

Ім'я користувача:
К-ра прогр. забезп. КС Tanashchyshena

ID перевірки:
1016348663

Дата перевірки:
11.06.2024 19:09:16 EEST

Тип перевірки:
Doc vs Internet + Library + DB

Дата звіту:
12.06.2024 07:01:24 EEST

ID користувача:
76848

Назва документа: Борсук А.Ю. - Антиплагіат

Кількість сторінок: 88 Кількість слів: 11947 Кількість символів: 95062 Розмір файлу: 9.94 MB ID файлу: 1016151936

Виявлено модифікації тексту (можуть впливати на відсоток схожості)

11.1% Схожість

Найбільша схожість: 11% з джерелом з Бібліотеки (ID файлу: 1015255623)

4.09% Джерела з Інтернету

1

Сторінка 90

11.1% Джерела з Бібліотеки

2

Сторінка 90

1.31% Цитат

Цитати

8

Сторінка 91

Не знайдено жодних посилань

5.88% Вилучень

Деякі джерела вилучено автоматично (фільтри вилучення: кількість знайдених слів є меншою за 10 слів та 2%)

4.24% Вилучення з Інтернету

303

Сторінка 92

5.06% Вилученого тексту з Бібліотеки

523

Сторінка 95

Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

Підозріле форматування

17
сторінок

Зміст

РЕФЕРАТ.....	2
ABSTRACT.....	4
ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ.....	5
ВСТУП.....	6
РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ СИСТЕМИ ВІДДАЛЕНОЇ ВЗАЄМОДІЇ З ВІРТУАЛЬНИМИ МАШИНАМИ.....	8
1.1. Постановка задачі.....	8
1.2. Аналіз аналогів програмного продукту.....	10
1.3. Вибір архітектури системи.....	14
1.4. Обґрунтування вибору інструментальних засобів та вимоги до апаратного забезпечення.....	20
Висновки до розділу 1.....	26
РОЗДІЛ 2. ПРОЕКТУВАННЯ СИСТЕМИ ВІДДАЛЕНОЇ ВЗАЄМОДІЇ З ВІРТУАЛЬНИМИ МАШИНАМИ.....	27
2.1. Визначення варіантів використання та об'єктно-орієнтованої структури системи.....	27
2.2. Розробка бази даних системи.....	40
2.3. Проектування та реалізація алгоритмів роботи системи.....	42
2.4. Реалізація додатку системи віддаленої взаємодії з віртуальними машинами.....	56
Висновки до розділу 2.....	59
РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З СИСТЕМОЮ ВІДДАЛЕНОЇ ВЗАЄМОДІЇ З ВІРТУАЛЬНОЮ МАШИНОЮ.....	60
3.1. Порядок встановлення та налаштування системи.....	60
3.2. Структура інтерфейсу. Інтерфейс та порядок роботи з системою.....	65
3.3. Тестування системи.....	83
Висновки до розділу 3.....	85
ВИСНОВКИ.....	86
ДОДАТКИ.....	89

РЕФЕРАТ

Випускна кваліфікаційна робота бакалавра складається з програмного комплексу системи віддаленої взаємодії з віртуальними машинами, з можливістю додавання кількох віртуальних машин. Робота системи полягає у наступних можливостях: створення декількох облікових записів в системі для одного телеграм акаунта, додавання декількох віртуальних машин прив'язаних до одного облікового запису, видалення облікових записів, видалення та редагування даних про віртуальні машини, використання SSH для виконання команд на віртуальній машині, використання SFTP для операцій над файловою системою машини, збір метрик, можливість зміни мови системи. Пояснювальна записка до випускної роботи містить 70 сторінок, 56 ілюстрацій та 12 таблиць.

Метою роботи є розробка системи призначеної для взаємодії користувача з його віртуальними машинами, можливість виконання команд, зняття метрик з віртуальної машини, яка буде зручною в користуванні з будь-якого девайсу користувача. Передбачено можливість повної взаємодії з машиною, окрім тих команд, які вимагають блокування основного потоку машини. Функція підняття віртуальних машин – відсутня.

В роботі визначено основні завдання на розробку системи, проаналізовано аналоги розробленої системи, обґрунтовано вибір архітектурного патерна для реалізації системи – N-tier та REST Api. Наведено сценарії роботи програмного комплексу, загальна структура програмного комплексу, опис бази даних системи (СУБД Microsoft SQL Server), опис алгоритмів взаємодії окремих модулів системи, опис об'єктної структури системи (діаграма класів), алгоритми роботи основних модулів, реалізованих з використанням мови програмування C# та Python. Проведене мануальне та Unit-тестування.

КЛЮЧОВІ СЛОВА: SSH, SFTP, ВІРТУАЛЬНА МАШИНА, LINUX,
МОНІТОРИНГ, ВІДДАЛЕНЕ АДМІНІСТРУВАННЯ, КЕРУВАННЯ
ВІРТУАЛЬНИМИ МАШИНАМИ, ВИКОННЯ КОМАНД, КОНТРОЛЬ
ПАНЕЛЬ ВІРТУАЛЬНИХ МАШИН.

ABSTRACT

The graduation thesis of the bachelor consists of a software complex of a system of remote interaction with virtual machines, with the possibility of adding several virtual machines. The operation of the system consists of the following possibilities: creating several accounts in the system for one Telegram account, adding several virtual ones tied to one account, deleting accounts, deleting and editing data about the virtual machine, using SSH to execute commands on the virtual machine, using SFTP to operations on the file system of the machine, collecting metrics, the ability to change the system language. The explanatory note to the final thesis contains 70 pages, 56 illustrations and 12 tables.

The purpose of the work is the development of a system designed for user interaction with its virtual machines, the ability to execute commands, remove metrics from a virtual machine, which will be convenient to use from any user device. The possibility of full interaction with the machine is provided, except for those commands that require blocking the main thread of the machine. The function of raising virtual machines is absent.

In the work, the main tasks for the development of the system are defined, analogs of the developed system are analyzed, the choice of an architectural pattern for the implementation of the system - N-tier and REST Api - is substantiated. The scenarios of the software complex, the general structure of the software complex, the description of the system database (DBMS Microsoft SQL Server), the description of the interaction algorithms of individual system modules, the description of the object structure of the system (class diagram), the work algorithms of the main modules implemented using the programming language are given C# and Python. Conducted manual and unit testing.

KEYWORDS: SSH, SFTP, VIRTUAL MACHINE, LINUX, MONITORING, REMOTE ADMINISTRATION, VIRTUAL MACHINE MANAGEMENT, COMMAND EXECUTION, VIRTUAL MACHINE CONTROL PANEL.

4

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

Арі	– набір визначень взаємодії різнотипного програмного забезпечення.
БД	– база даних
ІС	– інформаційна система
СКБД	– система керування базами даних
Лог	– службова або статистична інформація про подію

ВСТУП

Актуальність теми. Актуальність теми полягає в зростаючій децентралізації проектів та кількості віддалених віртуальних. Розробникам зараз потрібен зручний інструмент, який дасть їм змогу легко, швидко та зручно виконувати команди, збирати метрики, завантажувати файли на віртуальні машини. Оскільки децентралізація є неминучим процесом, адже вона дозволяє полегшити розробку проектів, полегшити їх розгортання та збільшити їх гнучкість, система надає змогу легко, з будь-якого девайсу взаємодіяти із ними та переглядати їх стан. Це стане перевагою для розробників, адже тепер можна буде спостерігати за своїми машинами з екрану телефона.

Мета випускної роботи. Метою випускної роботи є реалізація системи, яка дозволить користувачам взаємодіяти зі своїми віртуальними машинами. Основна мета полягає в створенні проекту, який надасть зручну можливість створювати користувачу декілька облікових записів, додавати необмежену кількість віртуальних машин до своїх них, адмініструвати свої акаунти та машини, а найголовніше виконувати на них команди за допомогою SSH підключення, виконувати операції над папками/файлами за допомогою SFTP підключення та збирати і подавати метрики віртуальної машини у зручному для користувача вигляді.

Поставлена мета випускної роботи, визначає наступні завдання:

- розробка модуля авторизації та реєстрації;
- розробка модуля видалення облікового запису;
- розробка модуля додання/редагування/видалення віртуальних машин;
- розробка клієнта, який відповідає за SSH підключення;
- розробка клієнта, який відповідає за SFTP підключення;

- розробка клієнта, який буде збирати метрики, систематизувати їх та надсилати на клієнтську частину;
- розробка модуля створення графіків;
- розробка захищеного Web Api;
- захист конфіденційної інформації користувача;
- локалізація системи;
- інтеграція з Telegram Bot Api;

Об'єкт дослідження. Об'єктом дослідження є система віддаленої взаємодії з віртуальними машинами, яка надає необхідний інструментарій для легкої взаємодії з ними, спостереженням за їх станом, а також інструментарій для адміністрування акаунтів та прив'язаних до них машин.

Предмет дослідження. Предметом дослідження є розробка та реалізація системи віддаленої взаємодії з віртуальними машинами, включаючи створення функціоналу, який дозволить створювати облікові записи всередині системи, прив'язувати до них віртуальні машини, адмініструвати облікові записи та машини, взаємодіяти із машинами, забезпечувати загальну картину стану віртуальної машини.

РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ СИСТЕМИ ВІДДАЛЕНОЇ ВЗАЄМОДІЇ З ВІРТУАЛЬНИМИ МАШИНАМИ

1.1. Постановка задачі

Темою випускної роботи є розробка системи віддаленої взаємодії з віртуальними машинами, призначеної для маніпуляції їх станом та вмістом, а також отримання загальної картини стану машини. Система буде надавати користувачу змогу повної взаємодії з машиною, яке буде реалізовано за допомогою SSH-клієнту, який є частиною застосунку. Даний клієнт дозволяє виконувати всі команди, окрім тих, які дають відповіді в реальному часі. Не менш важливою частиною системи буде SFTP-клієнт, який буде надавати користувачам можливість безпечно та захищено виконувати операції над папками та файлами віртуальної машини. Також система буде мати клієнт, який збиратиме дані з віртуальної машини та надсилатиме їх серверу на обробку, результати у зручній для читання формі будуть надсилатися користувачу.

У випускній роботі необхідно розробити засоби управління обліковими записами та прив'язаними до них віртуальними машинами, клієнти для зміни та отримання стану віртуальної машини. Для реалізації поставленого завдання, основними етапами є:

1. Розробити дизайн клієнтської частини застосунку.

1.1. Оскільки інформація система віддаленої взаємодії з віртуальними машинами буде мати інтеграцію з Telegram Bot Api, то інтерфейс користувача буде представлено у вигляді телеграм бота. Головним завданнями є формувати текст так, щоб його було легко та зручно читати, та забезпечити сценарність бота, щоб користувач не міг виконувати тих дій,

які не відповідають його стану, що буде досягнуто за допомогою ReplyKeyboardMarkups.

1.2. За допомогою функціоналу мови програмування C# забезпечити повну та коректну локалізацію користувацького інтерфейсу.

2. Реалізувати ядро системи управління обліковими записами та прив'язаними віртуальними машинами.

2.1. Написати програмні модулі для:

- модуль авторизації користувачів в системі;
- модуль реєстрації в системі;
- модуль видалення облікових записів користувача із системи;
- модуль виводу прив'язаних до акаунта віртуальних машин;
- модуль прив'язки віртуальної машини;
- модуль видалення віртуальної машини;
- модуль редагування інформації віртуальної машини;
- модуль зміни віртуальної машини;
- модуль зміни акаунта;
- модуль збереження користувачів та прив'язаних віртуальних машин в базі даних.

2.2. Реалізувати необхідні операції для всіх таблиць бази даних системи.

3. Реалізувати клієнти системи для взаємодії з віртуальними машинами та отримання інформації про її стан:

3.1. Написати програмні модулі для:

- SSH-клієнт для віддаленого виконання команд на віртуальній машині;
- SFTP-клієнт для віддаленого виконання операцій над папками/файлами системи;
- клієнт, який буде збирати метрики;

- модуль, який буде перетворювати зібрані метрики у графічний вигляд – діаграми та графіки;

4. Реалізувати структуру збереження даних наступного виду:

4.1. Дані про віртуальну машину (назва, ідентифікатор акаунта, ім'я користувач, зашифрований пароль, хост, порт);

4.2. Дані про користувача (ідентифікатор Telegram, ім'я користувача, хеш пароля, електронна адреса, нормалізоване ім'я користувача, нормалізована електронна адреса, код мови інтерфейсу);

5. Розробити план тестування системи управління контентом. Зокрема передбачити наступні види тестування:

5.1. Тестування функціональне.

5.2. Крос-девайсне тестування.

5.3. Тестування безпеки даних.

5.4. Тестування графічного інтерфейсу та “usability” тестування.

5.5. Unit-тестування системи.

Результатом реалізації поставленого завдання є розгорнута інформаційна система віддаленої взаємодії з віртуальними машинами, що містить в своєму складі серверну структуру збереження даних, багатокористувацький клієнтський додаток для реалізації функціональних можливостей, та засоби контролю та керування доступом.

