайтах та доступний простір виміряний у гігабайтах; пристрій охолодження має поточну швидкість обертів виміряну у обертах за хвилину (revolutions per minute або rpm). Всі ці характеристики можуть змінюватися з часом роботи програми, тому маємо це враховувати.

**Основні вимоги до програми**

Програма має працювати з операційною системою Windows, мати графічний інтерфейс користувача та використовувати системні методи для отримання необхідної інформацію про пристрої.

Як початкові дані програма має використовувати отриманні за запитом до системи дані про ресурси системи, а саме інформація про процесор, графічний процесор, оперативну пам’ять, дискові накопичувачі та пристрої охолодження. Для кожного пристрою програма має виводити його поточні характеристики.

Як вихідну інформація програма має виводити у графічний інтерфейс користувача всі зазначені характеристики для кожного пристрою у поточний момент часу:

1. Процесор

a. Назва

b. Кількість ядер

c. Кількість потоків

d. Тактова частота

e. Температура

f. Відсоток завантаженності

2. Графічний процесор

a. Назва

b. Тактова частота

c. Температура

d. Швидкість обертів вентиляторів

e. Обсяг оперативної пам’яті

3. Оперативна пам’ять

a. Загальний обсяг

b. Доступний обсяг

c. Швидкість

4. Дисковий накопичувач

a. Температура

b. Загальний обсяг

c. Доступний обсяг

5. Пристрій охолодження

a. Швидкість обертів

Ці дані програма має виводити у графічному вигляді засобами графічного інтерфейсу та оновлювати їх періодично для отримання точного статусу системи на кожен момент часу.

Графічний інтерфейс має складатися з елементів подання інформації у вигляді дерева або таблиці. Для кожного компоненту має бути виділено окреме місце інтерфейсу та розписані його поточні характеристики. Ці характеристики можна зазначити або як окремі таблиці у інтерфейсі, або у вигляді дерева де корінь це назва системи користувача.

**Алгоритм функціонування програми**

При запуску програма має зчитувати всю необхідну інформацію з системи. Це можна зробити вбудованими засобами системи Windows. Варіанти витягання необхідної інформації про систему наступні:

- Використати засоби WMI. Цей варіант є найбільш простим бо надає необхідну інформацію за доступом до відповідних змінних.

- Використання засобів Windows API. Цей варіант найскладніший бо його підтримка обраними засобами розробки обмежена.

- Доступ до системного регістру. Цей варіант є середнім за складністю і також надає інформацію за доступом до змінних.

При реалізації програми може виникнути потреба використанні комбінації усіх цих засобів аби отримати повну інформацію. Також при розробці може виникнути ситуація коли треба буде додавати додаткові спосби витяганні інформації якщо вже зазначені не надаватимуть якихось даних.

Після отримання всієї необхідної інформації про кожну з компонент системи програма має вивести її користувачеві у доступному та зрозумілому вигляді. Це має здійснюватись шляхом використання графічного інтерфейсу. Користувач має чітко бачити окремі компоненти та їх характеристики у кожен момент часу.

При використанні програма має додатково запитувати необхідну інформацію періодично та оновлювати інтерфейс з новиим даними. Така функція необхідна для отримання актуальної поточної інформації про систему. Це потрібно через те що використання системи може змінювати її характеристики. Наприклад навантаження на процесор може змінюватись при використанні різних програм, які потребують більше ресурсів; або обсяг доступної оперативної пам’яті може змінюватись при використанні програм які потребують більше доступної оперативної пам’яті для роботи.

**Обгрунтування вибору апаратно-технічних засобів, операційної системи та мови програмування**

Операційна система вибору для поточного проєкту є Windows 10 або 11. Обрана саме ця операційна система через використання її на машині розробки. Обрання іншої операційної системи потребуватиме використання засобів емуляторів або встановлення та використання іншої системи для розробки програмного забезпечення. Жоден з цих варіантів не є бажаним вибором.

Програма має розроблюватися інструментами розробки Python або C#. Ці засоби розробки надають доступ до усіх необхідних функцій системи, мають широку розповсюдженість серед розробників, надають багато доступних ресурсів у системі інтернет для використання та витягування потрібної інформації в процесі розробки. Також ці мови прогармування містять необхідні можливості створення графічних інтерфейсів користувача. Для мови програмування C# засобами розробки графічного інтерфейсу можуть бути Windows Forms або WPF. В разі використанні мови Python також доступні різні бібліотеки для написання графічного інтерфесу, як от PyQT або tkinter або NiceGUI.

