

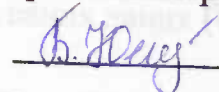
Міністерство освіти і науки України
Державний університет «Житомирська політехніка»
Факультет інформаційно-комп'ютерних технологій
Кафедра комп'ютерної інженерії та кібербезпеки

ПОЯСНЮВАЛЬНА ЗАПИСКА

до випускної кваліфікаційної роботи бакалавра
на тему: «Проект системи автоматизованого керування
середовищем віртуальних машин»

Виконав: студент 4-го курсу групи КІ-22-1

спеціальності 123 «Комп'ютерна інженерія»

 Юрій БАХУР

Керівник: доцент кафедри КІ та КБ, к.пед.н., доцент

 Олена ГОЛОВНЯ

Рецензент: доцент кафедри КІ та КБ, к.т.н.

 Олексій ШЕЛУХА

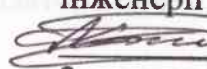
Житомир – 2026

Державний університет «Житомирська політехніка»
Факультет інформаційно-комп'ютерних технологій
Кафедра комп'ютерної інженерії та кібербезпеки
Спеціальність 123 «Комп'ютерна інженерія»
Освітня програма «Комп'ютерна інженерія»

ЗАТВЕРДЖУЮ

Завідувач

кафедри комп'ютерної
інженерії та кібербезпеки

 Андрій Єфіменко

2 лютого 2026 р.

ЗАВДАННЯ

на випускню кваліфікаційну роботу бакалавра

Студент: Бахур Юрій Вадимович

Тема роботи: «Проект системи автоматизованого керування середовищем віртуальних машин»

затверджена Наказом університету від 2 лютого 2026 р. №57/с

Термін здачі студентом закінченої роботи 19 червня 2026 р.

Вихідні дані роботи: архітектура утиліти командного рядка VBoxManage, системні API Операційної Системи Windows (планувальник завдань Windows Task Scheduler, механізми контролю облікових записів UAC), документація до мови сценаріїв PowerShell та специфікації фреймворку FastAPI для мови програмування Python; вимоги до крос-платформного кодування даних (UTF-8) при міжпроцесній взаємодії.

Об'єкт дослідження: процес адміністрування, забезпечення відмовостійкості, конфігурування та супроводження віртуальних інфраструктур на базі Hosted-гіпервізорів.

Предмет дослідження: методи, алгоритми, інструменти автоматизації життєвого циклу віртуальних машин (модулі PowerShell, системні COM-об'єкти Windows) та технології побудови веб-орієнтованих сервісів керування інфраструктурою (FastAPI, REST API).

Обладнання та програмне забезпечення:

1. Персональний комп'ютер із підтримкою віртуалізації з ОС Windows 10.

2. Мови програмування JavaScript та Python, мова сценаріїв PowerShell.
3. Мова гіпертекстової розмітки HTML і каскадні таблиці стилів CSS.
4. Гіпервізор Oracle VM VirtualBox 7.2.8 r173730 (Qt6.8.0 on windows), пакет інструментів командного рядка VBoxManage та компонент Extension Pack.
5. Гостьові ОС: дистрибутиви Linux та образи Windows.
6. Веб-фреймворк FastAPI разом із ASGI-сервером Uvicorn (для реалізації REST API та веб-інтерфейсу системи), шаблонізатор Jinja2 (для динамічного рендерингу HTML-сторінок керування).

Консультанти з випускної кваліфікаційної роботи із зазначенням розділів, що їх стосуються

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
1	Миколайчук В. В.	 11.06.2026	 12.06.2026
2	Миколайчук В. В.	 13.06.2026	 14.06.2026
3	Миколайчук В. В.	 15.06.2026	 16.06.2026

Студент  Юрій БАХУР
Керівник  Олена ГОЛОВНЯ

Календарний план

№ з/п	Назви етапів випускної кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1	Постановка завдання та аналіз предметної області	04.02.2026-18.02.2026	Виконано
2	Проектування архітектури системи	20.02.2026-26.03.2026	Виконано
3	Реалізація логіки керування (Backend)	02.03.2026-29.03.2026	Виконано
4	Розробка веб-інтерфейсу	04.04.2026-12.05.2026	Виконано
5	Тестування та налагодження програмного комплексу	13.05.2026-28.06.2026	Виконано
6	Оформлення пояснювальної записки	11.06.2026-16.06.2026	Виконано
7	Підготовка доповіді	17.06.2026-22.06.2026	Виконано
8	Захист	23.06.2026	Виконано

Студент  Юрій БАХУР
Керівник  Олена ГОЛОВНЯ

АНОТАЦІЯ

У кваліфікаційній роботі бакалавра представлено розробку проєкта гібридного програмного комплексу автоматизації, що містить набір спеціалізованих PowerShell-модулів для локального керування і бекапу віртуальних машин за розкладом, а також асинхронний вебсервер на FastAPI, який надає REST API та графічний інтерфейс для віддаленого адміністрування гіпервізора.

Пояснювальна записка складається з трьох розділів, висновків та списку використаної літератури. Кваліфікаційна робота містить 154 сторінки, 105 рисунків, 4 таблиці. Список використаних джерел налічує 21 найменувань і розміщений на 2 сторінках.

Метою кваліфікаційної роботи є розробка та впровадження крос-платформного програмного комплексу автоматизованого керування, моніторингу та резервного копіювання віртуальних машин у середовищі Oracle VirtualBox для оптимізації процесів адміністрування інфраструктури розробки та тестування.

У першому розділі виконано теоретико-аналітичний аналіз технологій віртуалізації на базі гіпервізорів II типу (hosted), досліджено низькорівневу архітектуру інструментів керування гіпервізора Oracle VM VirtualBox, проведено порівняльний аналіз існуючих рішень автоматизації та обґрунтовано вибір технологічного стека для створення власної гібридної системи.

У другому розділі виконано інженерне проєктування архітектури системи, розроблено п'ятиетапний асинхронний алгоритм міжпроцесної взаємодії (IPC) та структуру безкомпонентної інтеграції. Обґрунтовано ресурсно-орієнтовану специфікацію REST API, виконано модульну декомпозицію backend-частини та спроектовано ергономічний багатосторінковий веб-інтерфейс із системою інтерактивної візуалізації станів.

ККІКБ.КР.Б – 123 – 01 – ПЗ

Змн.	Арк.	№ докум.	Підпис	Дата			
Розроб.		Бахур Ю. В.		22.06.24	Проект системи автоматизованого керування середовищем віртуальних машин	Літ.	Арк.
Керівник		Головня О. С.		22.06.24			Аркушів
Реценз.		Шелуха О. О.		22.06.24			4
Н. контр.		Головня О. С.		22.06.24			154
Затверд.		Єфіменко А. А.		22.06.24		ФІКТ, Гр. КІ-22-1	

У третьому розділі наведено практичну програмну реалізацію автономного консольного інтерфейсу на базі PowerShell та асинхронного серверного шлюзу на базі FastAPI. Представлено результати функціонального тестування інфраструктури, описано логіку роботи алгоритму онлайн-валідації архівів Test-OvaBoot, а також наведено порівняльний аналіз часових витрат та оцінку ефективності автоматизованого резервного копіювання.

Ключові слова: ВІРТУАЛІЗАЦІЯ, ГІПЕРВІЗОР, ORACLE VM VIRTUAL-BOX, VBOXMANAGE, POWERSHELL, FASTAPI, REST API, РЕЗЕРВНЕ КОПІЮВАННЯ, АВТОМАТИЗАЦІЯ, ПЛАНУВАЛЬНИК ЗАВДАНЬ, ТЕСТУВАННЯ ЦІЛІСНОСТІ, JSON, UAC.

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				5
Змн.	Арк.	№ докум.	Підпис	Дата		

ANNOTATION

This bachelor's thesis presents the development of a hybrid automation software suite that includes a set of specialized PowerShell modules for local management and scheduled backups of virtual machines, as well as an asynchronous web server built on FastAPI, which provides a REST API and a graphical interface for remote administration of the hypervisor.

The explanatory note consists of three sections, conclusions, and a list of references. The thesis contains 154 pages, 105 figures, and 4 tables. The list of references includes 21 entries and is presented on 2 pages.

The purpose of this thesis is to develop and implement a cross-platform software suite for the automated management, monitoring, and backup of virtual machines in the Oracle VirtualBox environment, with the aim of optimizing the administration of development and testing infrastructure.

The first chapter presents a theoretical and analytical study of Hosted-type virtualization technologies, examines the low-level architecture of the Oracle VM VirtualBox hypervisor management tools, conducts a comparative analysis of existing automation solutions, and justifies the choice of a technology stack for creating a custom hybrid system.

In the second chapter, the system architecture was engineered; a five-stage asynchronous interprocess communication (IPC) algorithm and a componentless integration structure were developed. A resource-oriented REST API specification was justified, a modular decomposition of the backend was performed, and a user-friendly multi-page web interface with an interactive status visualization system was designed.

The third chapter presents a practical software implementation of a standalone console interface based on PowerShell and an asynchronous server gateway based on FastAPI. It presents the results of functional testing of the infrastructure, describes the logic behind the Test-OvaBoot algorithm for online archive validation, and provides a comparative analysis of time expenditures and an assessment of the effectiveness of automated backup.

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		6

Keywords: VIRTUALIZATION, HYPERVISOR, ORACLE VM
VIRTUALBOX, VBOXMANAGE, POWERSHELL, FASTAPI, REST API, BACKUP,
AUTOMATION, TASK SCHEDULER, INTEGRITY TESTING, JSON, UAC.

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				7
Змн.	Арк.	№ докум.	Підпис	Дата		

ЗМІСТ

ВСТУП	10
РОЗДІЛ 1. АНАЛІЗ МЕТОДІВ ТА ЗАСОБІВ АВТОМАТИЗАЦІЇ СЕРЕДОВИЩ ВІРТУАЛІЗАЦІЇ	12
1.1. Роль віртуалізації у сучасних процесах розробки та актуальність автоматизації Hosted-гіпервізорів.....	12
1.2. Аналіз архітектури, можливостей та обмежень утиліти VBoxManage	14
1.3. Порівняльний аналіз існуючих інструментів автоматизації та керування інфраструктурою	18
1.4. Обґрунтування гібридного підходу до архітектури системи	21
1.5. Обґрунтування вибору технологічного стека розробки.....	23
Висновки до першого розділу	26
РОЗДІЛ 2. ПРОЄКТУВАННЯ ТА СИСТЕМНА АРХІТЕКТУРА ПРОГРАМНОГО КОМПЛЕКСУ	28
2.1. Архітектура та функціональна схема взаємодії компонентів системи.....	28
2.2. Проєктування модульної структури backend-частини системи	31
2.3. Розробка специфікації REST API	33
2.4. Розробка інтерфейсу користувача	38
Висновки до другого розділу	42
РОЗДІЛ 3. ПРОГРАМНА РЕАЛІЗАЦІЯ, НАЛАШТУВАННЯ ТА ТЕСТУВАННЯ СИСТЕМИ	44
3.1. Реалізація та демонстрація автономного консольного інтерфейсу	44
3.2. Розробка серверного шлюзу інтеграції на базі програми main.py	49
3.3. Програмна реалізація та тестування функцій керування ВМ.....	52
3.4. Програмна реалізація модулів менеджменту контрольних точок	68
3.5. Реалізація підсистеми резервного копіювання, планування та валідації ..	70
Висновки до третього розділу	78
ВИСНОВКИ.....	82
СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ	85
ДОДАТКИ.....	87

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

ВМ – віртуальна машина

ІТ – інформаційні технології

ОС – операційна система

ПЗ – програмне забезпечення

СУБД – система управління базами даних

AJAX – Asynchronous JavaScript and XML

API – Application Programming Interface

CI/CD – Continuous Integration/Continuous Delivery

CLI – Command Line Interface

COM-об'єкти – Component Object Model

DevOps – Development + Operations

GUI – Graphical User Interface

IPC – Inter-Process Communication

JSON – JavaScript Object Notation

MS – Microsoft

PS – PowerShell

RDP – Remote Desktop Protocol

REST API – Representational State Transfer Application Programming Interface

SSH – Secure Shell

UAC – User Account Control

UI/UX – User Interface/User Experience

URI – Uniform Resource Identifier

UUID – Universally Unique Identifier

VBoxSVC – VirtualBox Service

VNC – Virtual Network Computing

VRDP – VirtualBox Remote Desktop Protocol

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		9

ВСТУП

Актуальність теми. У сучасних умовах розробки програмного забезпечення, тестування та системного адміністрування використання технологій віртуалізації є невіддільною частиною ІТ-інфраструктури. Проте ручне керування великою кількістю віртуальних машин (ВМ), налаштування їхніх мережевих параметрів, створення знімків стану (snapshots) та забезпечення регулярного резервного копіювання через стандартний графічний інтерфейс користувача (GUI) призводить до значних часових витрат і збільшує ризик виникнення помилок через «людський фактор». Організація автоматизованого контролю через інтерфейс командного рядка (CLI) є ефективним розв'язанням цієї проблеми, але вимагає від оператора глибоких знань синтаксису конкретного гіпервізора. Тому розробка інтегрованої системи, яка об'єднує гнучкість сценаріїв автоматизації ОС та доступність крос-платформного веб-інтерфейсу (REST API) для керування середовищем віртуальних машин, є актуальним науково-практичним завданням.

Метою кваліфікаційної роботи є розробка та впровадження крос-платформного програмного комплексу автоматизованого керування, моніторингу та резервного копіювання віртуальних машин у середовищі Oracle VirtualBox для оптимізації процесів адміністрування інфраструктури розробки та тестування.

Визначена мета обумовлює наступні завдання.

- Провести аналіз архітектури утиліти командного рядка VBoxManage та дослідити методи міжпроцесної взаємодії в операційних системах сімейства Windows.
- Спроектувати та реалізувати модульну архітектуру backend-компонентів мовою сценаріїв PowerShell для керування життєвим циклом ВМ, конфігурацією знімків стану та планування завдань резервного копіювання.
- Розробити прикладний програмний інтерфейс REST API на базі асинхронного вебфреймворку FastAPI (Python) з інтеграцією механізмів верифікації системних привілеїв, автоматичного підвищення прав доступу (UAC) та кросплатформного кодування текстових потоків (UTF-8).
- Створити інтерактивний консольний інтерфейс користувача та динамічну веб-панель моніторингу для забезпечення уніфікованого доступу до функцій системи.

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		10

– Провести комплексне тестування розробленого програмного забезпечення, оцінити його відмовостійкість та ефективність автоматизації рутинних операцій адміністрування.

Об’єктом дослідження є процес адміністрування, забезпечення відмовостійкості, конфігурування та супроводження віртуальних інфраструктур на базі Hosted-гіпервізорів.

Предметом дослідження є методи, алгоритми, інструменти автоматизації життєвого циклу віртуальних машин (модулі PowerShell, системні COM-об’єкти Windows) та технології побудови веб-орієнтованих сервісів керування інфраструктурою (FastAPI, REST API).

У процесі виконання кваліфікаційної роботи було розроблено проєкт гібридного програмного комплексу автоматизації, що містить набір спеціалізованих PowerShell-модулів для локального керування і бекапу віртуальних машин за розкладом, а також асинхронний вебсервер на FastAPI, який надає REST API та графічний інтерфейс для віддаленого адміністрування гіпервізора.

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		11

РОЗДІЛ 1. АНАЛІЗ МЕТОДІВ ТА ЗАСОБІВ АВТОМАТИЗАЦІЇ СЕРЕДОВИЩ ВІРТУАЛІЗАЦІЇ

1.1. Роль віртуалізації у сучасних процесах розробки та актуальність автоматизації Hosted-гіпервізорів

Впровадження технологій віртуалізації стало одним із ключових етапів еволюції сучасного сектору інформаційних технологій. На сьогодні архітектурні підходи до розробки, тестування, розгортання та супроводження програмного забезпечення нерозривно пов'язані з ізоляцією обчислювальних середовищ. Віртуалізація дозволяє абстрагувати фізичні апаратні ресурси комп'ютера (процесорний час, оперативну пам'ять, дисковий простір та мережеві інтерфейси), створюючи на їхній базі логічно відокремлені сутності – віртуальні машини, які функціонують як повністю незалежні комп'ютери зі своїми операційними системами та системним логічним оточенням [1].

У сучасних парадигмах розробки, таких як DevOps та CI/CD, роль віртуалізації є критично важливою, зокрема, з наступних причин:

1. Ізоляція та відтворюваність середовищ розробки. Створення складного корпоративного ПЗ вимагає використання специфічних версій бібліотек, СУБД та конфігурацій ОС. Запуск таких проєктів безпосередньо на хостовій машині розробника часто призводить до конфліктів залежностей. Віртуалізація вирішує цю проблему, надаючи кожному розробнику ізольовану «пісочницю» (sandbox), конфігурація якої повністю ідентична продуктивному серверу.

2. Безпека та мінімізація ризиків. Експерименти з системними утилітами, тестування потенційно небезпечного або нестабільного коду в межах ВМ мінімізують ризик пошкодження хостової операційної системи. Завдяки механізмам миттєвих знімків стану системи, розробник може гарантовано повернути ОС до робочої точки у разі виникнення критичного збою.

3. Кросплатформне тестування. Наявність однієї потужної робочої станції дозволяє одночасно розгорнути тестові стенди під керуванням різних сімейств ОС

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		12

(наприклад, різних дистрибутивів Linux та версій MS Windows) для перевірки коректності кросплатформної роботи програмних продуктів [2].

Усі гіпервізори – програмне забезпечення, що керує віртуальними машинами – поділяють на два основні типи. Гіпервізори 1-го типу (Bare-Metal, наприклад, VMware ESXi або Proxmox VE) встановлюються безпосередньо на «голе» залізо і використовуються на виділених серверах, наприклад, у дата-центрах. Натомість на локальних робочих станціях інженерів, розробників та системних адміністраторів переважають гіпервізори 2-го типу (Hosted), яскравим представником яких є Oracle VM VirtualBox. Вони функціонують як звичайні прикладні програми поверх вже існуючої хостової операційної системи (наприклад, Windows 10/11) [1].

Популярність Hosted-гіпервізорів на індивідуальних робочих місцях зумовлена їхньою доступністю, безкоштовністю (відкрита ліцензія), підтримкою широкого спектра гостьових ОС та відсутністю потреби у виділеному серверному обладнанні. Адміністрування таких середовищ зазвичай відбувається через стандартний графічний інтерфейс користувача. Проте, у міру масштабування завдань, ручне керування через GUI виявляє низку суттєвих недоліків, що й зумовлює високу актуальність автоматизації цих процесів.

– Часові витрати на рутинні операції. Створення нової ВМ, виділення ресурсів, підключення віртуальних дисків та образів інсталяційних ISO-дисків через графічне меню вимагає виконання десятків монотонних кліків мишкою. Якщо розробнику чи тестувальнику потрібно розгорнути багатокомпонентну інфраструктуру (наприклад, клієнт-серверний застосунок із кількома вузлами), ручний процес стає неефективним і уповільнює розробку.

– Відсутність вбудованих засобів планування резервного копіювання. У штатному GUI VirtualBox відсутня функція автоматичного створення бекапів за розкладом. Системний адміністратор або розробник змушений вручну вимикати машину та експортувати її, що через людський фактор часто призводить до забування виконання бекапів або втрати актуальних даних у разі апаратного збою диска.

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		13

– Складність керування знімками стану (snapshots). При активній розробці дерево знімків швидко розростається. Без консолідованої логіки перевірки та відкочування, ручна навігація між десятками snapshots через GUI підвищує ризик випадкового видалення критично важливої гілки розробки.

– Потреба у гнучкій інтеграції з іншими системами. Сучасний інженерний підхід вимагає можливості керувати інфраструктурою програмно. Стандартне графічне вікно гіпервізора неможливо інтегрувати у сторонній веб-ресурс, мобільний додаток чи систему моніторингу компанії.

Оскільки Hosted-гіпервізори розгортаються на локальних комп'ютерах користувачів (часто з відносно обмеженими апаратними ресурсами), впровадження важких корпоративних систем оркестрації є недоцільним та технічно неможливим. Постає гостра потреба у створенні полегшених, гнучких рішень, які автоматизують взаємодію з низькорівневим інструментарієм командного рядка гіпервізора.

Таким чином, актуальність теми дослідження полягає у необхідності вирішення суперечності між високою затребуваністю Hosted-віртуалізації (Oracle VirtualBox) на робочих місцях ІТ-фахівців та низькою ефективністю ручного керування нею через GUI. Розробка автоматизованої системи, що поєднує системні скрипти з REST API, дозволяє виключити людський фактор із процесів резервного копіювання, кардинально прискорити розгортання робочих середовищ і надати зручний інструмент програмного контролю над віртуальною інфраструктурою хоста.

1.2. Аналіз архітектури, можливостей та обмежень утиліти VBoxManage

Низькорівневим ядром і фундаментом для побудови будь-яких систем автоматизації в екосистемі Oracle VM VirtualBox є утиліта VBoxManage. Це універсальний інтерфейс командного рядка, який постачається разом із гіпервізором і забезпечує повний контроль над його програмно-апаратним рушієм. На відміну від стандартного графічного менеджера (VirtualBox.exe), який є лише візуальною оболонкою, орієнтованою на кінцевого користувача, VBoxManage надає прямий доступ до внутрішнього API віртуалізаційної платформи.

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		14

Архітектура взаємодії утиліти VBoxManage із компонентами гіпервізора базується на компонентній модельній архітектурі. Коли користувач або зовнішній скрипт викликає команду VBoxManage, утиліта не виконує операції віртуалізації безпосередньо. Вона виступає в ролі клієнта, який надсилає міжпроцесні запити (IPC) до фонового системного процесу VBoxSVC.

Процес VBoxSVC є центральним координатором усієї системи: він відстежує стан усіх зареєстрованих віртуальних машин, веде глобальний облік віртуальних жорстких дисків, оптичних образів, мережевих адаптерів та координує розподіл фізичних ресурсів хоста. Перевага такої архітектури полягає в тому, що всі зміни, внесені через командний рядок VBoxManage, миттєво синхронізуються з базою даних VBoxSVC і автоматично відображаються у графічному GUI (якщо він запущений), запобігаючи виникненню конфліктів та порушенню цілісності метаданих (конфігураційних файлів .vbox).

Діапазон можливостей VBoxManage повністю покриває життєвий цикл віртуальної інфраструктури, розділяючись на кілька великих категорій.

1. Керування станом та життєвим циклом ВМ. За допомогою підкоманд `createvm` та `registervm` реалізується програмне створення та реєстрація конфігурацій. Команди групи `startvm` та `controlvm` дозволяють змінювати поточний стан машини на льоту: запускати в автономному чи фоновому (`headless`) режимах, ставити на паузу (`pause`), відновлювати роботу (`resume`), ініціювати коректне вимкнення через ACPI-сигнал (`acpipowerbutton`) або здійснювати примусове аварійне відключення (`poweroff`).

2. Глибока конфігурація апаратних параметрів. Модифікація внутрішнього «заліза» гостьової системи здійснюється через команду `modifyvm`. Вона дозволяє динамічно або статично виділяти обсяг оперативної пам'яті (`--memory`), графічної пам'яті (`--vram`), кількість ядер процесора (`--cpus`), налаштовувати порядок завантаження пристроїв (`--boot1`, `--boot2`), а також гнучко конфігурувати мережеві інтерфейси (перемикання між ізольованим режимом NAT та мережевим мостом Bridged).

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		15

3. Керування віртуальними носіями даних. Інструментарій `createmedium` і `storageattach` відповідає за операції з дисковою підсистемою. Він дозволяє створювати віртуальні диски різних форматів (VDI, VMDK, VHD) із динамічним або фіксованим розподілом простору, визначати їхній максимальний розмір, а також підключати їх до програмних контролерів (IDE, SATA, SCSI, NVMe) разом із ISO-образами операційних систем.

4. Технологія знімків стану (Snapshots). Команда `snapshot` забезпечує роботу з контрольною точкою стану системи. Вона дозволяє заморозити поточний стан ВМ (`take`), видалити застарілі гілки дерева знімків (`delete`) або миттєво відкотити стан диска та пам'яті до збереженої раніше точки (`restore`).

5. Імпорт та експорт інфраструктури. Підкоманди `export` та `import` слугують для відчуження віртуальних машин та їхнього пакування в уніфікований кросплатформний формат OVA (Open Virtualization Format), що є критично важливим для міграції та створення резервних копій [3].

Попри колосальний функціонал, використання VBoxManage як самостійного інструменту для щоденного адміністрування та автоматизації має низку суттєвих обмежень, які ускладнюють його експлуатацію в реальних ізольованих середовищах.

1. Низька атомарність та надмірний синтаксис команд. Утиліта не має поняття високорівневих комплексних сценаріїв. Наприклад, щоб створити повноцінну працездатну віртуальну машину, адміністратору недостатньо виконати одну команду. Потрібно послідовно запустити ланцюжок інструкцій: створити каркас машини, зареєструвати її, модифікувати параметри RAM/CPU, створити файл віртуального диска, підключити сховище до віртуального контролера і зв'язати його з мережевим адаптером. Помилка або друкарська помилка на будь-якому з цих кроків ламає весь процес розгортання.

2. Відсутність вбудованої логіки планування (Scheduling). VBoxManage є виключно реактивною програмою. Вона виконує команду тут і зараз, але не має внутрішніх механізмів, таймерів чи фонових процесів для автоматичного виконання

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		16

завдань за часом. Організація регулярних операцій (таких як щотижневе створення бекапів) повністю перекладається на зовнішні інструменти операційної системи.

3. Проблеми синхронізації та блокувань об'єктів. Якщо віртуальна машина перебуває в активному стані або заблокована іншим процесом, прямий виклик багатьох команд VBoxManage завершується критичною помилкою (наприклад, неможливо змінити конфігурацію мережі чи обсяг RAM на увімкненій машині). Утиліта не вміє самостійно обробляти такі стани, ставити команди в чергу або безпечно вимикати машину для застосування налаштувань з наступним перезапуском.

4. Відсутність стандартизованого структурованого виводу. Більшість інформаційних команд (наприклад, VBoxManage showvminfo) повертають результат у вигляді сирого, неструктурованого текстового потоку (Plain Text), оптимізованого під читання людиною, а не машиною. Для того, щоб виокремити з цього тексту конкретне значення (наприклад, UUID знімка або поточний статус мережевого адаптера), розробнику доводиться писати складні регулярні вирази (Regex) та парсери, що знижує стабільність автоматизації при зміні версій гіпервізора.

5. Відсутність мережевої прозорості та веб-інтерфейсу. Утиліта жорстко прив'язана до локального оточення хоста. Виконати команду VBoxManage можна лише перебуваючи безпосередньо всередині термінала цієї ОС, що унеможливорює віддалений моніторинг або інтеграцію керування в сучасні централізовані веб-панелі управління IT-інфраструктурою підприємства.

Таким чином, детальний аналіз архітектури та обмежень VBoxManage показує, що хоча утиліта є потужним інструментом виконання низькорівневих команд, вона гостро потребує створення інтеграційної високорівневої надбудови. Розроблюваний у кваліфікаційній роботі програмний комплекс (шляхом зв'язування обробки помилок у PowerShell та трансляції даних через REST API на FastAPI у форматі JSON) покликаний нівелювати зазначені недоліки: структурувати хаотичний вивід утиліти, ізолювати синтаксичну складність команд від користувача та розширити можливості гіпервізора функціями автоматичного планування і мережевої доступності.

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		17

1.3. Порівняльний аналіз існуючих інструментів автоматизації та керування інфраструктурою

При проектуванні системи автоматизованого керування середовищем віртуальних машин для локальних робочих станцій необхідно проаналізувати існуючі програмні рішення, які частково або повністю покривають подібний функціонал. На сучасному ринку інформаційних технологій існує велика кількість інструментів для автоматизації розгортання, створення резервних копій та віддаленого доступу. Проте більшість із них розроблялися під специфічні корпоративні завдання і мають архітектурні особливості, які не завжди є оптимальними для використання індивідуальними розробниками на локальних хост-системах. Для обґрунтування розробки власного програмного комплексу доцільно розглянути три різні категорії систем: автоматизацію розгортання (Vagrant), системи резервного копіювання (Bareos) та агрегатори віддаленого доступу (MobaXterm, mRemoteNG).

Аналіз інструментів автоматизації розгортання (Vagrant)

Програмний продукт Vagrant від компанії HashiCorp є одним із найпопулярніших інструментів для автоматизації створення та конфігурування віртуальних середовищ розробки. В основі його філософії лежить концепція «Інфраструктура як код» (Infrastructure as Code, IaC). За допомогою текстового конфігураційного файлу Vagrantfile користувач може однією командою `vagrant up` завантажити базовий образ операційної системи (так званий «box»), розгорнути його в Oracle VirtualBox, налаштувати мережеві інтерфейси та запустити скрипти внутрішнього налаштування [4].

Обмеження. Попри високу ефективність, Vagrant орієнтований виключно на життєвий цикл «чистих» середовищ розробки, які створюються з нуля. Він не призначений для управління вже існуючими, налаштованими вручну або імпортованими раніше віртуальними машинами користувача. Крім того, Vagrant не має вбудованих графічних інтерфейсів (Web GUI), не надає REST API для сторонньої інтеграції «з коробки» та позбавлений штатних засобів автоматичного створення резервних копій у форматі .ova за гнучким календарним розкладом хостової операційної системи.

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		18

Аналіз систем резервного копіювання (Bareos)

Bareos (Backup Archiving Recovery Open Sourced) – це потужна надійна система резервного копіювання, архівації та відновлення даних із відкритим вихідним кодом. Вона базується на дворівневій архітектурі, що складається з кількох фонових служб: Director (керує логікою), Storage Daemon (відповідає за збереження файлів) та File Daemon (клієнтський агент, що встановлюється на цільову систему) [5].

Обмеження. Використання Bareos для автоматизації бекапів віртуальних машин на індивідуальній робочій станції розробника є технологічно надмірним та нерациональним. Система Bareos вимагає розгортання окремого виділеного сервера або постійно активних важких служб, налаштування реляційної бази даних (PostgreSQL або MySQL) та детального опису конфігураційних файлів директив.

Аналіз засобів віддаленого доступу та моніторингу (MobaXterm та mRemoteNG)

MobaXterm та mRemoteNG є універсальними менеджерами віддалених підключень, які об'єднують в одному графічному інтерфейсі роботу за протоколами SSH, RDP, VNC, Telnet, SFTP тощо. Вони дозволяють системним адміністраторам мати швидкий доступ до консолей або графічних столів десятків різних серверів та віртуальних машин з єдиної панелі вкладок [6-7].

Обмеження. Ці рішення є виключно термінальними клієнтами доступу. Вони починають працювати лише тоді, коли віртуальна машина вже створена, успішно запущена гіпервізором та має налаштовану мережу.

Обґрунтування розробки власного програмного комплексу

Аналіз готових рішень показує, що ринок насичений або занадто глобальними корпоративними продуктами (Bareos), або вузькоспеціалізованими утилітами (Vagrant, MobaXterm). Спроба об'єднати їх на одній локальній робочій станції призведе до значного оверхеду – перевитрат оперативної пам'яті та процесорного часу хоста, які є критично важливими для роботи самих віртуальних машин.

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		19

Тому виникає необхідність розробки полегшеної, інтегрованої системи, яка об'єднає ключові затребувані функції: автоматизацію розгортання, планування локальних бекапів і надання веб-інтерфейсу REST API, не створюючи при цьому надлишкового навантаження на ресурси ПК.

Функціональні можливості розробленого програмного комплексу та наявних аналогів наведені в таблиці 1.1 для наочного порівняння.

Таблиця 1.1

Порівняльний аналіз характеристик систем керування інфраструктурою

Функціональний критерій	Vagrant	Bareos	MobaXterm	Розроблювана система
Керування життєвим циклом ВМ (створення, запуск, зупинка)	Так (через код)	Ні	Ні	Так (Web-панель & CLI)
Зміна апаратних конфігурацій (RAM, CPU, мережеві адаптери)	Так (через IaC)	Ні	Ні	Так (Web-панель & CLI)
Локальне резервне копіювання (експорт в .ova за розкладом)	Ні	Частково (складно)	Ні	Так (інтеграція з Task Scheduler)
Управління знімками стану (snapshots гіпервізора)	Частково (CLI)	Ні	Ні	Так (повна підтримка дерева)
Наявність уніфікованого REST API для сторонньої інтеграції	Ні	Так (складне)	Ні	Так (на базі FastAPI)
Вимоги до ресурсів хоста (RAM, дисковий простір)	Середні	Високі (потребує БД)	Низькі	Низькі (асинхронне ядро)
Контроль системних привілеїв Windows (UAC інтеграція)	Частково	Ні	Ні	Так (автоматичне підвищення)

1.4. Обґрунтування гібридного підходу до архітектури системи

На основі детального аналізу низькорівневих можливостей утиліти командного рядка VBoxManage, а також критичного огляду існуючих засобів автоматизації та резервного копіювання, постає завдання формування оптимальної архітектурної концепції розроблюваного програмного комплексу. Головна суперечність сучасних рішень полягає у виборі між важковаговою корпоративною функціональністю та легкістю локального розгортання. Для ефективного керування середовищем віртуальних машин на базі індивідуальної робочої станції (зокрема, в умовах обмежених ресурсів хоста) у межах цієї кваліфікаційної роботи сформовано систему рекомендацій та обґрунтовано гібридний підхід до побудови.

При проектуванні полегшеного комплексу керування Hosted-віртуалізацією рекомендується відмовитися від агентської архітектури та сторонніх баз даних на користь нативних інструментів операційної системи хоста. Рекомендації базуються на трьох фундаментальних принципах.

1. Максимальне використання системного API хостової ОС. Оскільки Oracle VirtualBox на робочій станції функціонує в межах Win32-підсистеми Windows, будь-які операції автоматизації (наприклад, резервне копіювання за розкладом) мають спиратися на перевірені вбудовані служби, такі як Планувальник завдань, замість розгортання постійно активних сторонніх демонів-таймерів.

2. Асинхронна міжпроцесна взаємодія. Програмний шар, що комунікує з користувачем, не повинен блокувати потоки виконання під час тривалих низькорівневих операцій (таких як експорт .ova архіву розміром у десятки гігабайтів). Необхідно реалізувати асинхронну модель трансляції команд.

3. Строга типізація та структуризація даних. Текстовий вивід утиліти VBoxManage має примусово перетворюватися в уніфікований формат JSON на етапі зчитування, що дозволить уникнути помилок інтерпретації даних при зміні версій гіпервізора.

Під «гібридністю» розроблюваної системи розуміється поєднання двох технологічно різних, але взаємодоповнюючих рівнів: локально-сценарного ядра (скрипти PowerShell) та мережевого веб-орієнтованого сервісу (FastAPI / Python). Такий

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		21

підхід дозволяє нівелювати недоліки кожної технології окремо, максимізуючи їхні переваги.

Реалізація гібридного підходу забезпечує розроблюваній системі низку стратегічних інженерних переваг.

1. Поділ відповідальності (Separation of Concerns). Веб-сервер Python відповідає виключно за маршрутизацію запитів, рендеринг інтерфейсу (Jinja2) та архітектуру API, тоді як PowerShell повністю бере на себе низькорівневу взаємодію з залізом та гіпервізором.

2. Контроль системних привілеїв та архітектурна цілісність. Оскільки створення запланованих бекапів через модуль `backup.psm1` потребує взаємодії з системним API Windows Task Scheduler, запуск компонентів системи у непривілейному режимі унеможливорює виконання адміністративних дій. Для вирішення цієї проблеми веб-сервер FastAPI та інтерактивне PowerShell-ядро містять логічні блоки перевірки привілеїв під час запуску. Це дозволяє системі функціонувати як єдине ціле: користувач одноразово підтверджує підвищення прав через стандартне діалогове вікно UAC, після чого всі модулі, включно з асинхронними підпроцесами Python, отримують легітимний високопривілейований доступ до ресурсів хоста.

3. Кросплатформна сумісність даних. Проблема передачі текстових потоків між двома різними середовищами (Python та PowerShell) вирішується шляхом примусового встановлення UTF-8 кодування в CLI-сесії та передачі структурованих JSON-об'єктів, що забезпечує коректну обробку кирилиці в назвах віртуальних машин або знімків стану.

Таким чином, було обрано гібридний підхід до архітектури системи як найбільш раціональне інженерне рішення для автоматизації середовища віртуальних машин за заданих умов. Вона дозволяє створити легкий, безкоштовний, централізований інструмент керування, який не перевантажує операційну систему хоста, забезпечує надійне автоматичне резервне копіювання та надає сучасний програмний REST API для віддаленого або локального адміністрування інфраструктури.

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		22

1.5. Обґрунтування вибору технологічного стека розробки

Успішність практичної реалізації системи автоматизованого керування середовищем віртуальних машин та її подальша експлуатація безпосередньо залежать від збалансованого вибору технологічного стека. При проєктуванні архітектури програмного комплексу було висунуто низку жорстких технічних та економічних критеріїв: мінімальне споживання оперативної пам'яті (RAM) та процесорного часу (CPU) хост-системи, висока швидкість обробки асинхронних запитів, строга структуризація міжпроцесного обміну даними, а також використання виключно безкоштовного програмного забезпечення з відкритим вихідним кодом (Open Source), не враховуючи витрат на хостові та гостьові операційні системи. На основі цих вимог було сформовано та обґрунтовано дворівневу гібридну модель розробки.

Для безпосередньої взаємодії з Hosted-гіпервізором Oracle VirtualBox та операційною системою Windows 10/11 як базовий інструмент автоматизації було обрано мову сценаріїв Windows PowerShell. Вибір PowerShell замість компільованих бінарних мов (наприклад, C# чи C++) обґрунтований такими інженерними перевагами.

1. Глибока інтеграція з Win32-підсистемою. PowerShell має прямий доступ до об'єктної моделі інструментарію керування Windows (WMI/CIM) та системного реєстру, що дозволяє безпечно взаємодіяти з локальними службами та Планувальником завдань (Task Scheduler) через COM-об'єкти без розгортання додаткових фонових служб.

2. Пряме керування утилітою VBoxManage. Сценарії PowerShell дозволяють виконувати прямі виклики CLI-інтерфейсу гіпервізора, гнучко перехоплювати коди завершення процесів за допомогою системної змінної \$LASTEXITCODE, обробляти потоки помилок і виконувати складні маніпуляції з дисковими сховищами .vdi.

3. Модульність та гнучкість супроводження коду. Реалізація логіки у вигляді ізольованих сценарних модулів (.psm1) спрощує налагодження та розширення системи, унеможливаючи потребу в постійній перекомпіляції всього проєкту при зміні логіки окремої команди.

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		23

4. Управління привілеями (UAC Validation). Мова PowerShell володіє вбудованими засобами для перевірки та точкового запиту на підвищення прав облікового запису користувача до рівня адміністратора, що є критично важливим для легітимного створення та видалення системних завдань резервного копіювання [8].

Оскільки сценарії PowerShell є виключно консольними інструментами і не призначені для побудови сучасних мережесистем розподілених систем, поверх сценарного ядра розгорнуто веб-сервер на базі мови програмування Python та мікрофреймворку FastAPI. Вибір цього інструментарію має наступне технічне обґрунтування.

1. Асинхронна архітектура (ASGI). FastAPI базується на специфікації асинхронного серверного інтерфейсу (через веб-сервер uvicorn [17]) та бібліотеці asyncio. Це дозволяє серверу обробляти сотні вхідних HTTP-запитів в одному потоці, не блокуючи роботу системи під час виконання тривалих низькорівневих операцій (наприклад, під час експорту VM у формат .ova модулем Export-VM).

2. Знижене споживання системних ресурсів. На відміну від важких фреймворків (таких як Django або корпоративні Java/C# рішення), FastAPI споживає мінімальну кількість оперативної пам'яті хоста, що дозволяє виділити максимум апаратних ресурсів під потреби самих віртуальних машин.

3. Валідація даних через Pydantic. Використання бібліотеки Pydantic у межах FastAPI забезпечує строгу типізацію та автоматичну перевірку вхідних JSON-схем на етапі парсингу, запобігаючи передачі некоректних параметрів (наприклад, невалідних значень обсягу RAM або CPU) до PowerShell-скриптів [18].

4. Автоматична генерація документації. Фреймворк самостійно будує інтерактивну графічну документацію API за стандартами OpenAPI (Swagger UI), що значно полегшує інтеграцію системи з іншими сторонніми сервісами підприємства [9].

Взаємодія між сервером Python та ядром PowerShell реалізована через вбудований асинхронний модуль subprocess, який запускає ізольовані CLI-сесії, примусово встановлює кодування UTF-8 для коректної передачі кирилиці та перетворює результати роботи скриптів в уніфікований формат JSON за допомогою командлета ConvertTo-Json.

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		24

Для забезпечення кросплатформного доступу до панелі управління з будь-якого пристрою (через звичайний браузер) без необхідності встановлення стороннього клієнтського ПЗ на робочі місця користувачів, обрано класичний стек веб-технологій.

З метою мінімізації навантаження на клієнтську сторону та хост-систему було вирішено відмовитися від складних і важких компонентних фреймворків (React, Angular чи Vue) на користь архітектури багатосторінкового веб-додатка (Multi-Page Application, MPA) на базі чистої мови JavaScript (Vanilla JS). Графічне оформлення, сітка та адаптивність інтерфейсу реалізовані за допомогою популярного сучасного CSS-фреймворку Bootstrap. Вибір Bootstrap обґрунтований наявністю готових надійних стильових компонентів (карток для відображення ВМ, навігаційних панелей, індикаторів завантаження), що дозволяє створити інтуїтивно зрозумілий та повністю чутливий інтерфейс, який коректно відображається на моніторах з будь-якою роздільною здатністю [10].

Для забезпечення високого рівня інтерактивності та покращення користувацького досвіду (UX) при виникненні системних подій в архітектуру фронтенду інтегровано спеціалізовану JavaScript-бібліотеку SweetAlert2. Вона використовується для генерації сучасних, анімованих модальних вікон і сповіщень замість застарілих системних діалогів браузера. Це дозволяє в реальному часі наочно інформувати користувача про статус виконання критичних операцій: успішне завершення автоматичного бекапу, підтвердження незворотного видалення знімка стану (snapshot) або виведення деталізованого тексту інженерної помилки у разі збою на рівні PowerShell-ядра [11].