1.2. Аналіз аналогів програмного продукту

Система віддаленої взаємодії з віртуальними машинами забезпечує легку, швидку та надійну взаємодію користувача з його машинами. На сучасному ринку програмних засобів представлений набір інструментів та технологій, що допомагають вирішенню цих задач. Для розгляду аналогів, за критеріями популярності та позитивного рейтингу вибрані додатки наведені в табл. 1.1.

10

Назва	Розробник	Платформа	СУБД	Умови ліцензії
Virtualmin	Virtualmin, Inc.	Linux	MySQL, MariaDB	Virtualmin GPL (загальна публічна ліцензія)
Proxmox VE	Proxmox Server Solutions GmbH	Linux	SQLite, PostgreSQL	Відкрита ліцензія AGPLv3
Webmin	Jamie Cameron	Linux	MySQL, SQLite	Webmin GPL (загальна публічна ліцензія)
cPanel	cPanel, L.L.C.	Linux	MySQL	Комерційна ліцензія
Grafana	Grafana Labs	Крос-платформний	MySQL, PostgreSQL, InfluxDB	Apache License 2.0
OpenStack Horizon	OpenStack Foundation та спільнота розробників	Linux	MySQL, PostgreSQL	Відкрита ліцензія Apache 2.0
Virtkick	Virtkick Sp. z o.o.	Linux	PostgreSQL	Комерційна ліцензія

Таблиця 1.1 – Перелік аналогів системи

Virtualmin – це потужна контроль панель для керування хмарними обчисленнями та веб-хостингом на базі Linux. Він надає засоби для створення та керування віртуальними серверами, доменами, поштою та базами даних.

Proxmox VE – це віртуалізаційна платформа для управління віртуальними машинами та контейнерами на базі Linux. Він надає засоби для створення, управління та моніторингу віртуальних машин та ресурсів обчислювального центру.

Webmin – це проста у використанні контроль панель для керування сервера на базі Linux. Він надає інтерфейс для керування різними аспектами сервера, включаючи веб-сервери, пошту, бази даних та інші.

cPanel – це одна з найпопулярніших комерційних контроль панелей для веб-хостингу. Він надає засоби для керування веб-сайтами, поштою, базами даних та іншими аспектами веб-хостингу.

Grafana – це інструмент для моніторингу та візуалізації даних, який надає гнучкі можливості для створення графіків та дашбордів на основі різних джерел даних.

OpenStack Horizon – це веб-інтерфейс для керування хмарною інфраструктурою OpenStack. Він надає засоби для керування віртуальними машинами, мережами, об'єктами сховищами та іншими ресурсами хмарного середовища.

Virtkick – це інтерфейс для керування віртуальними машинами на базі KVM. Він надає простий і зручний інтерфейс для створення, управління та моніторингу віртуальних машин.

Порівняємо аналоги за не функціональними можливостями.

Програма	Версія для одного користувача	Мережева версія	Веб-інтерфейс	Інтеграція
Virtualmin	Virtualmin GPL	Ні	Так	Api, Cli, Whmcs
Proxmox VE	Proxmox VE	Так	Так	Rest Api, Cli
Webmin	Webmin	Ні	Так	Perl API, CLI
cPanel	cPanel	Так	Так	Api, Whmcs
Grafana	Grafana	Так	Так	Rest Api
OpenStack Horizon	Horizon	Так	Так	OpenStack API
Virtkick	Virtkick	Так	Так	Api

Таблиця 1.2 – Не функціональні можливості

Розглянемо аналоги в розрізі функціональності по підбору персоналу.

Програма	Авторизація	Прив'язка машин	Створення облікових записів	Виконання команд на машині	Виконання операцій над папками/файлами	Збір метрик	Вивід метрик у вигляді графіків
Virtualmin	Так	Так	Так	Ні	Ні	Ні	Ні
Proxmox VE	Так	Так	Так	Так	Так	Так	Так
Webmin	Так	Так	Так	Так	Так	Ні	Ні
cPanel	Так	Так	Так	Так	Так	Ні	Ні
Grafana	Ні	Ні	Ні	Ні	Ні	Так	Так
OpenStack Horizon	Так	Так	Так	Ні	Ні	Так	Так
Virtkick	Так	Так	Так	Ні	Ні	Ні	Ні

Таблиця 1.3 – Функціональні можливості

Переглянувши список аналогів інформаційної системи можна зробити висновки, що мало які конкуренти поєднують в собі увесь функціонал, що пропонує випускна робота – це свідчить про інноваційність продукту та те, що він зможе легко знайти свою аудиторію.

Таким чином, основним рисами нової системи мають бути: змога створювати декілька облікових записів для одного користувача та прив'язувати до них різні віртуальні машини, з метою розділення пріоритетів та цілей, наприклад, один акаунт для робочих машин, а інший для особистих, віддалене виконання команд на віртуальній машині, віддалене виконання операцій над її папками/файлами, збір метрик машини, задля забезпечення картини її стану та можливість зміни мови системи, щоб зробити користування застосунком зручнішим.

1.3. Вибір архітектури системи

Для розробки системи віддаленої взаємодії з віртуальними машинами був обраний архітектурний патерн N-tier. Цей підхід до створення програмного забезпечення передбачає поділ логіки, функцій та даних програми на кілька незалежних рівнів або шарів, кожен з яких виконує певні функції та має свою відповідальність. Це сприяє модульності, розширюваності та легкості підтримки програмного продукту.

Архітектура N-tier популярна в програмуванні з кількох причин. По-перше, вона забезпечує модульність, дозволяючи розбивати програмне забезпечення на окремі рівні. Це дає змогу розробникам працювати над різними компонентами системи незалежно один від одного, що полегшує розподіл завдань, спрощує тестування та підтримку системи.

По-друге, архітектура N-tier відзначається масштабованістю. Кожен рівень може бути масштабований незалежно від інших, що дозволяє гнучко налаштовувати ресурси для кожного рівня. Це забезпечує оптимальну продуктивність системи при збільшенні обсягу даних або навантаження.

Крім того, ця архітектура сприяє легкості підтримки та розширення системи. Розділення функціональності на різні рівні дозволяє розробникам зосередитися на конкретних завданнях без впливу на решту системи, що спрощує розширення функціоналу та виправлення помилок.

Також N-tier архітектура дозволяє розподіляти різні рівні системи на різних серверах або віртуальних машинах, що підвищує масштабованість та доступність системи, забезпечуючи резервування та відновлення даних і знижуючи ризик втрати інформації.

Архітектура N-tier має переваги у сфері безпеки, оскільки розділення функціональності на різні рівні дозволяє реалізувати механізми безпеки на

кожному рівні окремо, що допомагає захистити дані та функціонал системи від несанкціонованого доступу.

Всі ці переваги роблять архітектуру N-tier привабливою для розробників та компаній, що створюють програмне забезпечення. Вона забезпечує гнучкість, масштабованість, легкість підтримки, розширення та безпеку, що робить її ефективним вибором для багатьох проектів.

Основні переваги архітектури N-tier:

1. **Модульність:** розділення програмного забезпечення на окремі рівні дозволяє розробляти, тестувати та модифікувати кожен рівень незалежно від інших, що полегшує розподіл завдань між розробниками і забезпечує легкість підтримки та розширення системи.
2. **Масштабованість:** кожен рівень можна масштабувати незалежно від інших, що дозволяє збільшувати пропускну здатність, надійність або продуктивність лише на тих рівнях, які цього потребують, зберігаючи решту рівнів незмінними.
3. **Легкість супроводження:** розділення логіки та функцій на різні рівні спрощує виявлення й виправлення помилок, забезпечуючи більшу стабільність та надійність програми. Розробники можуть фокусуватися на конкретних рівнях, не впливаючи на решту системи.

Основні недоліки архітектури N-tier:

1. **Збільшений обсяг комунікації:** кожен рівень має взаємодіяти з іншими рівнями через інтерфейси або контракти, що може збільшити обсяг комунікації між рівнями і вплинути на продуктивність системи.

2. **Складність розгортання:** використання архітектури N-tier може ускладнити розгортання та налагодження системи, оскільки кожен рівень потребує окремих сервісів, налаштувань та залежностей.
3. **Збільшення витрат:** архітектура N-tier може вимагати більше ресурсів, таких як сервери та мережеве обладнання, що може призвести до збільшення витрат.



Рисунок 1.1 – Узагальнена архітектура N-tier

1. **Presentation Tier (Шар презентації):** Цей рівень відповідає за відображення та взаємодію з користувачем. Він забезпечує веб-інтерфейс або користувацький інтерфейс (UI), що дозволяє користувачам взаємодіяти з програмою. На цьому шарі розташовані такі компоненти, як веб-сторінки, форми та контролери.
2. **Business Logic Tier (Шар бізнес-логіки):** Цей рівень містить бізнес-логіку програми. Він відповідає за обробку бізнес-правил, логіки операцій та виконання розрахунків. Шар бізнес-логіки може включати

16

класи, сервіси, моделі даних та інші компоненти, які реалізують функціональність програми.

3. **Data Access Tier (Шар доступу до даних):** Цей рівень забезпечує доступ до даних та їх управління. Він виконує завдання, такі як підключення до бази даних, виконання запитів та оновлення даних. Шар доступу до даних може включати класи або компоненти, що взаємодіють з базою даних або іншими джерелами даних.

4. **Infrastructure Tier (Інфраструктурний шар):** Цей рівень надає загальні служби та інфраструктуру для функціонування програми. Він може включати такі компоненти, як логування, кешування, безпека, роутинг тощо. Інфраструктурний шар сприяє перевикористанню коду, централізованому керуванню та вирішенню загальних завдань.

Таким чином, цей архітектурний шаблон забезпечує всі необхідні технологічні можливості для створення гнучкого додатка відповідно до принципів SOLID. У нашому випадку цей підхід гарантує не лише успішну реалізацію проекту, але й його подальшу модифікацію та розвиток відповідно до змін вимог користувачів та процедур обробки інформації.

Архітектурою Web API застосунку було обрано архітектуру REST. Архітектура **REST(Representational State Transfer)** є підходом до розробки програмного забезпечення, що базується на принципах, які сприяють створенню розподілених систем. Основною ідеєю REST є використання стандартних протоколів комунікації, таких як HTTP, і уніфікованого набору операцій для взаємодії між клієнтом і сервером. Дозволяючи розділити клієнтську і серверну частини застосунку, REST спрощує розробку та підтримку системи, роблячи її більш масштабованою та гнучкою.



Рисунок 1.2 – Узагальнена архітектура Rest

Основні принципи REST включають:

1. **Ресурси (Resources):** Це центральні об'єкти в архітектурі REST. Кожен ресурс має свій унікальний ідентифікатор (URI), за яким до нього можна звертатися. Наприклад, у веб-системі ресурсами можуть бути статичні сторінки, користувачі, замовлення тощо.
2. **Методи (Methods):** REST використовує HTTP методи для виконання дій з ресурсами. Найпоширеніші методи включають GET (для отримання ресурсу), POST (для створення нового ресурсу), PUT (для оновлення існуючого ресурсу) та DELETE (для видалення ресурсу).
3. **Представлення(Representations):** Ресурси можуть мати різні представлення, такі як JSON, XML або HTML, в залежності від потреб клієнта та сервера. Наприклад, клієнт може вимагати JSON-представлення ресурсу, тоді як веб-браузер може відображати HTML-представлення того ж ресурсу.
4. **Самоповідомлення(Self-descriptive):** Кожен запит має містити достатньо інформації для розуміння того, як його обробити. Наприклад, використання заголовків HTTP (наприклад, Content-Type) і коду відповіді (наприклад, статус коду) дозволяє отримати необхідну інформацію про запит і відповідь.
5. **Без стану(Stateless):** Сервер не зберігає жодної інформації про стан клієнта між запитами. Це означає, що кожен запит повинен містити всю

необхідну інформацію для обробки, інакше сервер не може коректно відповісти.

Переваги архітектури REST включають:

1. **Простота:** REST є простим у розумінні і використанні завдяки своїм основним принципам та використанню стандартних протоколів.
2. **Масштабованість:** Розділення функціональності на ресурси дозволяє легко масштабувати систему шляхом додавання або заміни серверів, а також використання кешування для підвищення продуктивності.
3. **Незалежність від платформи:** REST не залежить від конкретних технологій або мов програмування, що дозволяє використовувати його в будь-яких середовищах.
4. **Підтримка кешування:** Використання HTTP дозволяє ефективно кешування ресурсів, що зменшує навантаження на сервер і покращує продуктивність застосунку.
5. **Можливість асинхронного обміну:** REST підтримує асинхронний обмін даними між клієнтом і сервером, що робить його ефективним для сучасних веб-застосунків, де час відповіді може варіюватися.

Спираючись саме на вищенаведені особливості та переваги архітектури Rest вона була обрана, як архітектура Web Api. Вона забезпечить асинхронність, надійність, масштабованість, що і потрібно, щоб користувач якомога швидше отримував відповідь на всі свої дії та не відчував ніяких затримок.