Скоріш за все для проєкту буде використана мова програмування Python, а разом з нею засоби розробки інтерфейсу NiceGUI. Використання Python для розробки обґрунтоване її знайомістю для розробника та доступністю джерел інформації і бібліотек для витягання необхідної інформації про систему. Мова програмування Python містить доступний та простий синтаксис що має спростити процес розробки та відповідно скоротити його загальний час.

Використання мови Python дозволяє застосувати методи розробки Microsoft Visual Studio Code які мають всі необхідні засоби та методи для швидкого та просто процесу розробки проєкту.

**Точно визначені системні функції програми**

Конкретні функції програми можуть змінюватись в процесі розробки програм. Найбільш необхідні функції програми такі:

- Витяганні інформації про систему. Ця функція має запитувати засоби системи про поточні характеристики кожного з компонентів. Така функція може бути реалізована багатьма окремими функціями або класами та модулями програми. Конкретний алгоритм роботи функції залежить від обрання даних які необхідно витягнути. При завершенні роботи ця функція має повертати всі необхідні дані у вигляді об’єкту JSON або словнику значень у вигляді ключ: значення.

- Виведення необхідної інформації. Ця функція має приймати як аргумент усі витягнені дані з попередньої функції та виводити їх на екран користувача. Виведення даних має здійснюватись через оголошення та налаштування відповідних засобів інтерфейсу користувача. Ці засоби мають бути використані з наявних методів обраної бібліотеки проєктування графічного інтерфейсу. Розробка цією функції також може передбачати використання багатьох класів та модулів коду для кращої модульності та незалежності кожної з компонентів системи.

- Оновлення даних. Ця функція має комбінувати роботу двох попередніх функцій. Такий підхід необхідний для забезпечення надання точної інформації про систему під час запиту а також під час використання програми. Ця функція може використати попередньо розроблені методи та функції першої функції а також методи другої функції для оновлення інтерфейсу користувача.

Комбінація цих трьох функцій має забезпечити коректну та точну роботу програми завдяки якій користувач може отримати необхідну поточну інформація про статус системи з деталями про кожен компонент, його роботу та його характеристики у необхідний момент часу.

**Обмеження на установку та використання**

Встановлення програмного застосунку не має займати багато часу оскільки застосунок має бути виконуваним у веб-браузері на системі користувача. Для запуску застосунку і його роботи на сторінці необхідно провести встановлення застосунку на сервер. На сервері потрібно виконати встановлення використаних програмних пакетів які застосунок використовує. Назви пакетів та їх версії мають бути прописані у окремому текстовому файлі з відповідною назвою. Після встановлення пакетів та запуску застосунку на сервері користувачі мають мати змогу використовувати застосунок через введення його адреси у пошуковий рядок системного браузера.

Використання застосунку має проходити із використанням простого світлого графічного інтерфейсу користувача на сторінці веб-браузера. Взаємодія з компонентами має бути виконана через графічні візуальні елементи сторінки користувача. Елементи також мають бути оновлені з певним інтервалом для забезпечення акутальності інформації на веб-сторінці користувача.

**ВИСНОВКИ**

По закінченню роботи було отримано документ що містить усю необхідну інформацію про проєкт та його можливі майбутні характеристики. Наявність такого документу технічного завдання може дозволити перейти до розробки програмного забезпечення із зазначеними вимогами.

Додаток Б – Текст програми

А1 – run.py

from collections import defaultdict

from datetime import datetime

from nicegui import ui

import win32api

import platform

import psutil

HEADING\_CLASSES='font-bold text-xl'

COLUMNS=[

{'name':'property','label':'Property','field':'property','align':'left'},

{'name':'value','label':'Value','field':'value','sortable':True},

]

uname=platform.uname()

boot\_time\_stamp=psutil.boot\_time()

boot\_time=datetime.fromtimestamp(boot\_time\_stamp)

cpu\_frequency=psutil.cpu\_freq()

system\_virtual\_memory=psutil.virtual\_memory()

swap=psutil.swap\_memory()

partitions=psutil.disk\_partitions()

disk\_io=psutil.disk\_io\_counters()

if\_addrs=psutil.net\_if\_addrs()

net\_io=psutil.net\_io\_counters()

sent,recv=net\_io.bytes\_sent,net\_io.bytes\_recv

def get\_rows(data:dict):

return [{'property':k.capitalize(),'value':v} for k,v in data.items()]

def update\_ui():

processor\_frequencies\_data={

'min':cpu\_frequency.min,

'max':cpu\_frequency.max,

'current':cpu\_frequency.current,

}

processor\_frequencies\_table.rows=get\_rows(processor\_frequencies\_data)

cpu\_usage=psutil.cpu\_percent(percpu=True)

processor\_usage\_data={

f'Core {i}': usage

for i,usage in enumerate(cpu\_usage)