Комунікація фронтенду з REST API сервера FastAPI відбувається асинхронно через вбудований браузерний API fetch() [19]. Це дозволяє відправляти фонові запити до backend-частини та динамічно оновлювати поточні статуси віртуальних машин (увімкнена / вимкнена) і хід створення резервних копій у реальному часі без повного перезавантаження веб-сторінки [9].

Таким чином, обраний технологічний стек є внутрішньо інтегрованим, технічно виправданим та економічно доцільним. Поєднання системних можливостей

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		25

мови PowerShell, асинхронної швидкодії фреймворку FastAPI (Python) та полегшеності фронтенд-рішень (HTML, Bootstrap, JavaScript, SweetAlert2) дозволяє створити високоефективний програмний комплекс. Він повністю нівелює недоліки ручного керування гіпервізором, забезпечуючи високу надійність обробки даних, безпеку через інтеграцію з Windows UAC [20] та миттєву візуальну чутливість інтерфейсу користувача.

Висновки до першого розділу

У першому розділі виконано теоретико-аналітичний аналіз технологій віртуалізації, досліджено інструменти керування гіпервізором та обґрунтовано концепцію системи. За результатами сформовано такі висновки.

1. Доведено актуальність автоматизації Hosted-гіпервізора Oracle VM VirtualBox. Ручне керування через GUI є неефективним через високі часові витрати, відсутність автоматичного планування бекапів та складність менеджменту знімків стану (snapshots).

2. Проаналізовано утиліту VBoxManage. Визначено, що вона надає повний низькорівневий контроль над ВМ, але має суттєві обмеження: складний синтаксис, відсутність вбудованого планувальника задач та неструктурований текстовий вивід, що потребує створення високорівневої програмної надбудови.

3. Проведено порівняльний аналіз існуючих рішень та виявлено їхні недоліки для локального використання. Vagrant орієнтований лише на створення ВМ з нуля; Vareos є занадто важким та складним; MobaXterm є лише терміналом доступу. Спроба їхнього сумісного використання створює суттєвий оверхед для RAM та CPU хоста, що обґрунтовує розробку власної легковагової системи.

4. Обґрунтовано концепцію гібридної архітектури. Запропоновано розподіл логіки: локально-сценарне ядро (Windows PowerShell) відповідає за взаємодію з системним API та гіпервізором, а мережевий сервіс (Python/FastAPI) – за маршрутизацію та інтерфейс. Взаємодія між ними реалізована через асинхронні підпроцеси із примусовим перетворенням даних у JSON.

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		26

5. Обґрунтовано вибір технологічного стека. Для бекенду обрано асинхронний фреймворк FastAPI (мінімальне споживання RAM, валідація Pydantic, документація Swagger UI). Для фронтенду обрано архітектуру MPA на базі HTML5, Vanilla JS, Bootstrap та бібліотеки SweetAlert2, що забезпечує адаптивність, безпеку (UAC) та інтерактивне оновлення таблиць без зайвих перезавантажень сторінок.

Сформована аналітична база дозволяє перейти до проєктування архітектури та алгоритмів системи у наступному розділі.

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		27

РОЗДІЛ 2. ПРОЄКТУВАННЯ ТА СИСТЕМНА АРХІТЕКТУРА ПРОГРАМНОГО КОМПЛЕКСУ

2.1. Архітектура та функціональна схема взаємодії компонентів системи

Проектування системи автоматизованого керування середовищем ВМ потребує створення чіткої архітектурної моделі, яка здатна об'єднати високорівневий користувацький інтерфейс із низькорівневим інструментарієм командного рядка гіпервізора Oracle VM VirtualBox. Враховуючи обґрунтований в розділі 1 гібридний підхід до архітектури системи, в основу проектування закладено триланкову (3-tier) архітектурну модель без використання зовнішніх агентів та баз даних. Це дозволяє досягти максимальної полегшеності програмного комплексу та виключити перевитрату апаратних ресурсів хост-системи.

Розроблювана система поділяється на три логічні рівні (ланки), кожен з яких виконує строго обмежений набір функцій.

1. Рівень представлення (Client/Frontend). Реалізований як веб-додаток за архітектурою Multiple Page Application (MPA) на базі HTML5, Bootstrap, JavaScript та SweetAlert2. Він функціонує в контексті веб-браузера користувача і відповідає за візуалізацію стану інфраструктури, перехоплення дій користувача та відображення інтерактивних сповіщень.

2. Серверний веб-орієнтований рівень (Backend/Middleware). Реалізований на базі асинхронного фреймворку FastAPI (Python). Він виступає в ролі центрального шлюзу інтеграції, забезпечує маршрутизацію HTTP-запитів, валідацію вхідних параметрів за допомогою моделей Pydantic, керування чергами завдань та генерацію інтерактивної документації OpenAPI.

3. Локально-сценарний системний рівень (Core/System Engine). Реалізований у вигляді набору ізольованих модулів сценаріїв Windows PowerShell (.psm1). Цей рівень володіє адміністративними привілеями (UAC context) і безпосередньо взаємодіє з підсистемою Windows Task Scheduler та утилітою командного рядка VBoxManage.

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		28

Взаємодія між компонентами системи має чітку послідовну логіку, побудовану на базі асинхронних міжпроцесних викликів. Загальний цикл обробки будь-якої команди користувача (наприклад, створення резервної копії VM) описується наступним ланцюжком взаємодії: Користувач (Браузер) → FastAPI (REST API) → PowerShell (Core) → VBoxManage → Гіпервізор.

Покроковий алгоритм функціонування схеми взаємодії компонентів системи включає такі етапи.

Етап 1 (Запит клієнта). Користувач натискає кнопку в інтерфейсі (наприклад, «Запустити VM»). JavaScript-скрипт фронтенду перехоплює подію та ініціює асинхронний HTTP-запит (за допомогою методу `fetch()`) до відповідного endpoint-маршруту на сервері FastAPI (наприклад, `POST /api/vms/{vm_name}/start`).

Етап 2 (Валідація та трансляція). FastAPI приймає HTTP-запит. Компонент Rудantic перевіряє вхідні змінні на відповідність типам даних. Якщо дані валідні, сервер за допомогою вбудованого асинхронного модуля `subprocess` запускає ізольований процес Windows PowerShell, передаючи йому як аргументи назву потрібного модуля (`vm.psm1`), внутрішню функцію (`Start-VM`) та параметри (назву VM та тип).

Етап 3 (Низькорівневе виконання). Запущений скрипт PowerShell імпортує необхідні системні директиви, перевіряє наявність адміністративних прав і формує кінцеву CLI-інструкцію для утиліти VBoxManage (в даному випадку: `VBoxManage startvm "vm_name" --type headless`). Утиліта через міжпроцесну взаємодію передає команду демону VBoxSVC, який безпосередньо змінює стан віртуальної машини в ядрі гіпервізора.

Етап 4 (Перетворення та зворотний зв'язок). Після виконання команди утиліти VBoxManage повертає текстовий вивід. Скрипт PowerShell перехоплює код завершення операції (`$LASTEXITCODE`), за допомогою регулярних виразів структурує Plain Text, перетворює його на об'єкт і за допомогою командлета `ConvertTo-Json` віддає в потік `stdout`.

Етап 5 (Рендеринг відповіді). FastAPI зчитує JSON-потік з `stdout`, завершує асинхронну сесію підпроцесу і повертає JSON-відповідь клієнту із HTTP-кодом 200 OK. Браузерний JavaScript-код приймає відповідь, а бібліотека SweetAlert2 миттєво

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		29

виводить красиве анімоване вікно про успішний запуск машини, змінюючи статус картки ВМ в інтерфейсі за допомогою Bootstrap-стилів.

При проєктуванні схеми взаємодії особливу увагу приділено логіці виконання операцій резервного копіювання, які через низькорівневий експорт образів у формат .ova є найбільш тривалими та ресурсомісткими в системі. Залежно від потреб користувача, архітектура системи передбачає три гнучкі режими взаємодії компонентів.

1. Миттєве створення (Backup Now). Користувач ініціює створення копії в поточний момент часу. Серверний веб-рівень транслює команду до PowerShell-модуля backup.psm1, який запускає процедуру експорту. При цьому клієнтський інтерфейс за допомогою Bootstrap-компонентів візуалізує процес очікування, а Sweet-Alert2 блокує випадкові повторні кліки користувача, відображаючи індикатор прогресу до моменту отримання фінального JSON-статусу від FastAPI.

2. Одноразове відкладене резервне копіювання. Користувач вказує конкретну дату та час у майбутньому. У цьому випадку FastAPI передає параметри в PS, який за допомогою COM-об'єктів реєструє одноразове завдання безпосередньо в Планувальнику завдань Windows. Веб-сервер миттєво повертає статус успішного планування, не навантажуючи хост-систему поточними обчисленнями.

3. Регулярне автоматичне копіювання (За розкладом). Користувач задає часовий інтервал (наприклад, щотижня чи щомісяця). Система за аналогічним принципом створює постійне заплановане завдання в ОС Windows. Надалі операційна система сама у визначений час викликає розроблений PowerShell-скрипт у фоновому автономному режимі, повністю виключаючи людський фактор та забезпечуючи збереження даних навіть без запущеного веб-інтерфейсу системи.

Такий підхід дозволив розмежувати навантаження на хост-систему: тривалі рутинні задачі делегуються системному планувальнику ОС, тоді як веб-інтерфейс залишається повністю чутливим і доступним для виконання інших операцій над віртуальними машинами.

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		30

2.2. Проєктування модульної структури backend-частини системи

Ефективність, відмовостійкість та легкість супроводження розроблюваного програмного комплексу безпосередньо залежать від декомпозиції його бізнес-логіки. При проєктуванні архітектури backend-частини системи було застосовано строгий модульний підхід на базі мови сценаріїв Windows PowerShell. Логіка взаємодії з Hosted-гіпервізором розмежована на п'ять функціональних модулів (.psm1), кожен з яких відповідає за ізольований спектр завдань та експортує строго визначений набір функцій для високорівневого веб-сервера чи CLI-сесій.

Кодова база сценарного ядра backend-частини організована у вигляді взаємопов'язаної модульної системи. Структура файлів має такий вигляд:

- `utils.psm1` – базові системні утиліти, парсинг даних та опитування ОС;
- `vm.psm1` – модуль керування життєвим циклом та конфігурацією VM;
- `snapshot.psm1` – модуль менеджменту знімків стану;
- `backup.psm1` – модуль інтеграції з Task Scheduler та створення .ova копій;
- `menu.psm1` – модуль консольного меню та інтерактивної валідації вибору.

Усі спеціалізовані модулі (`vm`, `snapshot`, `backup`) імпортують базовий модуль `utils.psm1` за допомогою директиви `Import-Module`, що забезпечує наскрізне використання спільних утиліт обробки команд.

Для детального аналізу спроектованої структури опишемо архітектурне призначення та ключові функції кожного розробленого модуля.

1. Модуль системних функцій (`utils.psm1`)

Він виступає фундаментом backend-частини. Його головна задача – абстрагувати низькорівневий текстовий вивід утиліти `VBoxManage` та системні виклики ОС у структуровані об'єкти. Ключові функції модуля є наступними.

- `Invoke-VBoxCommand`. Універсальний обгортковий метод для безпечного виконання команд `VBoxManage`. Він перехоплює системний код завершення `$LASTEXITCODE`, об'єднує потоки `stdout` і `stderr`, та повертає уніфікований об'єкт із прапором `Success`.
- `Get-VMs`. Повертає перелік зареєстрованих машин.
- `Get-VMState`. Повертає поточний статус VM (`running`, `paused`, `poweroff`).

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		31

- Get-Resources. За допомогою CIM-інстансів (Win32_ComputerSystem [12], Win32_Processor [13], Win32_LogicalDisk [14]) опитує хост-систему щодо вільного обсягу RAM, ядер CPU та пам'яті на дисках.

2. Модуль керування віртуальними машинами (vm.psm1)

Ізолює в собі логіку конфігурування та зміни станів VM.

- New-VM. Послідовно викликає ланцюжок інструкцій для створення каркаса машини, модифікації RAM/VRAM, генерації динамічного віртуального диску .vdi, ініціалізації контролерів зберігання даних та монтування ISO-образів.

- Start-VM та Stop-VM. Відповідають за запуск у фоновому режимі чи з графічним інтерфейсом та коректну зупинку через ACPI або примусове вимкнення.

- Set-VMNetwork. Налаштовує режими мережевих адаптерів (nat, bridged, hostonly) та зв'язує їх із конкретними фізичними картами хоста.

3. Модуль менеджменту контрольних точок (snapshot.psm1)

Реалізує повну підтримку дерева знімків стану гіпервізора. Містить три базові функції.

- New-Snapshot. Створює знімок стану системи заданої VM.

- Restore-Snapshot. Миттєвий відкат стану диска та пам'яті.

- Remove-Snapshot. Видалення гілки знімків із консолідацією даних.

Для уникнення конфліктів ідентифікації функції проєктуються із можливістю оперування як за назвою знімка, так і за його унікальним UUID.

4. Модуль резервного копіювання та планування (backup.psm1)

Включає складну логіку відчуження інфраструктури та автоматизації розкладу.

- Export-VM та Import-VM. Здійснюють пакування машин у кросплатформний формат прикладного ПЗ .ova та зворотне відновлення.

- Register-VMBackupTask. Створює динамічний сценарний блок (ScriptBlock) та реєструє фонові завдання у Планувальнику задач Windows, гнучко налаштовуючи тригери підтримуваних режимів: одноразово (Once), щодня (Daily), щотижня (Weekly) чи щомісяця (Weekly, WeeksInterval 4), а також при запуску операційної системи (AtStartup) та вході користувача в систему (AtLogOn) [21].

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		32

- Test-OvaBoot. Функція превентивного тестування бекапів, яка розгортає тимчасову тестову VM із бекапу, ініціює фоновий старт, валідує завантаження ядра операційної системи гостя через `guestproperty` та перетворює вердикт (Valid, Corrupted або Unknown при помилці під час тестування) у JSON-метадані.

5. Модуль інтерфейсу користувача (`menu.psm1`)

Описує структуру меню для CLI-режиму роботи. Завдяки функції `Get-User-Selection`, модуль забезпечує інтерактивний вибір елементів (конкретної VM або знімка) із динамічних масивів даних із захистом від некоректного введення користувача.

При проектуванні структури модулів вирішено ключову проблему утиліти `VBoxManage` – неструктурований вивід даних. Взаємодія між модулями базується на передачі строго типізованих об'єктів [`PSCustomObject`] [15].

Коли `FastAPI` викликає будь-яку функцію з цих модулів, результат роботи фіксується у вигляді об'єкта, який на виході автоматично конвертується у формат JSON за допомогою командлета `ConvertTo-Json`.

Переваги розробленої структури є наступними:

- Висока атомарність. Кожна функція виконує одну конкретну дію.
- Ізоляція помилок. Використання конструкцій `try-catch` всередині модулів (наприклад, при валідації бекапу чи роботі з дисками) запобігає падінню всього backend-ядра у разі системного збою.
- Уніфікація відповідей. Незалежно від того, яка операція виконується (створення знімка чи налаштування мережі), веб-сервер завжди отримує стандартизований JSON-пакет зі статусом успішності та текстовим повідомленням для виведення на фронтенді.

2.3. Розробка специфікації REST API

Для забезпечення мережевої прозорості, можливості віддаленого керування та інтеграції розробленого сценарного ядра з веб-інтерфейсом (фронтендом) спроектовано специфікацію REST API [16]. Веб-сервер на базі фреймворку `FastAPI` виступає в ролі архітектурного мосту: він приймає вхідні HTTP-запити від клієнта,

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		33

валідує їх за допомогою схем Pydantic, викликає відповідні функції з модулів PowerShell і повертає структуровану відповідь у форматі JSON.

REST API системи побудовано за принципом ресурсної орієнтації, де кожна сутність (віртуальна машина, хост-машина, бекап) має власну унікальну адресу та керується за допомогою стандартних HTTP-методів:

- GET – використовується для отримання даних (ресурсу) із сервера;
- POST – використовується для створення нового ресурсу на сервері чи запуску певної дії;
- PATCH – використовується для часткового оновлення наявного ресурсу;
- DELETE – використовується для видалення ресурсу із сервера.

REST API системи побудовано за принципом ресурсної орієнтації, де всі методи керування чітко розмежовані за трьома базовими архітектурними шляхами.

1. /api/vms – центральний маршрут для керування життєвим циклом, апаратними параметрами, мережею віртуальних машин, а також усіма залежними ресурсами (знімками стану та операціями негайного експорту/імпорту .ova файлів).

2. /api/backups – спеціалізований маршрут для взаємодії з Планувальником завдань Windows з метою створення, перегляду та видалення автоматичних бекапів, а також автоматизованої перевірки цілісності резервних копій.

3. /api/host – маршрут для збору телеметрії та діагностики поточної апаратної конфігурації фізичного комп'ютера, який виступає сервером.

Специфікація розроблених маршрутів REST API, відповідних HTTP-методів та їхньої взаємодії з функціями PowerShell-ядра наведена у таблицях 2.1-2.3.

Таблиця 2.1

Специфікація маршрутів REST API за маршрутом /api/vms

URL-маршрут	HTTP-метод	Модуль	Функціональне призначення операції
/api/vms	GET	utils.psm1	Отримання списку всіх наявних VM, їх поточного статусу та UUID.

URL-маршрут	HTTP-метод	Модуль	Функціональне призначення операції
/api/vms	POST	vm.psm1	Створення нової ВМ чи відновлення ВМ з бекапу.
/api/vms/{vm_name}/start	POST	vm.psm1	Запуск ВМ у фоновому режимі чи з GUI.
/api/vms/{vm_name}/stop	POST	vm.psm1	Вимкнення ВМ: безпечне (ACPI) чи миттєве (poweroff).
/api/vms/{vm_name}?mode={mode}	DELETE	vm.psm1	Видалення ВМ з очищенням чи без.
/api/vms/{vm_name}/details	GET	utils.psm1	Отримати масив параметрів
/api/vms/{vm_name}/full-details	GET	Відсутній	Отримання повного виводу команди showvminfo
/api/vms{vm_name}	PATCH	vm.psm1	Модифікація значення RAM, CPU, VRAM, типу мережі або ISO-образу.
/api/vms{name}/backups	POST	backup.psm1	Негайний експорт поточної конфігурації ВМ у файл .ova.
/api/vms/{vm}/snapshots	GET	snapshot.psm1	Отримання переліку знімків
/api/vms/{vm}/snapshots	POST	snapshot.psm1	Створення нового знімку ВМ
/api/vms/{vm}/snapshots/{uuid}/restore	POST	snapshot.psm1	Відновлення стану ВМ за певним знімком
/api/vms/{vm}/snapshots/{uuid}	DELETE	snapshot.psm1	Видалення певного знімку віртуальної машини

Таблиця 2.2

Специфікація маршрутів REST API за маршрутом /api/host

URL-маршрут	HTTP-метод	Модуль	Функціональне призначення операції
/api/host/network-interfaces ?type={type}	GET	utils.psm1	Отримання переліку мережевих інтерфейсів за типом.
/api/host/resources?drive=...	GET	utils.psm1	Отримання ресурсів хоста.
/api/host/drives	GET	utils.psm1	Отримання переліку дисків.
/api/host/folder-contents ?path={path}	GET	utils.psm1	Отримання переліку файлів та каталогів за шляхом.
/api/vms/{vm}/snapshots/ {uuid}? snap_name={...}	DELETE	snapshot.psm1	Видалення певного знімку віртуальної машини

Таблиця 2.3

Специфікація маршрутів REST API за маршрутом /api/backups

URL-маршрут	HTTP-метод	Модуль	Функціональне призначення операції
/api/backups?path={...}	GET	utils.psm1	Отримання переліку бекапів ВМ.
/api/backups/test-boot	POST	backup.psm1	Тестування резервної копії на цілісність за шляхом.
/api/backups/schedules	GET	utils.psm1	Отримання переліку запланованих бекапів ВМ.
/api/backups/schedules	POST	backup.psm1	Планування автоматичної копії
/api/backups/schedules/ {task_name}	DELETE	backup.psm1	Видалення запланованого бекапу ВМ за назвою.

На рівні REST API вхідні параметри структуруються у вигляді JSON-об'єктів (Request Body), які автоматично транслюються у PowerShell як іменовані аргументи, при методах POST та PATCH, а при GET та DELETE – параметри використовуються для фільтрації або пошуку даних (Query).

Коли користувач змінює конфігурацію заліза через Bootstrap-інтерфейс, фронтенд надсилає такий асинхронний пакет (JSON):

```
{  
  "vm_property": "--cpus",  
  "value": "8"  
}
```

FastAPI парсить цей об'єкт і запускає підпроцес PowerShell функцією run_ps: run_ps(cmd=f"Edit-VM -Name \"{name}\" -Property '{data.vm_property}' -Value {int(data.value)}", module="vm")

Текстовий потік stdout з консолі, згенерований функцією Invoke-VBoxCommand, завжди повертає клієнту передбачувану об'єктну структуру (JSON):

```
{  
  "Success": true,  
  "ExitCode": 0,  
  "Message": "Параметр '--cpus' ВМ '{name}' успішно змінено на '8'.",  
  "Timestamp": "2026-06-16 22:35:10"  
}
```

Логіка REST API повністю захищена від аварійних зупинок. Якщо користувач намагається виконати невалідну дію (наприклад, видалити знімок стану на запущеній машині або призначити обсяг RAM, що перевищує ресурси хоста), системний блок Try-Catch у PowerShell перехоплює код помилки.

Через потік виводу FastAPI транслює цю помилку на фронтенд у полі "Message", де бібліотека SweetAlert2 перетворює текстовий опис інженерної помилки на інформативне модальне вікно для користувача, зберігаючи при цьому стабільність та чутливість Single Page Application.

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		37

2.4. Розробка інтерфейсу користувача

Головною метою розробки інтерфейсу користувача (UI/UX) є створення єдиного, інтуїтивно зрозумілого та чутливого робочого простору адміністратора, який повністю приховує складність синтаксису командного рядка утиліти VBoxManage та автоматично адаптується під будь-які роздільні здатності моніторів. Відповідно до обраної концепції Multiple Page Application (MPA), взаємодія користувача з графічною оболонкою відбувається динамічно, без повного перезавантаження сторінок браузера, що значно прискорює відгук інтерфейсу (якщо не змінено сторінку).

2.4.1. Головна сторінка «Керування віртуальними машинами»

Головна сторінка системи (рис. 2.1) є основним робочим простором адміністратора. На ній розташовані дві кнопки «Створити VM» та «Оновити список» та таблиця, яка містить інформацію про віртуальні машини: назву VM, її статус, UUID та дії над VM.

На сторінці реалізовано кольорове кодування: статуси «Running», «Powered Off» та «Aborted» підсвічуються відповідними бейджами (зеленим, сірим та червоним), що дозволяє миттєво ідентифікувати стан системи.

У колонці «Дії» маємо кнопки «Запустити» або «Вимкнути» залежно від стану VM, «Інформація», яка виводить таблицю з параметрами VM (коротко) та містить опцію переглянути повний перелік усіх параметрів VM, «Редагувати» та іконку корзини, які є заблокованими за активної VM. Кожен елемент керування ініціює асинхронні HTTP-запити до REST API.

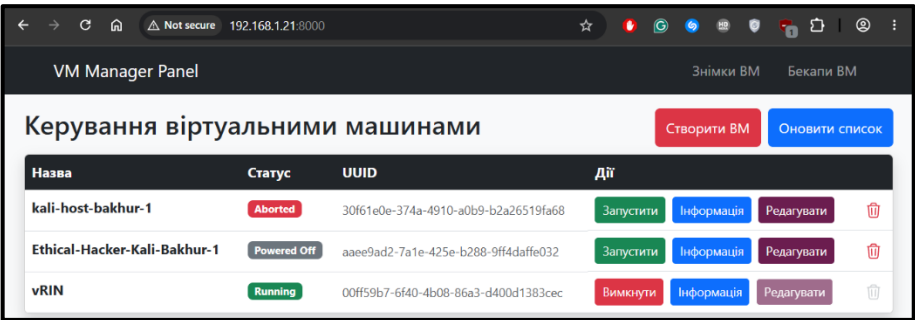
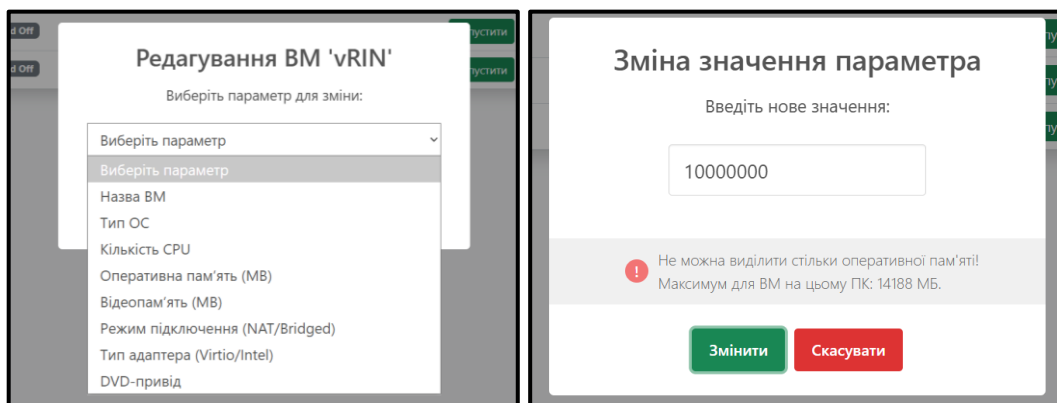


Рисунок 2.1 – Сторінка «Керування віртуальними машинами»

Для мінімізації кількості перезавантажень сторінок, процеси редагування, створення нових VM і виведення інформації винесено в окремі модальні вікна.

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		38

Форма передбачає інтерактивну валідацію: введення некоректних символів або перевищення лімітів оперативної пам'яті хоста блокується на рівні клієнтського JavaScript-коду перед відправкою на backend (рис. 2.2-2.3).



Рисунки 2.2-2.3 – Модальне вікно конфігурування параметрів VM (RAM)

2.4.2. Вікно менеджменту контрольних точок (Snapshot Manager)

При натисканні на елемент «Знімки VM» на навігаційній панелі, здійснено перехід на іншу сторінку (рис. 2.4). Для виведення заповнення таблиці знімків стану VM потрібно вибрати її у випадальному списку. Коли здійснено вибір, надсилається запит на `/api/vms/{name}/snapshots` та візуалізується отримана інформація, як можна побачити на рисунку 2.5.

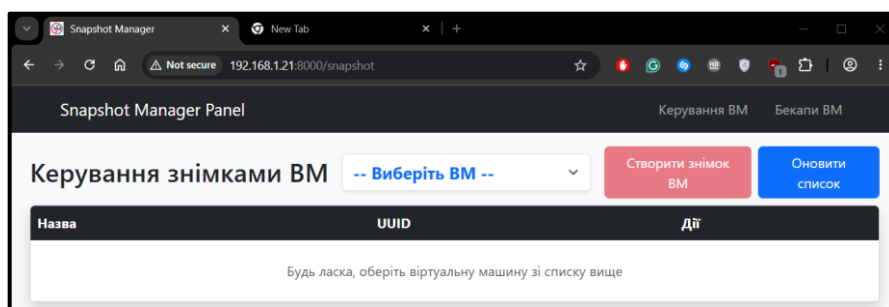


Рисунок 2.4 – Сторінка «Керування знімками VM»

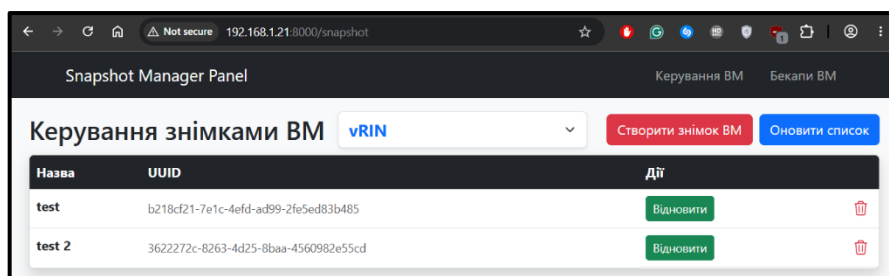
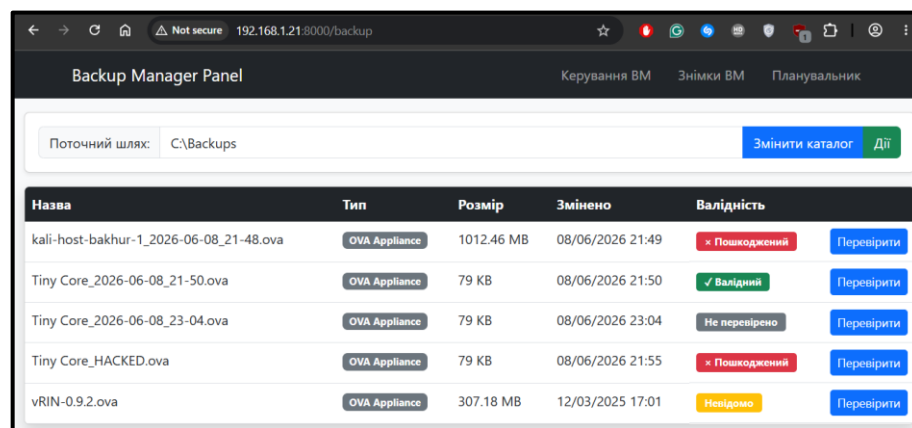


Рисунок 2.5 – Заповнена таблиця наявних знімків стану вибраної VM

Користувач має можливість відкотити стан системи до обраної точки або ініціювати видалення застарілої гілки знімків.

2.4.3. Сторінка «Керування бекапами VM»

Перехід на вкладку «Бекапи VM» відкриває сторінку, яка містить поле поточного шляху, відповідно до якого відбуватимуться всі дії, включаючи заповнення таблиці бекапами (рис. 2.6). Таблиця містить інформацію про назву бекапа, тип файлу, його розмір та дату останньої зміни, а також цілісність та кнопку для перевірки. Валідність має кольорове кодування: статуси «Валідний», «Не перевірений», «Пошкоджений» та «Невідомо» підсвічуються відповідними бейджами (зеленим, сірим, червоним та жовтим), що дозволяє миттєво ідентифікувати стан бекапів.

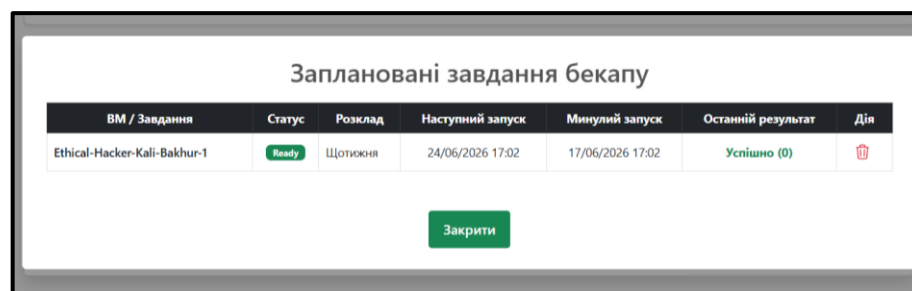


The screenshot shows a web interface titled "Backup Manager Panel" with a navigation bar containing "Керування VM", "Знімки VM", and "Планувальник". Below the navigation bar is a search bar with the text "Поточний шлях: C:\Backups" and buttons "Змінити каталог" and "Дії". The main content is a table with the following data:

Назва	Тип	Розмір	Змінено	Валідність	Дії
kali-host-bakhur-1_2026-06-08_21-48.ova	OVA Appliance	1012.46 MB	08/06/2026 21:49	✖ Пошкоджений	Перевірити
Tiny Core_2026-06-08_21-50.ova	OVA Appliance	79 KB	08/06/2026 21:50	✓ Валідний	Перевірити
Tiny Core_2026-06-08_23-04.ova	OVA Appliance	79 KB	08/06/2026 23:04	Не перевірено	Перевірити
Tiny Core_HACKED.ova	OVA Appliance	79 KB	08/06/2026 21:55	✖ Пошкоджений	Перевірити
vRIN-0.9.2.ova	OVA Appliance	307.18 MB	12/03/2025 17:01	Невідомо	Перевірити

Рисунок 2.6 – Сторінка «Керування бекапами VM»

Також на навігаційній панелі є нове посилання «Планувальник», яке відкриває модальне вікно (рис. 2.7), що міститиме таблицю запланованих автоматичних резервних копій VM. Таблиця має такі поля: назва VM, якої стосується завдання, розклад, дата наступного та минулого запуску, код останнього результату та дія «Видалити».



The screenshot shows a modal window titled "Заданні бекапу" (Backup Tasks). It contains a table with the following data:

VM / Завдання	Статус	Розклад	Наступний запуск	Минулий запуск	Останній результат	Дія
Ethical-Hacker-Kali-Bakhur-1	Ready	Щотижня	24/06/2026 17:02	17/06/2026 17:02	Успішно (0)	Видалити

Below the table is a green button labeled "Закрити" (Close).

Рисунок 2.7 – Модальне вікно «Планувальник»

2.4.4. Інтерфейс візуалізації фонових операцій (SweetAlert2)

Для забезпечення якісного UX-досвіду під час виконання операцій (наприклад, експорт .ova образів чи перевірка їх цілісності) застосовується блокуюче модальне вікно очікування (рис. 2.8).

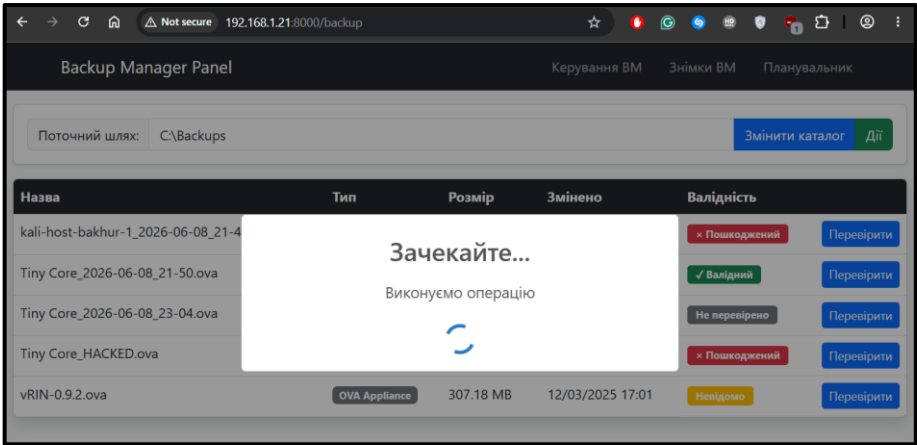


Рисунок 2.8 – Інформування про хід виконання асинхронного фонового процесу

Це вікно перехоплює фокус користувача, унеможливлюючи надсилення дублюючих запитів, і автоматично закривається після отримання фінального JSON-пакета з кодом завершення операції від FastAPI, виводячи інформацію про результат.

Оскільки операції конфігурування, створення знімків або планування розкладу вимагають введення додаткових параметрів, інтерфейс спирається на систему спливаючих модальних вікон Bootstrap (рис. 2.9). Завдяки цьому користувач не залишає сторінку додатка.



Рисунок 2.9 – Модальне вікно для вибору ОС при створенні VM

Для підтримки актуальності даних в інтерфейсі наявні кнопки «Оновити список», щоб завантажити дані без необхідності перезавантажувати сторінку. Розроблений користувацький інтерфейс забезпечує повну кросплатформність (доступ з будь-якого пристрою через браузер), високу швидкість взаємодії та максимальний комфорт при адмініструванні локальної віртуальної інфраструктури.

Висновки до другого розділу

У другому розділі кваліфікаційної роботи виконано повний комплекс інженерного проєктування та розробки системної архітектури програмного комплексу для автоматизованого керування віртуальною інфраструктурою. За результатами проведеного дослідження та проєктування сформовано такі науково-практичні висновки.

1. Обґрунтовано та спроектовано триланкову (3-tier) архітектурну модель безкомпонентної інтеграції, яка об'єднує веб-інтерфейс представлення (MPA), серверний аналітичний шлюз на базі FastAPI та локально-сценарне PowerShell-ядро взаємодії з гіпервізором Oracle VM VirtualBox через утиліту VBoxManage. Відсутність важковагових сторонніх баз даних та агентських служб дозволила звести до мінімуму навантаження на хост-систему розробника.

2. Розроблено п'ятиетапний асинхронний алгоритм міжпроцесної взаємодії, який забезпечує наскрізну трансляцію дій користувача (від клієнтського Fetch-запиту) до CLI-інструкцій демона VBoxSVC, парсинг неструктурованого консольного виводу за допомогою регулярних виразів та його зворотне перетворення у стандартизований JSON-потік.

3. Проєктуванням архітектури успішно вирішено проблему виконання тривалих і ресурсомістких операцій відчуження інфраструктури (експорт у формат .ova). Шляхом інтеграції PowerShell-модуля з нативним Планувальником завдань для Windows реалізовано три гнучкі режими резервного копіювання (миттєве, одноразове відкладене та перманентне за розкладом), що дозволило перенести обчислювальне навантаження на автономні фонові процеси ОС і зберегти чутливість веб-інтерфейсу додатка.

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		42

4. Спроектовано атомарну модульну структуру backend-частини на базі п'яти ізольованих PowerShell-модулів (utils.psm1, vm.psm1, snapshot.psm1, backup.psm1, menu.psm1). Впровадження універсального обгорткового методу Invoke-VBox-Command та конструкцій Try-Catch забезпечило повну ізоляцію інженерних помилок ОС і гіпервізора, високу ремонтпридатність коду та уніфікацію відповідей для веб-сервера.

5. Розроблено чітку ресурсно-орієнтовану специфікацію REST API, яка структурує методи взаємодії (GET, POST, PATCH, DELETE) навколо трьох базових архітектурних шляхів: /api/vms (керування життєвим циклом, параметрами та знімками VM), /api/backups (планування, видалення завдань та тест-драйв копій на цілісність) та /api/host (збір системної телеметрії). Організовано строгу двосторонню трансляцію обміну даними за допомогою JSON-пакетів та Pydantic-моделей валідації.

6. Спроектовано ергономічний інтерфейс користувача за архітектурою багатосторінкового веб-додатка (MPA) із серверною генерацією сторінок через шаблонізатор Jinja2 та застосуванням чистого JavaScript (Vanilla JS) і модульної сітки Bootstrap. Реалізовано чітке кольорове кодування статусів VM (Running, Powered Off, Aborted) та станів валідності архівів (Валідний, Пошкоджений, Не перевірено та Невідомо), що забезпечує миттєву візуальну ідентифікацію здоров'я інфраструктури.

7. Інтегровано систему динамічних модальних вікон Bootstrap (для конфігурування RAM/CPU, вибору ОС та перегляду планувальника) та інструментарій SweetAlert2 для інтерактивної візуалізації фонових операцій. Створений механізм блокування інтерфейсу під час виконання тривалого тестування цілісності бекапів за допомогою функції Test-OvaBoot повністю унеможливорює відправку дублюючих запитів клієнтом і гарантує високу UX-якість розробленого програмного комплексу.

Сформована архітектурна модель, модульна декомпозиція та специфікація API є вичерпним інженерно-технічним базисом для переходу до етапу безпосереднього налаштування та тестування компонентів системи.

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		43

РОЗДІЛ 3. ПРОГРАМНА РЕАЛІЗАЦІЯ, НАЛАШТУВАННЯ ТА ТЕСТУВАННЯ СИСТЕМИ

3.1. Реалізація та демонстрація автономного консольного інтерфейсу

Важливою архітектурною перевагою розробленого програмного комплексу є його автономність та стійкість до збоїв. Попри те, що основна взаємодія користувача із системою відбувається через веб-інтерфейс багатосторінкового додатка, утиліти та модулі бізнес-логіки спроектовані так, щоб функціонувати як незалежне консольне середовище. Це реалізовано за допомогою головного виконуваного скрипта `vm-manager.ps1` та спеціалізованого модуля `menu.psm1`.

Консольний інтерфейс є критично необхідним інструментом для системного адміністратора у випадках: проведення регламентних або аварійних робіт на хост-машині, коли веб-сервер FastAPI тимчасово зупинено; необхідності швидкого локального виконання операцій без запуску браузера та мережевого стека.

3.1.1. Програмна реалізація та обробка UAC-контексту

На початку виконання скрипт `vm-manager.ps1` здійснює обов'язкову перевірку поточних привілеїв безпеки в ОС Windows (рис. 3.1). Оскільки операції взаємодії з Планувальником завдань вимагають розширеного контексту безпеки, утиліта використовує класи `.NET` для перевірки ролі адміністратора.

У разі виявлення обмежених прав користувача, скрипт реалізує інтерактивний алгоритм самопідвищення привілеїв через механізм UAC (User Account Control) за допомогою командлета `Start-Process`:

```
Start-Process powershell -ArgumentList "-NoProfile -ExecutionPolicy Bypass -File `"$PSCommandPath`"" -Verb RunAs -ErrorAction Stop
```

Прапор `-Verb RunAs` ініціює виклик операційної системи для підвищення токена безпеки процесу, а параметри `-NoProfile -ExecutionPolicy Bypass` гарантують, що системні політики обмеження виконання скриптів на конкретному ПК не заблокують роботу ядра автоматизації. Прапор `-ErrorAction` з параметром `Stop` використовуються для зупинки дії для обробки системних помилок через блок `Try-Catch`. Роботу алгоритму самопідвищення привілеїв можна побачити на рисунках 3.2-3.3.