1.4. Обґрунтування вибору інструментальних засобів та вимоги до апаратного забезпечення

Для створення ІС, доцільно застосувати: мову програмування C# для основної частини програми, мову програмування Python для написання скрипту, що буде збирати метрики з віртуальної машини, Microsoft SQL Server, Redis, Docker, ORM-бібліотеку мови C# EntityFramework, бібліотеку для виконання SSH- та SFTP-підключень мови програмування C# SSH.NET та бібліотеки Telegram.Bot, яка дозволить інтегрувати в проект Telegram Bot Api. В якості IDE будуть використані Visual Studio 2022 та Visual Studio Code для C# та Python відповідно.

C# – це об'єктно-орієнтована мова програмування, розроблена компанією Microsoft. Синтаксис C# близький до C++ і Java. Мова має строгу статичну типізацію, підтримує поліморфізм перевантаження операторів, вказівники на функції-члени класів, атрибути, події, властивості, винятки, коментарі у форматі XML.

Python – це високорівнева, інтерпретована мова програмування, яка визначається своєю простотою та читабельністю коду. Структури даних високого рівня разом із динамічною семантикою та динамічним зв'язуванням роблять її привабливою для швидкої розробки програм, а також як засіб поєднання наявних компонентів.

Microsoft SQL Server – система управління базами даних, яка розробляється корпорацією Microsoft. Як сервер даних виконує головну функцію по збереженню та наданню даних у відповідь на запити інших застосунків, які можуть виконуватися як на тому ж самому сервері, так і у мережі.

Redis – розподілене сховище пар ключ-значення, які зберігаються в оперативній пам'яті, з можливістю забезпечувати довговічність зберігання на бажання користувача. Це програмне забезпечення з відкритим сирцевим кодом написане на ANSI C.

Docker – інструментарій для управління ізольованими Linux-контейнерами. Docker доповнює інструментарій LXC більш високорівневим API, що дозволяє керувати контейнерами на рівні ізоляції окремих процесів.

EntityFramework – це набір технологій в ADO.NET, які підтримують розробку програмно-орієнтованих на дані програмних додатків.

SSH.NET – бібліотека для роботи віддаленої взаємодії через SSH підключення. На створення цього проекту надихнула бібліотека Sharp.SSH, яка була перенесена з Java і не підтримувалася протягом тривалого часу. Ця бібліотека є повністю переписаною, без будь-яких залежностей від третіх сторін, з використанням паралелізму для досягнення найкращої продуктивності.

Telegram.Bot – найпопулярніший клієнт .NET для Telegram Bot API.

Visual Studio – серія продуктів фірми Майкрософт, які містять інтегроване середовище розробки програмного забезпечення та низку інших інструментальних засобів. Ці продукти дають змогу розробляти як консольні програми, так і програми з графічним інтерфейсом, включно з підтримкою технології Windows Forms, а також вебсайти, вебзастосунки, вебслужби як у рідному, так і в керованому кодах для всіх платформ, що підтримуються Microsoft Windows, Windows Mobile, Windows Phone, Windows CE, .NET Framework, .NET Compact Framework та Microsoft Silverlight.

Visual Studio Code (VS Code) – це безкоштовний та потужний текстовий редактор, розроблений компанією Microsoft. Він є одним з найпопулярніших редакторів серед розробників завдяки своїм функціональним можливостям та широким спектром плагінів. VS Code надає зручне та ергономічне робоче середовище для програмістів. Він підтримує багато мов програмування, включаючи JavaScript, Python, C++, Java, HTML, CSS та багато інших. Редактор має потужні функції редагування, такі як підсвічування синтаксису, автодоповнення коду, вбудовану підтримку Git та можливість відлагодження коду.

Порівняльний аналіз СКБД дозволяє раціонально вибрати систему керування базами даних для проекту. У якості альтернатив будуть розглянуті наступні СКБД: Microsoft SQL Server, MySQL, PostgreSQL. Всі 3 варіанти мають безкоштовну обмежену версію.

Усі обрані системи керування базами даних підходять для проведення аналізу й порівняння тому що реалізують не реляційну модель даних.

	Розмір БД	Розмір документа	Розмір рядка
Microsoft SQL Server	524,272 терабайти	2 гігабайти	8,000 байт
MySQL	64 терабайти	4 гігабайти	65,535 байт
PostgreSQL	Необмежено	1 гігабайт	1 гігабайт

Таблиця 1.4 – Максимально можливий обсяг збережених даних для кожної СКБД

За критерієм тригерів та збережених процедури всі альтернативи ідентичні. Усі підтримують тригери, процедури й функції.

	Windows	Linux	macOS	Android	Symbian
Microsoft SQL Server	+	-	-	-	-
MySQL	+/-	+	+	-	-
PostgreSQL	+	+	+	-	-

Таблиця 1.5 – Аналіз підтримуваних альтернативами операційних систем

Інформаційна система віддаленої взаємодії з віртуальними машинами містить у собі конфіденційну інформацію, для запобігання несанкціонованого доступу застосовуються різні способи захисту.

	Ідентифікація	Захист від brute-force	Шифрування	Сертифікація безпеки
Microsoft SQL Server	+	+	+	+
MySQL	+	+/-	+	-

PostgreSQL	+	+/-	+	-
------------	---	-----	---	---

Таблиця 1.6 – Аналіз систем забезпечення безпеки даних альтернативах

Проведений аналіз таких СКБД, як Microsoft SQL Server, MySQL, PostgreSQL показав, що для виконання поставлених задач найбільш підходящими є Microsoft SQL Server та PostgreSQL. Microsoft SQL Server має деякі переваги порівняно з MySQL та PostgreSQL. Microsoft SQL Server надає більш захищену модель зберігання даних, що допоможе зберегти конфіденційну інформацію користувачів недоторканою. Він також підтримує багато типів запитів і має потужні можливості для агрегаційних операцій. Також він має більший об'єм пам'яті на один запис 2 гігабайти проти 1 гігабайта в PostgreSQL. Крім того, Microsoft SQL Server має добре розвинену документацію, що полегшує розробку та підтримку. Однак, варто зазначити, що Microsoft SQL Server офіційно підтримується тільки на Windows, однак на таких платформах як Linux та macOS це вирішується за допомогою Docker.

	Microsoft SQL Server	MySQL	PostgreSQL
Логічна модель даних	Database Schema	Database Schema	Database Schema
Фізична модель даних	Tables and Indexes	Tables and Indexes	Tables and Indexes
Типи даних	Широкий спектр типів, включаючи INT, VARCHAR, DATETIME тощо.	Широкий спектр типів, включаючи INT, VARCHAR, DATETIME тощо.	Широкий спектр типів, включаючи INT, VARCHAR, TIMESTAMP тощо.
Індекси	Унікальний, зазвичай кластеризований, індекси стовпців.	Унікальний, неунікальний, повний текст тощо.	Унікальний, неунікальний, головний, зовнішній, частковий тощо.
Мови маніпулювання	T-SQL (Transact-SQL)	SQL	SQL
Вбудовані мови програмування	CLR (Common Language Runtime), можливість інтеграції з .NET	Немає	PL/pgSQL, PL/Python, PL/Perl тощо.
Генератор форм,	SQL Server Reporting	Немає	Немає

23

звітів	Services (SSRS)		
Транзакції	Підтримка ACID-властивостей (Atomicity, Consistency, Isolation, Durability) через транзакції.	Підтримка ACID-властивостей через транзакції.	Підтримка ACID-властивостей через транзакції.
Тригери, процедури, що зберігаються	Підтримка тригерів та Stored Procedures	Підтримка тригерів та Stored Procedures	Підтримка тригерів та Stored Procedures
Платформи	Windows, Linux, Docker	Windows, Linux, macOS тощо.	Windows, Linux, macOS тощо.
Область застосування	Великі корпоративні системи, веб-додатки, додатки для мобільних пристроїв тощо.	Веб-додатки, невеликі та середні проекти, IoT тощо.	Веб-додатки, аналітика, наукові дослідження, IoT тощо.
Особливості	Велика масштабованість, розширена підтримка BI, висока доступність тощо.	Простота використання, широкий спектр розширень, підтримка реплікації тощо.	Гнучкість в роботі з даними, висока продуктивність, підтримка JSON, XML тощо.

Таблиця 1.7 – Аналіз загальних показників СКБД

Отже було обрано для створення програмного комплексу такі інструментальні засоби та технології:

- мову основної частини програми C#;
- мову скриптів Python;
- ORM-бібліотеку EntityFramework;
- бібліотеку SSH.NET;
- бібліотеку Telegram.Bot;
- середовище керування базами даних Microsoft SQL Server;
- середовище керування базами даних Redis;
- середовище контейризації Docker;

Використано такі програмні продукти та інтегровані середовища розробки: середовище розробки Visual Studio Code 1.89.1, середовище розробки Visual Studio 2022 17.9.6.

Висновки до розділу 1

Проведений аналіз предметної області дозволив визначити основні напрямки системи віддаленої взаємодії з віртуальними машинами. Необхідно реалізувати клієнти для SSH- SFTP-підключення, клієнт, що буде збирати метрики. Розробити можливість створення декількох акаунтів та прив'язки обмеженої кількості різних віртуальних машин до акаунта. Надати інструментарій для адміністрування акаунтів та прив'язаних до них машин.

Огляд наявних аналогів показав, що основними функціями таких систем є: наявність модулів маніпуляції обліковими записами, маніпуляції прив'язаними до них віртуальними машинами, можливість виконувати команди на машині, виконувати операції з її файлами/папками, збір метрик та подача їх у вигляді графіків, локалізація системи.

В роботі обґрунтовано вибір архітектури для реалізації системи:

- мову основної частини програми C#;
- мову скриптів Python;
- ORM-бібліотеку EntityFramework;
- бібліотеку SSH.NET;
- бібліотеки Telegram.Bot;
- середовище керування базами даних Microsoft SQL Server;
- середовище керування базами даних Redis;
- середовище контейнеризації Docker;

Використано такі програмні продукти та інтегровані середовища розробки: середовище розробки Visual Studio Code 1.89.1, середовище розробки Visual Studio 2022 17.9.6.

РОЗДІЛ 2. ПРОЕКТУВАННЯ СИСТЕМИ ВІДДАЛЕНОЇ ВЗАЄМОДІЇ З ВІРТУАЛЬНИМИ МАШИНАМИ

2.1. **Визначення варіантів використання та об'єктно-орієнтованої структури системи**

Інформаційна система має основні цілі: проект створюється з метою надання зручного крос-девайсного ресурсу, який надасть можливість повної віддаленої взаємодії з віртуальними машинами, а також дозволить користувачам отримувати інформацію про стан їх у вигляді графіків.

Вимоги користувачів

Зовнішні користувачі мають доступ до наступних функцій:

1. Можливість мати декілька облікових записів.
2. Можливість прив'язувати необмежену кількість машин до одного облікового запису.
3. Зміни мови інтерфейсу.
4. Можливість захищеного віддаленого виконання команд.
5. Можливість захищеного віддаленого виконання операцій над папками/файлами.
6. Генерування токена безпеки, який буде потім використовуватися для виконання http-запитів на захищенні контролери.
7. Можливість збирати метрики, щоб моніторити стан машини.
8. Генерування графіків на основі зібраних метрик.
9. Безпека та конфіденційність.



Рисунок 2.1 – Діаграма варіантів використання системи

Функціональні вимоги:

1. Авторизація користувачів в системі: В системі повинна бути представлення можливість реєстрації та авторизації користувача.

2. Ведення обліку користувачів: зберігати інформацію про користувачів, їх облікових даних, за допомогою яких буде відбуватися автентифікація та авторизація користувачів у системі.

3. Ведення обліку віртуальних машин: зберігати всю необхідну інформацію для віддаленого підключення до віртуальної машини. ІС повинна відображати користувачу тільки ті віртуальні машини, які прив'язані до його акаунта.

4. Можливість маніпуляції обліковими записами та прив'язаними до них віртуальними машинами.

5. Безперервне та безперебійне підключення: ІС повинна забезпечувати постійне та безперебійне як SSH-, так і SFTP-підключення до віртуальної

машини. Це забезпечить безпеку та повному виконання з віртуальною машиною дій, обраних користувачем, та гарантує отримання очікуваного результату.

6. Збір метрик: ІС повинна вміти виконувати повний збір первинних метрик, щоб користувач міг оцінити стан, в якому зараз перебуває віртуальна машина.

7. Зміна мови: ІС повинна надавати змогу користувачам користуватися нею, тією мовою, якою їм було б найзручніше.

8. Можливість збереження інформації: Система повинна зберігати інформацію і надавати можливість користувачу керувати нею.

Нефункціональні вимоги:

1. Навчання роботи з системою:

- Час, необхідний для навчання роботи з інструментами інформаційної системи, складає 1-2 години для звичайних користувачів і 1 годину для досвідчених.
- Час відповіді системи на звичайні запити не повинен перевищувати 3 секунд, а на складні запити – 10 секунд.
- Інтерфейс інформаційної системи має бути інтуїтивно зрозумілим для користувача і не вимагати додаткової підготовки.
- Надійність.
- Доступність – час, необхідний для обслуговування системи, не повинен перевищувати 15% від загального робочого часу.
- Середній час безперервної роботи – 30 робочих днів.
- Максимально допустима кількість помилок і дефектів у роботі системи – 1 помилка на 1000 запитів користувачів.