}

processor\_usage\_table.rows=get\_rows(processor\_usage\_data)

cpu\_usage=psutil.cpu\_percent()

this\_time=datetime.now().timestamp()

processor\_usage\_plot.push([this\_time],[[cpu\_usage]])

system\_virtual\_memory=psutil.virtual\_memory()

virtual\_memory\_data={

'total':get\_formatted\_size(system\_virtual\_memory.total),

'available':get\_formatted\_size(system\_virtual\_memory.available),

'used':get\_formatted\_size(system\_virtual\_memory.used),

'percentage':f'{system\_virtual\_memory.percent}%',

}

virtual\_memory\_table.rows=get\_rows(virtual\_memory\_data)

virtual\_memory\_circle.value=system\_virtual\_memory.percent

memory\_usage\_plot.push([this\_time],[[system\_virtual\_memory.percent]])

swap=psutil.swap\_memory()

swap\_memory\_data={

'total':get\_formatted\_size(swap.total),

'free':get\_formatted\_size(swap.free),

'used':get\_formatted\_size(swap.used),

'percentage':f'{swap.percent}%',

}

swap\_memory\_table.rows=get\_rows(swap\_memory\_data)

swap\_memory\_circle.value=swap.percent

global sent,recv

new\_network\_io=psutil.net\_io\_counters()

us,ds=new\_network\_io.bytes\_sent-sent,new\_network\_io.bytes\_recv-recv

network\_speed\_plot.push([this\_time],[[ds/1],[us/1]])

sent,recv=new\_network\_io.bytes\_sent,new\_network\_io.bytes\_recv

def get\_formatted\_size(bytes,suffix='B'):

factor=1024

for unit in ['','K','M','G','T','P']:

if bytes<factor: return f'{bytes:.2f} {unit}{suffix}'

bytes/=factor

with ui.row().classes('flex gap-3'):

with ui.column():

system\_data={

'type':uname.system,

'user':uname.node,

'release':uname.release,

'version':uname.version,

'machine':uname.machine,

'booted':f'{boot\_time.day}.{boot\_time.month}.{boot\_time.year} {boot\_time.hour:02d}:{boot\_time.minute:02d}:{boot\_time.second:02d}'

}

system\_table=ui.table(columns=COLUMNS,rows=get\_rows(system\_data),row\_key='name',title='System')

with ui.column():

processor\_data={

'name':uname.processor,

'platform':uname.machine,

'cores':psutil.cpu\_count(logical=False),

'threads':psutil.cpu\_count(logical=True),

}

processor\_table=ui.table(columns=COLUMNS,rows=get\_rows(processor\_data),row\_key='name',title='Processor')

with ui.row().classes('flex w-full'):

with ui.column():

processor\_frequencies\_data={

'min':cpu\_frequency.min,

'max':cpu\_frequency.max,

'current':cpu\_frequency.current,

}

processor\_frequencies\_table=ui.table(columns=COLUMNS,rows=get\_rows(processor\_frequencies\_data),row\_key='name',title='Frequencies (MHz)')

ui.label('Processor Frequency')

processor\_frequencies\_circle=ui.circular\_progress(min=cpu\_frequency.min,max=cpu\_frequency.max,value=cpu\_frequency.current)

processor\_usage\_data={

f'Core {i}': usage

for i,usage in enumerate(psutil.cpu\_percent(percpu=True))