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		44

```

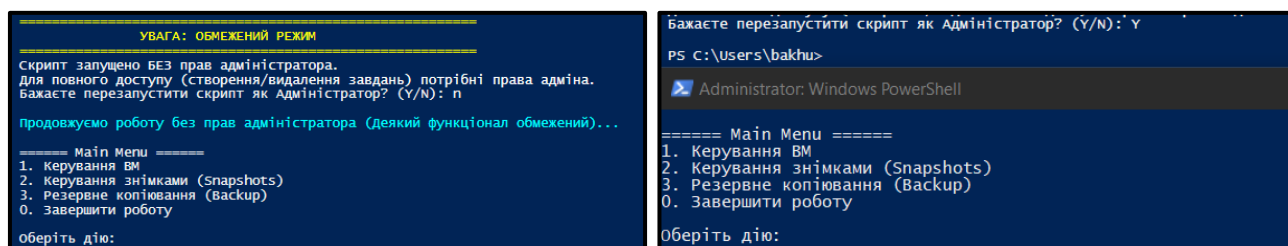
$isAdmin = ([Security.Principal.WindowsPrincipal][Security.Principal.WindowsIdentity]::GetCurrent()).IsInRole([Security.Principal.WindowsBuiltInRole]::Administrator)

if (-not $isAdmin) {
    Clear-Host
    Write-Host "===== УВАГА: ОБМЕЖЕНИЙ РЕЖИМ =====" -ForegroundColor Yellow
    Write-Host "Скрипт запущено БЕЗ прав адміністратора." -ForegroundColor Yellow
    Write-Host "Для повного доступу (створення/видалення завдань) потрібні права адміна." -ForegroundColor Yellow
    $choice = Read-Host "Бажаєте перезапустити скрипт як Адміністратор? (Y/N)"

    if ($choice -match "^[YyДдТтГг]$") {
        try {
            Start-Process powershell -ArgumentList "-NoProfile -ExecutionPolicy Bypass -File '$PSCommandPath'" -Verb RunAs -ErrorAction Stop
        } catch {
            Write-Host "`n[UAC] відмовлено в доступі. продовжуємо роботу в обмеженому режимі..." -ForegroundColor Red
            $IsReadOnlyMode = $true
            Start-Sleep -Seconds 2
        }
    } else {
        Write-Host "Продовжуємо роботу без прав адміністратора (деякий функціонал обмежений)..." -ForegroundColor Cyan
        $IsReadOnlyMode = $true
        Start-Sleep -Seconds 2
    }
}

```

Рисунок 3.1 – Перевірка роботи з правами адміністратора



Рисунки 3.2-3.3 – Виконання самопідвищення привілеїв залежно від вибору

Логіка головного меню та підменю побудована на базі вкладених циклів `do-while` та операторів вибору `switch`. Модуль `menu.psm1` надає декларативні функції `Show-MainMenu`, `Show-VMMenu`, `Show-MenuModify`, `Show-SnapshotMenu` та `Show-BackupMenu`, які транслюють у консоль текстові структури інтерфейсу (рис. 3.4).

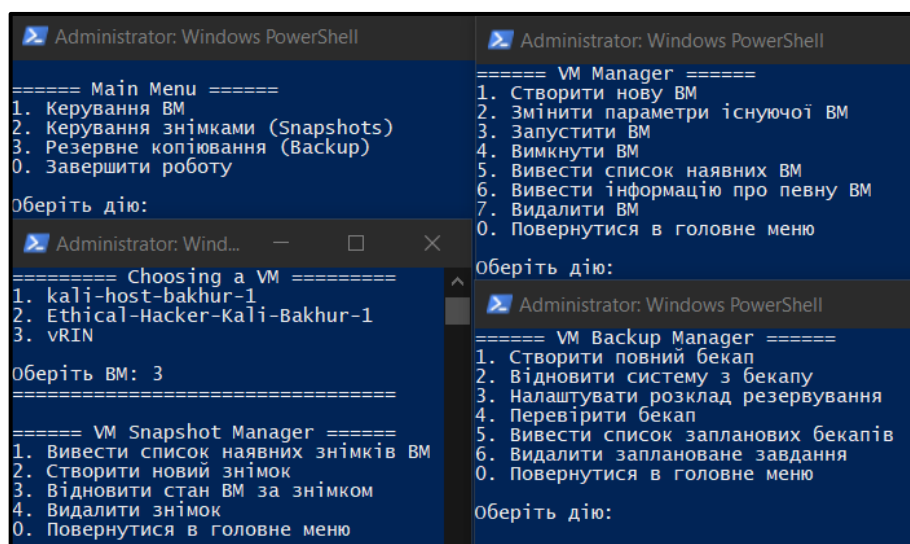


Рисунок 3.4 – Відображення меню вибору дій

Центральне місце в обробці дій користувача посідає розроблена уніфікована функція Get-UserSelection, яка абстрагує рутинні операції вибору об'єктів з динамічних масивів (рис. 3.5).

```
function Get-UserSelection {
    param(
        [Parameter(Mandatory=$true)]
        [AllowEmptyCollection()]
        $Collection,
        [string]$Message = "Оберіть номер елемента",
        [string]$EmptyMessage = "Список порожній"
    )

    if (-not $Collection -or $Collection.Count -eq 0) {
        Write-Host $EmptyMessage
        return $null
    }

    for ($i = 0; $i -lt $Collection.Count; $i++) {
        $displayName = $Collection[$i].Name -split ' ' | where-Object { $_ } | select-object -First 1
        if (-not $displayName) { $displayName = $Collection[$i] }
        Write-Host "$($i + 1). $displayName"
    }

    $input = (Read-Host "`n$Message").Trim()

    if ([int]::TryParse($input, [ref]$choice) {
        if ($choice -ge 1 -and $choice -le $Collection.Count) {
            return $Collection[$choice - 1]
        }
    }

    Write-Host "Неправильний вибір."
    return $null
}
```

Рисунок 3.5 – Синтаксис функції Get-UserSelection

Такий підхід повністю захищає сценарне ядро від критичних збоїв: якщо користувач введе нечислове значення, випадковий символ чи індекс, що виходить за межі наявного масиву VM чи знімків, функція перехопить помилку, виведе повідомлення «Неправильний вибір» і поверне контроль меню без падіння скрипта.

Також простою, але важливою для уникнення повторювання блоків коду є функція Out-Result, яка отримує результат виконання певної дії та виводить повідомлення про успіх чи помилку на екран консолі (рис. 3.6).

```
function Out-Result {
    param(
        [Parameter(Mandatory=$true)]
        $Result
    )

    if ($Result.Success) {
        Write-Host "`n$($Result.Message)"
    } else {
        Write-Host "`nПомилка: $($Result.Message)"
    }
}
```

Рисунок 3.6 – Синтаксис функції Out-Result

3.1.2. Функціональні підсистеми консольного інтерфейсу

Скрипт vm-manager.ps1 послідовно імпортує всі п'ять розроблених модулів системи та розподіляє логіку на три великі функціональні блоки (гілки):

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		46

1. Менеджер ВМ. Зв'язує інтерфейс із модулем `vm.psm1`. Дозволяє інтерактивно створити нову машину (із запитом через Read-Host імені, RAM, обсягу диска та ISO-образу), запустити ВМ у фоновому режимі (headless) чи з графічним інтерфейсом, виконати безпечне чи примусове вимкнення, змінити апаратні параметри заліза (CPU, RAM, VRAM, тип мережі, вставити/витягнути диск) або повністю видалити об'єкт із реєстрації гіпервізора.

2. Менеджер знімків стану ВМ. Інтегрується з `snapshot.psm1`. Користувач спочатку вибирає ВМ для роботи за допомогою функції `Select-VM`, після чого система виводить меню наявних дій.

3. Менеджер бекапів та планувальника. Забезпечує взаємодію з `backup.psm1`. Дозволяє виконати негайний експорт у файл `.ova`, перевірити цілісність наявних копій у директорії, вивести таблицю запланованих завдань із Task Scheduler за допомогою форматування `Format-Table -AutoSize` (із відображенням розкладу, часу минулого/наступного запусків та кодів результатів) або видалити наявне завдання.

3.1.3. Демонстрація функціонування консольного інтерфейсу

Для підтвердження працездатності та валідації сценарного ядра проведено серію тестових випробувань в автономному CLI-режимі.

Тест-кейс 1. Виведення списку ВМ та взаємодія з об'єктами.

При переході у підменю «Керування ВМ» (Пункт 1) та виклику операції «Вивести список наявних ВМ», ядро звертається до функції `Get-VMs` з модуля `utils.psm1`. Результат виконання утиліти `VBoxManage list vms` парситься і виводиться на екран у вигляді структурованої таблиці із поточними статусами машин у реальному часі (рис. 3.7).

```

Administrator: Windows PowerShell

===== VM Manager =====
1. Створити нову ВМ
2. Змінити параметри існуючої ВМ
3. Запустити ВМ
4. Вимкнути ВМ
5. Вивести список наявних ВМ
6. Вивести інформацію про певну ВМ
7. Видалити ВМ
0. Повернутися в головне меню

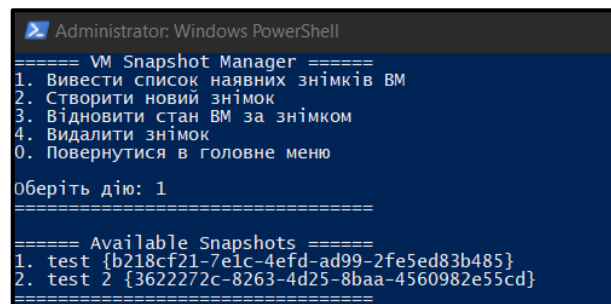
Оберіть дію: 5
=====

===== List of Available VMs =====
1. kali-host-bakhur-1 {30f61e0e-374a-4910-a0b9-b2a26519fa68} (Aborted)
2. Ethical-Hacker-Kali-Bakhur-1 {aaee9ad2-7a1e-425e-b288-9ff4daffe032} (Powered Off)
3. vRIN {00ff59b7-6f40-4b08-86a3-d400d1383cec} (Powered Off)
=====
  
```

Рисунок 3.7 – Список доступних ВМ

Тест-кейс 2. Виведення списку наявних знімків ВМ.

При переході у підменю «Керування знімками (Snapshots)» виводиться на екран нумерований список ВМ для вибору. Здійснивши вибір, бачимо меню, де викликаємо операцію «Вивести список наявних знімків ВМ». Ядро звертається до функції Get-VMSnapshotsList з модуля utils.psm1. Результат виконання утиліти VBoxManage snapshot "\$VMName" list парситься і виводиться на екран у вигляді структурованої таблиці (рис. 3.8).



```
Administrator: Windows PowerShell

===== VM Snapshot Manager =====
1. Вивести список наявних знімків ВМ
2. Створити новий знімок
3. Відновити стан ВМ за знімком
4. Видалити знімок
0. Повернутися в головне меню

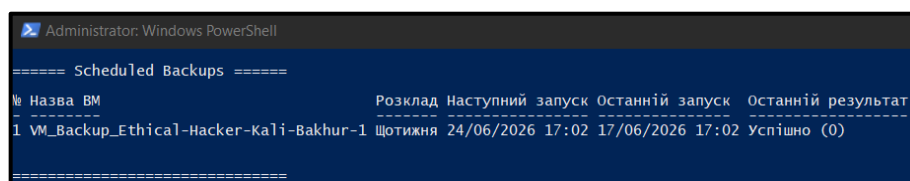
Оберіть дію: 1

===== Available Snapshots =====
1. test {b218cf21-7e1c-4efd-ad99-2fe5ed83b485}
2. test 2 {3622272c-8263-4d25-8baa-4560982e55cd}
```

Рисунок 3.8 – Список доступних знімків ВМ ‘vRIN’

Тест-кейс 3. Перегляд стану запланованих завдань.

При переході в меню резервного копіювання та виборі пункту перегляду запланованих завдань, скрипт через COM-об'єкт Schedule.Service опитує ОС Windows. На екран виводиться детальний звіт про роботу планувальника (рис. 3.9), де складні системні типи тригерів автоматично перекладені для зручності адміністратора на зрозумілу мову (наприклад, «Щотижня», «Одноразово», «При запуску операційної системи»).



```
Administrator: Windows PowerShell

===== Scheduled Backups =====

№ Назва ВМ Розклад Наступний запуск Останній запуск Останній результат
-----
1 VM_Backup_Ethical-Hacker-Kali-Bakhur-1 Щотижня 24/06/2026 17:02 17/06/2026 17:02 Успішно (0)
```

Рисунок 3.9 – Список запланованих бекапів ВМ

Таким чином, автономний консольний інтерфейс надає повний, відмовостійкий спектр інструментів для контролю інфраструктури, дублюючи логіку веб-додатка, і доводить високу стабільність та модульність усього розробленого програмного комплексу на рівні операційної системи хоста.

3.2. Розробка серверного шлюзу інтеграції на базі програми main.py

Центральним сполучним елементом, що забезпечує трансляцію дій користувача з веб-інтерфейсу (фронтенду) до низькорівневих модулів автоматизації хост-системи, є серверний інтеграційний шлюз, реалізований у файлі main.py. Веб-сервер побудовано на базі сучасного асинхронного Python-фреймворку FastAPI, що дозволяє обробляти HTTP-запити з мінімальними накладними витратами на процесорний час та оперативну пам'ять хоста.

Відповідно до архітектурної концепції Multi-Page Application (MPA), у файлі main.py реалізовано дві ключові ролі.

1. Рендеринг HTML-сторінок (серверний шаблонізатор). Використовується інтегрований інструментарій Jinja2Templates, який динамічно формує та наповнює інтерфейс даними на стороні сервера перед відправкою клієнту.

2. REST API Ендпоінти (JSON Gateway). Приймає структуровані асинхронні GET, POST, PATCH та DELETE запити від JavaScript-функцій інтерфейсу та повертає уніфіковані об'єкти стану системних операцій.

3.2.1. Ініціалізація додатка та механізм міжпроцесної взаємодії

Для початку роботи додатка використовуються наступні функції:

app = FastAPI() – ініціалізує головний об'єкт веб-додатка FastAPI. Через цей об'єкт app надалі реєструються маршрути (routes) та обробники запитів.

templates = Jinja2Templates(directory="templates") – створює об'єкт для керування HTML-шаблонами за допомогою двигуна Jinja2. Вказує фреймворку, що всі шаблони сайту (HTML-файли) зберігаються в папці з назвою templates у корені проєкту.

app.mount("/static", StaticFiles(directory="static"), name="static") – монтує окрему директорію для обслуговування статичного контенту.

Найважливішим інженерним рішенням у файлі main.py є розробка функції run_ps(cmd, module="utils"), яка реалізує міжпроцесну взаємодію між середовищем виконання Python та командною оболонкою Windows PowerShell Core. Вона змушує PowerShell працювати в кодуванні UTF-8, що є критичним для коректного відображення кирилиці.

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		49

Функція повністю абстрагує виклик сценарного ядра і має наступну програмну архітектуру (рис. 3.10).

```

async def run_ps(cmd, module="utils"):
    utf8_fix = '$OutputEncoding = [Console]::OutputEncoding = [System.Text.Encoding]::UTF8;'
    import_cmd = f'Import-Module ./modules/{module}.psm1;'
    full_cmd = f'{utf8_fix} {import_cmd} {cmd} | ConvertTo-Json'

    try:
        proc = await asyncio.create_subprocess_exec(
            program="powershell",
            args="-Command", full_cmd,
            stdout=asyncio.subprocess.PIPE,
            stderr=asyncio.subprocess.PIPE
        )

        stdout, stderr = await asyncio.wait_for(proc.communicate(), timeout=300)

        stdout_text = stdout.decode(encoding='utf-8', errors='ignore')
        stderr_text = stderr.decode(encoding='utf-8', errors='ignore')

        if stderr_text.strip():
            print(f"PowerShell Error: {stderr_text}")

        if not stdout_text.strip():
            return None

        return json.loads(stdout_text)
    except asyncio.TimeoutError:
        return {"Success": False, "Message": "Перевищено ліміт часу очікування операції (timeout)."}
    except Exception as e:
        return {"Success": False, "Message": f"Помилка виклику підпроцесу: {str(e)}"}

```

Рисунок 3.10 – Синтаксис функції run_ps

Параметр module автоматично підставляє назву цільового компонента (utils, vm, snapshot, backup), ізолюючи контекст виконання кожної команди.

Наприкінці виразу примусово викликається командлет | ConvertTo-Json, що змушує PowerShell транслювати строго типізовані об'єкти [PSCustomObject] у текстовий JSON-потік, який легко перетворюється в Python за допомогою json.loads().

Параметр timeout=300 обмежує час виконання фонових процесів до 5 хвилин. Це превентивний захід безпеки, який захищає веб-сервер від блокування потоків у випадку, якщо низькорівнева утиліта VBoxManage зависне під час тривалого експорту .ova архіву або у разі виникнення взаємного блокування процесів (deadlock).

3.2.2. Управління системними привілеями (UAC) та контекстом безпеки

Оскільки розроблена система автоматизації взаємодіє із захищеними компонентами ОС Windows (реєстрація фонових служб, створення завдань у системному каталозі Root Планувальника завдань Windows), сервер FastAPI вимагає найвищого рівня доступу.

Для автоматизації процесу розгортання в main.py інтегровано механізм взаємодії з Win32 API через бібліотеку ctypes. При старті програми функція is_admin()

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		50

опитує операційну систему (рис. 3.11). Якщо скрипт запущено у звичайному контексті користувача, додаток перехоплює виконання та за допомогою функції ShellExecuteW із дієсловом runas ініціює запит на примусовий перезапуск самого себе у новому процесі з підвищеним токеном адміністратора.

```
def is_admin():
    try:
        return ctypes.windll.shell32.IsUserAnAdmin()
    except:
        return False

if __name__ == "__main__":
    if not is_admin():
        if "--no-relaunch" not in sys.argv:
            script_path = os.path.abspath(sys.argv[0])
            working_dir = os.path.dirname(script_path)

            try:
                result = ctypes.windll.shell32.ShellExecuteW(
                    None, "runas", sys.executable,
                    f'"{script_path}" --no-relaunch', working_dir, 1
                )

                if result > 32:
                    sys.exit()
            except Exception as e:
                print(f"Запит на права адміністратора відхилено або стався збій: {e}")

    if is_admin():
        print("Скрипт запущено з правами адміністратора!")
    else:
        print("УВАГА: Скрипт запущено без прав адміністратора. Доступ до планувальника задач Windows буде обмежено.")
```

Рисунок 3.11 – Синтаксис функції is_admin та реалізація UAC

Прапор --no-relaunch виступає запобіжником від утворення нескінченного циклу рекурсивних перезапусків у випадку, якщо користувач відхилить запит UAC у діалоговому вікні операційної системи.

3.2.3. Алгоритм аварійного примусового завершення процесів

Під час тривалих фонових операцій (наприклад, верифікація цілісності віртуального диска або експорт великих ВМ) існує ймовірність некоректної поведінки гіпервізора або випадкових дій адміністратора. Для стабілізації системи в main.py розроблено функцію екстреного примусового очищення робочого середовища, яка викликається при запуску main.py (рис. 3.12).

```
def cleanup_zombie_processes():
    print("Очищення залишкових процесів...")
    processes_to_kill = ["VBoxManage.exe", "powershell.exe"]

    for proc in processes_to_kill:
        try:
            subprocess.run(
                args=["taskkill", "/F", "/IM", proc, "/T"],
                capture_output=True, check=False
            )
        except Exception as e:
            print(f"Не вдалося завершити {proc}: {e}")
```

Рисунок 3.12 – Синтаксис функції cleanup_zombie_processes

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		51

Функція викликає утиліту ОС Windows taskkill із системними аргументами:

/F – примусове завершення без очікування відповіді від процесу;

/IM – вибір цільових об'єктів за назвою їх виконуваного образу;

/T – рекурсивне знищення всього дерева дочірніх процесів, що гарантує повне очищення фонових екземплярів командного рядка PowerShell та низькорівневих демонів VBoxManage, які були породжені сервером FastAPI.

Таким чином, розроблений файл main.py є надійним, захищеним та високопродуктивним інтеграційним шлюзом. Він повністю ізолює клієнтську сторону від специфіки операційної системи, забезпечує безпечне підвищення привілеїв UAC та гарантує передбачувану асинхронну обробку даних.

3.3. Програмна реалізація та тестування функцій керування VM

Робочий простір керування життєвим циклом та апаратними конфігураціями обчислювальних вузлів зосереджений на сторінці vm.html. При переході на головну сторінку FastAPI за допомогою шаблонізатора Jinja2Templates повертає базову HTML-структуру (каркас сторінки). Після цього на стороні клієнта спрацьовує JS-скрипт, який через асинхронний API-запит звертається до бекенду. FastAPI викликає функцію Get-VMs модуля utils.psm1, отримує масив об'єктів наявних VM і повертає їх у форматі JSON, після чого браузер динамічно рендерить рядки таблиці.

Для виконання інженерних операцій (запуск, зупинка, модифікація параметрів) без повного перезавантаження сторінки, на фронтенді застосовано асинхронну технологію AJAX (за допомогою JavaScript API fetch()), яка відправляє JSON-пакети на відповідні REST API ендпоінти сервера.

3.3.1. Програмна реалізація асинхронного обробника

Функція handleAction є універсальним асинхронним обробником (хелпером) для відправки запитів на сервер (FastAPI) та візуалізації результатів користувачеві за допомогою бібліотеки SweetAlert2.

Вона повністю бере на себе рутинну роботу: показує індикатор завантаження, відправляє дані, обробляє відповіді чи помилки мережі та автоматично оновлює інтерфейс.

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		52

Вхідні параметри функції:

url – адреса ендпоінту на сервері (наприклад, /api/vms/{vm_name}/start).

method – HTTP-метод запиту; за замовчуванням це POST, але можна передати POST, DELETE чи PATCH.

payload – дані (об'єкт), які потрібно відправити на сервер у тілі JSON; за замовчуванням null (якщо дані не потрібні).

Покроковий алгоритм роботи є наступним.

1. Ініціалізація та візуальний лоадер. Функція скидає глобальний прапорець помилки (isError = false), який використовується при успішному запуску VM для виведення інформації про дистанційний доступ. Миттєво викликається Swal.fire з індикатором завантаження. Вікно блокує інтерфейс (allowOutsideClick: false), щоб користувач випадково не натиснув кнопку двічі, поки виконується операція на сервері (наприклад, запускається VM) (рис. 3.13).

```
async function handleAction(url, method = 'POST', payload = null) {
  isError = false;

  Swal.fire({
    title: 'Зачекайте...',
    text: 'Виконуємо операцію',
    allowOutsideClick: false,
    didOpen: () => {
      Swal.showLoading();
    }
  });
};
```

Рисунок 3.13 – Синтаксис функції handleAction (фрагмент)

2. Конфігурація та відправка запиту. Формується об'єкт options для нативного JS-методу fetch. Вказується метод та заголовок. Якщо є дані для відправки, об'єкт трансформується у JSON-рядок через JSON.stringify() і додається у тіло запиту. Рядок const response = await fetch(url, options); асинхронно відправляє запит на сервер та чекає на відповідь, не блокуючи головний потік браузера (рис. 3.14).

```
try {
  const options = {
    method: method,
    headers: { 'Content-Type': 'application/json' }
  };

  if (payload){
    options.body = JSON.stringify(payload)
  }

  const response = await fetch(url, options);
}
```

Рисунок 3.14 – Синтаксис функції handleAction (фрагмент)

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		53

3. Перевірка HTTP-статусу та парсинг. Якщо сервер повернув критичну помилку (наприклад, 500 Internal Server Error), спрацьовує перевірка `if (!response.ok)`, яка примусово викидає виняток (`throw new Error`). Якщо статус успішний, відповідь сервера парситься з JSON у звичайний JS-об'єкт `data`. У отриманому повідомленні (`data.Message`) всі символи переносу рядка `\n` замінюються на HTML-тег `<br>`, щоб текст красиво відображався у модальному вікні (рис. 3.15).

```
if (!response.ok) {
  throw new Error('Помилка сервера: ${response.status}');
}

const data = await response.json();

data.Message = data.Message.replace(/\n/g, '<br>');
```

Рисунок 3.15 – Синтаксис функції `handleAction` (фрагмент)

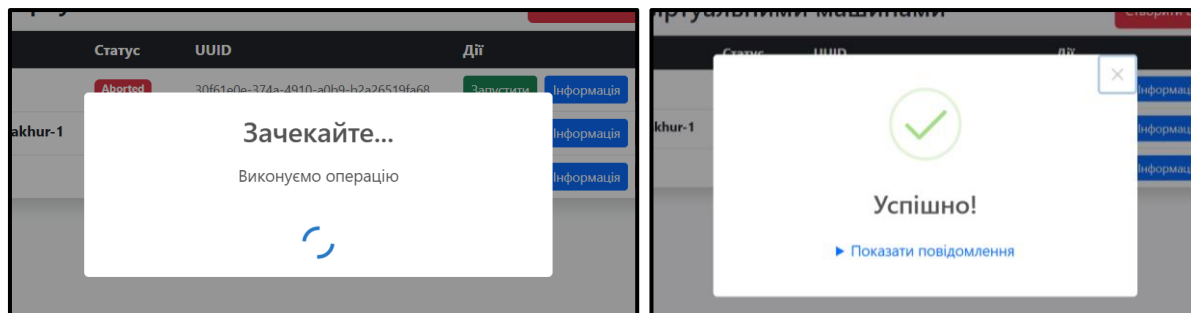
4. Обробка логічного результату від сервера. Сервер повертає JSON-структуру, яка містить прапорець успішності операції (`Success: true/false`). Залежно від успішності з'являється вікно успіху чи помилки з текстом відповіді, обгорнутим у допоміжну функцію `getMessageHtml()`. Кнопка підтвердження прихована, але є хрестик для закриття. При успіху викликається `await loadVMs()`, що змушує таблицю машин на сторінці миттєво оновитися та показати актуальний статус (рис. 3.16).

```
if (data.Success) {
  data.Message = data.Message || 'Дію виконано';
  Swal.fire({
    icon: 'success', title: 'Успішно!',
    html: getMessageHtml("повідомлення", data.Message),
    showConfirmButton: false, showCloseButton: true
  });
  await loadVMs();
} else {
  data.Message = data.Message || 'Щось пішло не так';
  Swal.fire({
    icon: 'error', title: 'Помилка виконання',
    html: getMessageHtml("помилку", data.Message),
    showConfirmButton: false, showCloseButton: true
  });
  isError = true;
}
} catch (error) {
  let text = 'Не вдалося з\'єднатися з сервером. Перевірте, чи запущено Python.';
  Swal.fire({
    icon: 'error', title: 'Помилка мережі',
    html: getMessageHtml("помилку", text),
    showConfirmButton: false, showCloseButton: true
  });
  console.error('Action error:', error);
  isError = true;
}
}
```

Рисунок 3.16 – Синтаксис функції `handleAction` (фрагмент)

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		54

Кожен елемент керування на сторінці працює за одним принципом: він викликає відповідну функцію, яка виконує зазначені дії згідно з операцією і завершується викликом `handleAction`, що надсилає асинхронний запит до сервера та виводить модальне вікно з результатом (рис. 3.17-3.18).



Рисунки 3.17-3.18 – Модальні вікна про виконання та успіх операції

3.3.2. Тестування та демонстрація роботи підсистеми керування VM

Для підтвердження коректності реалізації інтерфейсу та стабільності зв'язку компонентів системи було проведено серію функціональних тестів.

Тест-кейс 1. Візуалізація станів інфраструктури (Головна сторінка).

При переході на вкладку «Керування віртуальними машинами» користувачу демонструється інтерактивна таблиця (рис. 3.19). Система динамічно зчитує поточний стан ядра гіпервізора та застосовує колірне кодування бейджів для швидкої ідентифікації архітектури.

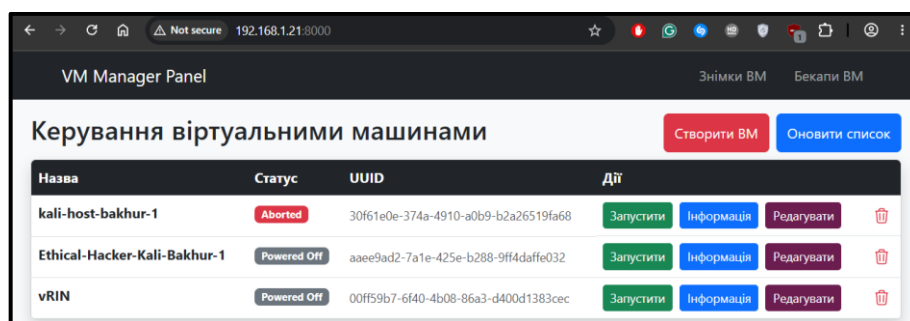


Рисунок 3.19 – Графічний інтерфейс сторінки керування VM

Машина у статусі Running підсвічуються зеленим маркером, а функціональні кнопки «Редагувати» та «Видалити (Корзина)» для них автоматично блокуються клієнтським скриптом (стають неактивними), що запобігає руйнуванню конфігураційних файлів запущеної ОС. Кнопка «Запустити» змінюється на «Вимкнути»

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		55

Тест-кейс 2. Запуск віртуальної машини.

Під час запуску ВМ перевіряється, чи достатньо вільних ресурсів. Якщо не вистачає обсягу вільної оперативної пам'яті чи дискового простору для ВМ, а також виділена максимальна кількість процесорів, перед виведенням модального вікна для вибору режиму візуалізації з'явиться попередження про недостатній обсяг ресурсів (рис. 3.20). Користувач може продовжити або скасувати операцію запуску.

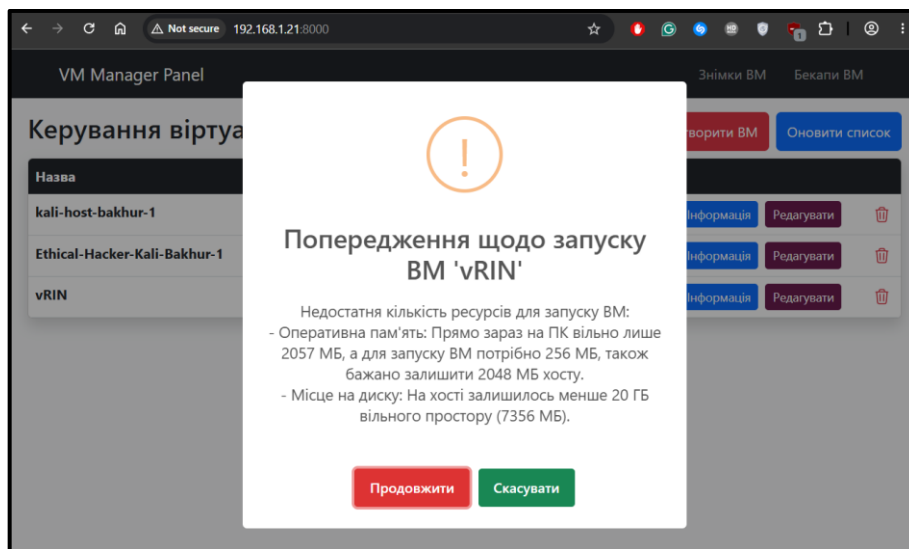


Рисунок 3.20 – Модальне вікно при недостатньому обсязі вільних ресурсів

Віртуальна машина може бути запущена з графічним інтерфейсом або у фоновому режимі (рис. 3.21). Користувач може працювати на ВМ локально з машини-хоста або дистанційно через локальну мережу (рис. 3.22). Якщо вибрано «З машини-хоста», машині присвоюється IP-адреса 127.0.0.1, що не дозволяє підключитися до неї через мережу (VRDP); при «Через мережу» – IP-адреса 0.0.0.0, яка дозволяє підключення за допомогою IP-адреси хоста та виділеного порту в діапазоні 5000-5999.

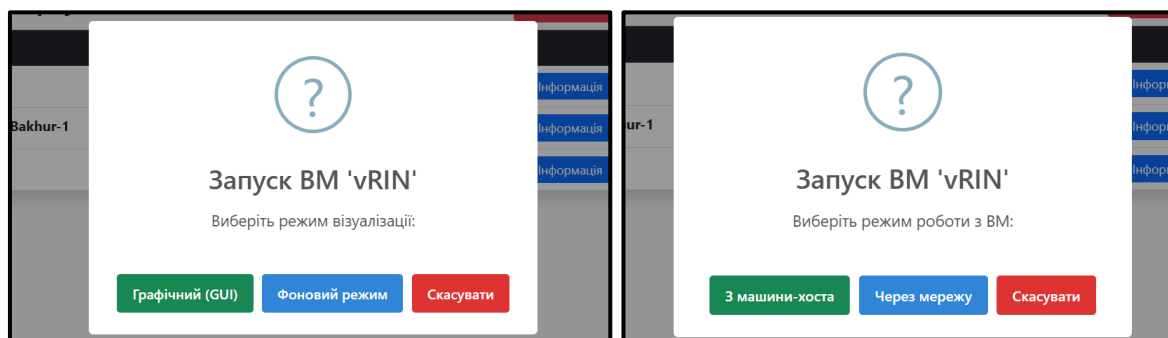


Рисунок 3.21-3.22 – Модальні вікна вибору режимів візуалізації та роботи

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		56

ВМ 'vRIN' успішно запущено, кнопки «Редагувати» та «Видалити» заблоковані, а «Запустити» змінено на «Вимкнути» (рис. 3.23). Після повідомлення про успіх з'являється вікно з інформацією про дистанційний доступ (рис. 3.24).

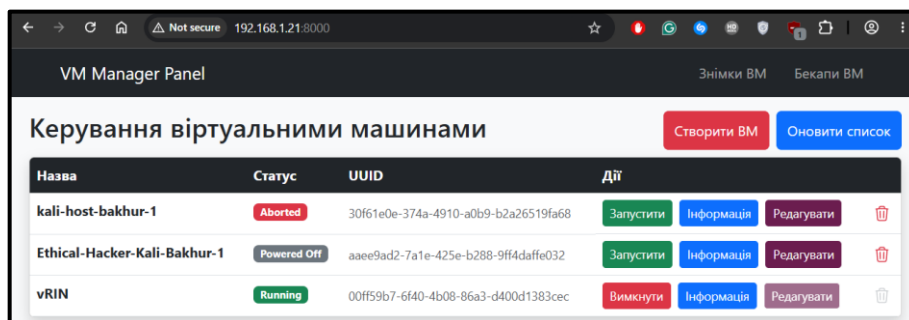


Рисунок 3.23 – Змінено статус ВМ

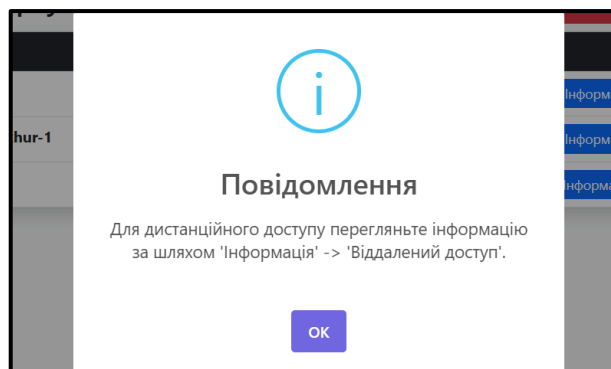
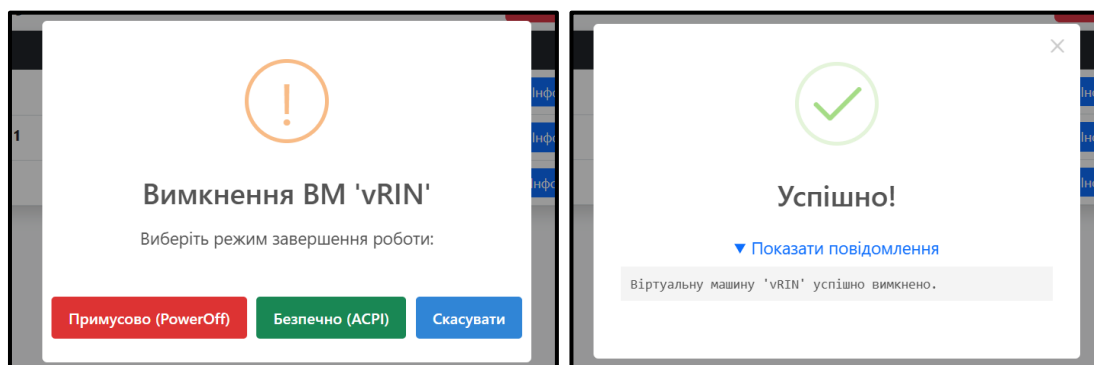


Рисунок 3.24 – Повідомлення про дистанційний доступ

Тест-кейс 3. Вимкнення віртуальної машини.

При натисненні на кнопку "Вимкнути" з'являється модальне вікно вибору режиму завершення роботи (рис. 3.25): примусово (моментально) чи безпечно (подається сигнал, щоб всі процеси закінчили свою роботу та зберегли файли). Після виконання дії виводиться вікно з результатом (рис. 3.26).



Рисунки 3.25-3.26 – Вимкнення віртуальної машини

Тест-кейс 4. Створення нової віртуальної машини.

При створенні нової VM потрібно заповнити ряд полів (рис. 3.27, 3.29-3.30), у кожного з яких є перевірка (рис. 3.28), тому не вдасться залишити поле порожнім або ввести обсяг пам'яті більший за можливий на хост-машині.

Рисунки 3.27-3.28 – Введення назви VM та спроба залишити поле порожнім

Рисунки 3.29-3.30 – Вибір ОС та виділення пам'яті при створенні VM

Вибір ISO-файлу для підключення відбувається через циклічне модальне вікно, яке з'являтиметься, поки не буде вибрано файл або натиснуто «Скасувати»: спочатку потрібно вибрати диск (рис. 3.31), далі з переліку вибрати потрібний файл для підключення або каталог для переходу (рис. 3.32-3.33).

Рисунок 3.31-3.32 – Вибір диску та каталогу для переходу

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				58
Змн.	Арк.	№ докум.	Підпис	Дата		

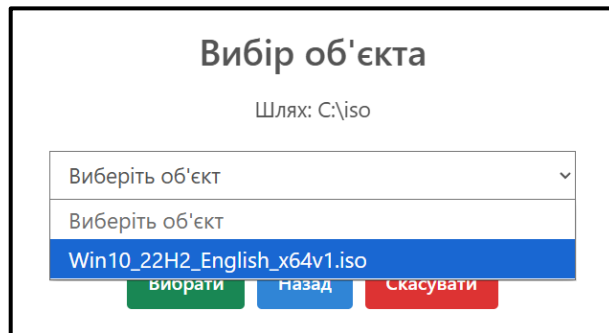


Рисунок 3.33 – Вибір файлу для підключення

Коли всі параметри задані, потрібно підтвердити або скасувати створення віртуальної машини із заданими параметрами (рис. 3.34). Після підтвердження бачимо вікно з результатом (рис. 3.35).

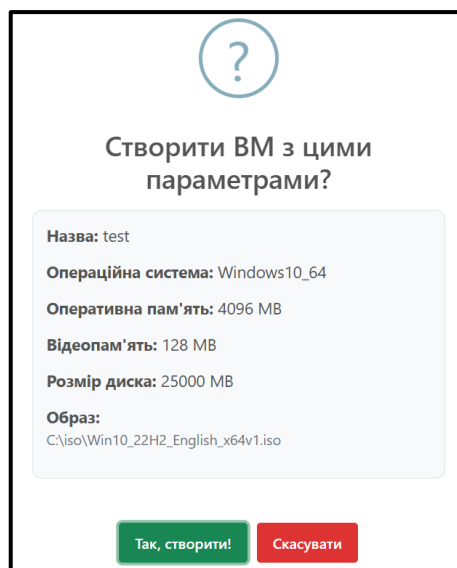


Рисунок 3.34 – Модальне вікно щодо

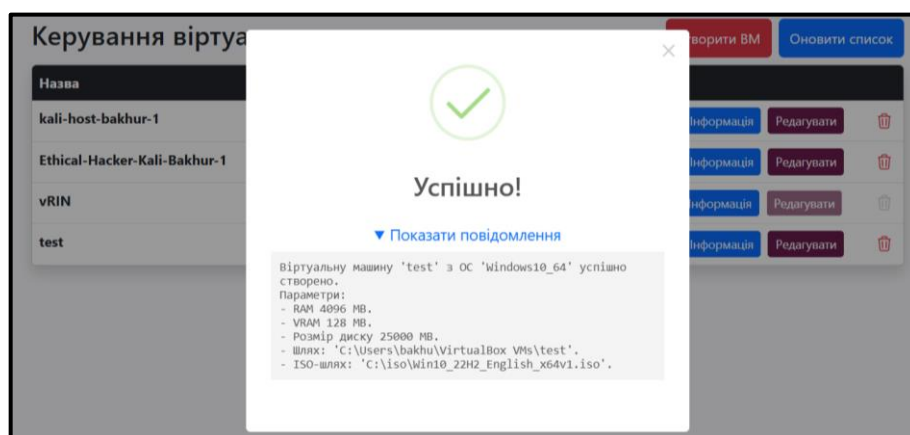
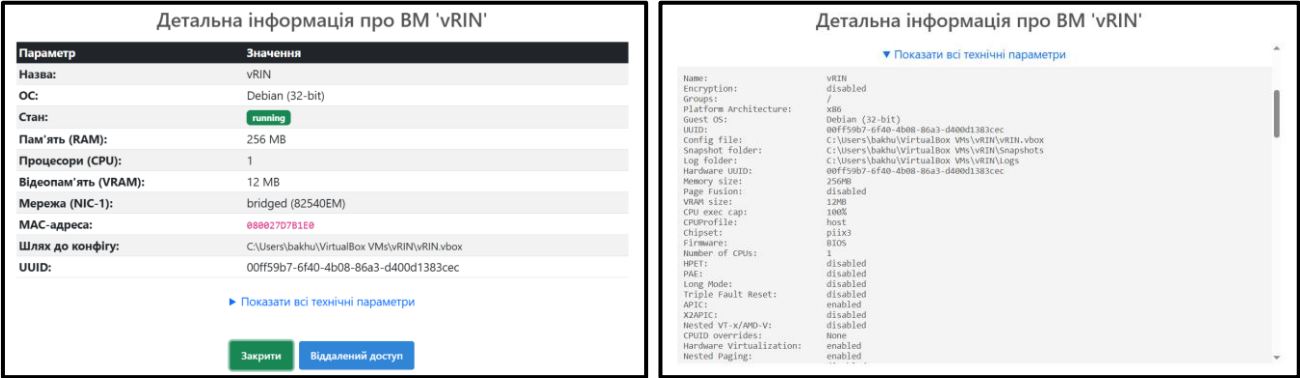


Рисунок 3.35 – Повідомлення про успішне створення ВМ

Тест-кейс 5. Виведення інформації про ВМ.