2. Продуктивність

Система повинна підтримувати мінімум 100 одночасно працюючих користувачів, пов'язаних з спільною базою даних.

3. Можливість експлуатації

- Масштабування – система повинна мати здатність підвищувати продуктивність із збільшенням кількості користувачів без негативного впливу на її роботу.
- Оновлення версій – оновлення версій повинно виконуватися автоматично, з урахуванням уподобань користувачів і розширення переліку запропонованих послуг.

Для реалізації проекту була вибрана архітектура N-tier та REST Api, та мови програмування C# й Python.

Аналіз функціональних вимог дозволив виділити наступні сутності, що забезпечать реалізацію програмного комплексу системи. Випускний проект було розділено на декілька менших проектів, з метою розділення відповідальності. Більшість проектів є бібліотеками класів, проте також є проекти Web Api та консольний проект, який є ботом. Діаграми класів кожного з проектів будуть представлені нижче.

Рисунок 2.2 – Діаграма класів проекту VMControlPanelTest

Рисунок 2.3 – Діаграма класів проекту Utilities

Рисунок 2.4 – Діаграма класів проекту Core

Рисунок 2.5 – Діаграма класів проекту Infrastructure

Рисунок 2.6 – Діаграма класів проекту UserInfrastructure

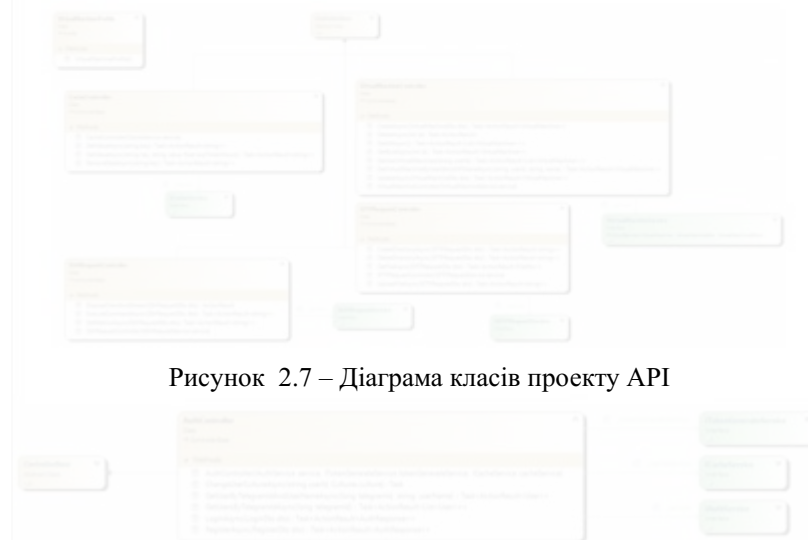


Рисунок 2.7 – Діаграма класів проекту API

Рисунок 2.8 – Діаграма класів проекту UserAPI

Рисунок 2.9 – Діаграма класів проекту Bot

Рисунок 2.10 – Діаграма класів модуля Commands проекту Bot

На даних рисунках можна виділити наступні класи:

- User – клас, який є репрезентацією колекції користувачів у БД та агрегує в собі всі основні параметри, характерні користувачеві;
- VirtualMachine – клас, який є репрезентацією колекції віртуальних машин у БД та агрегує в собі всі основні параметри необхідні для віддаленого під'єднання до віртуальної машини;
- RegisterDto, LoginDto – класи, що агрегують в собі всі дані необхідні для реєстрації та входу в систему відповідно;
- VirtualMachineDto – клас, який містить в собі всі дані, необхідні для створення об'єкту VirtualMachine;
- SSHRequestDto, SFTPRequestDto – класи, що містять в собі всі дані необхідні для виконання SSH- та SFTP-запитів відповідно;
- FileDto – клас, що містить в собі інформацію для маніпуляції файлами на віртуальній машині;
- GraphDto – клас, що містить в собі всю необхідну інформацію для побудови графіків;
- CpuMetricsDto, CpuPercentageDto, DiscMetricsDto, MemoryMetricsDto, NetMetricsDto, MetricsDto – класи модуля Metrics, які використовуються для зберігання та передачі інформації, що отримав клієнт збору метрик;
- CryptoService – клас, який відповідає за шифрування/розшифрування, кешування/розкешування конфіденційної інформації користувачів системи;
- CacheServiceTests – клас, в якому реалізовані тести сервісу кешування;
- UserServiceTests – клас, в якому реалізовані тести сервісу взаємодії з об'єктами сутності користувачів;
- VirtualMachineServiceTests – клас, в якому реалізовані тести сервісу взаємодії з об'єктами сутності віртуальних машин;

- SSHRequestServiceTests – клас, в якому реалізовані тести сервісу створення SSH-підключення, виконання команд та отримання метрик;
- SFTPRequestServiceTests – клас, в якому реалізовані тести сервісу створення SFTP-підключення та виконання операцій над файловою системою віртуальної машини;
- GraphServiceTests – клас, в якому реалізовані тести сервісу, який відповідає за створення графіків;
- UserDbContext – клас, який відповідає за під'єднання до бази даних та маніпуляції сутністю користувачів;
- IAuthService – інтерфейс, в якому описані всі методи, якими повинен володіти сервіс авторизації користувачів в системі;
- ITokenGenerateService – інтерфейс, в якому описані всі методи, якими повинен володіти сервіс генерування токена безпеки;
- AuthService – клас-сервіс, який відповідає за авторизацію користувача в системі, а також реалізує деякі запитати до бази даних користувачів;
- TokenGenerateService – клас-сервіс, який відповідає за генерування токена безпеки;
- UserConfiguration – клас-конфігурація, в якому описано як правильно зберігати поля класу User в сутність користувачів бази даних;
- TokenGenerateServiceConfiguration – клас-конфігурація, який містить дані для генерування правильного ключа безпеки;
- AppDbContext – клас, який відповідає за під'єднання до бази даних та маніпуляції сутністю віртуальних машин;
- ICrudService – дженерік інтерфейс, в якому описані CRUD методи;
- IVirtualMachineService – інтерфейс, в якому наведені всі необхідні методи маніпуляції віртуальними машинами, включно з CRUD методами;
- ICacheService – інтерфейс, в якому описані всі методи, якими повинен володіти сервіс кешування;

- ISSHRequestService – інтерфейс, в якому описані всі методи необхідні для успішного виконання SSH-запитів;
- ISFTPRequestService – інтерфейс, в якому описані всі методи необхідні для успішного виконання SFTP-запитів;
- VirtualMachineService – клас-сервіс, в якому реалізована вся логіка взаємодії та маніпуляції сутністю віртуальних машин;
- CacheService – клас-сервіс, який відповідає за кешування/розкешування даних;
- SSHRequestService – клас-сервіс, який реалізовує всю необхідну логіку для виконання SSH-запитів на віртуальній машині;
- SFTPRequestService – клас-сервіс, який реалізовує всю необхідну логіку для виконання SFTP-запитів на віртуальній машині;
- GraphsService – клас-сервіс, який реалізовує логіку побудови графіків;
- FileManager – клас-менеджер, який відповідає за взаємодію із вайлами;
- VirtualMachineConfiguration – клас-конфігурація, в якому описано як правильно зберігати поля класу VirtualMachine в сутність віртуальних машин бази даних;
- AuthController – клас-контролер, в якому описані всі ендпоїнти, які пов'язані зі взаємодією із сутністю користувачів, та логіка, яку необхідно виконати при отриманні http-запиту на відповідний ендпоїнт;
- VirtualMachineController – клас-контролер, в якому описані всі ендпоїнти, які пов'язані зі взаємодією із сутністю віртуальних машин, та логіка, яку необхідно виконати при отриманні http-запиту на відповідний ендпоїнт;
- CacheController – клас-контролер, в якому описані всі ендпоїнти, які пов'язані зі кешуванням даних, та логіка, яку необхідно виконати при отриманні http-запиту на відповідний ендпоїнт;

- SSHRequestController – клас-контролер, в якому описані всі ендпоїнти, які пов'язані зі виконанням SSH-запитів, та логіка, яку необхідно виконати при отриманні http-запиту на відповідний ендпоїнт;
- SFTPRequestController – клас-контролер, в якому описані всі ендпоїнти, які пов'язані зі виконанням SFTP-запитів, та логіка, яку необхідно виконати при отриманні http-запиту на відповідний ендпоїнт;
- VirtualMachineProfile – клас-конфігурація, необхідна для бібліотеки AutoMapper, в я ньому визначено як правильно поля класу VirtualMachineDto конвертувати в поля класу VirtualMachine;
- ITelegramBotHandlers – інтерфейс, в якому описані обробники, які необхідні для отримання та обробки повідомлень та помилок в Telegram боті;
- TelegramBotHandlers – клас, який реалізовує обробники повідомлень та помилок в Telegram боті;
- TelegramBot – дженерік клас-обгортка, для телеграм бота, головні ролі якого створити клієнт, який буде виконувати запити на Telegram Bot Api, та запустити прослуховування вхідних повідомлень;
- Command – абстрактний клас, який виступає базовим класом для сценаріїв реакції бота на повідомлення від користувача;
- MessageCommand – абстрактний клас, який виступає базовим класом для сценаріїв реакції бота на звичайні повідомлення користувача;
- CallbackQueryCommand – абстрактний клас, який виступає базовим класом для сценаріїв реакції бота на запити зворотного виклику від користувача;
- Класи модуля Commands – класи, що описують всі можливі реакції бота на дії користувача;
- ApiConfiguration – клас-конфігурація, який використовується базових адрес та ключів безпеки необхідних при виконання http-запитів;
- ListExtensions – клас-розширення, в якому реалізовані необхідні розширення для списків;

- RequestClient – базовий клас, який відповідає за виконання всіх http-запитів;
- CacheRequestClientExtensions – клас-розширення, в якому реалізовані запит на ендпоїнти, які відповідають за кешування даних;
- UserRequestClientExtensions – клас-розширення, в якому реалізовані запит на ендпоїнти, які відповідають за взаємодію та маніпулювання з сутністю користувачів;
- VirtualMachineRequestClientExtensions – клас-розширення, в якому реалізовані запит на ендпоїнти, які відповідають за взаємодію та маніпулювання з сутністю віртуальних машин;
- StateRequestClientExtensions – клас-розширення, в якому реалізовані запит на ендпоїнти, які відповідають за маніпуляцію станом користувача;
- SSHRequestRequestClientExtensions – клас-розширення, в якому реалізовані запит на ендпоїнти, які відповідають за виконання SSH-запитів;
- SFTPRequestRequestClientExtensions – клас-розширення, в якому реалізовані запит на ендпоїнти, які відповідають за виконання SFTP-запитів;
- OpenAIRequestClientExtensions – клас-розширення, в якому реалізовані запит на ендпоїнти платформи OpenAI;
- LocalizationManager – клас-менеджер, який відповідає за вибір локалізованого тексту, в залежності від локалізації, яку обрав користувач;
- Strings – файл ресурсів, в якому зберігаються увесь текст в англійському перекладі;
- Strings.uk – файл ресурсів, в якому зберігаються увесь текст в українському перекладі;
- State – клас, який описує стан, в якому зараз перебуває користувач;
- StateMachine – клас, який відповідає за маніпуляцію станом користувача;
- ConfigurationManager – клас-менеджер, який відповідає за конфігурування системи;

- Keyboards – клас, який відповідає за створення ReplyKeyboardMarkup;

- NoAuthCommands – клас, в якому зберігаються назви команд, яким не потрібна авторизація для виконання;

Таким чином, дана система реалізує поставлений перед нею функціонал.

2.2. Розробка бази даних системи

Реалізована база даних Microsoft SQL Server складається з 2 сутностей, які містять у собі дані про зареєстрованих користувачів та інформацію необхідну для підключення до віртуальних машин. База даних має назву VMControlPanel.

Назва таблиці	Призначення
Users	Колекція користувачів
VirtualMachines	Колекція даних для підключення до віртуальних машин

Таблиця 2.1 – Інформація про таблиці бази даних VMControlPanel

Структура бази даних наведена на рис. 2.11.



Рисунок 2.11 – Структурна схема БД

Далі наведено опис основних колекцій бази даних.