}

processor\_usage\_table=ui.table(columns=COLUMNS,rows=get\_rows(processor\_usage\_data),row\_key='name',title='Usage (%)').classes('flex-1')

with ui.column():

virtual\_memory\_data={

'total':get\_formatted\_size(system\_virtual\_memory.total),

'available':get\_formatted\_size(system\_virtual\_memory.available),

'used':get\_formatted\_size(system\_virtual\_memory.used),

'percentage':f'{system\_virtual\_memory.percent}%',

}

virtual\_memory\_table=ui.table(columns=COLUMNS,rows=get\_rows(virtual\_memory\_data),row\_key='name',title='Virtual Memory').classes('w-full')

ui.label('Virtual Memory Usage')

virtual\_memory\_circle=ui.circular\_progress(value=system\_virtual\_memory.percent,max=100)

swap\_memory\_data={

'total':get\_formatted\_size(swap.total),

'free':get\_formatted\_size(swap.free),

'used':get\_formatted\_size(swap.used),

'percentage':f'{swap.percent}%',

}

swap\_memory\_table=ui.table(columns=COLUMNS,rows=get\_rows(swap\_memory\_data),row\_key='name',title='Swap Memory')

ui.label('Swap Memory Usage')

swap\_memory\_circle=ui.circular\_progress(value=swap.percent,max=100)

with ui.column():

with ui.card():

ui.label('Disks').classes('q-table\_\_title')

disk\_tables=defaultdict(dict)

for partition in partitions:

try: usage\_data=psutil.disk\_usage(partition.mountpoint)

except: continue

partition\_name=win32api.GetVolumeInformation(partition.device)[0]

disk\_data={

'device':partition.device,

'name':partition\_name,

'file system':partition.fstype,

}

space\_data={

'total':get\_formatted\_size(usage\_data.total),

'used':get\_formatted\_size(usage\_data.used),

'free':get\_formatted\_size(usage\_data.free),

'percentage':f'{usage\_data.percent}%',

}

with ui.expansion(partition\_name):

disk\_tables[partition\_name]['disk']=ui.table(columns=COLUMNS,rows=get\_rows(disk\_data),row\_key='name',title=f'{partition\_name} Data')

disk\_tables[partition\_name]['space']=ui.table(columns=COLUMNS,rows=get\_rows(space\_data),row\_key='name',title=f'{partition\_name} Space')

ui.label(f'{partition\_name} Usage')

ui.circular\_progress(value=usage\_data.percent,max=100,min=0)

disks\_data={

'read':get\_formatted\_size(disk\_io.read\_bytes),

'write':get\_formatted\_size(disk\_io.write\_bytes),

}

disks\_table=ui.table(columns=COLUMNS,rows=get\_rows(disks\_data),row\_key='name').classes('w-full')

with ui.column():

with ui.card():

ui.label('Network').classes('q-table\_\_title')

network\_tables=defaultdict(str)

for interface\_name,interface\_addresses in if\_addrs.items():

interface\_addresses=[a for a in interface\_addresses if a.family.name=='AF\_INET' or a.family.name=='AF\_PACKET']

for address in interface\_addresses:

network\_data={

'IP Address' if address.family.name=='AF\_INET' else 'MAC Address':address.address,

'netmask':address.netmask,

'Broadcast IP' if address.family.name=='AF\_INET' else 'Broadcast MAC':address.broadcast,

}

with ui.expansion(interface\_name):

network\_tables[interface\_name]=ui.table(columns=COLUMNS,rows=get\_rows(network\_data),row\_key='name',title=f'{interface\_name} Data')

network\_data={

'sent':get\_formatted\_size(net\_io.bytes\_sent),

'received':get\_formatted\_size(net\_io.bytes\_recv),

}

network\_table=ui.table(columns=COLUMNS,rows=get\_rows(network\_data),row\_key='name').classes('w-full')

with ui.row():

processor\_usage\_plot=ui.line\_plot(n=1,figsize=(4.7,2.47)).with\_legend(['CPU Usage %'],loc='upper center',ncol=1)

processor\_usage\_plot.push([datetime.now().timestamp()],[[0]])

processor\_usage\_plot.push([datetime.now().timestamp()],[[100]])

memory\_usage\_plot=ui.line\_plot(n=1,figsize=(4.7,2.47)).with\_legend(['RAM Usage %'],loc='upper center',ncol=1)

memory\_usage\_plot.push([datetime.now().timestamp()],[[0]])

memory\_usage\_plot.push([datetime.now().timestamp()],[[100]])

network\_speed\_plot=ui.line\_plot(n=2,figsize=(4.7,2.47)).with\_legend(['Download Speed','Upload Speed'],loc='upper center',ncol=2)

network\_speed\_plot.push([datetime.now().timestamp()],[[0],[0]])

network\_speed\_plot.push([datetime.now().timestamp()],[[100],[100]])

ui.timer(1,update\_ui,active=True)

ui.run(title='System Resources Analysis',favicon='💻')

Додаток В – Методика та релузльтати тестування програми

Блок схема або структурна схема програми