Кнопка «Інформація» відкриває модальне вікно з таблицею параметрів ВМ (коротко) (рис. 3.36). Щоб переглянути повну інформацію про ВМ, потрібно натиснути на «Показати всі технічні параметри», де можна побачити повний вивід команди VBoxManage showvminfo '{name}' (рис. 3.37). При ввімкненій ВМ поруч з кнопкою «Закрити» з'являється кнопка «Віддалений доступ», яка при натисканні виводить вікно (рис. 3.38), де можна переглянути інформацію про підключення локально чи через мережу (VRDP, RDP, VNC чи SSH).



Рисунки 3.36-3.37 – Коротка та повна інформація про ВМ

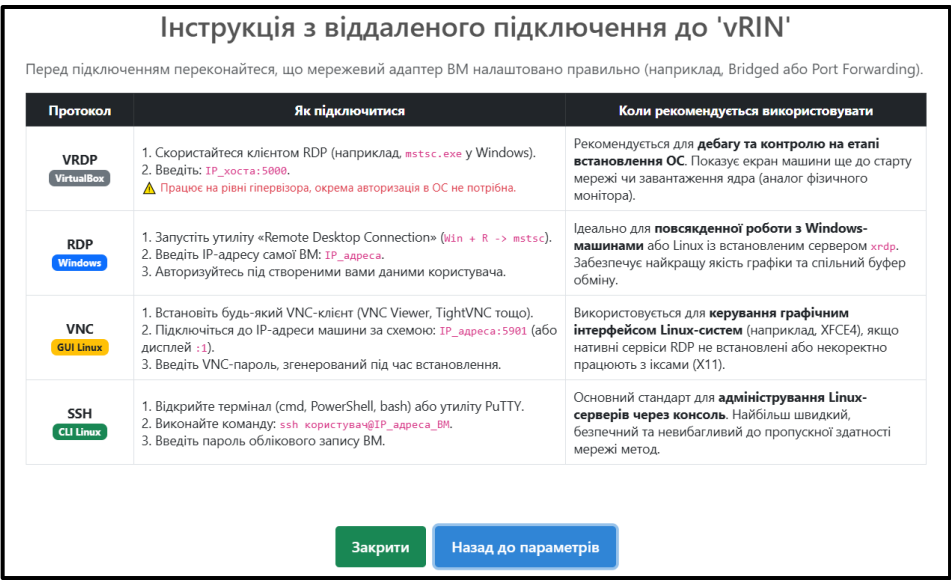


Рисунок 3.38 – Інформація про віддалене підключення

Тест-кейс 6. Редагування віртуальної машини.

Для редагування ВМ треба натиснути на «Редагувати», коли ВМ вимкнена, вибрати потрібне поле та заповнити відповідні поля (рис. 3.39-3.57).

Детальна інформація про ВМ 'test'

Параметр	Значення
Назва:	test
ОС:	Windows 10 (64-bit)
Стан:	poweroff
Пам'ять (RAM):	4096 MB
Процесори (CPU):	1
Відеопам'ять (VRAM):	128 MB
Мережа (NIC-1):	nat (82540EM)
MAC-адреса:	080027F8FA68
Шлях до конфігу:	C:\Users\bakhu\VirtualBox VMs\test\test.vbox
UUID:	eafd91e7-e79f-48f5-8851-749f986f2ce0

[▶ Показати всі технічні параметри](#)

Закрити

Рисунок 3.39. – Параметри ВМ до редагування

Редагування ВМ 'test'

Виберіть параметр для зміни:

Виберіть параметр

Виберіть параметр

Назва ВМ

Тип ОС

Кількість CPU

Оперативна пам'ять (MB)

Відеопам'ять (MB)

Режим підключення (NAT/Bridged)

Тип адаптера (Virtio/Intel)

DVD-привід

Рисунок 3.40 – Перелік параметрів, які можна змінити

Зміна назви ВМ 'test'

Введіть нову назву ВМ:

test (changed)

Змінити

Скасувати

Рисунок 3.41 – Зміна назви ВМ

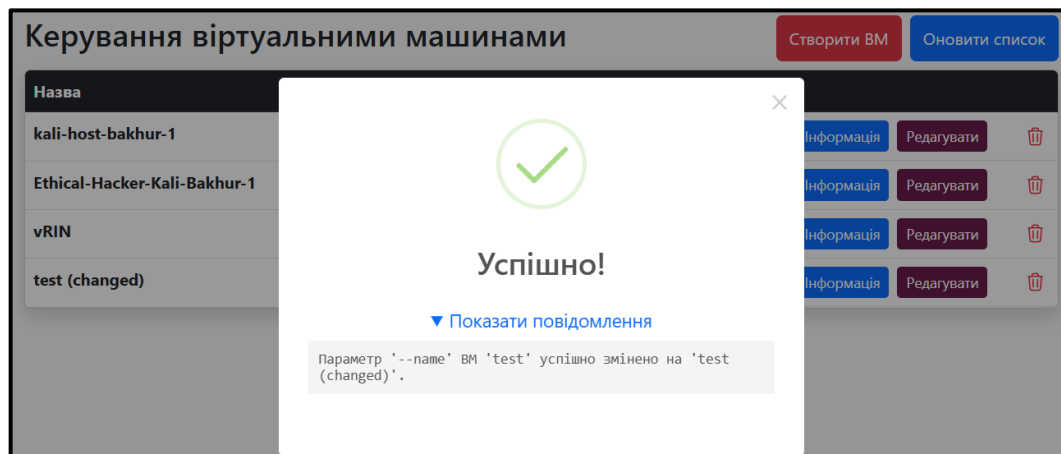


Рисунок 3.42 – Успішно змінено назву

Виберіть ОС

Виберіть операційну систему:

- ☐ Ubuntu (64-bit)
- ☐ Debian (64-bit)
- ☐ Fedora (64-bit)
- ☐ Windows 10 (64-bit)
- ☒ Windows 11 (64-bit)
- ☐ Other/Unknown (64-bit)

Вибрати
Скасувати

Рисунок 3.43 – Зміна операційної системи VM

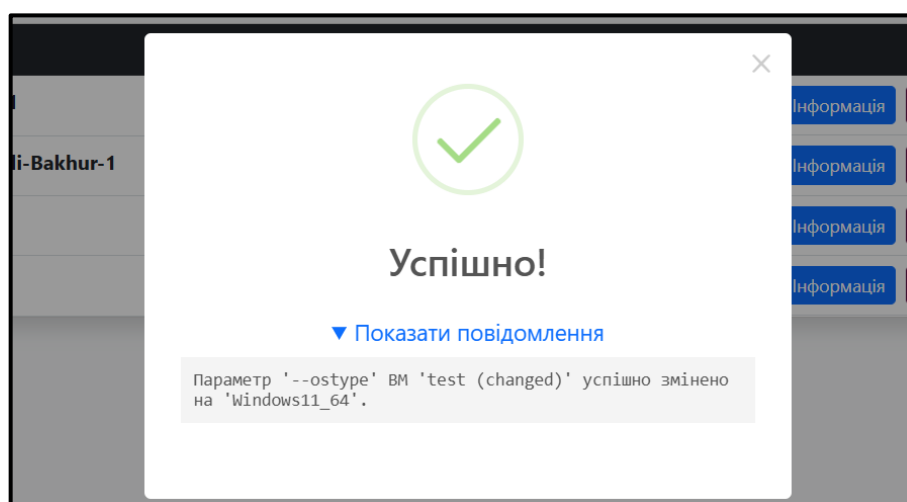


Рисунок 3.44 – Успішно змінено ОС

При редагуванні параметрів не можна залишити поля порожніми або ввести значення поза діапазоном (від мінімального до максимального можливого на хості) (рис. 3.45).

Зміна значення параметра

Введіть нове значення:

!

Неправильна кількість процесорів!
Кількість процесорів має бути від 1 до 12.

Змінити

Скасувати

✓

Успішно!

▼ Показати повідомлення

Параметр '--cpus' VM 'test (changed)' успішно змінено на '12'.

Рисунки 3.45-3.46 – Спроба змінити кількість процесорів поза межами та успіх

✓

Успішно!

▼ Показати повідомлення

Параметр '--memory' VM 'test (changed)' успішно змінено на '1280'.

✓

Успішно!

▼ Показати повідомлення

Параметр '--vram' VM 'test (changed)' успішно змінено на '130'.

Рисунки 3.47-3.48 – Успішно змінено обсяг оперативної пам'яті та відеопам'яті

Для зміни режиму підключення та типу адаптера потрібно: вибрати номер мережевого адаптера (від 1 до 8), далі бажаний режим (при режимах bridged та hostonly також потрібно вибрати фізичну карту) або тип.

Редагування VM 'test (changed)'

Виберіть параметр для зміни:

Режим підключення (NAT/Bridged) ▼

Далі

Скасувати

Вибір мережевого адаптера

Виберіть номер мережевого адаптера:

1 ▼

Вибрати

Скасувати

Рисунки 3.49-3.50 – Зміна режиму підключення та вибір мережевого адаптера

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		63

Зміна режиму підключення VM 'test (changed)'

Виберіть режим підключення з переліку:

Виберіть режим підключення

Виберіть режим підключення
Disabled Network Adapter
Not Attached
NAT
Bridged Adapter
Internal Network
Host-Only Adapter
Host-Only Network
Generic Driver
NAT Network
Cloud Network [EXPERIMENTAL]

Рисунок 3.51 – Перелік можливих режимів підключення

Вибір фізичної карти

Виберіть фізичну карту з переліку:

TP-Link Wireless USB Adapter

▼

Далі

Скасувати

Рисунок 3.52 – Вибір фізичної карти при режимі bridged

×

✓

Успішно!

▼ Показати повідомлення

Режим підключення мережі VM 'test (changed)' успішно змінено на 'bridged' (Адаптер: TP-Link Wireless USB Adapter).

Рисунок 3.53 – Успішно змінено режим підключення

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		64

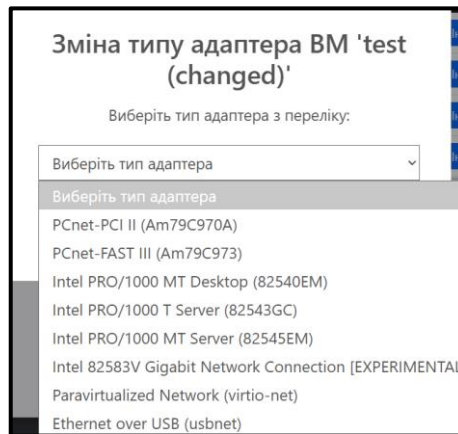


Рисунок 3.54 – Зміна типу адаптера VM

Зміна параметру «DVD-привід» відбувається таким ж способом, як при створенні VM.

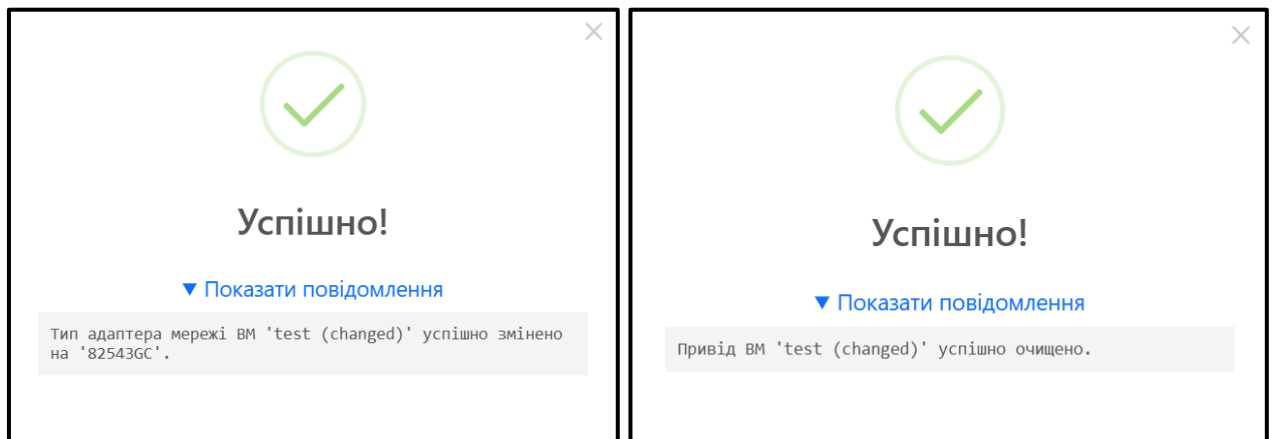


Рисунок 3.55-3.56 – Успішно змінено тип адаптера та витягнуто диск

Детальна інформація про VM 'test (changed)'	
Параметр	Значення
Назва:	test (changed)
ОС:	Windows 11 (64-bit)
Стан:	poweroff
Пам'ять (RAM):	1280 MB
Процесори (CPU):	12
Відеопам'ять (VRAM):	130 MB
Мережа (NIC-1):	bridged (82543GC)
MAC-адреса:	080027F8FA6B
Шлях до конфігу:	C:\Users\bakhu\VirtualBox VMs\test (changed)\test (changed).vbox
UUID:	eafd91e7-e79f-48f5-8851-749f986f2ce0
▶ Показати всі технічні параметри	
Закрити	

Рисунок 3.57 – Параметри VM після редагування

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		65

Тест-кейс 7. Видалення віртуальної машини.

Перед видаленням ВМ з'являється вікно (рис. 3.58), яке уточнює продовження операції. Якщо вибрано опцію «Так, видалити!», надається вибір: видалити ВМ повністю з файлами чи залишити їх (рис. 3.59). Після виконання операції виводиться вікно з результатом (рис. 3.60).

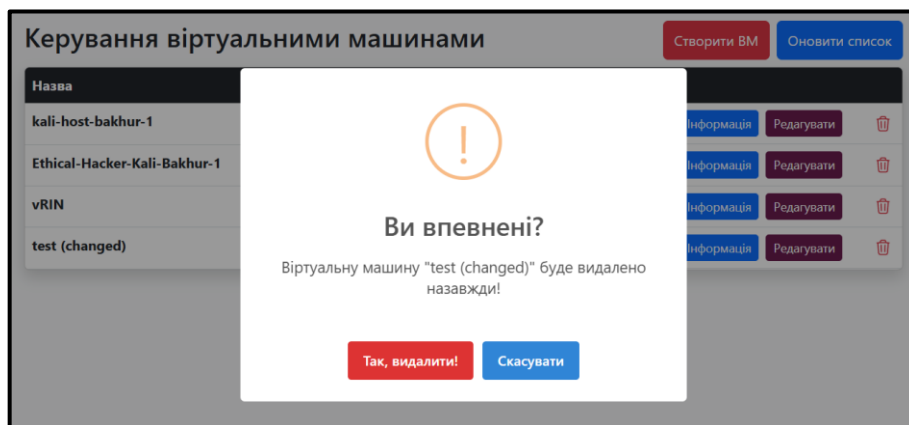


Рисунок 3.58 – Модальне вікно для уточнення

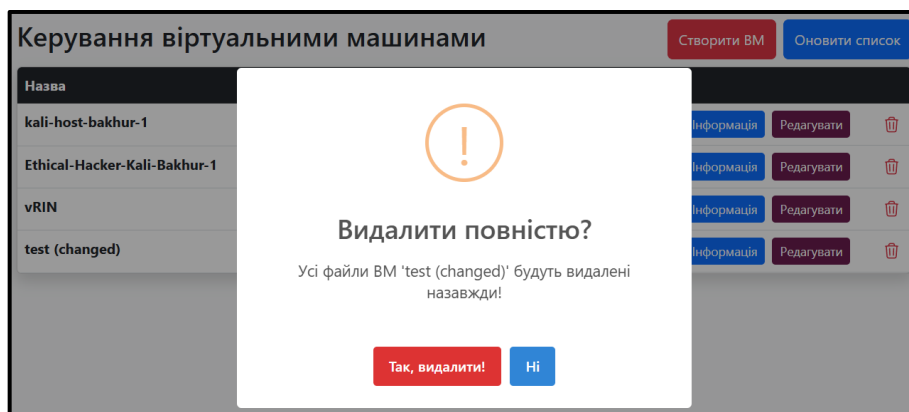


Рисунок 3.59 – Модальне вікно вибору режиму видалення

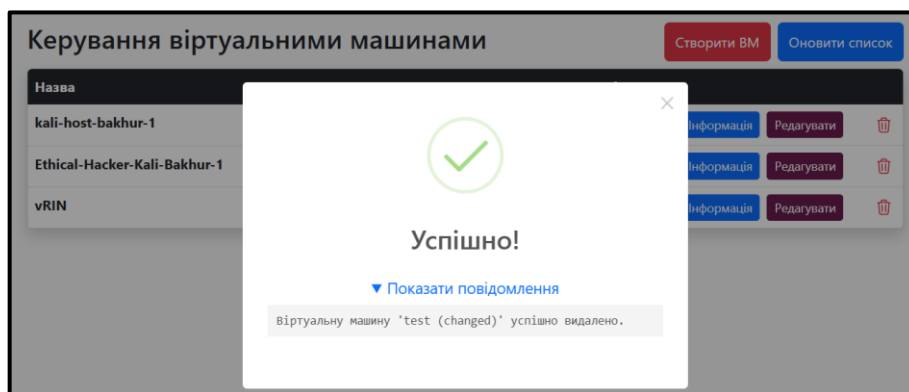


Рисунок 3.60 – Успішно видалено ВМ

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		66

Тест-кейс 8. Дистанційний доступ до ВМ.

Як зазначено в інструкції (рис. 3.61), потрібно переконатися, що принаймні один мережевий адаптер налаштований як Bridged, щоб пристрої в мережі могли підключитися через SSH, RDP чи VNC.

Підключення через VRDP за IP-адресою хоста та відповідним портом не потребує окремої авторизації (рис. 3.62-3.63), оскільки працює на рівні гіпервізора. На відміну від інших способів, через VRDP можна підключитися до ВМ, на якій ще не встановлено ОС, але для цього машина має бути запущена.

Інструкція з віддаленого підключення до 'vRIN'		
Перед підключенням переконайтеся, що мережевий адаптер ВМ налаштовано правильно (наприклад, Bridged або Port Forwarding).		
Протокол	Як підключитися	Коли рекомендується використовувати
VRDP VirtualBox	1. Скористайтесь клієнтом RDP (наприклад, <code>mstsc.exe</code> у Windows). 2. Введіть: <code>IP_хоста:5900</code> . ⚠️ Працює на рівні гіпервізора, окрема авторизація в ОС не потрібна.	Рекомендується для дебагу та контролю на етапі встановлення ОС . Показує екран машини ще до старту мережі чи завантаження ядра (аналог фізичного монітора).
RDP Windows	1. Запустіть утиліту «Remote Desktop Connection» (<code>Win + R -> mstsc</code>). 2. Введіть IP-адресу самої ВМ: <code>IP_адреса</code> . 3. Авторизуйтеся під створеними вами даними користувача.	Ідеально для повсякденної роботи з Windows-машинами або Linux із встановленим сервером <code>xrdp</code> . Забезпечує найкращу якість графіки та спільний буфер обміну.
VNC GUI Linux	1. Встановіть будь-який VNC-клієнт (VNC Viewer, TightVNC тощо). 2. Підключіться до IP-адреси машини за схемою: <code>IP_адреса:5901</code> (або дисплей -1). 3. Введіть VNC-пароль, згенерований під час встановлення.	Використовується для керування графічним інтерфейсом Linux-систем (наприклад, XFCE4), якщо нативні сервіси RDP не встановлені або некоректно працюють з іксами (X11).
SSH CLI Linux	1. Відкрийте термінал (<code>cmd</code> , <code>PowerShell</code> , <code>bash</code>) або утиліту PuTTY. 2. Виконайте команду: <code>ssh користувач@IP_адреса_ВМ</code> . 3. Введіть пароль облікового запису ВМ.	Основний стандарт для адміністрування Linux-серверів через консоль . Найбільш швидкий, безпечний та невибагливий до пропускової здатності мережі метод.

Рисунок 3.61 – Інструкція з віддаленого підключення до ВМ

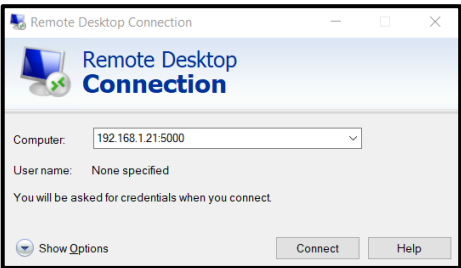


Рисунок 3.62 – Віддалене підключення до ВМ

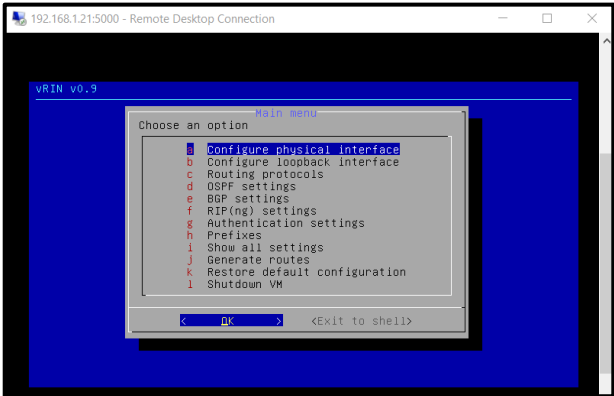


Рисунок 3.63 – Успішно здійснено підключення до ВМ

3.4. Програмна реалізація модулів менеджменту контрольних точок

Сторінка «Керування знімками ВМ» містить три дії: створити нову точку, відновити стан ВМ і видалити знімок (рис. 3.64).

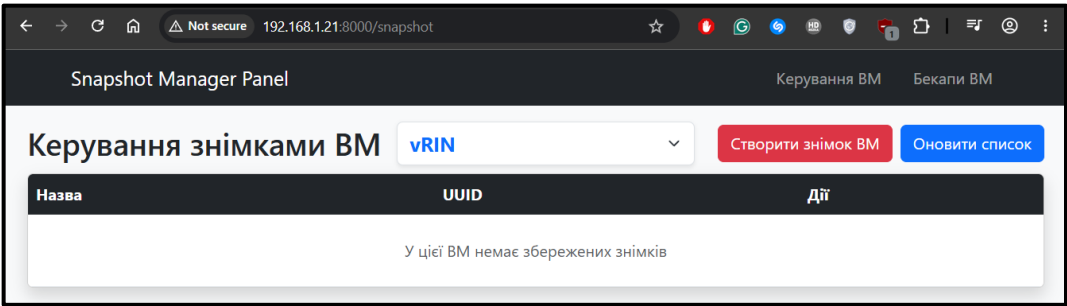


Рисунок 3.64 – Сторінка «Керування знімками ВМ»

При створенні потрібно задати ім'я знімку (рис. 3.65-3.66).

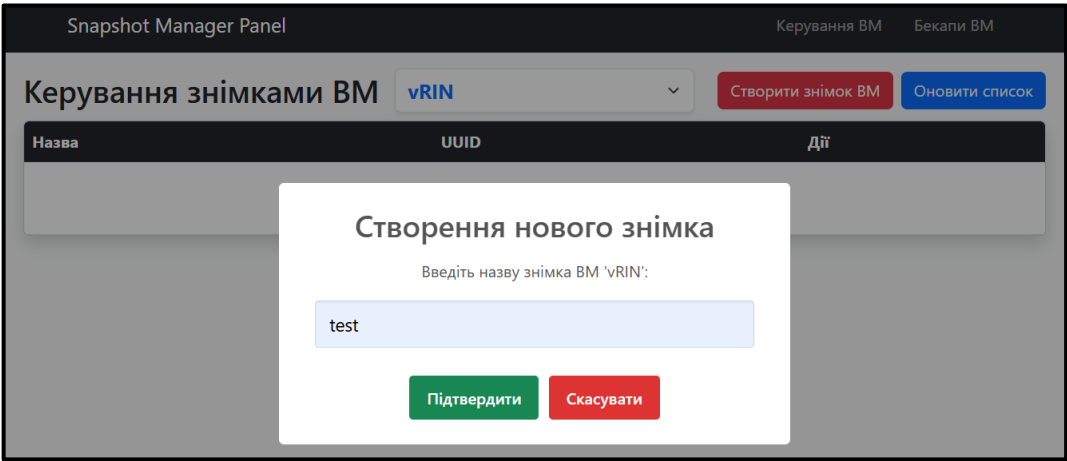


Рисунок 3.65 – Модальне вікно для введення назви знімку

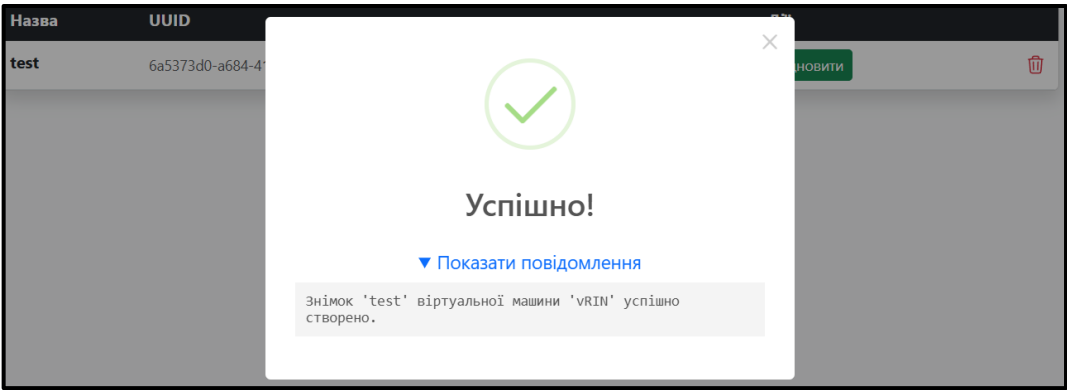


Рисунок 3.66 – Успішно створено ВМ

Для відновлення потрібно вибрати бажаний знімок, натиснути «Відновити» та підтвердити дію (рис. 3.67-3.68).

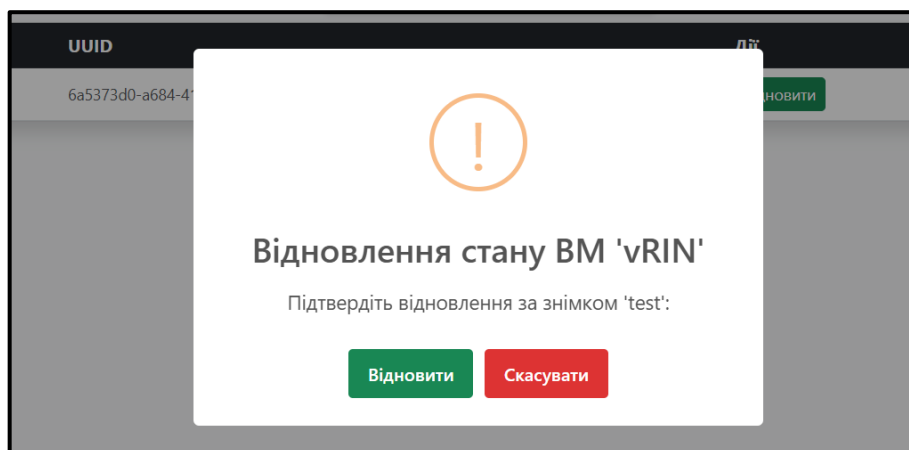


Рисунок 3.67 – Модальне вікно підтвердження дії

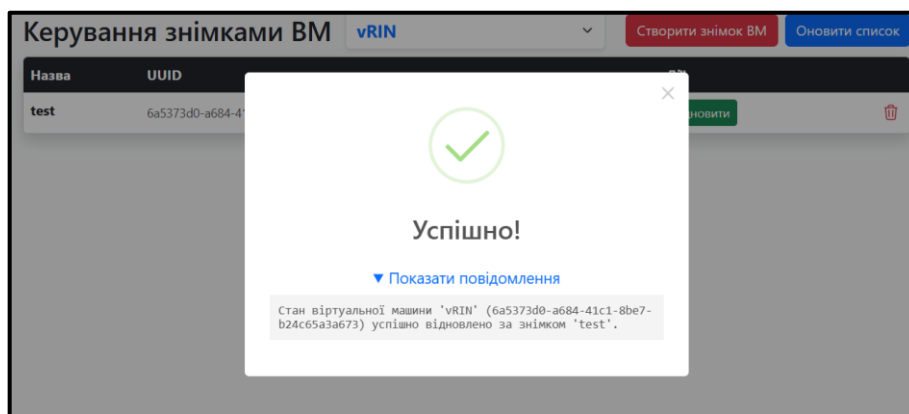


Рисунок 3.68 – Успішно відновлено стан ВМ

Аналогічно відновленню, для видалення потрібно вибрати бажаний знімок, натиснути «Видалити» та підтвердити дію в модальному вікні (рис. 3.69-3.70).

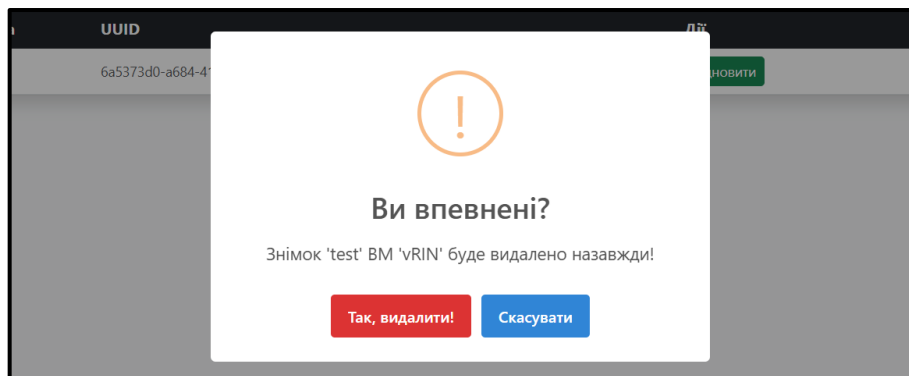


Рисунок 3.69 – Модальне вікно підтвердження дії

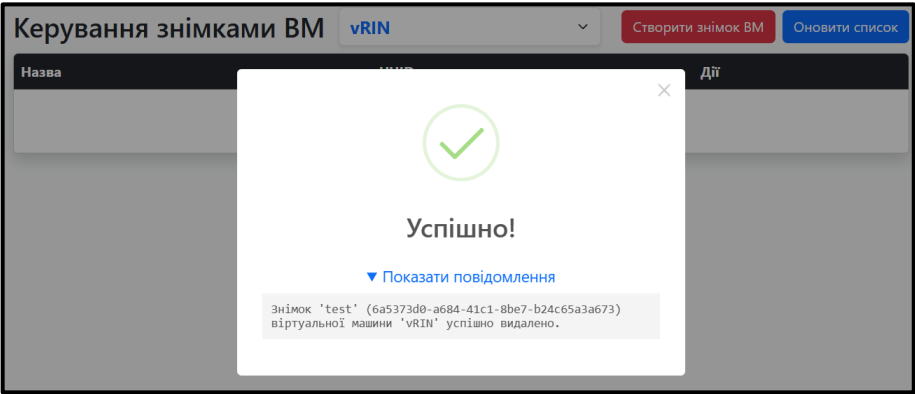


Рисунок 3.70 – Успішно видалено знімок VM

3.5. Реалізація підсистеми резервного копіювання, планування та валідації

Сторінка «Керування резервними копіями VM» містить таблицю з інформацією про бекапи за поточним шляхом, кнопки «Змінити каталог», «Дії», яка виводить модальне вікно з переліком дій, представлених кнопками (рис. 3.72), та «Перевірити» (рис. 3.71).

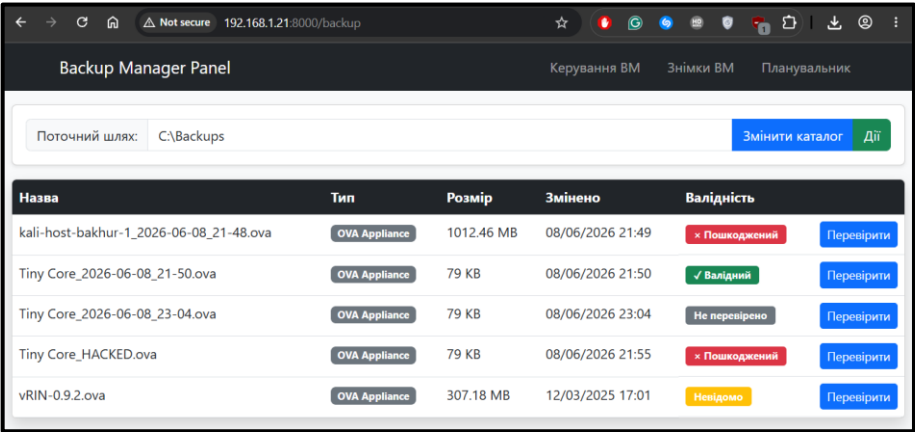


Рисунок 3.71 – Сторінка «Керування резервними копіями VM»

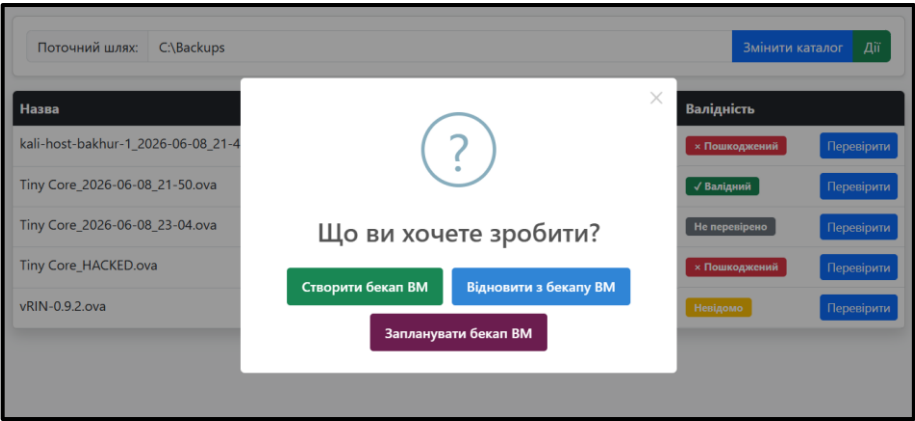


Рисунок 3.72 – Модальне вікно з наявними операціями

Зміна каталогу відбувається таким же чином, як і вибір ISO-файлу під час керування ВМ, але вибір файлу не змінює поточний шлях. Для того, щоб завершити вибір каталогу, потрібно перейти в бажану папку та натиснути «Вибрати», коли не вибрано жодного іншого файлу чи директорії (рис. 3.73).

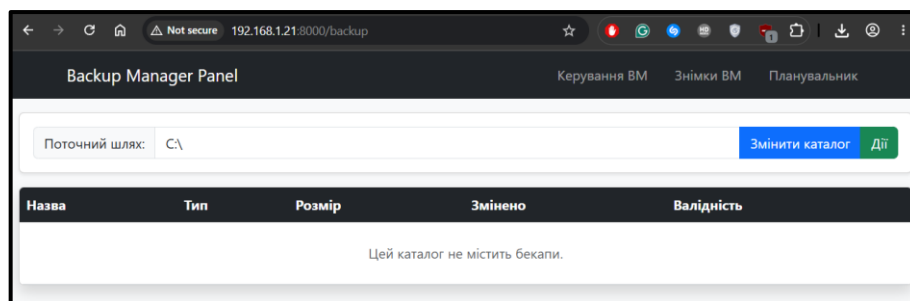


Рисунок 3.73 – Порожня таблиця через відсутність бекапів за шляхом

Створення миттєвої резервної копії потребує вибору бажаної ВМ (рис. 3.74), підтвердити вибір, натиснувши «Створити», та зачекати поки виконується операція (рис. 3.75). Створений файл матиме назву «Ім'я ВМ_рік-місяць-день_год-хв-сек» з розширенням .ova (рис. 3.76).

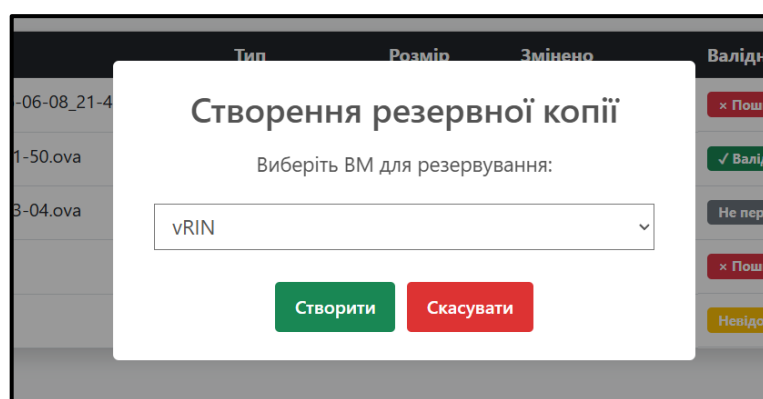


Рисунок 3.74 – Модальне вікно вибору ВМ для резервування

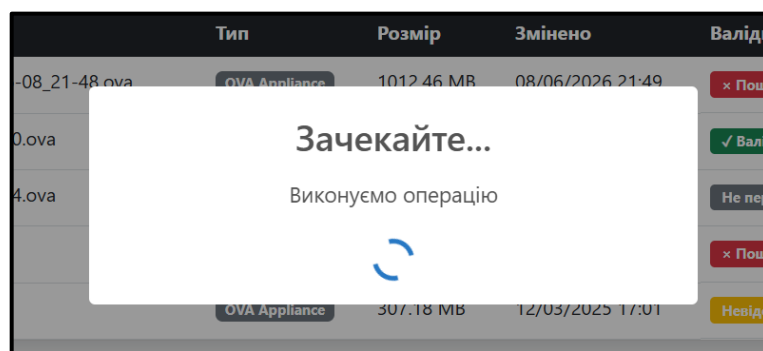


Рисунок 3.75 – Модальне вікно виконання операції

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				71
Змн.	Арк.	№ докум.	Підпис	Дата		

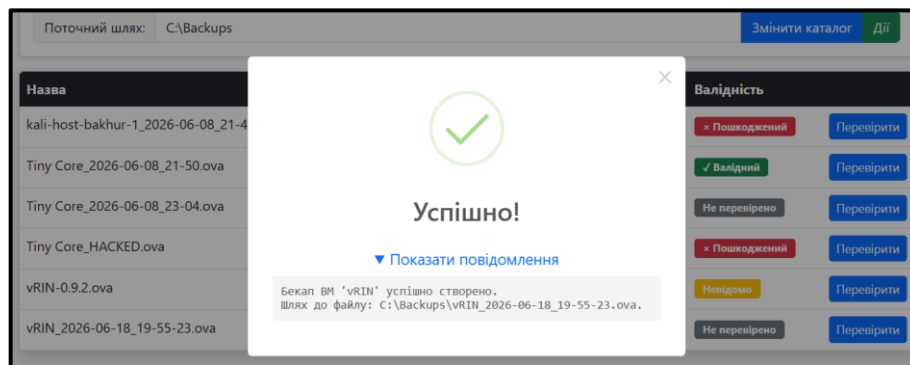


Рисунок 3.76 – Успішно створено новий бекап

Перевірка бекапу розпочинається з натискання на відповідну кнопку «Перевірити», тоді відбувається виконання функції Test-OvaBoot. Вона реалізує алгоритм автоматизованої перевірки цілісності резервних копій шляхом тестового розгортання віртуальної машини з .ova архіву у фоновому режимі, здійснюючи гнучкий аналіз помилок на кожному етапі взаємодії з гіпервізором.

Спочатку утиліта генерує унікальне ім'я для тимчасової VM на основі мітки часу Unix Epoch та намагається імпортувати архів, перехоплюючи інженерні збої середовища хоста (наприклад, брак вільного дискового простору, за якого фіксується статус Unknown) або ознаки фізичного руйнування самого файлу (Digest mismatch, за якого встановлюється статус Corrupted).

У разі успішного імпорту функція ініціює старт VM, паралельно аналізуючи критичні аспекти апаратного забезпечення (нестача RAM або конфлікти віртуальних мережевих карт), після чого у циклі опитує гостьові властивості (guestproperty) та конфігурацію (showvminfo) для фіксації ознак успішного завантаження ядра операційної системи.

Незалежно від результату верифікації, функція гарантує повне очищення хост-системи шляхом примусової зупинки (poweroff) та безповоротного видалення тимчасової машини разом із її дисками (unregistervm --delete-all), перетворюючи фінальний вердикт про працездатність бекапу (Valid, Corrupted або Unknown) у супутній конфігураційний файл метаданих JSON та повертаючи структуровану відповідь для відображення у веб-інтерфейсі додатка (рис. 3.79-3.81).

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		72

Можна переглянути вміст файлу .ova, змінивши розширення на .tar, і відкрити архів (рис. 3.77). Під час експортування ВМ у файл завдяки опціям --manifest та --options=manifest генерується файл .mf, у якому зберігається обчислена хеш-сума (рис. 3.78).

 Tiny Core_2026-06-08_23-04.mf	154	154	MF File	08/06/2026 23:04
 Tiny Core_2026-06-08_23-04.ovf	9,521	9,521	Open Virtualization Format	08/06/2026 23:04
 Tiny Core-disk002.vmdk	69,120	69,120	VMDK File	08/06/2026 23:04

Рисунок 3.77 – Архів містить файл з розширенням .mf

Tiny Core_2026-06-08_23-04.mf - Notepad				
File Edit Format View Help				
SHA1 (Tiny Core-disk002.vmdk) = 2838e50c0331d84c0080f29e521e337d309fe197				
SHA1 (Tiny Core_2026-06-08_23-04.ovf) = ad46b3dc414ee08db7357104c9f0ad597884e7d8				

Рисунок 3.78 – Вміст файлу з розширенням .mf містить обчислені хеш-суми

Назва	Тип	Розмір	Змінено	Валідність
kali-host-bakhur-1_2026-06-08_21-48.ova	OVA Appliance	1012.46 MB	08/06/2026 21:49	✖ Пошкоджений Перевірити
Tiny Core_2026-06-08_21-50.ova				✓ Валідний Перевірити
Tiny Core_2026-06-08_23-04.ova				Не перевірено Перевірити
Tiny Core_HACKED.ova				✖ Пошкоджений Перевірити
vRIN-0.9.2.ova	OVA Appliance	307.18 MB	12/03/2025 17:01	Невідомо Перевірити
vRIN_2026-06-18_19-55-23.ova	OVA Appliance	328.47 MB	18/06/2026 19:55	Не перевірено Перевірити

Зачекайте...