Колекція «Users» призначена збереження інформації про користувачів системи. В дану колекцію вводяться дані про користувача, такі як ідентифікатор Telegram, ім'я користувача, хеш пароля, електронна пошта, нормалізоване ім'я користувача та нормалізована електронна пошта. Структура колекції наведена нижче:

Назва	Тип даних	ПК	ЗП	Опис поля
Id	nvarchar(450)	+	+	Ідентифікатор
TelegramId	bigint	-	-	Ідентифікатор Telegram
UserName	nvarchar(max)	-	-	Ім'я користувача
PasswordHash	nvarchar(max)	-	-	Хеш пароля
Email	nvarchar(max)	-	-	Електронна пошта
NormalizedUserName	nvarchar(max)	-	-	Нормалізоване ім'я користувача
NormalizedEmail	nvarchar(max)	-	-	Нормалізована електронна пошта
Culture	nvarchar(max)	-	-	Код мови інтерфейсу

Таблиця 2.2 – Опис полів колекції «User»

Колекція «VirtualMachines» призначена для збереження інформації, необхідної для віддаленого підключення до віртуальної машини. В дану колекцію вводяться дані про віртуальну машину, такі як ім'я, ідентифікатор користувача, ім'я користувача, зашифрований пароль, хост та порт. Структура колекції наведена нижче:

Назва	Тип даних	ПК	ЗП	Опис поля
Id	int	+	-	Ідентифікатор
Name	nvarchar(max)	-	-	Ім'я
UserId	nvarchar(max)	-	+	Ідентифікатор користувача
UserName	nvarchar(max)	-	-	Ім'я користувача
PasswordEncrypted	nvarchar(max)	-	-	Зашифрований пароль
Host	nvarchar(max)	-	-	Хост
Port	int	-	-	Порт

Таблиця 2.3 – Опис полів колекції «VirtualMachines»

2.3.Проектування та реалізація алгоритмів роботи системи

Основними модулями системи є клієнти SSH- та SFTP-підключень, клієнт збору метрик, користувацький модуль адміністрування акаунтів і віртуальних машин.



Рисунок 2.12 – Діаграма активності

Після початку роботи із системою віддаленої взаємодії з віртуальними машинами (рис. 2.12) користувач може вибрати наступні дії з клавіатури: створити новий обліковий запис, виконати авторизацію в системі, видалити обліковий запис. Під час процесу створення нового облікового запису в користувача по чергово будуть запитані необхідні дані. Після успішного створення користувачеві знову будуть запропоновані три попередні дії. Під

час процесу видалення облікового запису система просить користувача обрати той із його записів, який він хоче видалити. Після чого користувачеві потрібно ввести пароль від нього, щоб система перевірила, що саме власник хоче видалити обліковий запис. Коли всі дані зібрані, система виконує спробу входу. Якщо вона успішна, то акаунт видаляється, якщо ні, то система просить користувача ввести пароль знову. Після видалення облікового запису користувачу доступні попередні три дії. Коли користувач успішно виконує авторизацію в системі, то йому будуть доступні дві нові дії, а саме: обрати для взаємодії необхідну віртуальну машину зі списку вже доданих та додати нову машину. Якщо користувач обрав «додати нову машину» і дані пройшли валідацію, інформація про віртуальну машину зберігається в базу даних, а користувачу виводиться повідомлення про успішність операції. Якщо дані не пройшли валідацію, то користувачу виводиться повідомлення про помилку.

Рисунок 2.13 – Діаграма авторизації в системі

Під час виконання авторизації (рис. 2.13) система просить користувача ввести ім'я та пароль облікового запису, в який користувач хоче виконати вхід. Після введення користувачем цих даних, система формує Dto-об'єкт,

який надсилає http-запитом на ендпоїнт контролера, що відповідає за авторизацію в системі. Контролер приймає цей запит та передає дані в сервіс, який надсилає запит до бази даних, щоб перевірити, чи є в ній користувач з такими даними. Після отримання відповіді від бази даних сервіс формує статус успішності входу та повертає його контролеру. Контролер перевіряє, чи вхід був успішним. Якщо так, то формує токен безпеки та кешує його, після чого повертає цей статус системі. Система перевіряє, чи вхід був успішним. Якщо так, то виводить користувачу повідомлення про успішний вхід, якщо ні, то повторює спробу входу.



Рисунок 2.14 – Діаграма реєстрації акаунта

Під час реєстрації користувачем акаунта (рис. 2.14) система просить ввести його наступні дані: ім'я акаунта, пароль та електронну пошту. Після введення користувач цих даних, система формує Dto-об'єкт, що містить ще додаткову інформацію у вигляді ідентифікатора Telegram користувача. Тоді вона надсилає цей об'єкт http-запитом на ендпоїнт контролера, що відповідає за реєстрацію акаунтів у системі. Контролер приймає цей запит та передає дані в сервіс, який надсилає запит до бази даних, щоб перевірити, чи є в ній

користувач з такими даними. Якщо в базі вже є такі дані, то сервіс повертає статус, що користувач вже зареєстрований. Якщо ж у базі немає даних про такого користувача, то ці дані зберігають у ній, і сервіс повертає статус успішної реєстрації акаунта. Система отримує даний статус та надсилає користувачу повідомлення, що відповідає йому.



Рисунок 2.15 – Діаграма видалення облікового запису

Під час видалення облікового запису (рис. 2.15) система повертає користувачу клавіатуру з усіма його записами, щоб він міг легко та зручно обрати той, який хоче видалити. Після обрання акаунта система просить користувача ввести пароль від нього, щоб впевнитись, що саме власник акаунта пробує його видалити. Після отримання всіх даних система формує Dto-об'єкт, що містить ще додаткову інформацію у вигляді ідентифікатора Telegram користувача. Тоді вона надсилає цей об'єкт http-запитом на ендпоінт контролера, що відповідає за видалення акаунта із системи. Контролер приймає цей запит та передає дані в сервіс, який надсилає запит до бази даних, щоб перевірити, чи є в ній користувач з такими даними. Якщо так,

сервіс видаляє обліковий запис, а система виводить користувачу повідомлення про успішне видалення. Якщо ні, то система просить користувача повторно ввести пароль та виконує всі дії, що йдуть після введення пароля.



Рисунок 2.16 – Діаграма додавання віртуальної машини

У процесі додавання віртуальної машини (рис. 2.16) система просить користувача ввести наступні дані: ім'я віртуальної машини, ім'я користувача на машині, пароль користувача, хост та порт. Після отримання всіх даних система формує Dto-об'єкт, що містить ще додаткову інформацію у вигляді ідентифікатора Telegram користувача та ідентифікатора акаунта. Тоді вона надсилає цей об'єкт http-запитом на ендпоінт контролера, що відповідає за додавання віртуальних машин. Контролер приймає цей запит та передає дані в сервіс, який надсилає запит до бази даних, щоб перевірити, чи є в ній віртуальна машина з такими даними. Якщо так, то сервіс передає повідомлення системі, що машина з такими даними вже створена, а система

45

виводить це повідомлення користувачу. Якщо ні, то сервіс зберігає дані про машину в базі, а система виводить користувачу повідомлення про успішне додавання віртуальної машини.

Після вибору необхідної віртуальної машини користувач може виконувати з нею наступні дії: виконати команду, додати/видалити директорію, завантажити/вивантажити файли, отримати метрики, оновити дані про віртуальну машину та видалити віртуальну машину.



Рисунок 2.17 – Діаграма віддаленого виконання команд на віртуальній машині

У процесі виконання команд на віртуальній машині (рис. 2.17) система просить користувача ввести команду, яку він хоче виконати. Після отримання команди система формує Dto-об'єкт, в який входить така інформація: команда, ідентифікатор користувача, ідентифікатор Telegram та інформація, яка потрібна для підключення до віртуальної машини. Система надсилає цей об'єкт на ендпоінт контролера, який відповідає за виконання команд на віртуальній машині. Контролер передає всі дані сервісу. Після отримання цього об'єкту SSH-сервіс перевіряє, чи вже є клієнт, який під'єднаний до цієї віртуальної машини. Якщо так, то просто обирає його, якщо ні, то створює новий, під'єднує його до віртуальної машини, додає в словник та обирає його

для роботи. Після обрання клієнта сервіс виконує дії аналогічні до тих, які виконував із клієнтом, але тепер вже з потоком консолі віртуальної машини. Коли сервіс готовий до передачі команди, він передає її на виконання віртуальній машині, яка повертає сервісу результат виконання у вигляді лога з консолі машини. Після отримання результату виконання команди SSH-сервіс передає цей результат системі, а система виводить його користувачу.



Рисунок 2.18 – Діаграма отримання метрик віртуальної машини

Під час отримання метрик віртуальної машини (рис. 2.18) система формує Dto-об'єкт з усією необхідною для підключення інформацією та надсилає його на ендпоінт контролера, який відповідає за отримання метрик. Контролер передає всі отриманні дані сервісу. Після отримання даних SSH-сервіс перевіряє, чи вже є клієнт, який під'єднаний до цієї віртуальної машини. Якщо так, то просто обирає його, якщо ні, то створює новий, під'єднує його до віртуальної машини, додає в словник та обирає його для роботи. Після обрання клієнта сервіс виконує дії аналогічні до тих, які виконував із клієнтом, але тепер вже з потоком консолі віртуальної машини. Коли сервіс готовий, він перевіряє, чи на віртуальній машині є клієнт для

збору метрик. Якщо ні, то він надсилає цей клієнт на віртуальну машину та збирає метрики. Якщо так, то клієнт зразу переходить до збирання метрик. Після збору метрик сервіс повертає дані системі, яка перетворює отримані числові дані на графіки та надсилає їх користувачу.



Рисунок 2.19 – Діаграма створення/видалення директорії на віртуальній машині

У процесі створення/видалення директорії на віртуальній машині (рис. 2.19) система просить користувача шлях до директорії. Після отримання шляху система формує Dto-об'єкт з усією необхідною інформацією для видалення/створення директорії та надсилає його на ендпоінт контролера, який відповідає за цю дію. Контролер передає всі отриманні дані сервісу. Після отримання даних SFTP-сервіс перевіряє, чи вже є клієнт, який під'єднаний до цієї віртуальної машини: якщо так, то просто обирає його, якщо ні, то створює новий, під'єднує його до віртуальної машини, додає в словник та обирає його для роботи. Коли сервіс готовий, то він виконує операцію видалення/створення директорії. У залежності від успіху операції сервіс повертає системі повідомлення про успішність її виконання, система виводить це повідомлення користувачу.



Рисунок 2.20 – Діаграма завантаження файлу з віртуальної машини

Під час завантаження файлу з віртуальної машини (рис. 2.20) система просить користувача шлях до файлу. Після отримання шляху вона формує Dto-об'єкт з усією необхідною інформацією для завантаження файлу та надсилає його на ендпоінт контролера, який відповідає за цю дію. Контролер передає всі отриманні дані сервісу. Після отримання даних SFTP-сервіс перевіряє, чи вже є клієнт, який під'єднаний до цієї віртуальної машини: якщо так, то просто обирає його, якщо ні, то створює новий, під'єднує його до віртуальної машини, додає в словник та обирає його для роботи. Коли сервіс готовий, він пробує завантажити файл. Якщо ця дія є успішною, то зберігає файл локально та формує Dto-об'єкт зі шляхом до нього, повідомленням користувачу та маркером, що файл завантажено, після чого передає цей об'єкт системі. Система надсилає користувачу файл із повідомленням, що сформував сервіс та видаляє його локально. Якщо дія не є успішною, то сервіс формує Dto-об'єкт із маркером, що файл не завантажено, повідомленням про помилку, після чого система виводить користувачу це повідомлення.

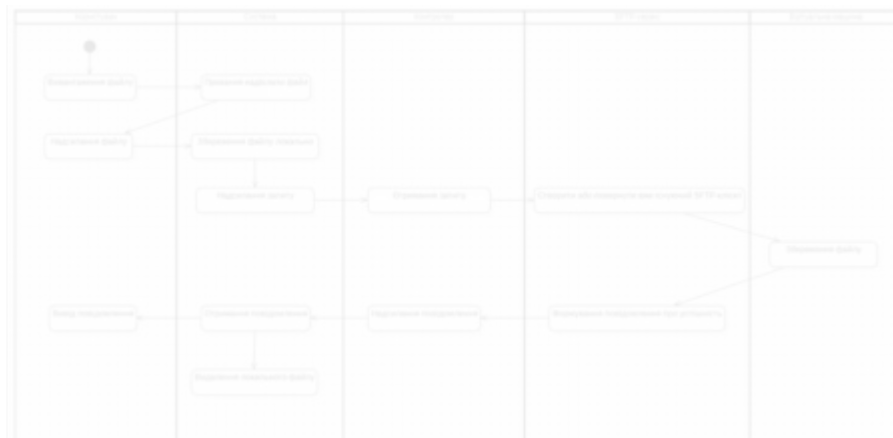


Рисунок 2.21 – Діаграма вивантаження файлу на віртуальну машину

У процесі вивантаження файлу на віртуальну машину (рис. 2.21) система просить користувача надіслати файл, який він хоче вивантажити. Після отримання файлу система зберігає його локально та формує Dto-об'єкт з усією необхідною інформацією для вивантаження та надсилає його на ендпоїнт контролера, який відповідає за цю дію. Контролер передає всі отриманні дані сервісу. Після отримання даних SFTP-сервіс перевіряє, чи вже є клієнт, який під'єднаний до цієї віртуальної машини: якщо так, то просто обирає його, якщо ні, то створює новий, під'єднує його до віртуальної машини, додає в словник та обирає його для роботи. Коли сервіс готовий, він пробує вивантажити файл, після чого повертає системі повідомлення про успішність виконання цієї дії, система виводить це повідомлення користувачу та видаляє локальний файл.