Виконуємо операцію

Рисунок 3.79 – Виконується тестування

Назва	Валідність
kali-host-bakhur-1_2026-06-08_21-4	✖ Пошкоджений Перевірити
Tiny Core_2026-06-08_21-50.ova	✓ Валідний Перевірити
Tiny Core_2026-06-08_23-04.ova	Не перевірено Перевірити
Tiny Core_HACKED.ova	✖ Пошкоджений Перевірити
vRIN-0.9.2.ova	Невідомо Перевірити
vRIN_2026-06-18_19-55-23.ova	✓ Валідний Перевірити

✕



Успішно!

▼ Показати повідомлення

Тест завантаження успішний! Віртуальна машина успішно ініціалізувалася у фоновому режимі, ядро ОС стартувало.

Назва

kali-host-bakhur-1_2026-06-08_21-48.ova

Tiny Core_2026-06-08_21-50.ova

Tiny Core_2026-06-08_23-04.ova

Tiny Core_HACKED.ova

vRIN-0.9.2.ova

vRIN_2026-06-18_19-55-23.ova

×

✖

Помилка виконання

▶ Показати помилку

Тестування перервано: На комп'ютері закінчилося місце на диску! Не вдалося розгорнути VM. Актуальний стан бекапу: Невідомо.

Валідність

✖ Пошкоджений

Перевірити

✓ Валідний

Перевірити

Не перевірено

Перевірити

✖ Пошкоджений

Перевірити

Невідомо

Перевірити

Невідомо

Перевірити

Рисунки 3.80-3.81 – Модальні вікна про успіх та помилку тестування

Відновлення VM з бекапу потребує вибрати бекап з поточного каталогу та підтвердити дію, натиснувши «Відновити» (рис. 3.82).

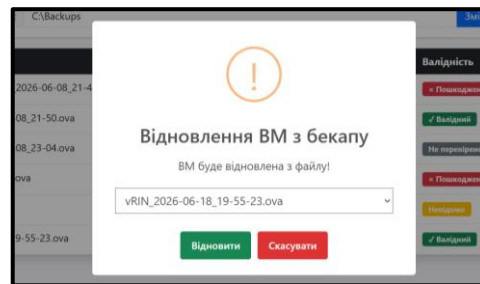
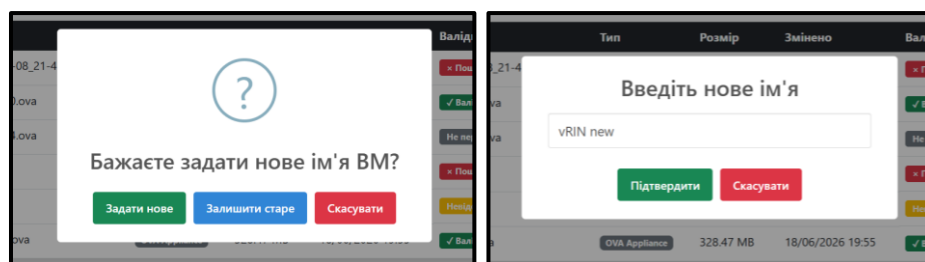


Рисунок 3.82 – Модальне вікно вибору файлу для відновлення VM

Є можливість задати нову назву VM або залишите стару (рис. 3.83-3.84).



Рисунки 3.83-3.84 – Модальне вікно вибору файлу для відновлення VM

Далі потрібно зачекати, поки завершиться відновлення (рис. 3.85). Для того, щоб побачити відновлену VM у переліку, потрібно перейти на сторінку «Керування VM» або оновити список уже запущеної.

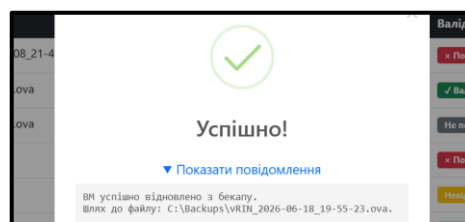


Рисунок 3.85 – Успішно відновлено VM з бекапу

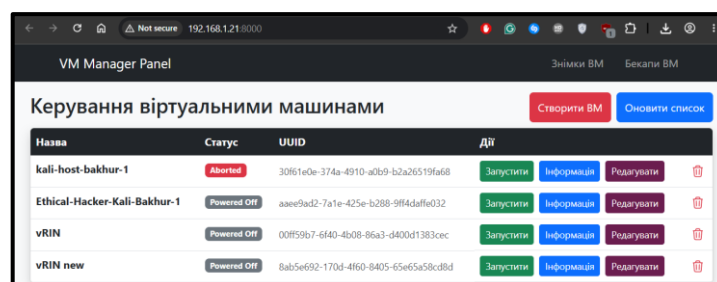


Рисунок 3.86 – Відновлена VM наявна в переліку

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		74

Запланування автоматичного бекапу потребує вибору VM (рис. 3.87), періодичності виконання (рис. 3.88) та встановлення часу (крім випадків, коли вибрано періодичність "при запуску ОС" або "при вході користувача в систему") (рис. 3.89). Бекап буде створюватися у поточному каталозі (рис. 3.90).

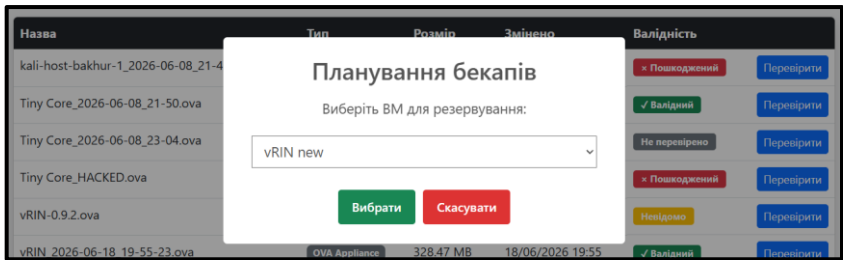


Рисунок 3.87 – Модальне вікно вибору VM для планування

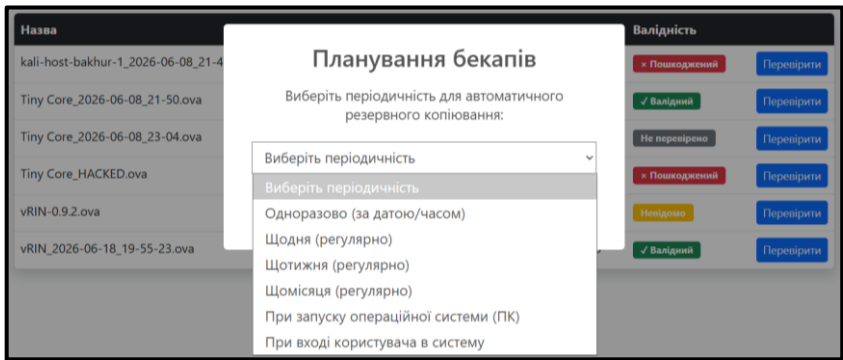


Рисунок 3.88 – Модальне вікно вибору періодичності

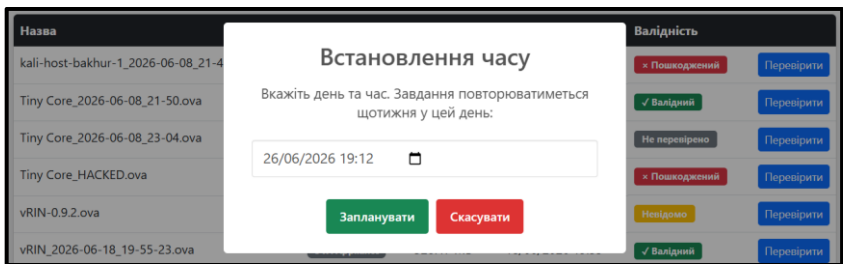


Рисунок 3.89 – Модальне вікно встановлення часу

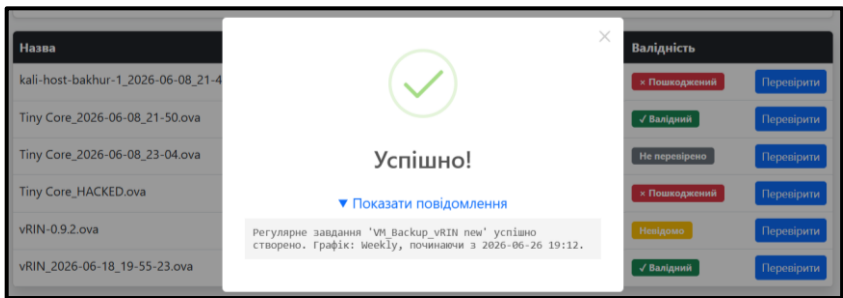


Рисунок 3.90 – Успішно заплановано автоматичний бекап

Для виведення переліку запланованих автоматичних бекапів потрібно натиснути на «Планувальник» на навігаційній панелі (наявний лише на сторінці «Керування бекапами VM») (рис. 3.91).

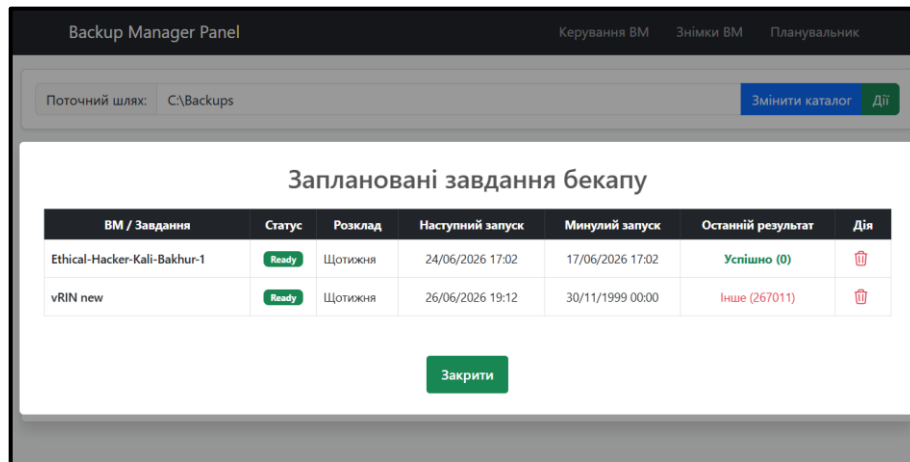


Рис. 3.91 – Перелік запланованих автоматичних бекапів

Для видалення завдання потрібно натиснути на відповідне "Видалити", підтвердити дію та зачекати, поки відбувається операція (рис. 3.92-3.93).

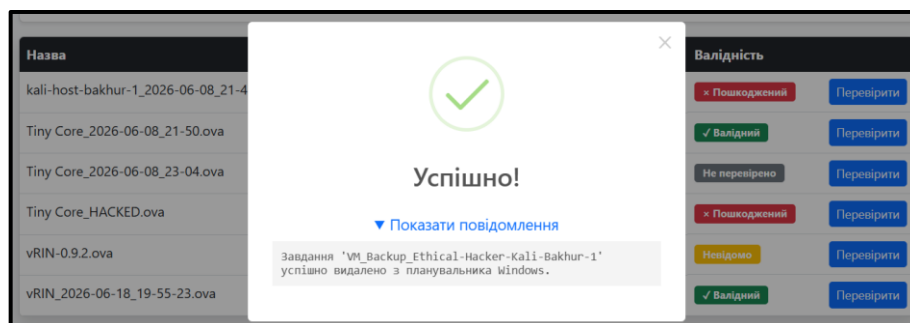


Рис. 3.92 – Успішно видалено завдання

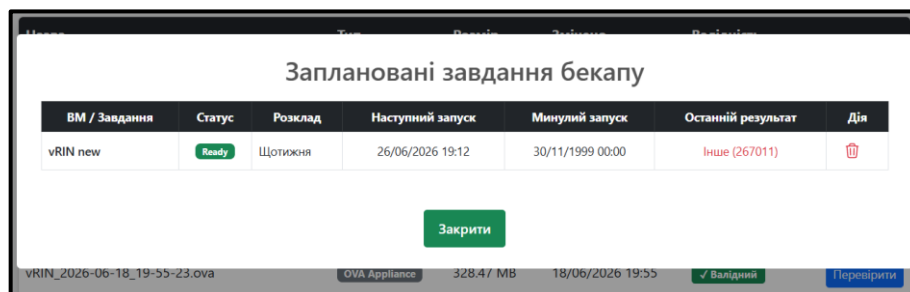


Рис. 3.93 – Перелік запланованих автоматичних бекапів після видалення

З метою оцінки практичної ефективності розробленого програмного комплексу було проведено порівняльний аналіз часових витрат оператора та швидкості ініціалізації резервного копіювання ВМ обсягом 3.8 Гб трьома різними методами: вручну через пряме введення команд у нативній CLI-консолі VBoxManage, через розроблений автономний консольний інтерфейс `vm-manager.ps1` та за допомогою графічного веб-інтерфейсу системи.

Експериментальні випробування показали, що пряме ручне адміністрування вимагає найбільше часу (≈ 100 секунд на підготовку через необхідність безпомилкового написання довгих інженерних конструкцій штибу `VBoxManage export` із ручним визначенням шляху, назви ВМ та параметрів для створення маніфесту), а також створює високі ризики синтаксичних помилок.

Бекап було створено за 03:52.27, з яких $\approx 01:40$ було витрачено на написання при тому, що користувач знає/має команду перед очима (рис. 3.94).

Рисунок 3.94 – Створення бекапу вручну

Натомість, використання розробленого інтерактивного CLI-меню скоротило час ініціалізації операції до ≈ 25 секунд завдяки автоматизованому вибору цільової ВМ із нумерованого списку та вбудованим механізмам екранування аргументів.

Бекап було створено за 02:06.88, з яких $\approx 00:25$ витрачено на перехід від головного меню до роботи зі резервними копіями та вибору ВМ (рис. 3.95).

Рисунок 3.95 – Створення бекапу через консольний інтерфейс

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				77
Змн.	Арк.	№ докум.	Підпис	Дата		

Найвищу ергономічність та UX-швидкість продемонстрував розроблений веб-інтерфейс: ініціалізація експорту в декілька кліки (перейти з головної сторінки до роботи з бекапами, вибрати дію та VM) дуже швидко запускає асинхронний ланцюжок REST API (POST /api/vms/{name}/backups) → FastAPI → backup.psm1, причому завдяки інтеграції з бібліотекою SweetAlert2 клієнтська сторона повністю блокується від повторних запитів і візуалізує хід процесу завантаження, а сам час безпосереднього пакування у кросплатформний формат .ova (усі три методи показали однакову швидкість чистої дискової утилізації $\approx 01:54.34$, оскільки спираються на однакове низькорівневе ядро гіпервізора) протікає стабільно, доводячи доцільність автоматизації рутинних операцій.

Бекап було створено за 02:01.03, з яких $\approx 00:07$ витрачено на перехід від головної сторінки до роботи зі резервними копіями та вибору VM (рис. 3.96).

Backup Manager Panel					Керування VM	Знімки VM	Планувальник
Поточний шлях: C:\Backups					Змінити каталог		
					Дії		
Назва	Тип	Розмір	Змінено	Валідність			
Tiny Core_2026-06-08_21-50.ova	OVA Appliance	79 KB	08/06/2026 21:50	✓ Валідний	Перевірити		
Tiny Core_2026-06-08_23-04.ova	OVA Appliance	79 KB	08/06/2026 23:04	Не перевірено	Перевірити		
Tiny Core_HACKED.ova	OVA Appliance	79 KB	08/06/2026 21:55	✗ Пошкоджений	Перевірити		
vm_2026-06-18_22-28-52.ova	OVA Appliance	1.42 GB	18/06/2026 22:30	Не перевірено	Перевірити		
vm_2026-06-18_22-32-32.ova	OVA Appliance	1.42 GB	18/06/2026 22:34	Не перевірено	Перевірити		
VM_Backup_vm.ova	OVA Appliance	1.42 GB	18/06/2026 22:26	Не перевірено	Перевірити		
vRIN-0.9.2.ova	OVA Appliance	307.18 MB	12/03/2025 17:01	Невідомо	Перевірити		
vRIN_2026-06-18_19-55-23.ova	OVA Appliance	328.47 MB	18/06/2026 19:55	✓ Валідний	Перевірити		

Рисунок 3.96 – Створення бекапу через веб-інтерфейс

Висновки до третього розділу

У третьому розділі виконано повний комплекс робіт із програмної реалізації, розгортання, конфігурування та всебічного функціонального тестування елементів автоматизованої системи керування віртуальною інфраструктурою. Проведені інженерні дослідження та практичні випробування дозволяють сформулювати такі висновки.

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		78

1. Реалізовано відмовостійкий автономний консольний інтерфейс на базі сценарного ядра Windows PowerShell (vm-manager.ps1 та menu.psm1), який повністю дублює бізнес-логіку системи. Доведено його практичну цінність як резервного каналу адміністрування під час регламентних робіт, коли веб-сервер зупинено. Впроваджено інтерактивні алгоритми автоматичного самопідвищення привілеїв безпеки (UAC-контекст за допомогою Start-Process -Verb RunAs), а також уніфіковані функції обробки та валідації введених користувачем даних (Get-UserSelection, Out-Result), що повністю захищає оболонку від непередбачуваних збоїв чи падіння скриптів у разі помилкового введення.

2. Спроековано та розроблено високопродуктивний серверний шлюз інтеграції на базі асинхронного Python-фреймворку FastAPI (main.py). У рамках архітектурної концепції багатосторінкових додатків (MPA) успішно поєднано серверний рендеринг сторінок за допомогою шаблонізатора Jinja2Templates та обробку динамічних асинхронних REST API ендпоінтів для обміну даними у форматі JSON. Центральним інженерним рішенням стало створення захищеного механізму міжпроцесної взаємодії за допомогою функції run_ps, яка примусово форсує UTF-8 локалізацію потоків введення-виведення, підтримує ізоляцію контексту модулів і реалізує захисні таймаути виконання (timeout=300) для запобігання зависанню веб-сервера. Також реалізовано превентивний алгоритм автоматичного очищення зомбі-процесів (cleanup_zombie_processes) через системну утиліту taskkill /F /IM /T.

3. Проведено повний цикл функціонального тестування підсистеми керування життєвим циклом та апаратними конфігураціями віртуальних машин (vm.html). Завдяки інтеграції нативного JavaScript API fetch() та універсального асинхронного обробника подій handleAction на базі бібліотеки SweetAlert2, вдалося досягти динамічного оновлення стану інтерфейсу та блокування елементів введення під час виконання тривалих операцій. Тест-кейси підтвердили стабільність створення, запуску (як у headless, так і в GUI режимах), модифікації заліза (CPU, RAM, VRAM, DVD-приводи, мережеві адаптери від 1 до 8), виведення повної телеметрії через showvminfo та безпечного/примусового вимкнення обчислювальних вузлів.

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		79

4. Успішно протестовано підсистему дистанційного доступу та менеджменту знімків стану (Snapshots). Експериментально доведено працездатність механізмів віддаленого підключення до гостьових ОС через інтегровані протоколи RDP чи VRDP, VNC та SSH при налаштуванні мережесхем адаптерів у режим мережевого мосту (Bridged Adapter). Модуль контрольних точок продемонстрував високу точність фіксації станів інфраструктури розробника, забезпечуючи миттєвий відкат до збережених точок або видалення застарілих гілок із консолідацією дискових шарів гіпервізора.

5. Реалізовано інноваційний алгоритм автоматизованої валідації бекапів Test-OvaBoot. Створена логіка дозволяє не просто перевіряти наявність архіву на диску, а здійснювати повноцінне «тест-драйв» розгортання ВМ у фоновому режимі із циклічним опитуванням гостьових властивостей (guestproperty enumerate) для детекції успішного старту ядра ОС. Алгоритм забезпечує гнучку діагностику середовища (диференціацію помилок браку пам'яті хоста, відсутності мережесхем інтерфейсів, пошкодження контрольних сум маніфесту .mf або переповнення сховища) та гарантує 100% очищення ресурсів після завершення перевірки шляхом виклику `unregistervm --delete-all`.

6. Експериментально доведено високу практичну та ергономічну ефективність розробленого комплексу шляхом проведення порівняльного хронометражу процедури створення резервних копій у формат .ova (обсяг тестової ВМ – 3.8 Гб). Результати випробувань продемонстрували такі показники.

- Пряме ручне введення в CLI VBoxManage: загальний час виконання становив 03:52.27, з яких понад 01:40 оператор витратив лише на безпомилковий набір довгої інженерної команди з параметрами маніфесту;

- Розроблений консольний інтерфейс `vm-manager.ps1`: час скоротився до 02:06.88, де на навігацію меню та інтерактивний вибір ВМ було витрачено всього 25 секунд;

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		80

- Розроблений графічний веб-інтерфейс системи: продемонстрував найкращий результат – 02:01.03, де на ініціалізацію резервування в кілька кліків миттєво через REST API було витрачено лише 7 секунд.

Враховуючи, що чистий час дискових операцій експорту в усіх трьох методах є приблизно однаковим ($\approx 01:54$, оскільки вони використовують одне низькорівневе ядро гіпервізора), автоматизація через веб-інтерфейс дозволила скоротити невиробничі часові витрати адміністратора на підготовку та запуск операції у 14 разів порівняно з ручним CLI-введенням (з 100 до 7 секунд), повністю нівелювавши людський фактор та ризики виникнення синтаксичних помилок.

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		81

ВИСНОВКИ

У кваліфікаційній роботі розв'язано актуальну науково-практичну задачу автоматизації процесів адміністрування, конфігурування, забезпечення відмовостійкості та створення резервних копій у середовищах віртуалізації на базі Hosted-гіпервізорів. На основі теоретичного аналізу, інженерного проектування та експериментальних випробувань сформульовано такі конкретні результати відповідно до поставлених на початку роботи завдань.

1. Досліджено архітектуру Hosted-гіпервізора Oracle VM VirtualBox та обґрунтовано концепцію автоматизації інфраструктури. Визначено повний перелік низькорівневих можливостей CLI-утиліти VBoxManage та зафіксовано її обмеження (складний синтаксис, брак планувальника завдань). Доведено неефективність сумісного використання важковагових сторонніх рішень (Vagrant, Bareos, MobaXterm) для локальних ПК через високий оверхед ресурсів хоста (RAM, CPU). Сформовано концепцію гібридної, безкомпонентної (3-tier) архітектури системи без використання громіздких баз даних чи агентських служб.

2. Розроблено та оптимізовано механізми міжпроцесної взаємодії (IPC) та крос-платформного кодування. Спроектовано та практично реалізовано п'ятиетапний асинхронний алгоритм трансляції дій користувача від клієнтських Fetch-запитів до інструкцій фонового демона VBoxSVC. Вирішено проблему крос-платформного кодування даних: розроблено функцію `run_ps`, яка примусово форсує UTF-8 локалізацію потоків введення-виведення командної оболонки Windows PowerShell, забезпечуючи коректний парсинг консольних відповідей за допомогою регулярних виразів та їх зворотнє перетворення в уніфікований текстовий JSON-потік для веб-сервера.

3. Забезпечено безпеку виконання та стійкість системи на рівні ОС Windows. Інтегровано системні API Windows для управління привілеями: впроваджено класи .NET та механізми ShellExecuteW (дієслово `runas`), які забезпечують автоматичне самопідвищення привілеїв безпеки (UAC-контекст) як для консольного ядра `vm-manager.ps1`, так і для серверного шлюзу. Також розроблено алгоритм аварійного

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		82

очищення робочого середовища (cleanup_zombie_processes), який через утиліту taskkill /F /IM /T рекурсивно знищує залишкові процеси PowerShell та VBoxManage, запобігаючи виникненню станів взаємного блокування (deadlocks) та витіканню оперативної пам'яті.

4. Спроековано та реалізовано модульне сценарне ядро на базі PowerShell. Розроблено структуру backend-частини, декомпозовану на п'ять ізольованих модулів: utils.psm1 (збір телеметрії хоста та ВМ), vm.psm1 (керування життєвим циклом та 8 мережевими адаптерами), snapshot.psm1 (контрольні точки), backup.psm1 (експорт та планувальник), menu.psm1 (консольний інтерфейс). Завдяки впровадженню універсального методу Invoke-VBoxCommand та блоків Try-Catch, досягнуто повної ізоляції внутрішніх помилок гіпервізора.

5. Розроблено ресурсно-орієнтований веб-серверний шлюз на базі REST API. На базі асинхронного Python-фреймворку FastAPI розроблено серверний шлюз (main.py), який об'єднує гнучку REST API маршрутизацію (ендпоінти груп /api/vms, /api/backups, /api/host) із двосторонньою валідацією типів через Pydantic-моделі та динамічний серверний рендеринг веб-сторінок за допомогою шаблонізатора Jinja2Templates.

6. Спроековано та впроваджено ергономічний, інтерактивний користувацький інтерфейс. Створено графічний веб-інтерфейс за архітектурою Multiple Page Application (HTML5, Bootstrap, Vanilla JS). Реалізовано кольорове кодування статусів інфраструктури (Running, Powered Off, Aborted) та станів архівів. Завдяки розробці універсального асинхронного JS-хелпера функції handleAction та бібліотеки SweetAlert2, впроваджено інтерактивні ладери та механізми блокування інтерфейсу під час тривалих операцій, що унеможливорює надсилання дублюючих HTTP-запитів клієнтом.

7. Вирішено проблему відмовостійкості, автоматичного планування та он-лайн-валідації резервних копій. Шляхом інтеграції з нативним Планувальником завдань ОС (через COM-об'єкт Schedule.Service) реалізовано три режими бекапування (миттєве, відкладене, за розкладом), перенісши ресурсомісткі процеси у фон.

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		83

Створено інноваційний алгоритм Test-OvaBoot, який здійснює «тест-драйв» розгортання ВМ з .ova архівів у headless-режимі з опитуванням guestproperty, точно діагностує пошкодження маніфестів .mf, брак RAM хоста чи переповнення дисків, фіксує статуси Valid / Corrupted та гарантуючи повне очищення хоста (unregistervm --delete-all).

8. Проведено експериментальну верифікацію та хронометраж ефективності системи. Функціональне тестування підтвердило 100% стабільність виконання всіх операцій над ВМ. Порівняльний хронометраж створення бекапу (ВМ обсягом 3.8 Гб) довів високу ергономічну цінність системи: завдяки автоматизації час ініціалізації експорту через розроблений веб-інтерфейс скоротився до 7 секунд проти 25 секунд у консольному меню та 100 секунд при ручному CLI-введенні (невиробничі часові витрати адміністратора зменшено у 14 разів), що повністю нівелює людський фактор та доводить досягнення мети кваліфікаційної роботи.

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		84

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Understanding virtualization // Red Hat : офіційний вебсайт. URL: <https://www.redhat.com/en/topics/virtualization>
2. Oracle VM VirtualBox. Chapter 1. First steps // Oracle VM VirtualBox Documentation : офіційна документація. URL: <https://www.virtualbox.org/manual/ch01.html>
3. Oracle VM VirtualBox. VBoxManage reference manual // Oracle VM VirtualBox Documentation : офіційна документація. URL: <https://www.virtualbox.org/manual/topics/vboxmanage.html#vboxmanage>
4. Vagrant Documentation // HashiCorp Developer : вебпортал для розробників. URL: <https://developer.hashicorp.com/vagrant>
5. What is Bareos? // Bareos Documentation : офіційна документація системи резервного копіювання. URL: <https://docs.bareos.org/IntroductionAndTutorial/What-IsBareos.html>
6. MobaXterm: Home Edition // Mobatek : офіційний вебсайт розробника. URL: <https://mobaxterm.mobatek.net/>
7. mRemoteNG Documentation v1.77.3-dev // Read the Docs : вебплатформа документації програмного забезпечення. URL: <https://mremoteng.readthedocs.io/en/v1.77.3-dev/>
8. PowerShell Documentation // Microsoft Learn : центральний вебпортал документації Microsoft. URL: <https://learn.microsoft.com/en-us/powershell/>
9. FastAPI Documentation // FastAPI : офіційний вебсайт вебфреймворку. URL: <https://fastapi.tiangolo.com/>
10. Bootstrap v5.0. Customize: Color // Bootstrap : офіційна документація графічного фреймворку. URL: <https://getbootstrap.com/docs/5.0/customize/color/>
11. SweetAlert2: Beautiful, responsive, customizable, accessible replacement for JavaScript's popup boxes // SweetAlert2 : офіційне вебкерівництво бібліотеки. URL: <https://sweetalert2.github.io/#examples>

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				85
Змн.	Арк.	№ докум.	Підпис	Дата		

12. Win32_ComputerSystem class // Microsoft Learn. Windows Win32 Provider : офіційна технічна документація. URL: <https://learn.microsoft.com/en-us/windows/win32/cimwin32prov/win32-computersystem>

13. Win32_Processor class // Microsoft Learn. Windows Win32 Provider : офіційна технічна документація. URL: <https://learn.microsoft.com/en-us/windows/win32/cimwin32prov/win32-processor>

14. Win32_LogicalDisk class // Microsoft Learn. Windows Win32 Provider : офіційна технічна документація. URL: <https://learn.microsoft.com/en-us/windows/win32/cimwin32prov/win32-logicaldisk>

15. Everything you wanted to know about [PSCustomObject] // Microsoft Learn. PowerShell Deep Dives : керівництво з розробки. URL: <https://learn.microsoft.com/en-us/powershell/scripting/learn/deep-dives/everything-about-pscustomobject>

16. API design guidance // Microsoft Learn. Azure Architecture Center : архітектурні рекомендації Microsoft. URL: <https://learn.microsoft.com/en-us/azure/architecture/best-practices/api-design>

17. Uvicorn: An ASGI web server, implementation for Python // Uvicorn Documentation : офіційна документація вебсервера. URL: <https://uvicorn.dev/>

18. Pydantic Concepts. Get started // Pydantic Documentation : офіційна документація бібліотеки валідації. URL: <https://pydantic.dev/docs/validation/latest/get-started/>

19. Fetch API // Сучасний підручник з JavaScript : навчально-довідковий веб-портал. URL: <https://uk.javascript.info/fetch-api>

20. How User Account Control works // Microsoft Learn. Windows Security Application Control : керівництво з безпеки Windows. URL: <https://learn.microsoft.com/en-us/windows/security/application-security/application-control/user-account-control/>

21. ScheduledTasks Module for Windows PowerShell // Microsoft Learn. PowerShell Module Reference : довідник командлетів. URL: <https://learn.microsoft.com/en-us/powershell/module/scheduledtasks/>

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		86

ДОДАТКИ

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				87
Змн.	Арк.	№ докум.	Підпис	Дата		

Вміст файлу vm-manager.ps1

```

Import-Module "$PSScriptRoot/modules/menu.psm1"
Import-Module "$PSScriptRoot/modules/vm.psm1"
Import-Module "$PSScriptRoot/modules/snapshot.psm1"
Import-Module "$PSScriptRoot/modules/backup.psm1"
Import-Module "$PSScriptRoot/modules/utils.psm1"

$isAdmin =
([Security.Principal.WindowsPrincipal][Security.Principal.WindowsIdentity]::GetCurrent()).IsInRole([Security.Principal.WindowsBuiltInRole]::Administrator)

if (-not $isAdmin) {
    Clear-Host
    Write-Host "===== " -ForegroundColor Yellow
    Write-Host "                УВАГА: ОБМЕЖЕНИЙ РЕЖИМ                " -ForegroundColor Yellow
    Write-Host "===== " -ForegroundColor Yellow
    Write-Host "Скрипт запущено БЕЗ прав адміністратора." -ForegroundColor white
    Write-Host "Для повного доступу (створення/видалення завдань) потрібні права адміна." -ForegroundColor white

    $choice = Read-Host "Бажаєте перезапустити скрипт як Адміністратор? (Y/N)"

    if ($choice -match "^[YyДдТт]$") {
        try {
            Start-Process powershell -ArgumentList "-NoProfile -ExecutionPolicy Bypass -File `"$PSCommandPath`" -Verb RunAs -ErrorAction Stop"
            Exit
        } catch {
            Write-Host "`n[UAC] Відмовлено в доступі. Продовжуємо роботу в обмеженому режимі..." -ForegroundColor Red
            $IsReadOnlyMode = $true
            Start-Sleep -Seconds 2
        }
    } else {
        Write-Host "`nПродовжуємо роботу без прав адміністратора (деякий функціонал обмежений)..." -ForegroundColor Cyan
        $IsReadOnlyMode = $true
        Start-Sleep -Seconds 2
    }
}

do {
    Show-MainMenu
    $choice = Read-Host "Оберіть дію"
    Write-Host "===== "

    switch ($choice) {
        "1" {
            do {
                Show-VMMenu
                $decision = Read-Host "Оберіть дію"
                Write-Host "===== "

                switch($decision) {
                    "1" {
                        Write-Host "`n===== VM Creator ====="

                        $vmName = Read-Host "Введіть назву VM"

                        Write-Host ""
                        $osList = @("Ubuntu_64", "Debian_64", "Fedora_64", "Windows10_64", "Windows11_64", "Other_64")
                        $osType = Get-UserSelection -Collection $osList -Message "Оберіть тип ОС"

                        if (-not $osType) { return }

                        $data = Get-Resources
                        $totalRAM = [int]($data.TotalRAM / (1024 * 1024))
                        $totalDisk = [int]($data.TotalDisk / (1024 * 1024))
                    }
                }
            }
        }
    }
}

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		88

```

$memoryMB = 0
do {
    $rawInput = Read-Host "Оперативна пам'ять (МБ, напр. 4096)"
    $parsedMemory = 0

    if ([int]::TryParse($rawInput, [ref]$parsedMemory)) {
        if ($parsedMemory -lt 3) {
            Write-Host "Помилка! Мінімальний обсяг пам'яті для ВМ –
4 МБ."
        } elseif ($parsedMemory -gt $totalRAM) {
            Write-Host "Помилка! Заданий обсяг пам'яті перевищує
максимальне значення на ПК ($totalRAM МБ)."
        } else {
            $memoryMB = $parsedMemory
        }
    } else {
        Write-Host "Помилка! Введіть коректне ціле число."
    }
} while ($memoryMB -eq 0)

$vramMB = 0
do {
    $rawInput = Read-Host "Відеопам'ять (МБ, від 0 до 256)"
    $parsedVram = 0

    if ([int]::TryParse($rawInput, [ref]$parsedVram) -and
    $parsedVram -ge 0 -and $parsedVram -le 256) {
        $vramMB = $parsedVram
    } else {
        Write-Host "Помилка! Відеопам'ять має бути цілим числом у
діапазоні від 0 до 256 МБ."
    }
} while ($vramMB -eq 0)

$diskSize = 0
do {
    $rawInput = Read-Host "Розмір диску (МБ, напр. 25000)"
    $parsedDisk = 0

    if ([int]::TryParse($rawInput, [ref]$parsedDisk) -and
    $parsedDisk -ge 1 -and $parsedDisk -lt $totalDisk) {
        $diskSize = $parsedDisk
    } else {
        Write-Host "Помилка! Розмір диску має бути числом у діапа-
зоні від 0 до $totalDisk МБ."
    }
} while ($diskSize -eq 0)

Add-Type -AssemblyName System.Windows.Forms
$browser = New-Object System.Windows.Forms.OpenFileDialog
$browser.Filter = "ISO Files (*.iso)|*.iso"

if ($browser.ShowDialog() -eq "OK") {
    $isoPath = $browser.FileName
} else {
    $isoPath = "emptydrive"
}

Write-Host "`nСтворюємо..."
$result = New-VM -Name $vmName -OSType $ostype -Memory $memoryMB -
VRAM $vramMB -DiskSize $diskSize -ISOPath $isoPath
Out-Result $result
Write-Host "====="

}

"2" {
    Write-Host "`n=== List of virtual machines ==="
    $vm = select-vm
    "====="
    if ($vm) {
        do {
            Show-MenuModify
            $ch = Read-Host "Введіть номер параметра"
            Write-Host "====="

            if ($ch -match '^[1-8]$') { Write-Host "`n===== VM"

            switch ($ch) {

                "1" {
                    $newName = Read-Host "Введіть нову назву ВМ"

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		89

```

value $newName                                $result = Edit-VM -Name $vm -Property "--name" -
                                                if ($result.Success) { $vm = $newName }
                                                Out-Result $result
    }
    "2" {
        $osList =
@("Ubuntu_64","Debian_64","Fedora_64","windows10_64","windows11_64","Other_64")
        $osType = Get-UserSelection -Collection $osList -
Message "Оберіть тип ОС"

        if ($osType) {
            $result = Edit-VM -Name $vm -Property "--
ostype" -value $osType
        }
        Out-Result $result
    }
    "3" {
        $data = Get-Resources
        $totalLogicalCPUs = $data.TotalLogicalCPUs

        $newCPUs = 0
        do {
            $rawInput = Read-Host "Введіть кількість проце-
корів (шт, від 1 до $totalLogicalCPUs)"
            $parsedCPUs = 0

            if ([int]::TryParse($rawInput,
[ref]$parsedCPUs)) {
                if ($parsedCPUs -lt 1) {
                    Write-Host "Помилка! Віртуальна машина
повинна мати хоча б 1 процесор."
                } elseif ($parsedCPUs -gt
$totalLogicalCPUs) {
                    Write-Host "Помилка! Не можна виділяти
більше ядер для ВМ, ніж наявно на хості. Максимум для цього ПК: $totalLogicalCPUs."
                } else {
                    $newCPUs = $parsedCPUs
                }
            } else {
                Write-Host "Помилка! Кількість процесорів
має бути цілим числом."
            }
        } while ($newCPUs -eq 0)

        $result = Edit-VM -Name $vm -Property "--cpus" -
        Out-Result $result
    }
    "4" {
        $data = Get-Resources
        $totalRAM = [int]($data.TotalRAM / (1024 * 1024))

        $newMem = 0
        do {
            $rawInput = Read-Host "Введіть новий розмір
оперативної пам'яті (МВ)"
            $parsedMemory = 0

            if ([int]::TryParse($rawInput,
[ref]$parsedMemory)) {
                if ($parsedMemory -lt 3) {
                    Write-Host "Помилка! Мінімальний обсяг
пам'яті для ВМ – 4 МВ."
                } elseif ($parsedMemory -gt $totalRAM) {
                    Write-Host "Помилка! Заданий обсяг па-
м'яті перевищує максимальне значення на ПК ($totalRAM МВ)."
                } else {
                    $newMem = $parsedMemory
                }
            } else {
                Write-Host "Помилка! Введіть коректне ціле
число."
            }
        } while ($newMem -eq 0)
    }

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		90

```

Value $newMem
$result = Edit-VM -Name $vm -Property "--memory" -
Out-Result $result
}
"5" {
    $newVRAM = 0
    do {
        $rawInput = Read-Host "Введіть новий розмір ві-
деопам'яті (MB)"
        $parsedVram = 0
        if ([int]::TryParse($rawInput,
[ref]$parsedVram) -and $parsedVram -ge 0 -and $parsedVram -le 256) {
            $newVRAM = $parsedVram
        } else {
            Write-Host "Помилка! Відеопам'ять має бути
цілим числом у діапазоні від 0 до 256 МБ."
        }
    } while ($newVRAM -eq 0)
}
Value $newVRAM
$result = Edit-VM -Name $vm -Property "--vram" -
Out-Result $result
}
"6" {
    $modes = @("none", "null", "nat", "bridged",
"intnet", "hostonly", "hostonlynet", "generic", "natnetwork", "cloud")
    $selectedMode = Get-UserSelection -Collection
    $modes -Message "Оберіть режим підключення"

    if (-not $selectedMode) { return }
    Write-Host ""
    $option = $null

    if ($selectedMode -eq "bridged") {
        $option = Get-UserSelection -Collection (Get-
HostNetworkInterfaces) -Message "Оберіть фізичну карту"
    } elseif ($selectedMode -eq "hostonly") {
        $option = Get-UserSelection -Collection (Get-
HostNetworkInterfaces hostonly) -Message "Оберіть Host-Only адаптер"
    }

    $result = Set-VMNetwork -Name $vm -Mode
    $selectedMode -OptionValue $option
    Out-Result $result
}
"7" {
    $types = @("Am79C970A", "Am79C973", "82540EM",
"82543GC", "82545EM", "82583V", "virtio", "usbnet")
    $selectedType = Get-UserSelection -Collection
    $types -Message "Оберіть тип адаптера"

    if ($selectedType) {
        $result = Set-VMAdapterType -Name $vm -
        Out-Result $result
    }
}
"8" {
    Write-Host "1. Підключити ISO файл"
    Write-Host "2. ВИТЯГТИ ДИСК (Empty Drive)"

    $isoChoice = Read-Host "`nВаш вибір"

    if ($isoChoice -eq "1") {
        Add-Type -AssemblyName System.Windows.Forms
        $FileBrowser = New-Object
        $FileBrowser.Filter = "ISO Files

system.windows.Forms.OpenFileDialog
(*.iso)|*.iso|All Files (*.*)|*.*"

        if ($FileBrowser.ShowDialog() -eq "OK") {
            $result = Set-VMIso -Name $vm -IsoPath
            Out-Result -Result $result
        }
    } elseif ($isoChoice -eq "2") {

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		91

```

"emptydrive"
$result = Set-VMiso -Name $vm -IsoPath
Out-Result -Result $result
}
}
"0" {
    Write-Host "= VM Modifier is closed ="
}
}
if ($ch -match '[1-8]$') { Write-Host
"===== " }
} while ($ch -ne "0")
Write-Host "===== "
}
}
"3" {
    Write-Host "`n==== VM Launcher ====="
    $vm = Select-VM
    if ($vm) {
        $state = Get-VMState $vm
        if ($state -eq "running"){
            Write-Host "`nВіртуальна машина '$vm' вже працює."
        } else {
            $resources = Get-Resources
            $freeRAM = [int]($resources.FreeRAM / (1024 * 1024))
            $freeDisk = [int]($resources.FreeDisk / (1024 * 1024))
            $totalLogicalCPUs = [int]$resources.TotalLogicalCPUs

            $data = Get-VMDetailedInfo -Name $vm

            $goodRAM = ([int]$data.memory -gt 3 -and [int]$data.memory
-le ($freeRAM - 2048))
            $goodDisk = (($freeDisk - 20480) -ge 0);
            $goodCPUs = ([int]$data.cpus -gt 0 -and [int]$data.cpus -lt
$totalLogicalCPUs);

            if ($goodRAM -eq $false -or $goodDisk -eq $false -or
$goodCPUs -eq $false) {
                Write-Host "Недостатня кількість ресурсів для запуску
ВМ:"

                if ($goodRAM -eq $false) {
                    Write-Host "- Оперативна пам'ять: Прямо зараз на ПК
вільно лише $freeRAM МБ, а для запуску ВМ потрібно $($data.memory) МБ."
                }

                if ($goodDisk -eq $false) {
                    Write-Host "- Місце на диску: На хості залишилось
менше 20 ГБ вільного простору ($freeDisk МБ)."
                }

                if ($goodCPUs -eq $false) {
                    Write-Host "- Кількість процесорів: ВМ потребує
$($data.cpus) ядер, але ПК має всього $totalLogicalCPUs потоків. Не слід запускати ВМ, ви-
діляючи їй усі ядра, інакше система зависне."
                }

                Write-Host "`nчи бажаєте продовжити запуск ВМ?`n1.
Так.`n0. Ні."

                if((Read-Host "Оберіть дію") -ne "1") {
                    Write-Host "===== "
                    break;
                }
            }

            Write-Host "`nчи бажаєте запустити ВМ у фоновому ре-
жимі?`n1. Так.`n0. Ні."

            if((Read-Host "Оберіть дію") -eq "1") {
                $type = "headless"
            } else {
                $type = "gui"
            }

            Write-Host "`nвиберіть режим роботи з ВМ:`n1. З машини-хо-
ста.`n0. через мережу."

            if((Read-Host "Оберіть дію") -eq "1") {

```

Бахур Ю. В.

Головня О. С.

Змн.

Арк.

№ докум.

Підпис

Дата

ККІКБ.КР.Б – 123 – 01 – ПЗ

Арк.

92

```

        $ipaddr = '127.0.0.1'
    } else {
        $ipaddr = '0.0.0.0'
        $adapter = 'TP-Link Wireless USB Adapter'
        $bridged = $false

        $details = Get-VMDetailedInfo -Name $vm
        for ($i = 0; $i -lt 8; $i++) {
            $nickey = "nic$(i + 1)"

            if ($details.$($nickey) -eq 'bridged') {
                $bridged = $true
                break
            }
        }
        if (-not $bridged) {
            Set-VMNetwork -Name $vm -Mode 'bridged' -
AdapterIndex 8 -OptionValue $adapter
        }
        $inputPort = Read-Host "`nВведіть бажаний порт для під-
ключення (5000-65535)"

        if ($inputPort) {
            $numericPort = [int]$inputPort

            if ($numericPort -ge 5000 -and $numericPort -le
65535) {
                $port = $inputPort
            } else {
                Write-Host "Вказано некоректний порт! Виставляю
дефолтний 5000."
                $port = '5000'
            }
        }
        VBoxManage modifyvm $vm --vrde on --vrdeport $port
    }

    VBoxManage modifyvm $vm --vrde-address=$ipaddress
    $result = Start-VM -Name $vm -Type $type
    Out-Result $result

    if ($result.Success -eq $true) {
        Write-Host "`nДля віддаленого підключення через VRDP
використайте: 192.168.1.21:$port"
    }
}
}
Write-Host "======"
}
"4" {
    Write-Host "`n==== VM Shutdowner ====="
    $vm = Select-VM

    if ($vm) {
        $state = Get-VMState $vm
        if ($state -eq "running"){
            Write-Host "`nЧи бажаєте виконати примусове вимкнення
ВМ?`n1. Так.`n0. Ні."

            if ((Read-Host "Оберіть дію") -eq "1") {
                $state = "poweroff"
            } else {
                $state = "acpipowerbutton"
            }

            $result = Stop-VM -Name $vm -State $state
            Out-Result $result
        } else {
            Write-Host "`nВіртуальна машина '$vm' не запущена."
        }
    }
    Write-Host "======"
}
"5" {
    Write-Host "`n==== List of Available VMs ====="
    $vmsList = Get-VMs

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				93
Змн.	Арк.	№ докум.	Підпис	Дата		

```

if ($vmsList.Count -eq 0) {
    Write-Host "Список порожній. Створіть свою першу ВМ."
} else {
    $i = 1
    foreach ($vm in $vmsList) {
        Write-Host "$i. $($vm.Name) { $($vm.UUID)} { $($vm.Status)}"
        $i++
    }
}
Write-Host "======"

"6" {
    Write-Host "`n===== VM Informer ====="
    $vm = Select-VM
    Write-Host "======"

    if ($vm) {
        Write-Host "`n===== VM Information ====="
        $info = Get-VMDetailedInfo -Name $vm

        if ($info) {
            Write-Host "1. Назва           : $($info.name)"
            Write-Host "2. ОС              : $($info.ostype)"
            Write-Host "3. Стан            : $($info.VMState)"
            Write-Host "4. Процесори (CPU) : $($info.cpus)"
            Write-Host "5. Пам'ять (RAM)   : $($info.memory) MB"
            Write-Host "6. Відеопам'ять    : $($info.vram) MB"
            Write-Host "7. Мережа (NIC 1)  : $($info.nic1)"

            Write-Host "8. MAC-адреса      : $($info.macaddress1)"
            Write-Host "9. Папка конфігу    : $($info.CfgFile)"
            Write-Host "10. UUID            : $($info.UUID)"

            Write-Host "======"
        }
    }

    "7" {
        Write-Host "`n===== VM Deleter ====="
        $vm = Select-VM
        if ($vm) {
            Write-Host "`нви впевнені, що бажаєте продовжити видалення ВМ '$vm'?`n1. Так.`n0. Ні."
            $isContinue = (Read-Host "Оберіть дію") -eq "1"

            if($isContinue) {
                $state = Get-VMState $vm
                if ($state -eq "running"){
                    Write-Host "`нВіртуальна машина '$vm' ввімкнена. Чи бажаєте виконати примусове вимкнення?`n1. Так.`n0. Ні."
                    if ((Read-Host "Оберіть дію") -eq "1") {
                        $result = Stop-VM -Name $vm -State "poweroff"

                        if ($result.Success) {
                            Write-Host "`n$($result.Message)" -NoNewline
                            Write-Host " чи бажаєте видалити файли?`n1. Так.`n0. Ні."
                            if((Read-Host "Оберіть дію") -eq "1") {
                                $mode = "--delete-all"
                            } else {
                                $mode = $null
                            }
                            $result = Remove-VM -Name $vm -Delete $mode
                            Out-Result $result
                        } else {
                            Write-Host "`нВидалення віртуальної машини '$vm' перервалося."
                            Write-Host "Помилка: $($result.Message)"
                        }
                    } else {
                        Write-Host "`нВидалення віртуальної машини '$vm' перервано."
                    }
                } else {
                    Write-Host "`нВіртуальна машина '$vm' вимкнена. Чи бажаєте видалити файли?`n1. Так.`n0. Ні."
                }
            }
        }
    }
}

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				94
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        if((Read-Host "Оберіть дію") -eq "1") {
            $mode = "--delete-all"
        } else {
            $mode = $null
        }
        $result = Remove-VM -Name $vm -Delete $mode
        Out-Result $result
    }
} else {
    Write-Host "`nВидалення віртуальної машини '$vm' перер-
вано."
}
}
Write-Host "======"
}
"0" {
    Write-Host "= VM Manager is closed ="
}
}
} while ($decision -ne "0")
Write-Host "======"
}
"2" {
    Write-Host "`n===== Choosing a VM ====="
    $vm = Select-VM
    Write-Host "======"
    if ($vm) {
        do {
            Show-SnapshotMenu
            $decision = Read-Host "Оберіть дію"
            Write-Host "======"
            switch($decision) {
                "1" {
                    Write-Host "`n===== Available Snapshots ====="
                    $snapshotsList = Get-VMSnapshotsList $vm
                    if ($snapshotsList.Count -eq 0) {
                        Write-Host "Список порожній. Створіть свій перший знімок"
                    }
                    else {
                        $i = 1
                        foreach ($snapshot in $snapshotsList) {
                            Write-Host "$i. $($snapshot.Name) {$($snapshot.UUID)}"
                            $i++
                        }
                    }
                }
                "2" {
                    Write-Host "`n===== VM Snapshot Creator ====="
                    $snapName = Read-Host "Введіть назву snapshot"
                    if ($snapName) {
                        $result = New-Snapshot -VMName $vm -Name $snapName
                        Out-Result $result
                    }
                }
                "3" {
                    Write-Host "`n===== VM Snapshot Recoverer ====="
                    $snapName = Select-Snapshot -vmName $vm
                    if ($snapName) {
                        $result = Restore-Snapshot -VMName $vm -Name $snapName
                        Out-Result $result
                    }
                }
                "4" {
                    Write-Host "`n===== VM Snapshot Deleter ====="
                    $snapName = Select-Snapshot -vmName $vm
                    if ($snapName) {
                        $result = Remove-Snapshot -VMName $vm -Name $snapName
                        Out-Result $result
                    }
                }
            }
        }
    }
}

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				95
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    }
    "0" {
        Write-Host "= VM Snapshot Manager is closed ="
    }
}
Write-Host "===== "
} while ($decision -ne "0")
}
"3" {
    $backupFolder = "C:\Backups"
    New-Item -Path $backupFolder -ItemType Directory -Force > $null
    do {
        Show-BackupMenu
        $decision = Read-Host "Оберіть дію"
        Write-Host "===== "

        switch($decision) {
            "1" {
                Write-Host "`n===== VM Backup Creator ====="
                $vm = Select-VM
                if ($vm) {
                    $path = "$backupFolder"
                    $result = Export-VM -VMName $vm -BackupPath $path
                    Out-Result $result
                }
            }
            "2" {
                Write-Host "`n===== VM Backup Restorer ====="
                $files = Get-ChildItem -Path $backupFolder -Filter *.ova
                $selectedFile = Get-UserSelection -Collection ($files) -Message
                "Оберіть бекап для відновлення"
                $file = "$backupFolder\$selectedFile"

                if ($file) {
                    Write-Host "`нБажаєте змінити ім'я ВМ?`n1. Так.`n0. Ні."
                    $toChange = (Read-Host "Оберіть дію") -eq "1"
                    if ($toChange) {
                        $newName = Read-Host "Нове ім'я віртуальної машини"
                        $result = Import-VM -SelectedFile $file -NewName $newName
                    } else {
                        $result = Import-VM -SelectedFile $file
                    }
                    Out-Result $result
                }
            }
            "3" {
                Write-Host "`n===== VM Backup Scheduler ====="
                $vm = Select-VM
                if ($vm) {
                    Write-Host ""
                    $path = "$backupFolder"

                    $types = @("Once", "Daily", "Weekly", "Monthly", "AtStartup",
"АtLogOn")
                    $selectedType = Get-UserSelection -Collection $types -Message
                    "Оберіть періодичність для автоматичного резервного копіювання"
                    if (-not $selectedType) { return }

                    if ($selectedType -eq "AtStartup" -or $selectedType -eq
"АtLogOn") {
                        $result = Register-VMBackupTask -VMName $vm -BackupPath
$path -ScheduleType $selectedType
                    } else {
                        $selectedTime = Read-Host "`нВведіть час та дату в форматі
'РРРР-ММ-ДД ГГ:ХВ' (наприклад: 2026-05-30 17:31)"
                        $result = Register-VMBackupTask -VMName $vm -BackupPath
$path -ScheduleType $selectedType -StartTime $selectedTime
                    }
                    Out-Result $result
                }
            }
            "4" {
                Write-Host "`n===== VM Backup Checker ====="
                $files = Get-ChildItem -Path $backupFolder -Filter *.ova

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				96
Змн.	Арк.	№ докум.	Підпис	Дата		

```

$selectedFile = Get-UserSelection -Collection ($files) -Message
"Оберіть бекап для перевірки"
Write-Host "=====
if ($selectedFile) {
    Write-Host "`nЗачекайте! Виконується тестування..."
    $result = Test-OvaBoot -OvaPath "$backupFolder\$selectedFile"
    Out-Result $result
}
}

"5" {
    Write-Host "`n===== Scheduled Backups ====="
    $backupsList = Get-ScheduledBackups

    if ($backupsList.Count -eq 0) {
        Write-Host "Список порожній. Заплануйте свій перший бекап ВМ."
    } else {
        $tableData = @()
        $i = 1

        foreach ($backup in $backupsList) {
            $resultText = "Інше ($($backup.LastResult))"
            if ($backup.LastResult -eq 0) {
                $resultText = "Успішно (0)"
            }

            $row = [PSCustomObject]@{
                "№" = $i
                "Назва ВМ" = $backup.Name
                "Розклад" = $backup.Schedule
                "Наступний запуск" = $backup.NextRun
                "Останній запуск" = $backup.LastRun
                "Останній результат" = $resultText
            }

            $tableData += $row
            $i++
        }

        $tableData | Format-Table -AutoSize
    }
}

"6" {
    Write-Host "`n== Scheduled Backups Deleter =="
    $backupsList = Get-ScheduledBackups

    $backupsCollection = @()
    foreach ($backup in $backupsList) {
        $backupsCollection += $backup.Name
    }

    $chosenBackup = Get-UserSelection -Collection ($backupsCollection)
    -Message "Оберіть завдання для видалення"

    if ($chosenBackup) {
        $result = Unregister-VMBackupTask -TaskName $chosenBackup
        Out-Result $result
    }
}

"0" {
    Write-Host "= VM Backup Manager is closed ="
}

} if ($decision -match '^[1-6]$') { Write-Host
"=====
} while ($decision -ne "0")
Write-Host "=====
}

"0" {
    Write-Host "= Main Menu is closed ="
}
} while ($choice -ne "0")
Write-Host "=====

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		97

Вміст PowerShell-модулів

Файл vm.psm1

```
Import-Module "$PSScriptRoot/utils.psm1"
```

```
function New-VM {
    param(
        [Parameter(Mandatory=$true)] [string]$Name,
        [Parameter(Mandatory=$true)] [string]$OSType,
        [int]$Memory = 2048,
        [int]$VRAM = 128,
        [int]$DiskSize = 20000,
        [string]$ISOPath = "emptydrive"
    )

    $result = Invoke-VBoxCommand -Command "createvm" -Arguments "--name", $Name, "--
ostype", $OSType, "--register"
    if (-not $result.Success) { return $result }

    $result = Invoke-VBoxCommand -Command "modifyvm" -Arguments $Name, "--memory", $Memory,
"--vram", $VRAM, "--nic1", "nat", "--boot1", "dvd", "--boot2", "disk"
    if (-not $result.Success) { return $result }

    $vdiDir = "$env:USERPROFILE\VirtualBox VMS\$Name"
    New-Item -Path $vdiDir -ItemType Directory -Force > $null
    $vdiPath = "$vdiDir\$Name.vdi"

    $result = Invoke-VBoxCommand -Command "createmedium" -Arguments "disk", "--filename",
$vdiPath, "--size", $DiskSize, "--format", "VDI"
    if (-not $result.Success) { return $result }

    $result = Invoke-VBoxCommand -Command "storagectl" -Arguments $Name, "--name", "SATA
Controller", "--add", "sata", "--controller", "IntelAhci"
    if (-not $result.Success) { return $result }

    $result = Invoke-VBoxCommand -Command "storageattach" -Arguments $Name, "--storagectl",
"SATA Controller", "--port", "0", "--device", "0", "--type", "hdd", "--medium", $vdiPath
    if (-not $result.Success) { return $result }

    # $result = Invoke-VBoxCommand -Command "storagectl" -Arguments $Name, "--name", "IDE
Controller", "--add", "ide"
    # if (-not $result.Success) { return $result }

    if ([string]::IsNullOrEmpty($ISOPath)) {
        $isoTarget = "emptydrive"
    } else {
        $isoTarget = $ISOPath
    }
    $result = Invoke-VBoxCommand -Command "storageattach" -Arguments $Name, "--storagectl",
"SATA Controller", "--port", "1", "--device", "0", "--type", "dvddrive", "--medium",
$isoTarget
    if (-not $result.Success) { return $result }

    $result.Message = "Віртуальну машину '$Name' з ОС '$OSType' успішно створено. \nПараме-
три: \n- RAM $Memory MB. \n- VRAM $VRAM MB. \n- Розмір диску $DiskSize MB. \n- Шлях:
'$vdiDir'. \n- ISO-шлях: '$isoTarget'."
    return $result
}

function Edit-VM {
    param(
        [Parameter(Mandatory=$true)] [string]$Name,
        [Parameter(Mandatory=$true)] [string]$Property,
        [Parameter(Mandatory=$true)] $Value
    )

    $result = Invoke-VBoxCommand -Command "modifyvm" -Arguments $Name, $Property, $Value

    if ($result.Success) {
        $result.Message = "Параметр '$Property' VM '$Name' успішно змінено на '$Value'."
    }
    return $result
}

function Start-VM {
    param(
        [Parameter(Mandatory=$true)]
        [string]$Name,
```

Бахур Ю. В.

Головня О. С.

Змн.

Арк.

№ докум.

Підпис

Дата

ККІКБ.КР.Б – 123 – 01 – ПЗ

Арк.

98

```

    [string]$Type = "gui"
)
$result = Invoke-VBoxCommand -Command "startvm" -Arguments $Name, "--type", $Type
if ($result.Success) {
    $result.Message = "Віртуальну машину '$Name' успішно запущено."
}
return $result
}

function Stop-VM {
    param(
        [Parameter(Mandatory=$true)]
        [string]$Name,

        [string]$State = "acpipowerbutton"
    )

    $result = Invoke-VBoxCommand -Command "controlvm" -Arguments $Name, $State
    if ($result.Success) {
        $result.Message = "Віртуальну машину '$Name' успішно вимкнено."
    }
    return $result
}

function Remove-VM {
    param(
        [Parameter(Mandatory=$true)]
        [string]$Name,

        [string]$Delete = $null
    )

    $result = Invoke-VBoxCommand -Command "unregistervm" -Arguments $Name, $Delete
    if ($result.Success) {
        $result.Message = "Віртуальну машину '$Name' успішно видалено."
    }
    return $result
}

function Set-VMNetwork {
    param(
        [Parameter(Mandatory=$true)] [string]$Name,
        [Parameter(Mandatory=$true)]
        [ValidateSet("none", "null", "nat", "bridged", "intnet", "hostonly", "hostonlynet",
"generic", "natnetwork", "cloud")]
        [string]$Mode,
        [string]$OptionValue = $null, # Назва фізичного або віртуального адаптера
        [int]$AdapterIndex = 1
    )

    $result = Invoke-VBoxCommand -Command "modifyvm" -Arguments "`"$Name`"", "--
nic$AdapterIndex", $Mode

    if (-not $result.Success) { return $result }

    switch ($Mode) {
        "bridged" {
            if (-not $OptionValue) { return @{ Success=$false; Message="Для режиму Bridged
потрібно вказати інтерфейс." } }
            $result = Invoke-VBoxCommand -Command "modifyvm" -Arguments $Name, "--
bridgeadapter$AdapterIndex", $OptionValue
        }
        "hostonly" {
            if (-not $OptionValue) { return @{ Success=$false; Message="Для режиму Host-
Only потрібно вказати адаптер." } }
            $result = Invoke-VBoxCommand -Command "modifyvm" -Arguments $Name, "--
hostonlyadapter$AdapterIndex", $OptionValue
        }
        "intnet" {
            if ($OptionValue) {
                $result = Invoke-VBoxCommand -Command "modifyvm" -Arguments $Name, "--
intnet$AdapterIndex", $OptionValue
            }
        }
    }
}

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				99
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    }
}

if ($result.Success) {
    if($OptionValue) {
        $result.Message = "Режим підключення мережі ВМ '$Name' успішно змінено на '$Mode' (Адаптер: $OptionValue)."
    } else {
        $result.Message = "Режим підключення мережі ВМ '$Name' успішно змінено на '$Mode'."
    }
}

return $result
}

function Set-VMAdapterType {
    param(
        [Parameter(Mandatory=$true)] [string]$Name,
        [int]$AdapterIndex = 1,
        [Parameter(Mandatory=$true)]
        [ValidateSet("Am79C970A", "Am79C973", "82540EM", "82543GC", "82545EM", "82583V",
"virtio", "usbnet")]
        $AdapterType
    )

    $typeArg = "--nictype$AdapterIndex"
    $result = Invoke-VBoxCommand -Command "modifyvm" -Arguments $Name, $typeArg,
$AdapterType

    if ($result.Success) {
        $result.Message = "Тип адаптера мережі ВМ '$Name' успішно змінено на '$AdapterType'."
    }

    return $result
}

function Set-VMIso {
    param(
        [Parameter(Mandatory=$true)] [string]$Name,
        [Parameter(Mandatory=$false)] [string]$IsoPath = "emptydrive",
        [string]$ControllerName = "SATA Controller"
    )

    if ($IsoPath -eq "" -or $IsoPath -eq "emptydrive") {
        $target = "emptydrive"
    } else {
        $target = $IsoPath
    }

    $result = Invoke-VBoxCommand -Command "storageattach" -Arguments $Name,
        "--storagectl", $ControllerName,
        "--port", "1", "--device", "0",
        "--type", "dvd drive",
        "--medium", $target

    if ($result.Success) {
        if ($target -eq "emptydrive") {
            $result.Message = "Привід ВМ '$Name' успішно очищено."
        } else {
            $result.Message = "Образ '$IsoPath' підключено до ВМ '$Name'."
        }
    }

    return $result
}

Export-ModuleMember -Function New-VM, Edit-VM, Start-VM, Stop-VM, Remove-VM, Set-VMNetwork,
Set-VMAdapterType, Set-VMIso

```

Файл snapshot.psm1

```
Import-Module "$PSScriptRoot/utils.psm1"
```

```

function New-Snapshot {
    param(
        [Parameter(Mandatory=$true)]
        [string]$VMName,

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		100

```

        [Parameter(Mandatory=$true)]
        [string]$Name
    )

    $result = Invoke-VBoxCommand -Command "snapshot" -Arguments $VMName, "take", $Name
    if ($result.Success) {
        $result.Message = "Знімок '$Name' віртуальної машини '$VMName' успішно створено."
    }

    return $result
}

function Restore-Snapshot {
    param(
        [Parameter(Mandatory=$true)]
        [string]$VMName,

        [Parameter(Mandatory=$true)]
        [string]$Name,

        [string]$UUID
    )

    if ($UUID) {
        $snapName = $UUID
        $outputMessage = " ($UUID)"
    } else {
        $snapName = $Name
        $outputMessage = ""
    }

    $result = Invoke-VBoxCommand -Command "snapshot" -Arguments $VMName, "restore",
    $snapName

    if ($result.Success) {
        $result.Message = "Стан віртуальної машини '$VMName'$outputMessage успішно віднов-
лено за знімком '$Name'."
    }

    return $result
}

function Remove-Snapshot {
    param(
        [Parameter(Mandatory=$true)]
        [string]$VMName,

        [Parameter(Mandatory=$true)]
        [string]$Name,

        [string]$UUID
    )

    if ($UUID) {
        $snapName = $UUID
        $outputMessage = " ($UUID)"
    } else {
        $snapName = $Name
        $outputMessage = ""
    }

    $result = Invoke-VBoxCommand -Command "snapshot" -Arguments $VMName, "delete",
    $snapName

    if ($result.Success) {
        $result.Message = "Знімок '$Name'$outputMessage віртуальної машини '$VMName' успі-
шно видалено."
    }

    return $result
}

Export-ModuleMember -Function New-Snapshot, Restore-Snapshot, Remove-Snapshot

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		101

Файл backup.psm1

```
Import-Module "$PSScriptRoot/utils.psm1"
```

```
function Export-VM {
```

```
    param(
```

```
        [Parameter(Mandatory=$true)] [string] $VMName,
```

```
        [Parameter(Mandatory=$true)] [string] $BackupPath
```

```
    )
```

```
    $date = Get-Date -Format "yyyy-MM-dd_HH-mm-ss"
```

```
    $backupName = "$BackupPath\${VMName}_$date.ova"
```

```
    $result = Invoke-VBoxCommand -Command "export" -Arguments $VMName, "--output",  
$BackupName, "--manifest", "--options=manifest"
```

```
    if ($result.Success) {
```

```
        $result.Message = "Бекап VM '$VMName' успішно створено. `н шлях до файлу:  
$BackupName."
```

```
    }
```

```
    return $result
```

```
}
```

```
function Import-VM {
```

```
    param(
```

```
        [Parameter(Mandatory=$true)] [string] $SelectedFile,
```

```
        [string] $NewName
```

```
    )
```

```
    if($NewName) {
```

```
        $result = Invoke-VBoxCommand -Command "import" -Arguments $selectedFile, "--vsys",  
"0", "--vmname", $NewName, "--eula=accept"
```

```
    } else {
```

```
        $result = Invoke-VBoxCommand -Command "import" -Arguments $selectedFile, "--vsys",  
"0", "--eula=accept"
```

```
    }
```

```
    if ($result.Success) {
```

```
        $result.Message = "VM успішно відновлено з бекапу. `н шлях до файлу: $SelectedFile."
```

```
    }
```

```
    return $result
```

```
}
```

```
function Register-VMBackupTask {
```

```
    param(
```

```
        [Parameter(Mandatory=$true)] [string] $VMName,
```

```
        [Parameter(Mandatory=$true)] [string] $BackupPath,
```

```
        [Parameter(Mandatory=$false)] [string] $ScheduleType = "Daily",
```

```
        [Parameter(Mandatory=$false)] [string] $StartTime = "02:00"
```

```
    )
```

```
    $ModulePathBackup = (Resolve-Path "./modules/backup.psm1" -ErrorAction  
SilentlyContinue).Path
```

```
    if (-not $ModulePathBackup) {
```

```
        $ModulePathBackup = "C:\Users\bakhu\Desktop\учб\4 курс\2 семестр\кваліфікаційна ро-  
бота\modules\backup.psm1"
```

```
    }
```

```
    $ModulePathUtils = (Resolve-Path "./modules/utils.psm1" -ErrorAction  
SilentlyContinue).Path
```

```
    if (-not $ModulePathUtils) {
```

```
        $ModulePathUtils = "C:\Users\bakhu\Desktop\учб\4 курс\2 семестр\кваліфікаційна ро-  
бота\modules\utils.psm1"
```

```
    }
```

```
    $ScriptBlock = "Import-Module '$ModulePathUtils'; Import-Module '$ModulePathBackup';  
Export-VM -VMName '$VMName' -BackupPath '$BackupPath'"
```

```
    $TaskArguments = "-NoProfile -windowStyle Hidden -Command `"$ScriptBlock`""
```

```
    $ParsedTime = Get-Date $StartTime
```

```
    $TriggerArgs = @{}
```

```
    switch ($ScheduleType) {
```

```
        "Once" {
```

```
            $TriggerArgs.Add('Once', $true)
```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		102

```

        $TriggerArgs.Add('At', $ParsedTime)
    }
    "Daily" {
        $TriggerArgs.Add('Daily', $true)
        $TriggerArgs.Add('At', $ParsedTime)
    }
    "weekly" {
        $DayOfWeek = $ParsedTime.DayOfWeek.ToString()

        $TriggerArgs.Add('weekly', $true)
        $TriggerArgs.Add('At', $ParsedTime)
        $TriggerArgs.Add('DaysOfWeek', $DayOfWeek)
    }
    "Monthly" {
        $DayOfWeek = $ParsedTime.DayOfWeek.ToString()

        $TriggerArgs.Add('weekly', $true)
        $TriggerArgs.Add('At', $ParsedTime)
        $TriggerArgs.Add('DaysOfWeek', $DayOfWeek)
        $TriggerArgs.Add('WeeksInterval', 4)
    }
    "AtStartup" {
        $TriggerArgs.Add('AtStartup', $true)
    }
    "AtLogOn" {
        $TriggerArgs.Add('AtLogOn', $true)
    }
    default {
        $TriggerArgs.Add('Daily', $true)
        $TriggerArgs.Add('At', (Get-Date $StartTime))
    }
}

$Trigger = New-ScheduledTaskTrigger @TriggerArgs
if ($ScheduleType -eq 'AtStartup' -or $ScheduleType -eq 'AtLogOn') {
    $resMessage = "$ScheduleType"
} else {
    $resMessage = "$ScheduleType, починаючи з $StartTime"
}

$Action = New-ScheduledTaskAction -Execute "powershell.exe" -Argument $TaskArguments

$TaskName = "VM_Backup_{$VMName}"
$Settings = New-ScheduledTaskSettingsSet -AllowStartIfOnBatteries -
DontStopIfGoingOnBatteries

try {
    Register-ScheduledTask -TaskName $TaskName -Trigger $Trigger -Action $Action -
Settings $Settings -User $env:USERNAME -Force -ErrorAction Stop | Out-Null

    return @{ "Success" = $true; "Message" = "Регулярне завдання '$TaskName' успішно
створено. Графік: $resMessage." }
} catch {
    $errorMessage = $_.ToString().Trim()

    if ($errorMessage -like "*Access is denied*") {
        $additionalMessage = "Відмовлено в доступі. Переконайтеся, що FastAPI сервер
запущено від імені Адміністратора. n"
    }

    return @{ "Success" = $false; "Message" = "{$additionalMessage}помилка:
$errorMessage" }
}

function Test-OvaBoot {
    param (
        [string]$OvaPath
    )

    $epoch = [DateTimeOffset]::UtcNow.ToUnixTimeSeconds()
    $testVmName = "Backup-Test_{$epoch}"
    $statusFilePath = "{$OvaPath}.json"

    try {
        $importRes = VBoxManage import $OvaPath --vsys 0 --vmname $testVmName --eula=accept
        $importOutput = $importRes -join "`n"
        if ($LASTEXITCODE -ne 0) {

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		103

```

if ($importOutput -like '*VERR_DISK_FULL*' -or $importOutput -like '*Ran out of
disk space*') {
    $jsonContent = @{} | ConvertTo-Json
    Set-Content -Path $statusFilePath -value $jsonContent -Encoding UTF8

    return @{
        Success = $false
        Message = "Тестування перервано: На комп'ютері закінчилося місце на
диску! Не вдалося розгорнути ВМ. Актуальний стан бекапу: Невідомо."
    }
}

if ($importOutput -like '*Digest mismatch*' -or $importOutput -like
'*VERR_TAR_BAD_CHKSUM*' -or $importOutput -like '*VERR_TAR_CHKSUM_MISMATCH*' -or
$importOutput -like '*Appliance read failed*' -or ($importOutput -like
'*VBOX_E_FILE_ERROR*' -and $importOutput -notlike '*VERR_DISK_FULL*')) {
    $jsonContent = @{} | ConvertTo-Json
    Set-Content -Path $statusFilePath -value $jsonContent -Encoding UTF8
    return @{ Success = $false; Message = "Бекап пошкоджено! Структура архіву
або контрольні суми файлів (Digest mismatch) не пройшли перевірку." }
} else {
    $jsonContent = @{} | ConvertTo-Json
    Set-Content -Path $statusFilePath -value $jsonContent -Encoding UTF8

    return @{
        Success = $false
        Message = "Не вдалося виконати імпорт через технічну помилку оточення
гіпервізора. Статус бекапу не змінено. Деталі: $importOutput"
    }
}
}

$startRes = VBoxManage startvm $testVmName --type headless 2>&1
$startOutput = $startRes -join "`n"
if ($LASTEXITCODE -ne 0) {
    $null = VBoxManage controlvm $testVmName poweroff 2>&1
    Start-Sleep -Seconds 2
    $null = VBoxManage unregistervm $testVmName --delete-all 2>&1

    if ($startOutput -like '*NO_MEMORY*' -or $startOutput -like '*OUT_OF_MEMORY*')
{
        $jsonContent = @{} | ConvertTo-Json
        Set-Content -Path $statusFilePath -value $jsonContent -Encoding UTF8

        return @{
            Success = $false
            Message = "Не вдалося запустити тест через брак оперативної пам'яті
(RAM) на хост-машині. Спробуйте закрити інші програми."
        }
    } elseif ($startOutput -like '*session was aborted*' -or $startOutput -like
'*E_FAIL (0x80004005)*') {
        $jsonContent = @{} | ConvertTo-Json
        Set-Content -Path $statusFilePath -value $jsonContent -Encoding UTF8

        return @{
            Success = $false
            Message = "Помилка конфігурації: Віртуальний мережевий адаптер, описа-
ний в бекапі, відсутній на цьому комп'ютері (Session aborted). Для тесту знадобиться пере-
визначити мережеву карту."
        }
    } else {
        $jsonContent = @{} | ConvertTo-Json
        Set-Content -Path $statusFilePath -value $jsonContent -Encoding UTF8

        return @{ Success = $false; Message = "Машина не змогла стартувати через
внутрішнє пошкодження конфігурації бекапу." }
    }
}

$bootSuccessful = $false
$maxAttempts = 12

for ($attempt = 1; $attempt -le $maxAttempts; $attempt++) {
    Start-Sleep -Seconds 5

    $guestRes = VBoxManage guestproperty enumerate $testVmName 2>&1
    $guestOutput = $guestRes -join "`n"

    $stateRes = VBoxManage showvminfo $testVmName --machinereadable 2>&1
    $stateOutput = $stateRes -join "`n"

```

```

        if ($stateOutput -like '*VMState="running"*') {
            if ($guestOutput -like '*/VirtualBox/GuestInfo/OS/Release*' -or $attempt -
gt 6) {
                $bootSuccessful = $true
                break
            }
        }
    }

    $null = VBoxManage controlvm $testVmName poweroff 2>&1
    Start-Sleep -Seconds 2
    $null = VBoxManage unregistervm $testVmName --delete-all 2>&1

    if ($bootSuccessful) {
        $jsonContent = @{ Status = "Valid" } | ConvertTo-Json
        Set-Content -Path $statusFilePath -Value $jsonContent -Encoding UTF8

        return @{
            Success = $true
            Message = "Тест завантаження успішний! Віртуальна машина успішно ініціалі-
зувалася у фоновому режимі, ядро ОС стартувало."
        }
    } else {
        $jsonContent = @{ Status = "Corrupted" } | ConvertTo-Json
        Set-Content -Path $statusFilePath -Value $jsonContent -Encoding UTF8

        return @{
            Success = $false
            Message = "Машина запустилася, але ОС не подала ознак життя за відведений
час (можливий збій завантажувача або BSOD/kernel panic)."
        }
    }
}
catch {
    $null = VBoxManage unregistervm $testVmName --delete-all 2>&1

    $jsonContent = @{ Status = "Unknown" } | ConvertTo-Json
    Set-Content -Path $statusFilePath -Value $jsonContent -Encoding UTF8

    return @{ Success = $false; Message = "Критичний збій скрипта валідації:
$(($_.Exception.Message))" }
}

function Unregister-VMBackupTask {
    param(
        [Parameter(Mandatory=$true)] [string]$TaskName
    )

    $taskExists = Get-ScheduledTask -TaskName $TaskName -ErrorAction SilentlyContinue

    if (-not $taskExists) {
        return @{
            Success = $false
            Message = "Завдання з назвою '$TaskName' не знайдено в системі."
        }
    }

    try {
        2>$null Unregister-ScheduledTask -TaskName $TaskName -Confirm:$false -ErrorAction Stop

        $checkDeleted = Get-ScheduledTask -TaskName $TaskName -ErrorAction SilentlyContinue

        if (-not $checkDeleted) {
            return @{
                Success = $true
                Message = "Завдання '$TaskName' успішно видалено з планувальника windows."
            }
        } else {
            return @{
                Success = $false
                Message = "Не вдалося видалити завдання. Можливо, воно заблоковане систе-
мою."
            }
        }
    }
    catch {
        return @{
            Success = $false

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				105
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        Message = "Критичний збій при видаленні завдання: $($_.Exception.Message)"
    }
}

Export-ModuleMember -Function Export-VM, Import-VM, Register-VMBackupTask, Test-OvaBoot,
Unregister-VMBackupTask

```

Файл menu.psm1

```

function Show-MainMenu {
    Write-Host "`n===== Main Menu ====="
    Write-Host "1. Керування ВМ"
    Write-Host "2. Керування знімками (Snapshots)"
    Write-Host "3. Резервне копіювання (Backup)"
    Write-Host "0. Завершити роботу`n"
}

function Show-VMMenu {
    Write-Host "`n===== VM Manager ====="
    Write-Host "1. Створити нову ВМ"
    Write-Host "2. Змінити параметри існуючої ВМ"
    Write-Host "3. Запустити ВМ"
    Write-Host "4. Вимкнути ВМ"
    Write-Host "5. Вивести список наявних ВМ"
    Write-Host "6. Вивести інформацію про певну ВМ"
    Write-Host "7. Видалити ВМ"
    Write-Host "0. Повернутися в головне меню`n"
}

function Show-MenuModify {
    Write-Host "`n===== VM Modifier ====="
    Write-Host "1. Назва ВМ"
    Write-Host "2. Тип ОС"
    Write-Host "3. Кількість CPU"
    Write-Host "4. Оперативна пам'ять (MB)"
    Write-Host "5. Відеопам'ять (MB)"
    Write-Host "6. Режим підключення (NAT/Bridged)"
    Write-Host "7. Тип адаптера (Virtio/Intel)"
    Write-Host "8. DVD-привід"
    Write-Host "0. Повернутися в головне меню`n"
}

function Show-SnapshotMenu {
    Write-Host "`n===== VM Snapshot Manager ====="
    Write-Host "1. Вивести список наявних знімків ВМ"
    Write-Host "2. Створити новий знімок"
    Write-Host "3. Відновити стан ВМ за знімком"
    Write-Host "4. Видалити знімок"
    Write-Host "0. Повернутися в головне меню`n"
}

function Show-BackupMenu {
    Write-Host "`n===== VM Backup Manager ====="
    Write-Host "1. Створити повний бекап"
    Write-Host "2. Відновити систему з бекапу"
    Write-Host "3. Налаштувати розклад резервування"
    Write-Host "4. Перевірити бекап"
    Write-Host "5. Вивести список запланованих бекапів"
    Write-Host "6. Видалити заплановане завдання"
    Write-Host "0. Повернутися в головне меню`n"
}

function Get-UserSelection {
    param(
        [Parameter(Mandatory=$true)]
        [AllowEmptyCollection()]
        $Collection,
        [string]$Message = "Оберіть номер елемента",
        [string]$EmptyMessage = "Список порожній"
    )

    if (-not $Collection -or $Collection.Count -eq 0) {
        Write-Host $EmptyMessage
        return $null
    }

    for ($i = 0; $i -lt $Collection.Count; $i++) {
        $displayName = $Collection[$i].Name -split ' ' | where-object { $_ } | select-
Object -First 1
    }
}

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		106

```

        if (-not $displayName) { $displayName = $Collection[$i] }
        write-Host "$($i + 1). $displayName"
    }
    $input = (Read-Host "`n$Message").Trim()
    if ([int]::TryParse($input, [ref]$choice)) {
        if ($choice -ge 1 -and $choice -le $Collection.Count) {
            return $Collection[$choice - 1]
        }
    }
    write-Host "Неправильний вибір."
    return $null
}

function Select-VM {
    $vms = @(Get-VMs)
    $selected = Get-UserSelection -Collection $vms -Message "Оберіть ВМ" -EmptyMessage "Немає доступних ВМ"
    return $selected.Name
}

function Select-Snapshot {
    param($VMName)

    $snapshots = @(Get-VMSnapshotsList -VMName $VMName)
    $selected = Get-UserSelection -Collection $snapshots -Message "Оберіть знімок" -EmptyMessage "У цієї ВМ немає знімків"

    return $selected.Name
}

function Out-Result {
    param(
        [Parameter(Mandatory=$true)]
        $Result
    )

    if ($Result.Success) {
        write-Host "`n$($Result.Message)"
    } else {
        write-Host "`нПомилка: $($Result.Message)"
    }
}

Export-ModuleMember -Function Show-MainMenu, Show-VMMenu, Show-MenuModify, Show-SnapshotMenu, Show-BackupMenu, Get-UserSelection, Select-VM, Select-Snapshot, Out-Result

```

Файл utils.psm1

```

function Invoke-VBoxCommand {
    param(
        [Parameter(Mandatory=$true)]
        [string]$Command,

        [Parameter(ValueFromRemainingArguments=$true)]
        [string[]]$Arguments
    )

    $output = & VBoxManage $Command $Arguments 2>&1

    $result = [PSCustomObject]@{
        Success = ($LASTEXITCODE -eq 0)
        ExitCode = $LASTEXITCODE
        Message = ($output -join "`n").Trim()
        Timestamp = Get-Date -Format "yyyy-MM-dd HH:mm:ss"
    }

    return $result
}

function Get-VMs {
    $rawVms = VBoxManage list vms
    if (-not $rawVms) { return @() }

    $vms = foreach ($line in $rawVms) {
        if ($line -match '".{1,}(.+)" \{(.+)\}') {
            $name = $matches[1]
        }
    }
}

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				107
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        $uuid = $matches[2]

        $vmInfo = VBoxManage showvminfo $uuid --machinereadable
        $stateLine = $vmInfo | Select-String "VMState="

        $status = ($stateLine.ToString() -split '')[1]

        $friendlyStatus = switch ($status) {
            "running"      { "Running" }
            "poweroff"     { "Powered off" }
            "paused"       { "Paused" }
            "saved"        { "Saved" }
            "aborted"      { "Aborted" }
            default        { $status }
        }

        [PSCustomObject]@{
            Name     = $name
            UUID     = $uuid
            Status   = $friendlyStatus
        }
    }
}

return @($vms)
}

function Get-VMState {
    param([Parameter(Mandatory=$true)]$VMName)

    $info = VBoxManage showvminfo $VMName --machinereadable 2>&1
    if ($LASTEXITCODE -ne 0) { return "unknown" }

    $stateLine = $info | Select-String "VMState="
    return ($stateLine.ToString() -split '')[1]
}

function Get-VMSnapshotsList {
    param([Parameter(Mandatory=$true)]$VMName)

    $raw = VBoxManage snapshot "$VMName" list --machinereadable 2>&1
    if ($LASTEXITCODE -ne 0 -or -not $raw) { return @() }

    $snapshots = @()
    $data = @{}

    $i = 0
    foreach ($line in $raw) {
        if ($line -match '^Snapshot(Name|UUID)(.*)"(.+)"') {
            $type = $matches[1] # Name або UUID
            $value = $matches[3] # Саме значення в лапках

            if (-not $data.ContainsKey($i)) { $data[$i] = @{} }

            $data[$i].$type = $value
            if ($type -eq "UUID") { $i++ }
        }
    }

    foreach ($k in ($data.Keys | Sort-Object { [int]$_ })) {
        $snapshots += [PSCustomObject]@{
            Name = $data[$k].Name
            UUID = $data[$k].UUID
        }
    }

    return $snapshots
}

function Get-VMDetailedInfo {
    param([Parameter(Mandatory=$true)]$Name)