Рисунок 2.22 – Діаграма оновлення даних віртуальної машини

Під час оновлення даних віртуальної машини (рис. 2.22) система запитує ті ж дані, що і при її додаванні. Після отримання всіх даних система формує Dto-об'єкт, що містить ще додаткову інформацію у вигляді ідентифікатора Telegram користувача та ідентифікатора акаунта. Тоді вона надсилає цей об'єкт http-запитом на ендпоінт контролера, що відповідає за оновлення даних віртуальної машини. Контролер приймає цей запит та передає дані в сервіс, який надсилає запит до бази даних, щоб перевірити, чи є в ній віртуальна машина з такими даними. Якщо так, то сервіс оновлює дані віртуальної машин, а система виводить користувачу повідомлення про успішне оновлення даних. Якщо ні, то сервіс повертає системі повідомлення про помилку, а система виводить це повідомлення користувачу.



Рисунок 2.23 – Діаграма видалення віртуальної машини

У процесі видалення віртуальної машини (рис. 2.23) система запитує, чи користувач впевнений, що хоче видалити машину. Після отримання ствердної відповіді система формує Dto-об'єкт, що містить ще додаткову інформацію у вигляді ідентифікатора Telegram користувача та ідентифікатора акаунта. Тоді вона надсилає цей об'єкт http-запитом на ендпоінт контролера, що відповідає за оновлення даних віртуальної машини. Контролер приймає цей запит та передає дані в сервіс, який надсилає запит до бази даних, щоб перевірити, чи є в ній віртуальна машина з такими даними. Якщо так, то сервіс видаляє дані віртуальної машин, а система виводить користувачу повідомлення про успішне видалення віртуальної машини. Якщо ні, то сервіс повертає системі повідомлення про помилку, а система виводить це повідомлення користувачу.



Рисунок 2.24 – Діаграма генерування та збереження токена

Як було зазначено вище, система при успішному вході створює токен безпеки, який в подальшому використовується для виконання HTTP-запитів на захищені Web API (рис. 2.24). У токені зберігаються унікальну для проєкту дані про видавця та аудиторію токена, а також про ключ підпису видавця. Ці дані допоможуть ідентифікувати токен, який був виданий саме нашим проєкту, а не будь-яким іншим. Після створення цей токен кешується в базі даних за унікальним ключем. Будь-який користувач, що володіє даним токеном зможе виконувати запити на захищені Web API проєкту.



Рисунок 2.25 – Виконання запитів на Web API

Коли користувач пробує виконати будь-яку дію в системі, то вона перевіряє, чи ця дія потребує авторизації в системі (рис. 2.25). Якщо ні, то система виконує цю дію та повертає користувачу відповідне повідомлення. Якщо так, то система надсилає запит на контролер по отриманню токена за його унікальним ключем. Система передає цей ключ сервісу, який отримує з бази закешовані дані, які зберігалися за цим ключем, потім розкешовує їх та повертає контролеру. Контролер передає ці дані системі, і система перевіряє, чи є ці дані токеном безпеки. Якщо так, то система виконує дію та виводить користувачу відповідне повідомлення. Якщо ні, то система просить

користувача авторизуватися в системі, щоб токен було згенеровано та закешовано.

2.4. Реалізація додатку системи віддаленої взаємодії з віртуальними машинами

Для з'єднання з базою даних, налаштуванням параметрів з'єднання та реалізації системи було використано ORM-бібліотеку EntityFrameworkCore. EntityFrameworkCore – це платформа об'єктно-реляційного відображення (ORM) з відкритим кодом для ADO.NET. Спочатку він постачався як невід'ємна частина .NET Framework, однак, починаючи з EntityFramework версії 6.0, він постачався окремо від .NET Framework. Новий фреймворк, відомий як EntityFramework Core, був представлений у 2016 році зі схожими, але не повними функціями. Нумерація версій цього фреймворку перезапущена з 1.0, а остання версія EF Core – 8.0. Основні переваги EntityFrameworkCore: крос-платформність, підтримка різних баз даних, продуктивність, моделювання за допомогою коду (Code First), міграції, підтримка LINQ, відстеження змін (Change Tracking), гнучкість конфігурації, розширюваність та підтримка спадкування (Inheritance). Для з'єднання додатку з базою даних використовуємо технологію ORM.

Для досягнення SSH- та SFTP-з'єднання було використано бібліотеку SSH.NET. SSH.NET – бібліотека для віддаленої взаємодії через SSH підключення. На створення цього проекту надихнула бібліотека Sharp.SSH, яка була перенесена з Java і не підтримувалася протягом тривалого часу. Ця бібліотека є повністю переписаною, без будь-яких залежностей від третіх сторін, з використанням паралелізму для досягнення найкращої продуктивності. Основні переваги SSH.NET: простота використання, відкритий код, підтримка різних версій SSH, широкий функціонал, безпека, крос-платформність, підтримка асинхронних операцій, хороша документація, мінімальні залежності.

Для побудови серверної частини було використано ASP.NET Core. ASP.NET Core – вільне та відкрите програмне забезпечення каркаса

вебзастосунків, з продуктивністю вищою ніж у ASP.NET, розробленою корпорацією Microsoft і співтовариством. Це модульна структура, яка працює як на повній платформі .NET Framework, так і на платформі .NET Core.

ASP.NET Core є крос-платформним фреймворком, що працює на Windows, macOS і Linux, дозволяючи розробникам використовувати будь-яку операційну систему для розробки та розгортання додатків. Він оптимізований для високої продуктивності та масштабованості, демонструючи одну з найвищих швидкостей серед веб-фреймворків. Завдяки модульній архітектурі розробники можуть включати лише необхідні компоненти, що зменшує обсяг додатків та покращує продуктивність. ASP.NET Core повністю підтримує створення мікросервісів та їх розгортання в контейнерах, таких як Docker, що полегшує створення розподілених систем. Єдина платформа дозволяє створювати різні типи додатків, включаючи вебдодатки, API та серверні додатки для хмарних сервісів. Підтримка асинхронного програмування за допомогою `async/await` підвищує продуктивність при обробці великої кількості запитів. Вбудована система залежностей (Dependency Injection) спрощує керування залежностями та покращує тестованість коду. ASP.NET Core також забезпечує високий рівень безпеки, надаючи засоби для аутентифікації та авторизації, захисту від міжсайтових атак (XSS) та інших загроз. Регулярні оновлення та активна підтримка від Microsoft і спільноти гарантують своєчасне впровадження нових функцій і виправлення помилок.

Використання бібліотеки Telegram.Bot для інтеграції проєкту з Telegram Bot API виявилось надзвичайно корисним завдяки кільком ключовим аспектам. Дана бібліотека значно спрощує процес взаємодії з Telegram Bot API. Замість того, щоб самостійно формувати та відправляти HTTP-запити, розробники можуть використовувати зручний і зрозумілий інтерфейс, який пропонує бібліотека. Це дозволяє зосередитися на бізнес-логіці бота, не витрачаючи час на деталі роботи з протоколом HTTP.

Також варто зазначити, що Telegram.Bot надає широкий спектр функціональних можливостей для створення ботів будь-якої складності. Він підтримує всі основні методи Telegram Bot API, включаючи відправку повідомлень, обробку команд, керування мультимедійними файлами та багато іншого. Завдяки цьому розробники можуть створювати багатофункціональні боти, які відповідають усім вимогам користувачів.

Отже, використання бібліотеки Telegram.Bot для інтеграції з Telegram Bot API не лише спрощує процес розробки, але й забезпечує високу гнучкість та функціональність створених ботів.

Висновки до розділу 2

Система віддаленої взаємодії з віртуальними машинами реалізовує функціонал ресурсу, який надасть можливість повної віддаленої взаємодії з машинами, а також дозволить користувачам отримувати інформацію про стан машини у вигляді графіків. Також дана система реалізовує менеджмент акаунтів користувача та прив'язаних до них віртуальних машин.

У розділі реалізована база даних у відповідності до вимог реляційної моделі забезпечує збереження та колективний доступ до інформації системи. База даних складається з 2 колекцій: Users та VirtualMachines.

Було розглянуто алгоритми основних процесів ІС та проаналізовано взаємодію її класів у процесі виконання основних дій. Було побудовано графіки взаємодії та детально описано кожен аспект функціоналу системи.

РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З СИСТЕМОЮ ВІДДАЛЕНОЇ ВЗАЄМОДІЇ З ВІРТУАЛЬНОЮ МАШИНОЮ

3.1. Порядок встановлення та налаштування системи

Система віддаленої взаємодії з віртуальними машинами – це застосунок, створений за допомогою мов програмування C# та Python. Він має конфігураційні файли, в яких прописуються такі дані, як рядки підключення до баз даних, конфігурація логування та інші необхідні конфігураційні дані. Перед перенесенням системи на хостинг необхідно змінити дані в конфігураційних файлах. Система використовує Microsoft SQL Server та Redis бази даних. Переносячи Microsoft SQL Server, необхідно скопіювати і його вміст. Після цього треба оновити конфігураційні файли. Порядок дій в такому випадку наступний:

1. Створення копії (бекап) бази даних на старому сервері. Якщо поточний провайдер надає інструмент для створення Microsoft SQL Server копій, тоді можна скористатись ними. Також це можна зробити вручну: достатньо ввести в командну консоль команду `sqlcmd -S [server_name] -Q "BACKUP DATABASE [database_name] TO DISK='C:\Backup\backup_file.bak' WITH INIT"`, а для відновлення бази із бекапу потрібно ввести команду `sqlcmd -S [server_name] -Q "RESTORE DATABASE [database_name] FROM DISK='C:\Backup\backup_file.bak' WITH RECOVERY`.

2. Створення бази даних на сервері. Як тільки скачування закінчиться, визначити назву бази. Перед відновленням потрібно створити базу на сервері та користувача бази.

Рисунок 3.1 – Діаграма розгортання системи

3. Імпорт даних бази на сервер. Після створення порожньої бази даних, на сервері слід ввести команду `sqlcmd -S [server_name] -Q "RESTORE DATABASE [database_name] FROM DISK='C:\Backup\backup_file.bak' WITH RECOVERY`. Почнеться процес відновлення бази даних. Він може зайняти деякий час, якщо розмір бази занадто великий. Тепер, коли база даних відновлена, потрібно відкрити конфігураційний файл системи та змінити значення хоста бази даних, ім'я бази даних, ім'я користувача бази даних і пароль бази даних.

Параметри налаштування бази даних зберігаються у файлі appsettings.Development.json:

```
"ConnectionStrings": {
  "MSSQL": "Server=127.0.0.1,5434;User=
Id=SA;Password=Strong2@PWD12;Database=VMControlPanel;TrustServerCertificate=True
;",
  "Redis": "localhost:6379"
}
```

Параметри моделей зберігаються у відповідних класах.

Параметри моделі User зберігаються в UserConfiguration.cs:

```
using Core.Entities;
using Microsoft.EntityFrameworkCore;
using Microsoft.EntityFrameworkCore.Metadata.Builders;
using Microsoft.EntityFrameworkCore.Storage.ValueConversion;

namespace UserInfrastructure.Configurations
{
    public class UserConfiguration : IEntityTypeConfiguration<User>
    {
        public void Configure(EntityTypeBuilder<User> builder)
        {
            builder.HasKey(_ => _.Id);

            builder.Property(_ => _.Id)
                .IsRequired();

            builder.Property(_ => _.TelegramId)
                .IsRequired();

            builder.Property(_ => _.UserName)
                .IsRequired();

            builder.Property(_ => _.PasswordHash)
                .IsRequired();

            builder.Property(_ => _.Email)
                .IsRequired();

            builder.Property(_ => _.NormalizedUserName)
                .IsRequired();

            builder.Property(_ => _.NormalizedEmail)
                .IsRequired();

            builder.Property(_ => _.Culture)
                .HasConversion(new EnumToStringConverter<Culture>())
                .IsRequired();
        }
    }
}
```

61

```
}  
}  
}
```

Параметри моделі VirtualMachine зберігаються в

VirtualMachineConfiguration.cs:

```
using Core.Entities;  
using Microsoft.EntityFrameworkCore;  
using Microsoft.EntityFrameworkCore.Metadata.Builders;  
  
namespace Infrastructure.Configurations  
{  
    public class VirtualMachineConfiguration :  
        IEntityTypeConfiguration<VirtualMachine>  
    {  
        public void Configure(EntityTypeBuilder<VirtualMachine> builder)  
        {  
            builder.Property(_ => _.Id)  
                .IsRequired()  
                .UseIdentityColumn();  
  
            builder.Property(_ => _.Name)  
                .IsRequired();  
  
            builder.Property(_ => _.UserId)  
                .IsRequired();  
  
            builder.Property(_ => _.UserName)  
                .IsRequired();  
  
            builder.Property(_ => _.PasswordEncrypted)  
                .IsRequired();  
  
            builder.Property(_ => _.Host)  
                .IsRequired();  
  
            builder.Property(_ => _.Port)  
                .IsRequired();  
        }  
    }  
}
```

62

```
        builder.Ignore(_ => _.Password);  
    }  
}  
}
```

З метою забезпечення можливості дистанційної роботи і незалежності від операційної системи користувача, даний програмний комплекс потребує наявності сервера і девайса користувача. При необхідності комп'ютер користувача може виконувати і роль сервера.