    $rawInfo = VBoxManage showvminfo $Name --machinereadable 2>&1

    if ($LASTEXITCODE -ne 0) { return $null }

    $infoTable = @{}
    foreach ($line in $rawInfo) {
        if ($line -match '^"?(["=]+)"?="?("[^"]*)"?'$') {
            $key = $matches[1]
            $value = $matches[2]
        }
    }

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		108

```

        $infoTable[$key] = $value
    }
}

return [PSCustomObject]$infoTable
}

function Get-HostNetworkInterfaces {
    param(
        [string]$Type = "bridged"
    )

    if ($Type -eq "hostonly") {
        $rawList = VBoxManage list hostonlyifs
    } else {
        $rawList = VBoxManage list bridgedifs
    }

    $interfaces = @()

    foreach ($line in $rawList) {
        if ($line -match '^Name:\s+(.+)$') {
            $interfaces += $matches[1].Trim()
        }
    }

    return $interfaces
}

function Get-IsoPath {
    Add-Type -AssemblyName System.Windows.Forms
    $browser = New-Object System.Windows.Forms.OpenFileDialog
    $browser.Filter = "ISO Files (*.iso)|*.iso"
    $browser.Title = "Обрати ISO образ"

    $browser.ShowHelp = $true

    if ($browser.ShowDialog() -eq "OK") {
        return @{ Success = $true; Path = $browser.FileName }
    } else {
        return @{ Success = $false; Path = "emptydrive" }
    }
}

function Get-Drives {
    $list = Get-PSDrive -PSProvider FileSystem | Sort-Object Name

    $drives = @()

    foreach ($drive in $list) {
        $freeGB = "--"
        if ($drive.Free -ne $null) {
            $freeGB = "$([math]::round($drive.Free / 1GB, 2)) GB"
        }

        $usedGB = "--"
        if ($drive.Used -ne $null) {
            $usedGB = "$([math]::round($drive.Used / 1GB, 2)) GB"
        }

        $drives += [PSCustomObject]@{
            Name      = $drive.Name
            Used       = $usedGB
            Free       = $freeGB
            Provider   = $drive.Provider.Name
            Root       = $drive.Root
            Type       = "Folder"
        }
    }

    return $drives
}

function Get-FolderChildren {
    param([string]$Path)
    try {
        $items = @(Get-ChildItem -Path $Path | Select-Object Name,
            @{Name="Type"; Expression={if($_.PSIsContainer){"Folder"}else{"File"}}},
            @{Name="Size"; Expression={
                if ($_.PSIsContainer) {
                    "--"
                } else {
                    $len = $_.Length
                }
            }}
        )
    } catch {
        $items = @()
    }
}

```

		Бахур Ю. В.			ККИКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		109

```

        if ($len -ge 1GB) {
            "$([math]::round($len / 1GB, 2)) GB"
        } elseif ($len -ge 1MB) {
            "$([math]::round($len / 1MB, 2)) MB"
        } elseif ($len -ge 1KB) {
            "$([math]::round($len / 1KB, 2)) KB"
        } else {
            "$len Bytes"
        }
    }
}

},
    @{Name="LastWriteTime"; Expression={$_.LastWriteTime.ToString("dd/MM/yyyy
HH:mm")}}}
    return $items
} catch {
    return @()
}
}

function Get-Resources {
    param([string]$Drive)
    if (-not $Drive) { $Drive = 'C' }

    $ComputerInfo = Get-CimInstance -ClassName Win32_ComputerSystem
    $TotalRAM = [int64]$ComputerInfo.TotalPhysicalMemory
    $OSInfo = Get-CimInstance -ClassName Win32_OperatingSystem
    $FreeRAM = [int64]$OSInfo.FreePhysicalMemory * 1024

    $DiskInfo = Get-CimInstance -ClassName Win32_LogicalDisk -Filter "DeviceID='${Drive}':"
    $TotalDisk = [int64]$DiskInfo.Size
    $FreeDisk = $DiskInfo.FreeSpace

    $CPUInfo = Get-CimInstance -ClassName Win32_Processor
    $TotalLogicalCPUs = [int]$CPUInfo.NumberOfLogicalProcessors

    return [PSCustomObject]@{
        TotalRAM = $TotalRAM
        FreeRAM = $FreeRAM
        TotalDisk = $TotalDisk
        FreeDisk = $FreeDisk
        TotalLogicalCPUs = $TotalLogicalCPUs
    }
}

function Get-BackupsList {
    param (
        [string]$BackupPath
    )

    if (-not (Test-Path $BackupPath)) {
        return "[]"
    }

    $backups = @()

    $ovaFiles = Get-ChildItem -Path $BackupPath -Filter "*.ova"

    foreach ($file in $ovaFiles) {
        $validationStatus = "Unverified"

        $statusFilePath = "$($file.FullName).json"

        if (Test-Path $statusFilePath) {
            try {
                $meta = Get-Content -Raw -Path $statusFilePath | ConvertFrom-Json
                if ($meta.Status) {
                    $validationStatus = $meta.Status
                }
            } catch {
                $validationStatus = "Error Reading Status"
            }
        }

        $sizeStr = ""
        $len = $file.Length
        if ($len -ge 1GB) {
            $sizeStr = "$([math]::round($len / 1GB, 2)) GB"
        } elseif ($len -ge 1MB) {
            $sizeStr = "$([math]::round($len / 1MB, 2)) MB"
        }
    }
}

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		110

```

    } elseif ($len -ge 1KB) {
        $sizeStr = "$([math]::round($len / 1KB, 2)) KB"
    } else {
        $sizeStr = "$len Bytes"
    }

    $backupItem = [PSCustomObject]@{
        Name      = $file.Name
        Type      = "OVA Appliance"
        Size      = $sizeStr
        Modified   = $file.LastWriteTime.ToString("dd/MM/yyyy HH:mm")
        Validity   = $validationStatus
    }

    $backups += $backupItem
}

return $backups
}

function Get-ScheduledBackups {
    $list = @()
    $tasks = Get-ScheduledTask -TaskName "VM_Backup_*"

    foreach ($task in $tasks) {
        $name = $task.TaskName
        $info = Get-ScheduledTaskInfo -TaskName $name

        $triggerNames = foreach ($trigger in $task.Triggers) {
            $className = $trigger.CimClass.CimClassName

            switch -wildcard ($className) {
                "*Daily*" { "Щодня" }
                "*weekly*" {
                    if ($trigger.WeeksInterval -gt 1) {
                        $days = $trigger.WeeksInterval * 7
                        "кожні $days дн. (раз на $($trigger.WeeksInterval) тиж.)"
                    } else {
                        "Щотижня"
                    }
                }
                "*Monthly*" { "Щомісяця" }
                "*Once*" { "Одноразово" }
                "*Boot*" { "При запуску операційної системи (ПК)" }
                "*Logon*" { "При вході користувача у систему" }
                default { "За розкладом" }
            }
        }

        $scheduleText = $triggerNames -join ", "

        $item = [PSCustomObject]@{
            Name      = $name
            State     = $task.State.ToString()
            Schedule  = $scheduleText
            NextRun   = if ($info.NextRunTime) { $info.NextRunTime.ToString("dd/MM/yyyy HH:mm") } else { "Н/д" }
            LastRun   = if ($info.LastRunTime) { $info.LastRunTime.ToString("dd/MM/yyyy HH:mm") } else { "Н/д" }
            LastResult = $info.LastTaskResult
        }

        $list += $item
    }

    return $list
}

```

Export-ModuleMember -Function Invoke-VBoxCommand, Get-VMs, Get-VMState, Get-VMSnapshotsList, Get-VMDetailedInfo, Get-HostNetworkInterfaces, Get-IsoPath, Get-Drives, Get-FolderChildren, Get-Resources, Get-BackupsList, Get-ScheduledBackups

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		111

Вміст файлу main.py

```

import asyncio
import ctypes
import os
import socket
import subprocess
import json
import sys
from typing import Optional

from fastapi import FastAPI, Request, Body
from fastapi.responses import HTMLResponse
from fastapi.templating import Jinja2Templates
from pydantic import BaseModel
from fastapi.staticfiles import StaticFiles

app = FastAPI()
templates = Jinja2Templates(directory="templates")
app.mount("/static", StaticFiles(directory="static"), name="static")

# uvicorn main:app --host 0.0.0.0 --port 8000 --reload
# uvicorn main:app --reload

async def run_ps(cmd, module="utils"):
    utf8_fix = '$OutputEncoding = [Console]::OutputEncoding = [System.Text.Encoding]::UTF8;'
    import_cmd = f"Import-Module ./modules/{module}.psm1;"
    full_cmd = f"{utf8_fix} {import_cmd} {cmd} | ConvertTo-Json"

    try:
        proc = await asyncio.create_subprocess_exec(
            "powershell", "-Command", full_cmd,
            stdout=asyncio.subprocess.PIPE,
            stderr=asyncio.subprocess.PIPE
        )

        stdout, stderr = await asyncio.wait_for(proc.communicate(), timeout=300)

        stdout_text = stdout.decode('utf-8', errors='ignore')
        stderr_text = stderr.decode('utf-8', errors='ignore')

        if stderr_text.strip():
            print(f"PowerShell Error: {stderr_text}")

        if not stdout_text.strip():
            return None

        return json.loads(stdout_text)
    except asyncio.TimeoutError:
        return {"Success": False, "Message": "Перевищено ліміт часу очікування операції (timeout)."}
    except Exception as e:
        return {"Success": False, "Message": f"Помилка виклику підпроцесу: {str(e)}"}

@app.get("/", response_class=HTMLResponse)
async def index(request: Request):
    return templates.TemplateResponse(name="vm.html", request=request)

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		112

```

@app.get("/backup", response_class=HTMLResponse)
async def backup(request: Request):
    return templates.TemplateResponse(name="backup.html", request=request)

@app.get("/snapshot", response_class=HTMLResponse)
async def snapshot(request: Request):
    return templates.TemplateResponse(name="snapshot.html", request=request)

@app.get("/api/vms")
async def get_vms():
    return await run_ps(cmd="Get-VMs", module="utils")

class VMCreateRequest(BaseModel):
    from_backup: bool = False

    name: Optional[str] = None
    os: Optional[str] = None
    memory: Optional[int] = None
    vram: Optional[int] = None
    diskSize: Optional[int] = None
    iso: Optional[str] = None

    new_name: Optional[str] = None
    path: Optional[str] = None
    file_name: Optional[str] = None

@app.post("/api/vms")
async def create_or_import_vm(data: VMCreateRequest):
    if data.from_backup:
        if data.new_name:
            return await run_ps(cmd=f"Import-VM -SelectedFile
'{data.path}\\{data.file_name}' -NewName '{data.new_name}'",
                                module="backup")
        else:
            return await run_ps(cmd=f"Import-VM -SelectedFile
'{data.path}\\{data.file_name}'", module="backup")
    else:
        command = (f"New-VM -Name '{data.name}' -OSType '{data.os}' -Memory
{data.memory} -VRAM {data.vram} "
                    f"-DiskSize {data.diskSize} -ISOPath '{data.iso}'")
        return await run_ps(cmd=command, module="vm")

class VMStartRequest(BaseModel):
    type_option: str
    location: str

@app.post("/api/vms/{name}/start")
async def start_vm(name: str, data: VMStartRequest):
    if data.location == 'remote':
        ipaddress = '0.0.0.0'

        adapter = 'TP-Link Wireless USB Adapter'
        bridged = False
        details = await run_ps(cmd=f"Get-VMDetailedInfo -Name '{name}'",
                                module="utils")
        for i in range(8):
            if details[f'nic{i + 1}'] == 'bridged':
                bridged = True

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				113
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        break
    if bridged is False:
        await run_ps(cmd=f"Set-VMNetwork -Name '{name}' -Mode 'bridged' -
AdapterIndex 8"
                        f" -OptionValue '{adapter}'", module="vm")
        assigned_port = get_free_port(start_port=5000)

        await run_ps(cmd=f"VBoxManage modifyvm '{name}' --vrde on --vrdeport
'{assigned_port}'")
    else:
        ipaddress = '127.0.0.1'

        await run_ps(cmd=f"VBoxManage modifyvm '{name}' --vrde-address='{ipaddress}'")
        return await run_ps(cmd=f"Start-VM -Name '{name}' -Type '{data.type_option}'",
module="vm")

def get_free_port(start_port: int = 5000) -> int:
    port = start_port
    while port < 6000:
        with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
            try:
                s.bind(("0.0.0.0", port))
                return port
            except socket.error:
                port += 1
    return start_port

@app.post("/api/vms/{name}/stop")
async def stop_vm(name: str, mode: str = Body(..., embed=True)):
    return await run_ps(cmd=f"Stop-VM -Name '{name}' -State '{mode}'",
module="vm")

@app.delete("/api/vms/{name}")
async def delete_vm(name: str, mode: str):
    if mode == "all":
        return await run_ps(cmd=f"Remove-VM -Name '{name}' -Delete '--delete-
all'", module="vm")
    else:
        return await run_ps(cmd=f"Remove-VM -Name '{name}'", module="vm")

@app.get("/api/vms/{name}/details")
async def vm_details(name: str):
    vm_data = await run_ps(cmd=f"Get-VMDetailedInfo -Name '{name}'",
module="utils")

    vm_data["ip_address"] = None

    if vm_data.get("VMState") == "running":
        try:
            cmd = f'VBoxManage guestproperty get "{name}"
"/VirtualBox/GuestInfo/Net/0/V4/IP"'
            result = subprocess.run(cmd, shell=True, capture_output=True,
text=True, encoding="utf-8", timeout=300)
            output = result.stdout.strip()

            if "Value:" in output:
                real_ip = output.replace("Value:", "").strip()
                vm_data["ip_address"] = real_ip
        except Exception as e:
            print(f"Не вдалося отримати IP: {e}")

```

```

    return vm_data

@app.get("/api/vms/{name}/full-details")
async def full_vm_details(name: str):
    result = subprocess.run(
        ["powershell", "-Command", f"VBoxManage showvminfo '{name}'",
        capture_output=True, text=True, encoding='utf-8', timeout=300
    )
    return result.stdout.strip()

class VMUpdateRequest(BaseModel):
    vm_property: str
    value: str
    netint: Optional[str] = None
    index: Optional[str] = None

@app.patch("/api/vms/{name}")
async def modify_vm(name: str, data: VMUpdateRequest):
    if data.vm_property in ['--name', '--ostype', '--cpus', '--memory', '--vram']:
        if data.vm_property in ['--cpus', '--memory', '--vram']:
            value_str = f"-Value {int(data.value)}"
        else:
            value_str = f"-Value '{data.value}'"
        return await run_ps(cmd=f"Edit-VM -Name '{name}' -Property '{data.vm_property}' {value_str}", module="vm")
    elif data.vm_property == 'mode':
        if data.value in ['bridged', 'hostonly']:
            return await run_ps(cmd=f"Set-VMNetwork -Name '{name}' -Mode '{data.value}' -OptionValue '{data.netint}' "
                                f"-AdapterIndex {data.index}", module="vm")
        else:
            return await run_ps(cmd=f"Set-VMNetwork -Name '{name}' -Mode '{data.value}' "
                                f"-AdapterIndex {data.index}", module="vm")
    elif data.vm_property == 'type':
        return await run_ps(cmd=f"Set-VMAdapterType -Name '{name}' -AdapterType '{data.value}'", module="vm")
    else:
        return await run_ps(cmd=f"Set-VMIso -Name '{name}' -IsoPath '{data.value}'", module="vm")

@app.get("/api/host/network-interfaces")
async def network_interfaces(type: str):
    return await run_ps(cmd=f"Get-HostNetworkInterfaces -Type '{type}'",
    module="utils")

@app.get("/api/host/resources")
async def resources(drive: str):
    return await run_ps(cmd=f"Get-Resources -Drive '{drive}'", module="utils")

@app.get("/api/host/drives")
async def drives():
    data = await run_ps(cmd=f"Get-Drives", module="utils")

    if data is None:
        return []

    return data if isinstance(data, list) else [data]

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		115

```

@app.get("/api/host/folder-contents")
async def folder_contents(path: str):
    if path:
        data = await run_ps(cmd=f"Get-FolderChildren -Path '{path}'",
module="utils")

        if data is None:
            return []

        return data if isinstance(data, list) else [data]

@app.post("/api/vms/{vm_name}/backups")
async def create_backup(vm_name: str, path: str = Body(..., embed=True)):
    return await run_ps(cmd=f"Export-VM -VMName '{vm_name}' -BackupPath '{path}'",
module="backup")

class BackupScheduleRequest(BaseModel):
    vm_name: str
    schedule_type: str
    start_time: str
    path: str

@app.post("/api/backups/schedules")
async def schedule_backup(data: BackupScheduleRequest):
    return await run_ps(cmd=f"Register-VMBackupTask -VMName '{data.vm_name}' -
BackupPath '{data.path}' "
                        f"-ScheduleType '{data.schedule_type}' -StartTime
'{data.start_time}'", module="backup")

@app.get("/api/vms/{vm_name}/snapshots")
async def snapshots(vm_name: str):
    data = await run_ps(cmd=f"Get-VMSnapshotsList -VMName '{vm_name}'",
module="utils")

    if data is None:
        return []

    return data if isinstance(data, list) else [data]

@app.post("/api/vms/{vm_name}/snapshots")
async def create_snapshot(vm_name: str, snap_name: str = Body(..., embed=True)):
    return await run_ps(cmd=f"New-Snapshot -VMName '{vm_name}' -Name
'{snap_name}'", module="snapshot")

@app.post("/api/vms/{vm_name}/snapshots/{uuid}/restore")
async def restore_snapshot(vm_name: str, uuid: str, snap_name: str = Body(...,
embed=True)):
    return await run_ps(cmd=f"Restore-Snapshot -VMName '{vm_name}' -Name
'{snap_name}' -UUID '{uuid}'", module="snapshot")

@app.delete("/api/vms/{vm_name}/snapshots/{uuid}")
async def delete_snapshot(vm_name: str, uuid: str, snap_name: str):
    return await run_ps(cmd=f"Remove-Snapshot -VMName '{vm_name}' -Name
'{snap_name}' -UUID '{uuid}'", module="snapshot")

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				116
Змн.	Арк.	№ докум.	Підпис	Дата		

```

@app.get("/api/backups")
async def backups(path: str):
    return await run_ps(cmd=f"Get-BackupsList -BackupPath '{path}'",
module="utils")

@app.post("/api/backups/test-boot")
async def boot_backup_testing(path: str = Body(..., embed=True)):
    return await run_ps(cmd=f"Test-OvaBoot -OvaPath '{path}'", module="backup")

@app.get("/api/backups/schedules")
async def backups_schedules():
    data = await run_ps(cmd=f"Get-ScheduledBackups", module="utils")

    if data is None:
        return []

    return data if isinstance(data, list) else [data]

@app.delete("/api/backups/schedules/{task_name}")
async def delete_backup_schedule(task_name: str):
    return await run_ps(cmd=f"Unregister-VMBackupTask -TaskName '{task_name}'",
module="backup")

def cleanup_zombie_processes():
    print("Очищення залишкових процесів...")
    processes_to_kill = ["VBoxManage.exe", "powershell.exe"]

    for proc in processes_to_kill:
        try:
            # /F - примусово, /IM - за іменем образу, /T - дерево процесів
            subprocess.run(["taskkill", "/F", "/IM", proc, "/T"],
capture_output=True, check=False)
        except Exception as e:
            print(f"Не вдалося завершити {proc}: {e}")

def is_admin():
    try:
        return ctypes.windll.shell32.IsUserAnAdmin()
    except:
        return False

if __name__ == "__main__":
    if not is_admin():
        if "--no-relaunch" not in sys.argv:
            script_path = os.path.abspath(sys.argv[0])
            working_dir = os.path.dirname(script_path)

            try:
                result = ctypes.windll.shell32.ShellExecuteW(
                    None, "runas", sys.executable,
                    f'"{script_path}" --no-relaunch', working_dir, 1
                )

                if result > 32:
                    sys.exit()
            except Exception as e:
                print(f"Запит на права адміністратора відхилено або стався збій:

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		117

```

{e}")

if is_admin():
    print("Скрипт запущено з правами адміністратора!")
else:
    print("УВАГА: Скрипт запущено без прав адміністратора. Доступ до планува-
льника задач Windows буде обмежено.")

import uvicorn

cleanup_zombie_processes()
uvicorn.run(app, host="0.0.0.0", port=8000, reload=False)

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		118

Вміст HTML-файлів

Файл vm.html

```

<!DOCTYPE html>
<html lang="uk">
<head>
  <meta charset="UTF-8">
  <title>VM Manager</title>
  <link rel="icon" type="image/x-icon" href="/static/icon.ico">
  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"
rel="stylesheet">
  <script src="https://cdn.jsdelivr.net/npm/sweetalert2@11"></script>
  <style>
    .swal2-radio {
      display: flex !important;
      flex-direction: column !important;
      align-items: flex-start !important;
      margin-left: 20% !important;
      gap: 10px;
    }

    .swal2-radio label {
      font-size: 1.1rem !important;
      cursor: pointer;
    }

    .swal2-radio input {
      margin-right: 15px !important;
      transform: scale(1.2);
    }

    .card.shadow {
      max-height: calc(100vh - 150px);
      display: flex;
      flex-direction: column;
    }

    .card.shadow .table-responsive,
    .card.shadow {
      overflow-y: auto;
    }

    .table thead th {
      position: sticky;
      top: 0;
      z-index: 2;
      background-color: #212529;
    }
  </style>
</head>
<body class="bg-light">
  <nav class="navbar navbar-expand-lg navbar-dark bg-dark mb-3">
    <div class="container">
      <a class="navbar-brand" href="/vm">VM Manager Panel</a>
      <div class="navbar-nav flex-row gap-3">
        <a class="nav-link" href="/snapshot">Знімки ВМ</a>
        <a class="nav-link" href="/backup">Бекапи ВМ</a>
      </div>
    </div>
  </nav>
  <div class="container-fluid px-2 px-md-4 mt-2">

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				119
Змн.	Арк.	№ докум.	Підпис	Дата		

```

<div class="d-flex justify-content-between mb-2">
  <h2>Керування віртуальними машинами</h2>
  <div class="d-flex gap-2">
    <button class="btn btn-danger" onclick="createVM()">Створити
    <button class="btn btn-primary" onclick="loadVMs()">Оновити спи-
    сок</button>
  </div>
</div>

<div class="card shadow">
  <div class="table-responsive" style="overflow-y: auto;">
    <table class="table table-hover m-0">
      <thead class="table-dark">
        <tr>
          <th>Назва</th>
          <th>Статус</th>
          <th>UUID</th>
          <th>Дії</th>
        </tr>
      </thead>
      <tbody id="vm-list"></tbody>
    </table>
  </div>
</div>

<script>
  let totalRAM, freeRAM, totalDisk, freeDisk, totalLogicalCPUs;
  let isError = false;
  async function loadVMs() {
    const container = document.getElementById('vm-list');

    container.innerHTML = `
      <tr>
        <td colspan="4" class="text-center py-4">
          <div class="spinner-border text-primary"
            role="status"></div>
          <span class="ms-2">Завантаження списку віртуальних ма-
            шин...</span>
        </td>
      </tr>
    `;

    try {
      const res = await fetch('/api/vms');
      let data = await res.json();

      if (!data) {
        data = [];
      }

      if (data.length === 0) {
        container.innerHTML = `
          <tr>
            <td colspan="4" class="text-center text-muted py-4">
              Віртуальні машини відсутні. Створіть нову VM за
              допомогою кнопки вище.
            </td>
          </tr>
        `;
        return;
      }
    }
  }

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				120
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        container.innerHTML = '';

        data.forEach(vm => {
            const isRunning = vm.Status === 'Running';

            let statusBadge = 'bg-secondary';
            if (isRunning) statusBadge = 'bg-success';
            if (vm.Status === 'Aborted') statusBadge = 'bg-danger';

            const btnClass = isRunning ? 'btn-danger' : 'btn-success';
            const btnText = isRunning ? 'Вимкнути' : 'Запустити';
            const btnAction = isRunning ? `stopVM('${vm.Name}')` :
            `startVM('${vm.Name}')`;
            const trashIconClass = isRunning ? 'text-secondary opacity-50'
            : 'text-danger';

            rowHtml = `
                <tr>
                    <td><strong>${vm.Name}</strong></td>
                    <td><span class="badge
                    ${statusBadge}">${vm.Status}</span></td>
                    <td class="align-middle text-muted
                    small">${vm.UUID}</td>
                    <td class="d-flex justify-content-between align-items-
                    center">
                        <div class="d-flex gap-2">
                            <button class="btn btn-sm ${btnClass}"
                            onclick="${btnAction}">${btnText}</button>
                            <button class="btn btn-sm btn-primary"
                            onclick="showVMDetails('${vm.Name}')">Інформація</button>
                            <button class="btn btn-sm"
                            onclick="editVM('${vm.Name}')" ${isRunning ? 'disabled' : ''} style="color: white;
                            background-color: #6B1D4F">Редагувати</button>
                        </div>
                        <button class="btn border-0 p-1 ms-2"
                        style="background: transparent; box-shadow: none;"
                        onclick="deleteVM('${vm.Name}')"
                        ${isRunning ? 'disabled' : ''} title="Видалити">
                            <svg xmlns="http://www.w3.org/2000/svg"
                            width="20" height="20" fill="none"
                            viewBox="0 0 24 24" stroke-width="1.5"
                            stroke="currentColor"
                            class="${trashIconClass}" style="display:
                            block;">
                                <path stroke-linecap="round" stroke-
                                linejoin="round" d="m14.74 9-.346 9m-4.788
                                0L9.26 9m9.968-3.21c.342.052.682.107
                                1.022.166m-1.022-.165L18.16 19.673a2.25
                                2.25 0 0 1-2.244 2.077H8.084a2.25 2.25 0 0
                                1-2.244-2.077L4.772 5.79m14.456 0a48.108 48.108 0 0 0-3.478-.397m-12
                                .562c.34-.059.68-.114 1.022-.165m0 0a48.11
                                48.11 0 0 1 3.478-.397m7.5 0v-.916c0-1.18-
                                .91-2.164-2.09-2.201a51.964 51.964 0 0
                                0-3.32 0c-1.18.037-2.09 1.022-2.09
                                2.201v.916m7.5 0a48.667 48.667 0 0 0-7.5 0" />
                            </svg>
                        </button>
                    </td>
                </tr>
            `;
        });
    
```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				121
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        container.innerHTML += rowHtml
    });
} catch (error) {
    console.error("Помилка завантаження ВМ:", error);
    container.innerHTML = `
        <tr>
            <td colspan="4" class="text-center text-danger py-4">
                Помилка завантаження даних з сервера. Перевірте роботу
                бекенду.
            </td>
        </tr>
    `;
}

async function startVM(name) {
    await promptResources();

    let toContinue = false, message = "Недостатня кількість ресурсів для
запуску ВМ:";

    const response = await fetch(`/api/vms/${name}/details`);
    const data = await response.json();

    let goodRAM = (data.memory > 3 && data.memory <= (freeRAM - 2048));
    let goodDisk = ((freeDisk - 20480) >= 0);
    let goodCPUs = (data.cpus > 0 && data.cpus < totalLogicalCPUs);

    if (!goodRAM || !goodDisk || !goodCPUs) {
        if (!goodRAM) {
            message += `<br>- Оперативна пам'ять: Прямо зараз на ПК вільно
лише ${freeRAM.toFixed(0)} МБ, а для запуску ВМ потрібно ${data.memory} МБ, також
бажано залишити 2048 МБ хосту.`;
        }

        if (!goodDisk) {
            message += `<br>- Місце на диску: На хості залишилось менше 20
ГБ вільного простору (${freeDisk.toFixed(0)} МБ).`;
        }

        if (!goodCPUs) {
            message += `<br>- Кількість процесорів: ВМ потребує
${data.cpus} ядер, але ПК має всього ${totalLogicalCPUs} потоків. Не слід запус-
кати ВМ, виділяючи їй усі ядра, інакше система зависне.`;
        }

        const res = await Swal.fire({
            title: `Попередження щодо запуску ВМ '${name}'`,
            html: message,
            icon: 'warning',
            showCancelButton: true,
            confirmButtonText: 'Продовжити',
            cancelButtonText: 'Скасувати',
            confirmButtonColor: '#d33',
            cancelButtonColor: '#198754'
        });

        if (res.isConfirmed) toContinue = true;
    } else {
        toContinue = true;
    }

    if (toContinue) {
        let type_option = null, location = null;

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				122
Змн.	Арк.	№ докум.	Підпис	Дата		

```

await Swal.fire({
  title: `Запуск ВМ '${name}'`,
  text: "Виберіть режим візуалізації:",
  icon: 'question',
  showCancelButton: true,
  showDenyButton: true,
  confirmButtonText: "Графічний (GUI)",
  denyButtonText: "Фоновий режим",
  cancelButtonText: 'Скасувати',
  confirmButtonColor: '#198754',
  denyButtonColor: '#3085d6',
  cancelButtonColor: '#d33'
}).then((result) => {
  if (result.isConfirmed) type_option = 'gui';
  else if (result.isDenied) type_option = 'headless';
});
if (!type_option) return;

await Swal.fire({
  title: `Запуск ВМ '${name}'`,
  text: "Виберіть режим роботи з ВМ:",
  icon: 'question',
  showCancelButton: true,
  showDenyButton: true,
  confirmButtonText: "З машини-хоста",
  denyButtonText: "Через мережу",
  cancelButtonText: 'Скасувати',
  confirmButtonColor: '#198754',
  denyButtonColor: '#3085d6',
  cancelButtonColor: '#d33'
}).then((result) => {
  if (result.isConfirmed) location = 'local';
  else if (result.isDenied) location = 'remote';
});
if (!location) return;

const finalData = {
  type_option: type_option,
  location: location
};

await handleAction(`/api/vms/${name}/start`, 'POST', finalData);

if (!isError) {
  Swal.fire('Повідомлення', "Для дистанційного доступу перегля-
  ньте інформацію за шляхом 'Інформація' -> 'Віддалений доступ'.", 'info');
}
}

}

async function stopVM(name) {
  let mode = null;

  let result = await Swal.fire({
    title: `Вимкнення ВМ '${name}'`,
    text: "Виберіть режим завершення роботи:",
    icon: 'warning',
    showCancelButton: true,
    showDenyButton: true,
    confirmButtonText: 'Примусово (PowerOff)',
    denyButtonText: 'Безпечно (ACPI)',
    cancelButtonText: 'Скасувати',
    confirmButtonColor: '#d33',
  });

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				123
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        denyButtonColor: '#198754',
        cancelButtonColor: '#3085d6'
    });

    if (result.isConfirmed) mode = "poweroff";
    else if (result.isDenied) mode = "acpipowerbutton";
    else return;

    await handleAction(`/api/vms/${name}/stop`, 'POST', {mode: mode});
}

async function createVM() {
    let vmName, osType, ram, vram, diskSize, isoPath

    const {value} = await Swal.fire({
        title: "Створення нової ВМ",
        inputLabel: "Введіть назву ВМ:",
        input: 'text',
        showCancelButton: true,
        confirmButtonText: 'Підтвердити',
        cancelButtonText: 'Скасувати',
        confirmButtonColor: '#198754',
        cancelButtonColor: '#d33',
        inputValueValidator: (value) => {
            if (!value) {
                return 'ВМ має мати назву!';
            }
        }
    });
    vmName = value;

    if (vmName) {
        osType = await promptOSType();
    }

    if (osType) {
        const {value: formValues} = await Swal.fire({
            title: "Створення нової ВМ",
            html: `
                <div style="text-align: left;">
                    <label class="form-label" style="font-weight:
bold;">Оперативна пам'ять (MB)</label>
                    <input id="swal-ram" type="number" class="swal2-input"
value="4096" min="512" step="256">

                    <label class="form-label" style="font-weight: bold;
margin-top: 15px; display: block;">Відеопам'ять (MB)</label>
                    <input id="swal-vram" type="number" class="swal2-
input" value="128" min="4" max="256" step="4">

                    <label class="form-label" style="font-weight: bold;
margin-top: 15px; display: block;">Розмір диска (MB)</label>
                    <input id="swal-disk" type="number" class="swal2-
input" value="25000" min="1000" step="1000">
                </div>
            `,
            focusConfirm: false,
            showCancelButton: true,
            confirmButtonText: 'Підтвердити',
            cancelButtonText: 'Скасувати',
            confirmButtonColor: '#198754',
            cancelButtonColor: '#d33',
            preConfirm: () => {
                const ram = document.getElementById('swal-ram').value;

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				124
Змн.	Арк.	№ докум.	Підпис	Дата		

```

const vram = document.getElementById('swal-vram').value;
const disk = document.getElementById('swal-disk').value;

let goodRAM = (ram > 3 && ram < totalRAM);
let goodVRAM = (vram >= 0 && vram <= 256);
let goodDisk = (disk > 0 && disk < totalDisk);

if (!ram || !vram || !disk) {
    Swal.showValidationMessage('Усі поля мають бути запов-
нені!');

    return false;
}

if (!goodRAM) {
    Swal.showValidationMessage(`Не можна виділити стільки
оперативної пам'яті!<br>Максимум для ВМ на цьому ПК: ${totalRAM.toFixed(0)} МБ.`);
    return false;
}

if (!goodVRAM) {
    Swal.showValidationMessage('Неправильний обсяг відео-
пам\''яті!<br>Обсяг відеопам\''яті має бути від 0 до 256 МБ.');
```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				125
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        Swal.fire({
            title: 'Створити ВМ з цими параметрами?',
            text: "Зібрані дані:",
            html: `
                <div style="text-align: left; background: #f8f9fa;
padding: 15px; border-radius: 10px; border: 1px solid #dee2e6;">
                    <p><strong>Назва:</strong> ${finalData.name}</p>
                    <p><strong>Операційна система:</strong>
${finalData.os}</p>
                    <p><strong>Оперативна пам'ять:</strong>
${finalData.memory} MB</p>
                    <p><strong>Відеопам'ять:</strong> ${finalData.vram}
MB</p>
                    <p><strong>Розмір диска:</strong>
${finalData.diskSize} MB</p>
                    <p><strong>Образ:</strong> <br>
                    <small style="word-break: break-all; color:
#6c757d;">${finalData.iso}</small>
                    </p>
                </div>
            `,
            icon: 'question',
            showCancelButton: true,
            confirmButtonText: 'Так, створити!',
            cancelButtonText: 'Скасувати',
            confirmButtonColor: '#198754',
            cancelButtonColor: '#d33',
        }).then((result) => {
            if (result.isConfirmed) {
                handleAction('/api/vms', 'POST', finalData);
            }
        });
    }

    async function deleteVM(name){
        const result = await Swal.fire({
            title: 'Ви впевнені?',
            text: `Віртуальну машину "${name}" буде видалено назавжди!`,
            icon: 'warning',
            showCancelButton: true,
            confirmButtonColor: '#d33',
            cancelButtonColor: '#3085d6',
            confirmButtonText: 'Так, видалити!',
            cancelButtonText: 'Скасувати'
        });

        if (result.isConfirmed) {
            Swal.fire({
                title: 'Видалити повністю?',
                text: `Усі файли ВМ "${name}" будуть видалені назавжди!`,
                icon: 'warning',
                showDenyButton: true,
                confirmButtonColor: '#d33',
                denyButtonColor: '#3085d6',
                confirmButtonText: 'Так, видалити!',
                denyButtonText: 'Hi'
            }).then((res) => {
                if (res.isConfirmed) {
                    handleAction(`/api/vms/${name}?mode=all`, 'DELETE');
                } else if (res.isDenied) {
                    handleAction(`/api/vms/${name}?mode=none`, 'DELETE');
                }
            });
        }
    }
}

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		126

```

    }
}

async function handleAction(url, method = 'POST', payload = null) {
    isError = false;

    Swal.fire({
        title: 'Зачекайте...',
        text: 'Виконуємо операцію',
        allowOutsideClick: false,
        didOpen: () => {
            Swal.showLoading();
        }
    });

    try {
        const options = {
            method: method,
            headers: { 'Content-Type': 'application/json' }
        };

        if (payload) {
            options.body = JSON.stringify(payload)
        }

        const response = await fetch(url, options);

        if (!response.ok) {
            throw new Error(`Помилка сервера: ${response.status}`);
        }

        const data = await response.json();

        data.Message = data.Message.replace(/\n/g, '<br>');

        if (data.Success) {
            data.Message = data.Message || 'Дію виконано'
            Swal.fire({
                icon: 'success', title: 'Успішно!',
                html: getMessageHtml("повідомлення", data.Message),
                showConfirmButton: false, showCloseButton: true
            });
            await loadVMs();
        } else {
            data.Message = data.Message || 'Щось пішло не так'
            Swal.fire({
                icon: 'error', title: 'Помилка виконання',
                html: getMessageHtml("помилку", data.Message),
                showConfirmButton: false, showCloseButton: true
            });
            isError = true;
        }
    } catch (error) {
        let text = 'Не вдалося з\'єднатися з сервером. Перевірте, чи запу-
щено Python.'
        Swal.fire({
            icon: 'error', title: 'Помилка мережі',
            html: getMessageHtml("помилку", text),
            showConfirmButton: false, showCloseButton: true
        });
        console.error('Action error:', error);
        isError = true;
    }
}

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				127
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    }

    const getMessageHtml = (info, message) => `
      <details>
        <summary class="mt-2 text-primary" style="cursor:pointer;">Показати ${info}</summary>
        <pre class="mt-2" style="font-size: 14px; background: #f4f4f4; padding: 10px; white-space: pre-wrap; text-align: left;">${message}</pre>
      </details>
    `;

    async function showVMDetails(name) {
      const response = await fetch(`/api/vms/${name}/details`);
      const data = await response.json();

      const answer = await fetch(`/api/vms/${name}/full-details`);
      const info = await answer.json();

      let isRunning = (data.VMState === 'running');

      Swal.fire({
        title: `Детальна інформація про ВМ '${name}'`,
        width: '1000px',
        html: `
          <div style="text-align: left; max-height: 500px; overflow-y: auto;">
            <table class="table table-sm table-striped">
              <thead>
                <tr>
                  <th class="table-dark">Параметр</th>
                  <th class="table-dark">Значення</th>
                </tr>
              </thead>
              <tbody>
                <tr><td><b>Назва:</b></td><td>${data.name}</td></tr>

                <tr><td><b>ОС:</b></td><td>${data.ostype}</td></tr>
                <tr><td><b>Стан:</b></td><td><span class="badge ${data.VMState === 'running' ? 'bg-success' : 'bg-danger'}">${data.VMState}</span></td></tr>
                <tr><td><b>Пам'ять (RAM):</b></td><td>${data.memory} MB</td></tr>
                <tr><td><b>Процесори (CPU):</b></td><td>${data.cpus}</td></tr>
                <tr><td><b>Відеопам'ять (VRAM):</b></td><td>${data.vram} MB</td></tr>
                <tr><td><b>Мережа (NIC-1):</b></td><td>${data.nic1} (${data.nictype1})</td></tr>
                <tr><td><b>MAC-адреса:</b></td><td><code>${data.macaddress1}</code></td></tr>
                <tr><td><b>Шлях до конфігурації:</b></td><td><small>${data.CfgFile ? data.CfgFile.replaceAll('\\', '\\\\') : ''}</small></td></tr>

                <tr><td><b>UUID:</b></td><td>${data.UUID}</td></tr>
              </tbody>
            </table>
            <div style="display: flex; flex-direction: column; align-items: center;">
              <details style="width: 100%;">
                <summary class="mt-2 text-primary" style="cursor:pointer; text-align: center; list-style-position: inside;">Показати

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		128

```

всі технічні параметри</summary>
        <pre class="mt-2" style="font-size: 14px;
background: #f4f4f4; padding: 10px; text-align: left;">${info}</pre>
        </details>
    </div>
</div>
    ,
    confirmButtonText: 'Закрити',
    denyButtonText: 'Віддалений доступ',
    showDenyButton: isRunning,
    confirmButtonColor: '#198754',
    denyButtonColor: '#3085d6'
  }).then((result) => {
    if (result.isDenied) {
      let vrdeport = 'порт_VRDP';
      if (data.vrde === 'on'){
        vrdeport = data.vrdeport;
      }

      Swal.fire({
        title: `Інструкція з віддаленого підключення до
'${name}'`,
        width: '1050px',
        html: `
          <div style="text-align: left;">
            <p class="text-muted small">Перед підключенням пе-
реконайтеся, що мережевий адаптер ВМ налаштовано правильно (наприклад, Bridged або
Port Forwarding).</p>
            <table class="table table-bordered table-hover
align-middle" style="font-size: 14px;">
              <thead class="table-dark text-center">
                <tr>
                  <th style="width: 12%;">Протокол</th>
                  <th style="width: 48%;">Як підключи-
тися</th>
                  <th style="width: 40%;">Коли рекомен-
дується використовувати</th>
                </tr>
              </thead>
              <tbody>
                <tr>
                  <td class="text-
center"><b>VRDP</b><br><span class="badge bg-secondary">VirtualBox</span></td>
                  <td>
                    1. Скористайтесь клієнтом RDP (на-
приклад, <code>mstsc.exe</code> у Windows).<br>
                    2. Введіть: <code>IP_хо-
ста:${vrdeport}</code>.<br>
                    <span class="text-danger small">⚠️
Працює на рівні гіпервізора, окрема авторизація в ОС не потрібна.</span>
                  </td>
                  <td>
                    Рекомендується для <b>дебагу та
контролю на етапі встановлення ОС</b>. Показує екран машини ще до старту мережі чи
завантаження ядра (аналог фізичного монітора).
                  </td>
                </tr>
                <tr>
                  <td class="text-
center"><b>RDP</b><br><span class="badge bg-primary">Windows</span></td>
                  <td>
                    1. Запустіть утиліту «Remote
Desktop Connection» (<code>Win + R -> mstsc</code>).<br>
                    2. Введіть IP-адресу самої ВМ:

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				129
Змн.	Арк.	№ докум.	Підпис	Дата		

```

<code>${data.ip_address ? data.ip_address : 'IP_адреса'}</code>.<br>
3. Авторизуйтесь під створеними
вами даними користувача.
</td>
<td>
Ідеально для <b>повсякденної ро-
боти з Windows-машинами</b> або Linux із встановленим сервером <code>xrdp</code>.
Забезпечує найкращу якість графіки та спільний буфер обміну.
</td>
</tr>
<tr>
<td class="text-
center"><b>VNC</b><br><span class="badge bg-warning text-dark">GUI
Linux</span></td>
<td>
1. Встановіть будь-який VNC-клієнт
(VNC Viewer, TightVNC тощо).<br>
2. Підключіться до IP-адреси ма-
шини за схемою: <code>${data.ip_address ? data.ip_address : 'IP_адреса'}:5901</code> (або дисплей <code>:1</code>).<br>
3. Введіть VNC-пароль, згенерована
ний під час встановлення.
</td>
<td>
Використовується для <b>керування
графічним інтерфейсом Linux-систем</b> (наприклад, XFCE4), якщо нативні сервіси
RDP не встановлені або некоректно працюють з іксами (X11).
</td>
</tr>
<tr>
<td class="text-
center"><b>SSH</b><br><span class="badge bg-success">CLI Linux</span></td>
<td>
1. Відкрийте термінал (cmd,
PowerShell, bash) або утиліту PuTTY.<br>
2. Виконайте команду: <code>ssh
користувач@${data.ip_address ? data.ip_address : 'IP_адреса_ВМ'}</code>.<br>
3. Введіть пароль облікового за-
пису ВМ.
</td>
<td>
Основний стандарт для <b>адмініст-
рування Linux-серверів через консоль</b>. Найбільш швидкий, безпечний та невибаг-
ливий до пропускну здатності мережі метод.
</td>
</tr>
</tbody>
</table>
</div>
,
confirmButtonText: 'Назад до параметрів',
confirmButtonColor: '#3085d6',
cancelButtonText: 'Закрити',
showCancelButton: true,
cancelButtonColor: '#198754',
reverseButtons: true
}).then((subResult) => {
if (subResult.isConfirmed) {
showVMDetails(name);
}
});
}
}
}

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		130

```

async function editVM(name){
  const result = await Swal.fire({
    title: `Редагування ВМ '${name}'`,
    text: "Виберіть параметр для зміни:",
    input: 'select',
    inputOptions: {
      '--name': 'Назва ВМ',
      '--ostype': 'Тип ОС',
      '--cpus': 'Кількість CPU',
      '--memory': 'Оперативна пам'ять (МБ)',
      '--vram': 'Відеопам'ять (МБ)',
      'mode': 'Режим підключення (NAT/Bridged)',
      'type': 'Тип адаптера (Virtio/Intel)',
      'dvd': 'DVD-привід'
    },
    inputPlaceholder: 'Виберіть параметр',
    showCancelButton: true,
    confirmButtonText: 'Далі',
    cancelButtonText: 'Скасувати',
    confirmButtonColor: '#198754',
    cancelButtonColor: '#d33',
    inputValidator: (value) => {
      if (!value) {
        return 'Параметр має бути вибраний!';
      }
    }
  });

  if (result.isConfirmed) {
    await promptResources();
    await promptForValue(name, result.value);
  }
}

async function promptForValue(name, property) {
  let inputValue = null, networkInterface = null, adapterIndex = null;

  if (property === '--name') {
    const {value} = await Swal.fire({
      title: `Зміна назви ВМ '${name}'`,
      text: 'Введіть нову назву ВМ:',
      input: 'text',
      inputValue: '',
      inputValidator: (value) => {
        if (!value) {
          return 'Поле має бути заповнене!';
        }
      },
      showCancelButton: true,
      confirmButtonText: 'Змінити',
      cancelButtonText: 'Скасувати',
      confirmButtonColor: '#198754',
      cancelButtonColor: '#d33'
    });
    if (value) inputValue = value;
  } else if (property === '--ostype') {
    inputValue = await promptOSType();
  } else if ([ '--cpus', '--memory', '--vram' ].includes(property)) {
    const { value } = await Swal.fire({
      title: 'Зміна значення параметра',
      html: `
        <div style="text-align: center;">

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		131

```

        <label class="form-label">Введіть нове значення:</label>
        <div style="text-align: center;">
            <input id="swal-val" type="number" class="swal2-input" value="" min="0" step="1">
        </div>
    </div>
    `,
    focusConfirm: false,
    showCancelButton: true,
    confirmButtonText: 'Змінити',
    cancelButtonText: 'Скасувати',
    confirmButtonColor: '#198754',
    cancelButtonColor: '#d33',
    preConfirm: () => {
        let message, isGood;

        const value = document.getElementById('swal-val').value;

        if (!value) {
            Swal.showValidationMessage('Поле має бути заповнене!');

            return false;
        }

        if (property === '--memory') {
            message = `Не можна виділити стільки оперативної пам'яті!<br>Максимум для ВМ на цьому ПК: ${totalRAM.toFixed(0)} МБ.`;
            isGood = (value > 3 && value < totalRAM);
        } else if (property === '--vram') {
            message = 'Неправильний обсяг відеопам'яті!<br>Обсяг відеопам'яті має бути від 0 до 256 МБ.';
            isGood = (value >= 0 && value <= 256);
        } else {
            message = `Неправильна кількість процесорів!<br>Кількість процесорів має бути від 1 до ${totalLogicalCPUs}.`;
            isGood = (value > 0 && value <= totalLogicalCPUs);
        }

        if (!isGood) {
            Swal.showValidationMessage(message);
            return false;
        }

        return value;
    }
});
if (value) inputValue = value;
} else if (['mode', 'type'].includes(property)) {
    const res = await Swal.fire({
        title: 'Вибір мережевого адаптера',
        text: 'Виберіть номер мережевого адаптера:',
        input: 'select',
        inputOptions: ['1', '2', '3', '4', '5', '6', '7', '8'],
        inputPlaceholder: 'Виберіть номер адаптера',
        showCancelButton: true,
        inputValidator: (value) => {
            if (!value) {
                return 'Номер адаптера має бути вибраний!';
            }
        },
        confirmButtonText: 'Вибрати',
        cancelButtonText: 'Скасувати',
        confirmButtonColor: '#198754',

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				132
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        cancelButtonColor: '#d33'
    });
    if (res.isConfirmed) adapterIndex = parseInt(res.value) + 1;
    else return;

    let inputTitle, inputText, inputObj, inputHold, inputValid;
    if (property === 'mode') {
        inputTitle = `Зміна режиму підключення ВМ '${name}'`;
        inputText = "Виберіть режим підключення з переліку:";
        inputObj = {
            'none': 'Disabled Network Adapter',
            'null': 'Not Attached',
            'nat': 'NAT',
            'bridged': 'Bridged Adapter',
            'intnet': 'Internal Network',
            'hostonly': 'Host-Only Adapter',
            'hostonlynet': 'Host-Only Network',
            'generic': 'Generic Driver',
            'natnetwork': 'NAT Network',
            'cloud': 'Cloud Network [EXPERIMENTAL]'
        };
        inputHold = "Виберіть режим підключення";
        inputValid = "Режим підключення";
    } else {
        inputTitle = `Зміна типу адаптера ВМ '${name}'`;
        inputText = "Виберіть тип адаптера з переліку:";
        inputObj = {
            'Am79C970A': 'PCnet-PCI II (Am79C970A)',
            'Am79C973': 'PCnet-FAST III (Am79C973)',
            '82540EM': 'Intel PRO/1000 MT Desktop (82540EM)',
            '82543GC': 'Intel PRO/1000 T Server (82543GC)',
            '82545EM': 'Intel PRO/1000 MT Server (82545EM)',
            '82583V': 'Intel 82583V Gigabit Network Connection
[EXPERIMENTAL]',
            'virtio': 'Paravirtualized Network (virtio-net)',
            'usbnet': 'Ethernet over USB (usbnet)'
        };
        inputHold = "Виберіть тип адаптера";
        inputValid = "Тип адаптера";
    }
    const result = await Swal.fire({
        title: inputTitle,
        text: inputText,
        input: 'select',
        inputOptions: inputObj,
        inputPlaceholder: inputHold,
        showCancelButton: true,
        inputValue: (value) => {
            if (!value) {
                return `${inputValid} має бути вибраний!`;
            }
        },
        confirmButtonText: 'Вибрати',
        cancelButtonText: 'Скасувати',
        confirmButtonColor: '#198754',
        cancelButtonColor: '#d33'
    });
    if (result.isConfirmed) {
        inputValue = result.value;
        if (['bridged', 'hostonly'].includes(inputValue)) {
            networkInterface = await
promptNetworkInterface(inputValue);
        }
    }
}

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				133
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    } else {
      const result = await Swal.fire({
        title: `DVD-привід ВМ '${name}'`,
        text: "Виберіть дію:",
        icon: 'question',
        showCancelButton: true,
        showDenyButton: true,
        confirmButtonText: 'Вставити диск',
        denyButtonText: 'Витягти диск',
        cancelButtonText: 'Скасувати',
        confirmButtonColor: '#198754',
        denyButtonColor: '#3085d6',
        cancelButtonColor: '#d33'
      });

      if (result.isConfirmed) {
        const chosenPath = await browseForObject();

        if (chosenPath) {
          inputValue = chosenPath;
        } else {
          return;
        }
      } else if (result.isDenied) {
        inputValue = "emptydrive";
      } else {
        return;
      }
    }

    if (inputValue) {
      const payload = {
        vm_property: String(property),
        value: String(inputValue),
        netint: networkInterface ? String(networkInterface) : null,
        index: adapterIndex ? String(adapterIndex) : null
      };

      await handleAction(`/api/vms/${name}`, 'PATCH', payload);
    }
  }

  async function promptNetworkInterface(val) {
    let inputTitle, inputText, adapter = 'Adapter 1';

    const response = await fetch(`/api/host/network-
interfaces?type=${val}`);
    const adapters = await response.json();

    if (val === 'bridged') {
      inputTitle = "Вибір фізичної карти";
      inputText = "Виберіть фізичну карту з переліку:";
    } else {
      inputTitle = "Вибір Host-Only адаптера";
      inputText = "Виберіть Host-Only адаптер з переліку:";
    }

    const { value: value } = await Swal.fire({
      title: inputTitle,
      text: inputText,
      input: 'select',
      inputOptions: adapters,
      inputPlaceholder: 'Виберіть параметр',
      showCancelButton: true,

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				134
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        confirmButtonText: 'Далі',
        cancelButtonText: 'Скасувати',
        confirmButtonColor: '#198754',
        cancelButtonColor: '#d33',
        inputValidator: (value) => {
            if (!value) {
                return 'Потрібно вибрати параметр!';
            }
        }
    });

    if (value) {
        adapter = adapters[value];
    }

    return adapter;
}

async function browseForObject() {
    let path = [];

    while (true) {
        if (path.length === 0) {
            let drives = [];

            const response = await fetch(`/api/host/drives`);
            const data = await response.json();
            for (let i = 0; i < data.length; i++) {
                drives.push(data[i].Root.slice(0, -1))
            }

            const result = await Swal.fire({
                title: "Вибір диску",
                text: "Виберіть диск з переліку:",
                input: 'select',
                inputOptions: drives,
                inputPlaceholder: 'Виберіть диск',
                showCancelButton: true,
                confirmButtonText: 'Вибрати',
                cancelButtonText: 'Скасувати',
                confirmButtonColor: '#198754',
                cancelButtonColor: '#d33',
                inputValidator: (value) => {
                    if (!value) {
                        return 'Потрібно вибрати диск!';
                    }
                }
            });

            if (result.isConfirmed) path.push(drives[result.value]);
            else return null;
        } else {
            if (path.length === 1 && !path[0].includes("\\\\")) path[0] =
`_${path[0]}\\`;
            else if (path.length === 2 && path[0].includes("\\\\")) path[0]
= path[0].slice(0, -1);
            let currentPath = path.join('\\');
            const response = await fetch(`/api/host/folder-
contents?path=${encodeURIComponent(currentPath)}`);
            const data = await response.json();

            let objects = []
            for (let i = 0; i < data.length; i++) {
                objects.push(data[i].Name)
            }
        }
    }
}

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		135

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				136
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        return 'Потрібно вибрати операційну систему!';
    }
    });
    return osType;
}

async function promptResources() {
    Swal.fire({
        title: 'Зачекайте...',
        text: 'Виконуємо операцію',
        allowOutsideClick: false,
        didOpen: () => {
            Swal.showLoading();
        }
    });

    const BYTES_IN_MB = 1024 * 1024;

    const reply = await fetch('/api/host/resources?drive=C');
    const resources = await reply.json();

    totalRAM = resources.TotalRAM / BYTES_IN_MB; // B -> MB
    freeRAM = resources.FreeRAM / BYTES_IN_MB; // B -> MB
    totalDisk = resources.TotalDisk / BYTES_IN_MB; // B -> MB
    freeDisk = resources.FreeDisk / BYTES_IN_MB; // B -> MB

    totalLogicalCPUs = resources.TotalLogicalCPUs;
}

document.addEventListener('DOMContentLoaded', loadVMs);
</script>
</body>
</html>

```

Файл snapshot.html

```

<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <title>Snapshot Manager</title>
    <link rel="icon" type="image/x-icon" href="/static/icon.ico">
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"
rel="stylesheet">
    <script src="https://cdn.jsdelivr.net/npm/sweetalert2@11"></script>
    <style>
        .card.shadow {
            max-height: calc(100vh - 150px);
            display: flex;
            flex-direction: column;
        }

        .card.shadow .table-responsive,
        .card.shadow {
            overflow-y: auto;
        }

        .table thead th {
            position: sticky;
            top: 0;
            z-index: 2;
            background-color: #212529;

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		137

```

    }
  </style>
</head>
<body class="bg-light">
  <nav class="navbar navbar-expand-lg navbar-dark bg-dark mb-3">
    <div class="container">
      <a class="navbar-brand" href="/snapshot">Snapshot Manager Panel</a>
      <div class="navbar-nav flex-row gap-3">
        <a class="nav-link" href="/">Керування ВМ</a>
        <a class="nav-link" href="/backup">Бекапи ВМ</a>
      </div>
    </div>
  </nav>
  <div class="container-fluid px-2 px-md-4 mt-2">
    <div class="d-flex flex-column flex-md-row justify-content-between align-items-md-center gap-3 mb-2">
      <div class="d-flex align-items-center flex-row gap-3">
        <h2 class="m-0 text-nowrap">Керування знімками ВМ</h2>
        <select class="form-select fs-5 fw-bold text-primary shadow-sm"
id="vm-snapshot-select" style="max-width: 300px;">
          <option value="" selected disabled>Завантаження ВМ...</option>
        </select>
      </div>
      <div class="d-flex gap-2">
        <button class="btn btn-danger" id="btn-create-snapshot"
onclick="createSnapshot()" disabled>Створити знімок ВМ</button>
        <button class="btn btn-primary" id="btn-refresh-snapshots"
onclick="loadSnapshots()">Оновити список</button>
      </div>
    </div>

    <div class="card shadow">
      <div class="table-responsive" style="overflow-y: auto;">
        <table class="table table-hover m-0">
          <thead class="table-dark">
            <tr>
              <th>Назва</th>
              <th>UUID</th>
              <th>Дії</th>
            </tr>
          </thead>
          <tbody id="snapshot-list"></tbody>
        </table>
      </div>
    </div>
  </div>

  <script>
    let vmSelect, snapshotList, btnCreate, btnRefresh;
    let currentVM;

    async function createSnapshot() {
      const result = await Swal.fire({
        title: "Створення нового знімка",
        inputLabel: `Введіть назву знімка ВМ '${currentVM}':`,
        input: 'text',
        showCancelButton: true,
        confirmButtonText: 'Підтвердити',
        cancelButtonText: 'Скасувати',
        confirmButtonColor: '#198754',
        cancelButtonColor: '#d33',
        inputValue: (value) => {
          if (!value) {
            return 'Знімок має мати назву!';
          }
        }
      });
    }
  </script>

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		138

```

    }
  });

  if (result.isConfirmed) await
handleAction(`/api/vms/${currentVM}/snapshots`, 'POST', {snap_name:
result.value});
}

async function restoreSnapshot(name, uuid) {
  await Swal.fire({
    title: `Відновлення стану ВМ '${currentVM}'`,
    text: `Підтвердіть відновлення за знімком '${name}':`,
    icon: 'warning',
    showCancelButton: true,
    confirmButtonText: 'Відновити',
    cancelButtonText: 'Скасувати',
    confirmButtonColor: '#198754',
    cancelButtonColor: '#d33'
  }).then((result) => {
    if (result.isConfirmed)
handleAction(`/api/vms/${currentVM}/snapshots/${uuid}/restore`, 'POST',
{snap_name: name});
  });
}

async function deleteSnapshot(name, uuid){
  const result = await Swal.fire({
    title: 'Ви впевнені?',
    text: `Знімок '${name}' ВМ '${currentVM}' буде видалено назав-
жди!`,
    icon: 'warning',
    showCancelButton: true,
    confirmButtonColor: '#d33',
    cancelButtonColor: '#3085d6',
    confirmButtonText: 'Так, видалити!',
    cancelButtonText: 'Скасувати'
  });

  if (result.isConfirmed) {
    await
handleAction(`/api/vms/${currentVM}/snapshots/${uuid}?snap_name=${name}`,
'DELETE');
  }
}

async function handleAction(url, method = 'POST', payload = null) {
  Swal.fire({
    title: 'Зачекайте...',
    text: 'Виконуємо операцію',
    allowOutsideClick: false,
    didOpen: () => {
      Swal.showLoading();
    }
  });

  try {
    const options = {
      method: method,
      headers: { 'Content-Type': 'application/json' }
    };

    if (payload){
      options.body = JSON.stringify(payload)
    }
  }
}

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		139

```

    }

    const response = await fetch(url, options);

    if (!response.ok) {
        throw new Error(`Помилка сервера: ${response.status}`);
    }

    const data = await response.json();

    data.Message = data.Message.replace(/\n/g, '<br>');

    if (data.Success) {
        data.Message = data.Message || 'Дію виконано'
        Swal.fire({
            icon: 'success',
            title: 'Успішно!',
            html: getMessageHtml("повідомлення", data.Message),
            showConfirmButton: false,
            showCloseButton: true
        });

        await loadSnapshots(currentVM);

    } else {
        data.Message = data.Message || 'Щось пішло не так'
        Swal.fire({
            icon: 'error',
            title: 'Помилка виконання',
            html: getMessageHtml("помилку", data.Message),
            showConfirmButton: false,
            showCloseButton: true
        });
    }
} catch (error) {
    let text = 'Не вдалося з'єднатися з сервером. Перевірте, чи запущено Python.'
    Swal.fire({
        icon: 'error',
        title: 'Помилка мережі',
        html: getMessageHtml("помилку", text),
        showConfirmButton: false,
        showCloseButton: true
    });
    console.error('Action error:', error);
}

const getMessageHtml = (info, message) => `
    <details>
        <summary class="mt-2 text-primary" style="cursor:pointer;">Показати ${info}</summary>
        <pre class="mt-2" style="font-size: 14px; background: #f4f4f4; padding: 10px; white-space: pre-wrap; text-align: left">${message}</pre>
    </details>
`;

document.addEventListener("DOMContentLoaded", async () => {
    vmSelect = document.getElementById("vm-snapshot-select");
    snapshotList = document.getElementById("snapshot-list");
    btnCreate = document.getElementById("btn-create-snapshot");
    btnRefresh = document.getElementById("btn-refresh-snapshots");

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		140

```

vmSelect.addEventListener("change", () => {
    currentVM = vmSelect.value;
    loadSnapshots(vmSelect.value);
});

btnRefresh.addEventListener("click", () => {
    loadVirtualMachines();
});

await loadVirtualMachines();
});

async function loadVirtualMachines() {
    const previouslySelectedVM = vmSelect.value;

    try {
        const response = await fetch('/api/vms');
        const vms = await response.json();

        vmSelect.replaceChildren();

        vmSelect.innerHTML = '<option value="" disabled>-- Виберіть ВМ --</option>';

        let stillExists = false;

        vms.forEach(vm => {
            const option = document.createElement('option');
            option.value = vm.Name;
            option.textContent = vm.Name;

            if (vm.Name === previouslySelectedVM) {
                option.selected = true;
                stillExists = true;
            }

            vmSelect.appendChild(option);
        });

        if (stillExists && previouslySelectedVM) {
            await loadSnapshots(previouslySelectedVM);
        } else {
            currentVM = "";
            vmSelect.value = "";
            btnCreate.disabled = true;
            snapshotList.innerHTML = '<tr><td colspan="3" class="text-center text-muted py-4">Будь ласка, оберіть віртуальну машину зі списку вище</td></tr>';
        }

    } catch (error) {
        console.error("Помилка завантаження ВМ:", error);
        vmSelect.innerHTML = '<option value="" disabled>Помилка завантаження ВМ</option>';
    }
}

async function loadSnapshots(vmName) {
    if (!vmName) return;

    snapshotList.innerHTML = '<tr><td colspan="3" class="text-center py-4"><div class="spinner-border text-primary" role="status"></div> Завантаження знімків...</td></tr>';
    btnCreate.disabled = false;

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		141

```

try {
  const response = await fetch(`/api/vms/${vmName}/snapshots`);
  const snapshots = await response.json();

  if (snapshots.length === 0) {
    snapshotList.innerHTML = '<tr><td colspan="3" class="text-center text-muted py-4">У цієї ВМ немає збережених знімків</td></tr>';
    return;
  }

  snapshotList.innerHTML = '';
  snapshots.forEach(snap => {
    rowHtml = `
      <tr>
        <td><strong>${snap.Name}</strong></td>
        <td class="align-middle text-muted small">${snap.UUID}</td>
        <td class="d-flex justify-content-between align-items-center">
          <div class="d-flex gap-2">
            <button class="btn btn-sm btn-success"
              onclick="restoreSnapshot('${snap.Name}', '${snap.UUID}')">Відновити</button>
            <button class="btn border-0 p-1 ms-2"
              style="background: transparent; box-shadow: none;"
              onclick="deleteSnapshot('${snap.Name}', '${snap.UUID}')" title="Видалити">
              <svg xmlns="http://www.w3.org/2000/svg" width="20"
                height="20" fill="none"
                viewBox="0 0 24 24" stroke-width="1.5"
                stroke="currentColor"
                class="text-danger" style="display: block;">
                <path stroke-linecap="round" stroke-
                  linejoin="round" d="m14.74 9-.346 9m-4.788
                    0L9.26 9m9.968-3.21c.342.052.682.107
                    1.022.166m-1.022-.165L18.16 19.673a2.25
                    2.25 0 0 1-2.244 2.077H8.084a2.25 2.25 0 0 1-
                    2.244-2.077L4.772 5.79m14.456
                    0a48.108 48.108 0 0 0-3.478-.397m-12 .562c.34-
                    .059.68-.114 1.022-.165m0 0a48.11
                    48.11 0 0 1 3.478-.397m7.5 0v-.916c0-1.18-.91-
                    2.164-2.09-2.201a51.964 51.964 0 0
                    0-3.32 0c-1.18.037-2.09 1.022-2.09
                    2.201v.916m7.5 0a48.667 48.667 0 0 0-7.5 0" />
              </svg>
            </button>
          </td>
        </tr>
      `;
    snapshotList.insertAdjacentHTML('beforeend', rowHtml);
  });
} catch (error) {
  snapshotList.innerHTML = '<tr><td colspan="3" class="text-center text-danger py-4">Помилка завантаження даних з сервера</td></tr>';
}
</script>
</body>
</html>

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		142

Файл backup.html

```
<!DOCTYPE html>
<html lang="uk">
<head>
  <meta charset="UTF-8">
  <title>Backup Manager</title>
  <link rel="icon" type="image/x-icon" href="/static/icon.ico">
  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"
rel="stylesheet">
  <script src="https://cdn.jsdelivr.net/npm/sweetalert2@11"></script>
  <style>
    .card.shadow {
      max-height: calc(100vh - 175px);
      display: flex;
      flex-direction: column;
    }

    .card.shadow .table-responsive,
    .card.shadow {
      overflow-y: auto;
    }

    .table thead th {
      position: sticky;
      top: 0;
      z-index: 2;
      background-color: #212529;
    }
  </style>
</head>
<body class="bg-light">
  <nav class="navbar navbar-expand-lg navbar-dark bg-dark mb-3">
    <div class="container">
      <a class="navbar-brand" href="/backup">Backup Manager Panel</a>
      <div class="navbar-nav flex-row gap-3">
        <a class="nav-link" href="/">Керування ВМ</a>
        <a class="nav-link" href="/snapshot">Знімки ВМ</a>
        <a class="nav-link" onclick="schedulerMenu()" style="cursor:
pointer;">Планувальник</a>
      </div>
    </div>
  </nav>
  <div class="container-fluid">
    <div class="card shadow-sm mb-3">
      <div class="card-body">
        <div class="input-group">
          <span class="input-group-text">Поточний шлях:</span>
          <input type="text" id="currentPath" class="form-control"
value="" readonly>
          <button class="btn btn-primary" onclick="pickDirectory()">Змі-
нити каталог</button>
          <button class="btn btn-success"
onclick="showActionMenu()">Дії</button>
        </div>
      </div>
    </div>
    <div class="card shadow">
      <div class="table-responsive" style="overflow-y: auto;">
        <table class="table table-hover m-0" id="backupsTable">
          <thead class="table-dark">
            <tr>
              <th>Назва</th>
            </tr>
          </thead>
        </table>
      </div>
    </div>
  </div>
```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		143

```

        <th>Тип</th>
        <th>Розмір</th>
        <th>Змінено</th>
        <th>Валідність</th>
    </tr>
</thead>
<tbody></tbody>
</table>
</div>
</div>
</div>
</div>

<script>
    let currentPath = "C:\\Backups";
    document.getElementById('currentPath').value = currentPath;

    let path = currentPath.length !== 0? currentPath.split("\\") : [];
    let backupsList = [];
    async function browseForFolder() {
        while (true) {
            if (path.length === 0) {
                let drives = [];

                const response = await fetch(`/api/host/drives`);
                const data = await response.json();
                for (let i = 0; i < data.length; i++){
                    drives.push(data[i].Root.slice(0, -1))
                }

                const result = await Swal.fire({
                    title: "Вибір диску",
                    text: "Виберіть диск з переліку:",
                    input: 'select',
                    inputOptions: drives,
                    inputPlaceholder: 'Виберіть диск',
                    showCancelButton: true,
                    confirmButtonText: 'Вибрати',
                    cancelButtonText: 'Скасувати',
                    confirmButtonColor: '#198754',
                    cancelButtonColor: '#d33',
                    inputValidator: (value) => {
                        if (!value) {
                            return 'Потрібно вибрати диск!';
                        }
                    }
                });
                if (result.isConfirmed) path.push(drives[result.value]);
                else return null;
            } else {
                if (path.length === 1 && !path[0].includes("\\")) path[0] =
` ${path[0]} \\ `;
                else if (path.length === 2 && path[0].includes("\\")) path[0]
= path[0].slice(0, -1);
                let currentPath = path.join('\\');
                const response = await fetch(`/api/host/folder-
contents?path=${encodeURIComponent(currentPath)}`);
                const data = await response.json();

                let objects = []
                for (let i = 0; i < data.length; i++){
                    objects.push(data[i].Name)
                }

                const result = await Swal.fire({

```

Бахур Ю. В.

Головня О. С.

Змн.

Арк.

№ докум.

Підпис

Дата

ККІКБ.КР.Б – 123 – 01 – ПЗ

Арк.

144

```

        title: "Вибір директорії",
        text: `Шлях: ${currentPath}`,
        input: 'select',
        inputOptions: objects,
        inputPlaceholder: 'Виберіть каталог',
        showCancelButton: true,
        showDenyButton: true,
        confirmButtonText: 'Вибрати',
        denyButtonText: 'Назад',
        cancelButtonText: 'Скасувати',
        confirmButtonColor: '#198754',
        denyButtonColor: '#3085d6',
        cancelButtonColor: '#d33'
    });

    if (result.isConfirmed) {
        const selectedIndex = result.value;

        if (selectedIndex !== "") {
            const chosenObject = data[selectedIndex];

            if (chosenObject.Type === 'Folder') {
                path.push(chosenObject.Name);
            }
        } else {
            return path.join('\\');
        }
    } else if (result.isDenied) {
        path.pop();
    } else if (result.isDismissed) {
        return null;
    }
}

}

}

async function pickDirectory() {
    const path = await browseForFolder();
    if (path) {
        currentPath = path;
        document.getElementById('currentPath').value = path;
        await loadBackups();
    }
}

async function loadBackups() {
    backupsList = []

    const tbody = document.querySelector('#backupsTable tbody');

    tbody.innerHTML = `
        <tr>
            <td colspan="5" class="text-center py-4">
                <div class="spinner-border text-primary"
role="status"></div>
                <span class="ms-2">Зчитування вмісту директорії...</span>
            </td>
        </tr>
    `;

    try {
        const res = await
fetch(`/api/backups?path=${encodeURIComponent(currentPath)}`);
        const backups = await res.json();

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		145

```
tbody.innerHTML = '';

if (!backups || backups.length === 0) {
    tbody.innerHTML = `
        <tr>
            <td colspan="5" class="text-center text-muted py-4">
                Цей каталог не містить бекапи.
            </td>
        </tr>
    `;
    return;
}

backups.forEach(file => {
    backupsList.push(file.Name);

    let validityBadge = '';

    if (file.Validity === 'Valid') {
        validityBadge = `<span class="badge bg-success"
style="padding: 5px 10px; border-radius: 4px;">✓ Валідний</span>`;
    } else if (file.Validity === 'Corrupted') {
        validityBadge = `<span class="badge bg-danger"
style="padding: 5px 10px; border-radius: 4px;">× Пошкоджений</span>`;
    } else if (file.Validity === 'Unknown') {
        validityBadge = `<span class="badge bg-warning"
style="padding: 5px 10px; border-radius: 4px;">Невідомо</span>`;
    } else {
        validityBadge = `<span class="badge bg-secondary"
style="padding: 5px 10px; border-radius: 4px;">Не перевірено</span>`;
    }

    const rowHtml = `
        <tr>
            <td>${file.Name}</td>
            <td><span class="badge bg-
secondary">${file.Type}</span></td>
            <td>${file.Size}</td>
            <td>${file.Modified}</td>
            <td class="d-flex justify-content-between align-items-
center">${validityBadge}
                <button class="btn btn-sm btn-primary"
onclick="verifyBackup('${file.Name}')">Перевірити</button>
            </td>
        </tr>
    `;
    tbody.innerHTML += rowHtml;
});
} catch (error) {
    console.error("Помилка завантаження бекапів:", error);
    tbody.innerHTML = `
        <tr>
            <td colspan="5" class="text-center text-danger py-4">
                Помилка зчитування каталогу. Перевірте правильність
                шляху або права доступу.
            </td>
        </tr>
    `;
}

function showActionMenu() {
    Swal.fire({
```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		146

```

        title: 'Що ви хочете зробити?',
        icon: 'question',
        showDenyButton: true,
        showCancelButton: true,
        confirmButtonText: 'Створити бекап ВМ',
        denyButtonText: 'Відновити з бекапу ВМ',
        cancelButtonText: 'Запланувати бекап ВМ',
        confirmButtonColor: '#198754',
        denyButtonColor: '#3085d6',
        cancelButtonColor: '#6B1D4F',
        showCloseButton: true,

    }).then((result) => {
        if (result.isConfirmed) {
            promptCreate();
        } else if (result.isDenied) {
            promptRestore();
        } else if (result.isDismissed) {
            if (result.dismiss === Swal.DismissReason.cancel) {
                promptSchedule();
            }
        }
    });
}

async function promptCreate() {
    let vmsList = await getVms();

    const result = await Swal.fire({
        title: 'Створення резервної копії',
        text: 'Виберіть ВМ для резервування:',
        input: "select",
        inputOptions: vmsList,
        inputPlaceholder: 'Виберіть ВМ',
        showCancelButton: true,
        confirmButtonText: 'Створити',
        cancelButtonText: 'Скасувати',
        confirmButtonColor: '#198754',
        cancelButtonColor: '#d33',
        inputValueValidator: (value) => {
            if (!value) {
                return 'ВМ має бути вибрана!';
            }
        }
    });

    if (result.isConfirmed) {
        await handleAction(`/api/vms/${vmsList[result.value]}/backups`,
        'POST', { path: currentPath });
    }
}

async function promptRestore() {
    const { value: fileName } = await Swal.fire({
        title: 'Відновлення ВМ з бекапу',
        text: 'ВМ буде відновлена з файлу!',
        input: 'select',
        inputOptions: backupsList,
        inputPlaceholder: 'Виберіть бекап в поточному каталозі',
        icon: 'warning',
        showCancelButton: true,
        confirmButtonText: 'Відновити',
        cancelButtonText: 'Скасувати',
        confirmButtonColor: '#198754',
    });

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		147

```

        cancelButtonColor: '#d33',
        inputValidator: (value) => {
            if (!value) {
                return 'Файл має бути вибраний!';
            }
        }
    });

    if (fileName) {
        Swal.fire({
            title: 'Бажаєте задати нове ім\'я ВМ?',
            icon: 'question',
            showDenyButton: true,
            showCancelButton: true,
            confirmButtonText: 'Задати нове',
            denyButtonText: 'Залишити старе',
            cancelButtonText: 'Скасувати',
            confirmButtonColor: '#198754',
            denyButtonColor: '#3085d6',
            cancelButtonColor: '#dc3545'
        }).then(async (result) => {
            if (result.isConfirmed) {
                const res = await Swal.fire({
                    title: 'Введіть нове ім\'я',
                    input: "text",
                    inputValue: '',
                    inputValidator: (value) => {
                        if (!value) {
                            return 'Поле має бути заповнене!';
                        }
                    },
                    showCancelButton: true,
                    confirmButtonText: 'Підтвердити',
                    cancelButtonText: 'Скасувати',
                    confirmButtonColor: '#198754',
                    cancelButtonColor: '#dc3545'
                });

                if (res.isConfirmed) {
                    const payload = {
                        from_backup: true,
                        new_name: res.value,
                        path: currentPath,
                        file_name: backupsList[fileName]
                    };
                    await handleAction('/api/vms', 'POST', payload);
                }
            } else if (result.isDenied) {
                const payload = {
                    from_backup: true,
                    path: currentPath,
                    file_name: backupsList[fileName]
                };
                await handleAction('/api/vms', 'POST', payload);
            }
        });
    }
}

async function promptSchedule() {
    let vmName, scheduleType, startTime;
    let vmsList = await getVms();

    const result = await Swal.fire({

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		148

```

        title: 'Планування бекапів',
        text: 'Виберіть ВМ для резервування:',
        input: "select",
        inputOptions: vmsList,
        inputPlaceholder: 'Виберіть ВМ',
        showCancelButton: true,
        confirmButtonText: 'Вибрати',
        cancelButtonText: 'Скасувати',
        confirmButtonColor: '#198754',
        cancelButtonColor: '#d33',
        inputValidator: (value) => {
            if (!value) {
                return 'ВМ має бути вибрана!';
            }
        }
    });

    if (result.isConfirmed) {
        vmName = vmsList[result.value];
    } else {
        return;
    }

    const optionsList = {
        'Once': 'Одноразово (за датою/часом)',
        'Daily': 'Щодня (регулярно)',
        'Weekly': 'Щотижня (регулярно)',
        'Monthly': 'Щомісяця (регулярно)',
        'AtStartup': 'При запуску операційної системи (ПК)',
        'AtLogOn': 'При вході користувача в систему'
    };

    const res = await Swal.fire({
        title: 'Планування бекапів',
        text: 'Виберіть періодичність для автоматичного резервного копію-
вання:',
        input: "select",
        inputOptions: optionsList,
        inputPlaceholder: 'Виберіть періодичність',
        showCancelButton: true,
        confirmButtonText: 'Вибрати',
        cancelButtonText: 'Скасувати',
        confirmButtonColor: '#198754',
        cancelButtonColor: '#d33',
        inputValidator: (value) => {
            if (!value) {
                return 'Періодичність має бути вибрана!';
            }
        }
    });

    if (res.isConfirmed) {
        scheduleType = res.value;
    } else {
        return;
    }

    if (scheduleType !== 'AtStartup' && scheduleType !== 'AtLogOn') {
        let labelText;
        if (scheduleType === 'Daily') labelText = 'Вкажіть час щоденного
запуску завдання:';
        if (scheduleType === 'Once') labelText = 'Вкажіть дату та час для
одноразового бекапу:';
        if (scheduleType === 'Weekly') labelText = 'Вкажіть день та час.

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		149

```

Завдання повторюватиметься щотижня у цей день:';
    if (scheduleType === 'Monthly') labelText = 'Вкажіть дату та час.
Завдання повторюватиметься щомісяця (через 28 дні):';

    const timeResult = await Swal.fire({
      title: 'Встановлення часу',
      text: labelText,
      input: 'datetime-local',
      inputPlaceholder: 'Виберіть час',
      showCancelButton: true,
      confirmButtonText: 'Запланувати',
      cancelButtonText: 'Скасувати',
      confirmButtonColor: '#198754',
      cancelButtonColor: '#d33',
      inputValue: new Date().toISOString().slice(0, 16),
      inputValidator: (value) => {
        if (!value) {
          return 'Потрібно вказати дату і час!';
        }
      }
    });
    if (timeResult.isConfirmed) {
      startTime = timeResult.value;
      startTime = startTime.replace('T', ' ');
    } else {
      return;
    }
  } else {
    startTime = "02:00";
  }

  const finalData = {
    vm_name: vmName,
    schedule_type: scheduleType,
    start_time: startTime,
    path: currentPath
  };

  await handleAction(`/api/backups/schedules`, 'POST', finalData);
}

async function getVms() {
  const res = await fetch(`/api/vms`);
  const vms = await res.json();

  let vmsList = []
  for (let i = 0; i < vms.length; i++) {
    vmsList.push(vms[i].Name)
  }

  return vmsList;
}

async function handleAction(url, method = 'POST', payload = null,
errorLoad = false) {
  Swal.fire({
    title: 'Зачекайте...',
    text: 'Виконуємо операцію',
    allowOutsideClick: false,
    didOpen: () => {
      Swal.showLoading();
    }
  });
}

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				150
Змн.	Арк.	№ докум.	Підпис	Дата		

```

try {
  const options = {
    method: method,
    headers: { 'Content-Type': 'application/json' }
  };

  if (payload){
    options.body = JSON.stringify(payload)
  }

  const response = await fetch(url, options);

  if (!response.ok) {
    throw new Error(`Помилка сервера: ${response.status}`);
  }

  const data = await response.json();

  data.Message = data.Message.replace(/\n/g, '<br>');

  if (data.Success) {
    data.Message = data.Message || 'Дію виконано'
    Swal.fire({
      icon: 'success',
      title: 'Успішно!',
      html: getMessageHtml("повідомлення", data.Message),
      showConfirmButton: false,
      showCloseButton: true
    });

    await loadBackups();
  } else {
    data.Message = data.Message || 'Щось пішло не так'
    Swal.fire({
      icon: 'error',
      title: 'Помилка виконання',
      html: getMessageHtml("помилку", data.Message),
      showConfirmButton: false,
      showCloseButton: true
    });
    if (errorLoad) await loadBackups();
  }
} catch (error) {
  let text = 'Не вдалося з'єднатися з сервером. Перевірте, чи запущено Python.'
  Swal.fire({
    icon: 'error',
    title: 'Помилка мережі',
    html: getMessageHtml("помилку", text),
    showConfirmButton: false,
    showCloseButton: true
  });
  console.error('Action error:', error);
  if (errorLoad) await loadBackups();
}

const getMessageHtml = (info, message) => `
  <details>
    <summary class="mt-2 text-primary" style="cursor:pointer;">Показати ${info}</summary>
    <pre class="mt-2" style="font-size: 14px; background: #f4f4f4; padding: 10px; white-space: pre-wrap; text-align:

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		151

```

left">${message}</pre>
</details>
`;

    async function verifyBackup(fileName) {
        await handleAction(`/api/backups/test-boot`, 'POST', {path:
[currentPath, fileName].join("\\")}, true);
    }

    async function schedulerMenu() {
        Swal.fire({
            title: 'Зачекайте...',
            text: 'Виконуємо операцію',
            allowOutsideClick: false,
            didOpen: () => {
                Swal.showLoading();
            }
        });

        const response = await fetch(`/api/backups/schedules`);
        const tasks = await response.json();

        await showScheduledBackupsAlert(tasks);
    }

    async function showScheduledBackupsAlert(tasks) {
        if (!tasks || tasks.length === 0) {
            Swal.fire({
                title: 'Планувальник завдань',
                text: 'Запланованих бекапів не знайдено.',
                icon: 'info',
                confirmButtonColor: '#3085d6'
            });
            return;
        }
        console.log(tasks)
        let tableRows = '';
        tasks.forEach(task => {
            let stateBadge = '';
            if (task.State === 'Ready') {
                stateBadge = `<span class="badge bg-success" style="padding:
4px 8px;">${task.State}</span>`;
            } else if (task.State === 'Disabled') {
                stateBadge = `<span class="badge bg-secondary" style="padding:
4px 8px;">${task.State}</span>`;
            } else {
                stateBadge = `<span class="badge bg-warning text-dark"
style="padding: 4px 8px;">${task.State}</span>`;
            }

            let resultText = task.LastResult === 0
                ? `<span style="color: #198754; font-weight: bold;">Успішно
(${task.LastResult})</span>`
                : `<span style="color: #dc3545;">Інше
(${task.LastResult})</span>`;

            tableRows += `
                <tr>
                    <td style="text-align: left; font-weight:
500;">${task.Name.replace('VM Backup_', '')}</td>
                    <td class="text-center align-middle">${stateBadge}</td>
                    <td>${task.Schedule}</td>
                    <td class="text-center align-middle">${task.NextRun}</td>
                    <td class="text-center align-middle">${task.LastRun}</td>

```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		152

```
 ${resultText}</td> |
```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				
Змн.	Арк.	№ докум.	Підпис	Дата		153

```
    }

    document.addEventListener('DOMContentLoaded', loadBackups);
</script>
</body>
```

		Бахур Ю. В.			ККІКБ.КР.Б – 123 – 01 – ПЗ	Арк.
		Головня О. С.				154
Змн.	Арк.	№ докум.	Підпис	Дата		