Розроблена програма буде використовуватися на будь-яких девайсах сумісних із Telegram.

Для десктоп версії Telegram під керування Windows необхідні наступні системні параметри:

- Процесор з тактовою частотою не менше 800 MHz
- Не менше 128 Mb операційної пам'яті
- Вільного місця на диску 49 Mb
- Архітектура системи: x64
- Версія Windows: не нижче Windows 7

Для десктоп версії Telegram під керування macOS необхідні наступні системні параметри:

- macOS 10.13 або новіша

Для десктоп версії Telegram під керування Linux-системи необхідні наступні системні параметри:

- Будь-який дистрибутив Linux
- Опціонально: Snapd або Flatpak

Для Android версії Telegram необхідний смартфон під керуванням Android версії 6.0 або новіше. Для iOS версії Telegram необхідний смартфон під керуванням iOS версії 11.0 або новіше. Для Telegram Web версії необхідний будь-який сумісний браузер, наприклад Google Chrome версії 49 або новіше.

3.2. Структура інтерфейсу. Інтерфейс та порядок роботи з системою

Коли користувач починає роботу із ботом, його зустрічає стандартний для всіх Telegram ботів інтерфейс (рис. 3.2). Тут присутні наступні елементи: ім'я бота (рис. 3.2.1), інформація про бота (рис. 3.2.2), яка містить ім'я бота, фото профіля, ім'я користувача, можливість вимкнути сповіщення, можливість додати до групи та можливості поскаржитися на бота або зупинити його роботу, кнопка «РОЗПОЧАТИ» (рис. 3.2.3), яка розпочне роботу із ботом, надіславши йому команду `/start`, панель із додатковим функціоналом (рис. 3.2.4) та поле, в якому будуть відображатися повідомлення (рис. 3.2.5).

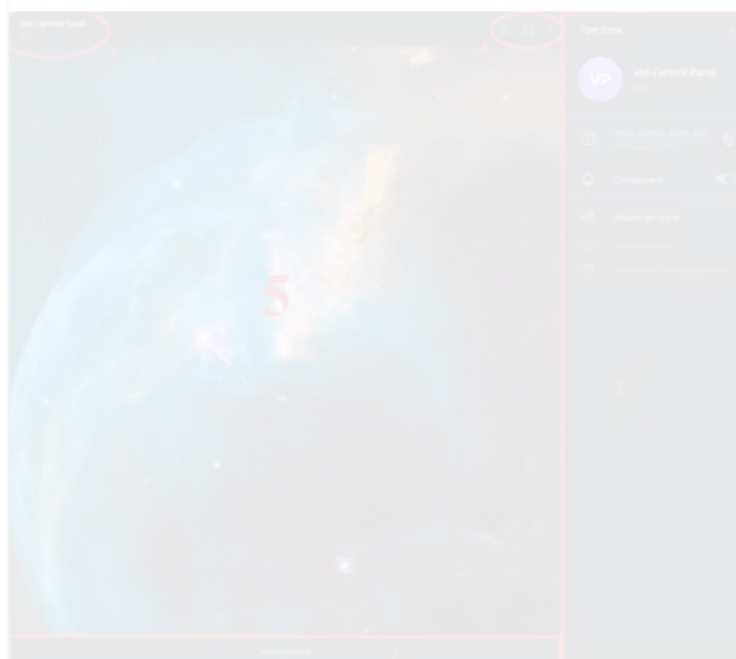


Рисунок 3.2 – Початковий інтерфейс бота

Панель додаткового функціоналу має наступний вигляд (рис. 3.3). Першим функціоналом, який доступний користувачу, є пошук повідомлень в чаті за ключовими словами. Для цього необхідно натиснути на іконку, схожу на лупу, після чого ввести ключові слова. Наступна іконка, схожа на вікно розділене вертикальною лінією, прикріплює/відкріплює інформацію про бота (рис. 3.2.2). При натисканні на іконку, яка має вигляд трьох вертикальних крапок, користувачеві відкривається вікно із наступним функціоналом: налаштування сповіщень, перегляд профілю бота, очищення історії повідомлень та видалення чату із ботом.

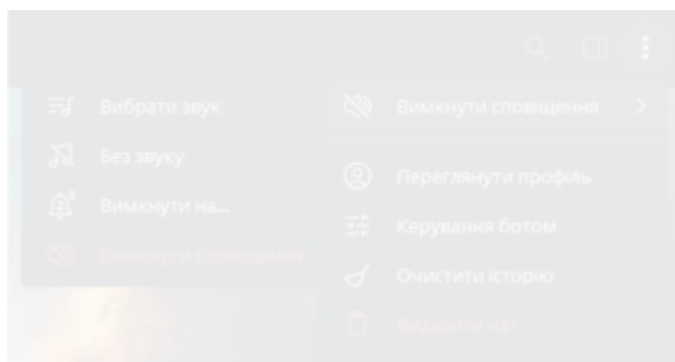


Рисунок 3.3 – Панель додаткового функціоналу

При натисканні користувачем кнопки «РОЗПОЧАТИ», система виконує запит до бази даних, щоб перевірити, чи в користувача вже є зареєстровані облікові записи. Якщо вони в користувача є, то система виводить привітальне повідомлення та список імен акаунтів, щоб користувачу було легше зорієнтуватися (рис. 3.4). Якщо в користувача немає зареєстрованих акаунтів, то система виводить привітальне повідомлення та повідомлення про те, що в нього немає зареєстрованих облікових записів (рис. 3.5).

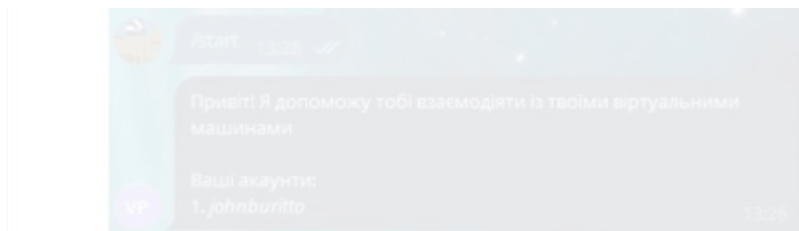


Рисунок 3.4 – Вивід, якщо в користувача є зареєстровані акаунти

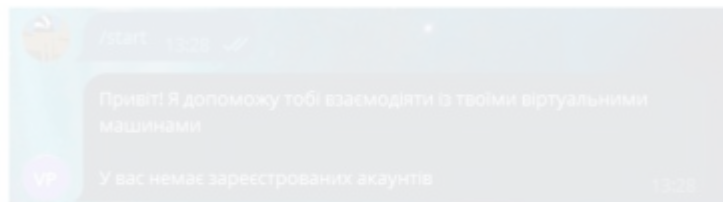


Рисунок 3.5 – Вивід, якщо в користувача немає зареєстрованих акаунтів

Після привітального повідомлення від системи у користувача є наступні опції для взаємодії з віртуальною системою (рис. 3.6).



Рисунок 3.6 – Опції взаємодії

Після натискання користувачем опції «Створити акаунт», система попросить придумати та ввести наступні дані: ім'я користувача, пароль та електронну адресу (рис. 3.7). При успішній реєстрації користувачеві виведеться повідомлення про це. Якщо під час процесу реєстрації станеться помилка, то система повідомить про це. Також на початку реєстрації система переведе користувача в стан реєстрації, це означає, що система не буде реагувати на інші команди, доки процес реєстрації не закінчиться, а всі дані, що користувач ввів, будуть розцінені системою як ті, що були запитані у нього. На кожному кроці реєстрації в користувача буде можливість відмінити

66

процес, натиснувши на кнопку «✕ Відмінити» (рис. 3.8). Користувачу буде виведено повідомлення про відміну дії.

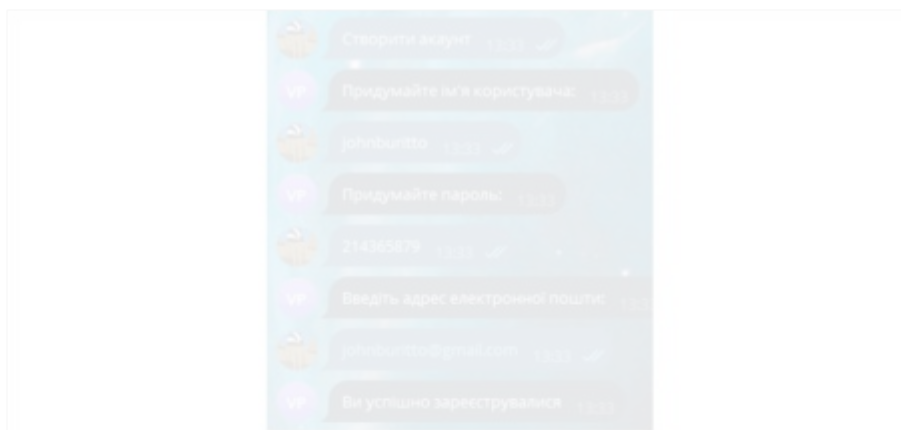


Рисунок 3.7 – Реєстрація в системі



Рисунок 3.8 – Кнопка відміни

Після натискання команди «Видалити акаунт», система виводить користувачу клавіатуру з усіма обліковими записами (рис. 3.9), які є у нього, щоб він обрав той, який хоче видалити. Після вибору акаунта система просить користувача ввести пароль від облікового запису, щоб перевірити, чи саме власник хоче видалити акаунт, а не хтось інший. Після введення паролю система виконує спробу входу, щоб перевірити, чи дані вірні: якщо так, то акаунт видаляється (рис. 3.10), якщо ні, то система просить знову ввести пароль (рис. 3.10). Користувач може відмінити дію, натиснувши на кнопку «✕ Відмінити» (рис. 3.8).

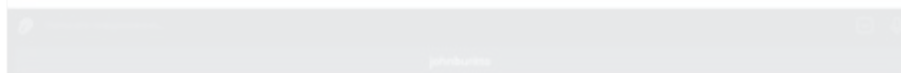


Рисунок 3.9 – Акаунти користувача



Рисунок 3.10 – Видалення акаунта

Після натискання користувачем опції «Увійти в акаунт», система попросить користувача ввести наступні дані для ідентифікації та авторизації в системі: ім'я користувача та пароль (рис. 3.11). При успішному вході в систему користувачеві виведеться повідомлення про це, також система генерує токен доступу, який буде використаний для доступу до захищених API. Якщо користувач введе невірні дані, то система почне заново процес запиту даних для входу. Кількість спроб входу є необмеженою. Як і у випадку із реєстрацією, на початку процесу входу в акаунт система переводить користувача в стан входу, користувач може відмінити вхід, натиснувши кнопку «✕ Відмінити» (рис. 3.8). Користувачу буде виведено повідомлення про відміну дії (рис. 3.12).

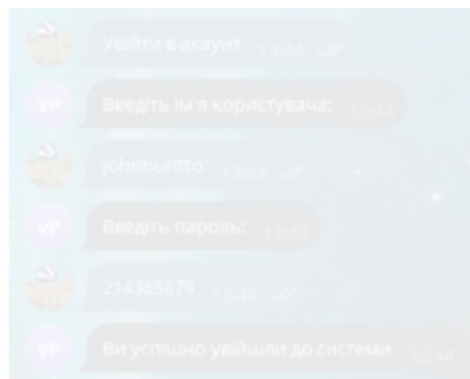


Рисунок 3.11 – Вхід в систему

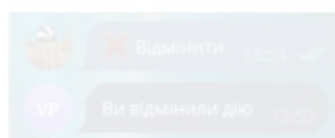


Рисунок 3.12 – Повідомлення про відміну дії

Після входу в систему користувачу буде виведено список всіх віртуальних машин, прив'язаних до його акаунта, та опцію «+ Додати нову машину» (рис. 3.13). Якщо ж до акаунта користувача не прив'язана жодна машина, то в нього буде доступна тільки опція додавання віртуальної машини (рис. 3.14).



Рисунок 3.13 – Список віртуальних машин користувача

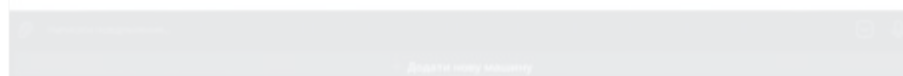


Рисунок 3.14 – Опція додавання віртуальної машини

При виборі користувачем опції «✚ Додати нову машину» система переводить його в стан додавання віртуальної машини та запитує наступні дані: ім'я віртуальної машини, ім'я користувача на віртуальній машині, пароль користувача на віртуальній машині, хост та порт хоста. При успішному додаванні машини система виводить повідомлення про це (рис. 3.15) та переводить користувача до вибору віртуальної машини для взаємодії (рис. 3.13). Додання машини можна відмінити, натиснувши кнопку «✕ Відмінити» (рис. 3.8).

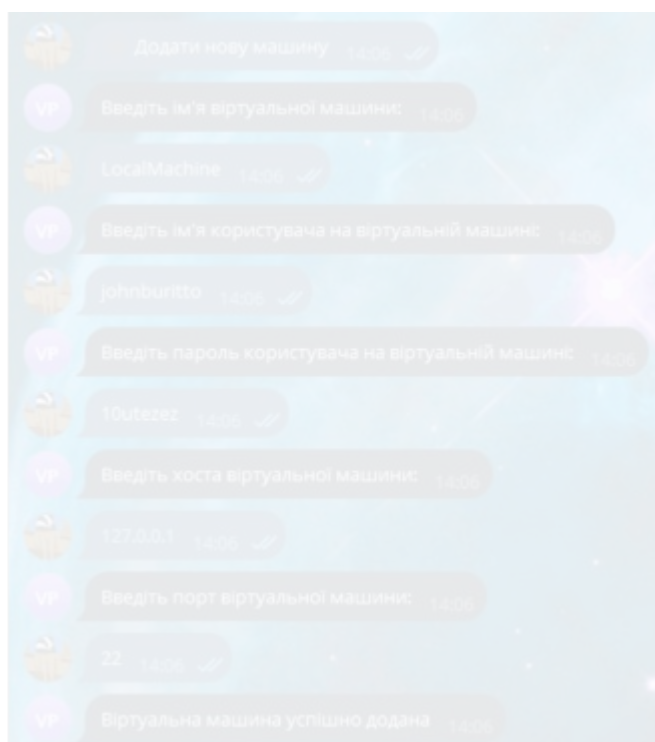


Рисунок 3.15 – Додання віртуальної машини

Після вибору користувачем віртуальної машини для взаємодії (рис. 3.16), йому відображаються всі опції, які він може виконати із нею та акаунтом (рис. 3.17).

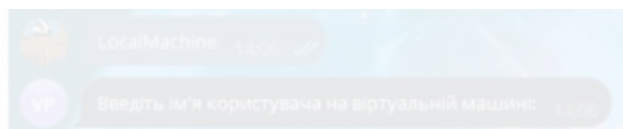


Рисунок 3.16 – Вибір віртуальної машини для взаємодії



Рисунок 3.17 – Опції взаємодії користувача із віртуальною машиною та акаунтом

При виборі користувачем опції «Виконати команду», система переводить користувача в стан виконання команд, де будь-яке повідомлення користувача буде розцінене як команда та буде передане віртуальній машині на виконання (рис. 3.18). Команда буде виконана за вище згаданим алгоритмом, а користувачу буде повернута відповідь віртуальної машини (рис. 2.17). Користувач може в будь-який момент відмінити виконання команд, натиснувши кнопку «✕ Відмінити» (рис. 3.8). При цьому система збереже свій стан на тому етапі, коли користувач натиснув цю кнопку.



```
Виконувати команди 14:07 ✓
Введіть команду: 14:07
neofetch 14:07 ✓

johnburitto@DESKTOP-148441K
-----
OS: Ubuntu 22.04.3 LTS on windows 10 x86_64
Kernel: 5.15.146.1-microsoft-standard-WSL2
Uptime: 1 hour, 16 mins
Packages: 812 (dpkg), 6 (snap)
Shell: bash 5.1.16
Terminal: /dev/pts/2
CPU: Intel i7-10750H (12) @ 2.592GHz
GPU: 5320-00-00-0 Microsoft Corporation Device 008e
Memory: 2418MiB / 15867MiB

johnburitto@DESKTOP-148441K ~ johnburitto@DESKTOP-148441K ~$
```

Рисунок 3.18 – Виконання команди

При виборі користувачем команди «Створити директорію», система просить користувача ввести ім'я директорії, яку необхідно створити. Після отримання імені система виконує SFTP-підключення до віртуальної машини та створює директорію (рис. 3.19).

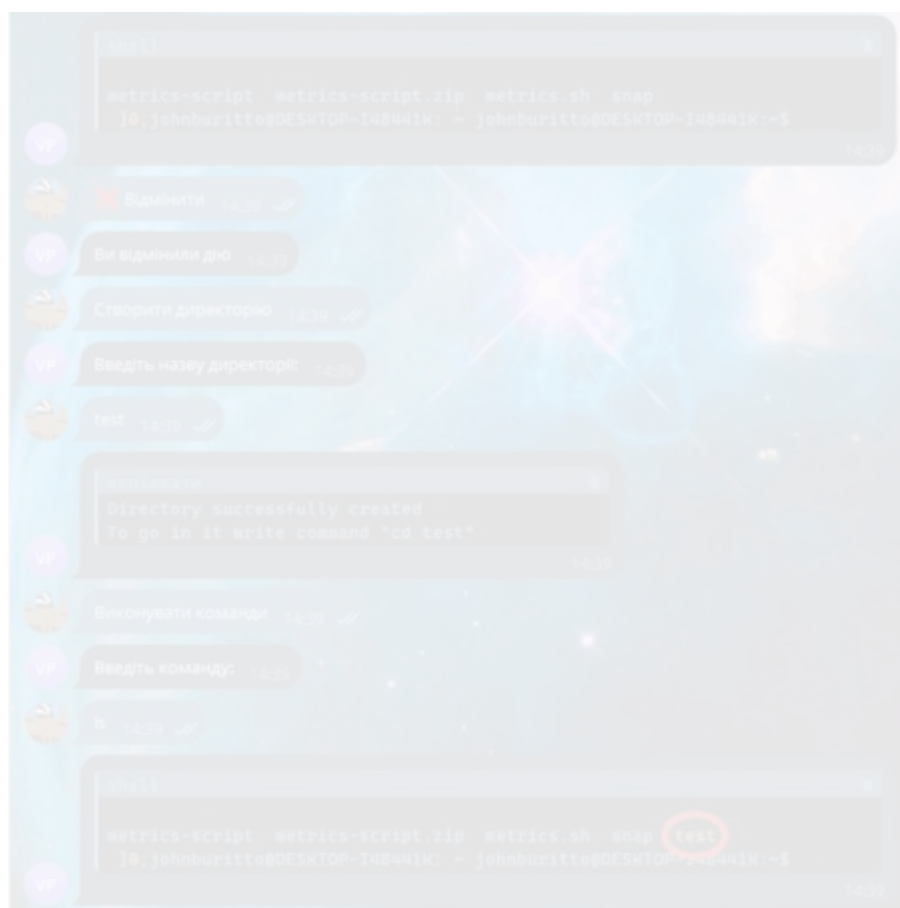


Рисунок 3.19 – Створення директорії

При виборі користувачем команди «Видалити директорію», система просить користувача ввести ім'я директорії, яку необхідно видалити. Після

отримання імені система виконує SFTP-підключення до віртуальної машини та видаляє директорію (рис. 3.20).

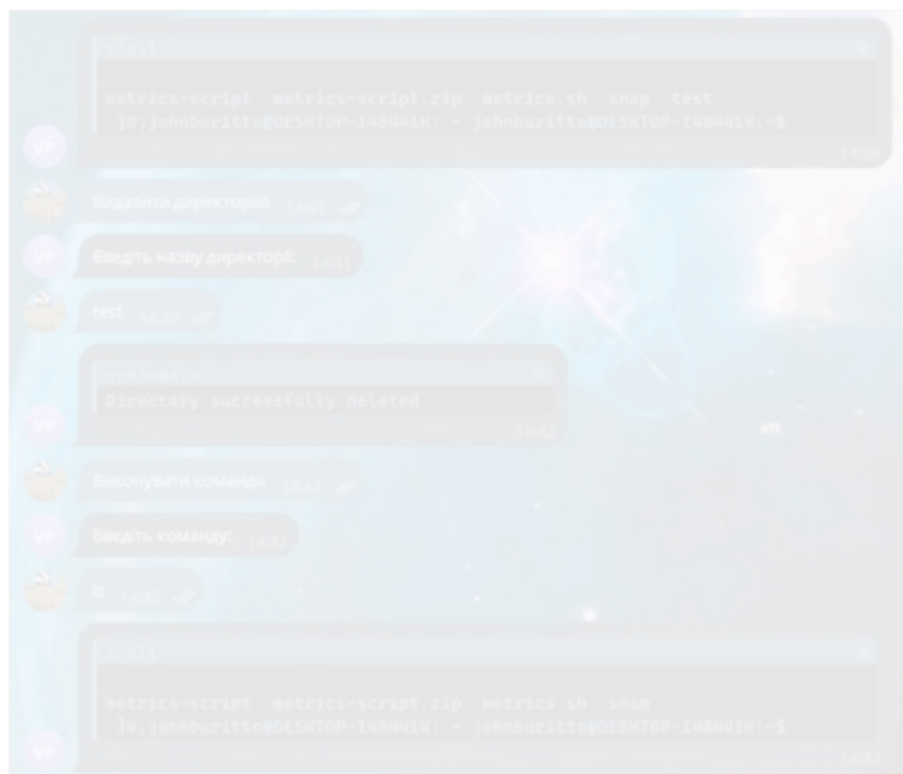


Рисунок 3.20 – Видалення директорії

При виборі користувачем опції «Завантажити файл», система просить вказати шлях до файлу на віртуальній машині, який необхідно завантажити. Після отримання шляху система виконує SFTP-підключення до машини, завантажує файл у своє локальне сховище, надсилає його користувачу та видаляє із локального сховища (рис. 3.21). Якщо під час завантаження файлу сталася помилка, то система виводить повідомлення про це (рис. 3.22).

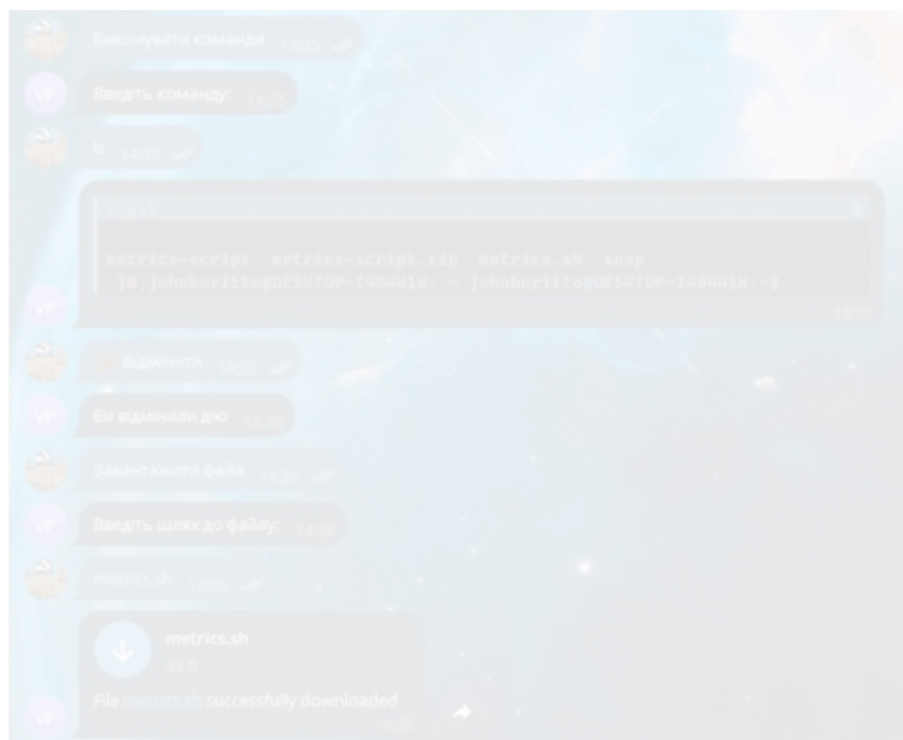


Рисунок 3.21 – Завантаження файлу

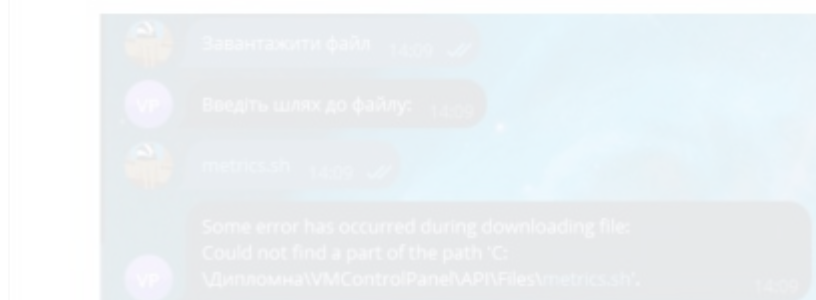


Рисунок 3.22 – Помилка при завантаженні файлу

При виборі користувачем опції «Вивантажити файл», система просить надіслати боту файл, який необхідно вивантажити на віртуальну машину. Після отримання файлу система завантажує його у своє локальне сховище,

виконує SFTP-підключення до машини та вивантажує файл на машину, після чого видаляє цей файл із локального сховища (рис. 3.23).

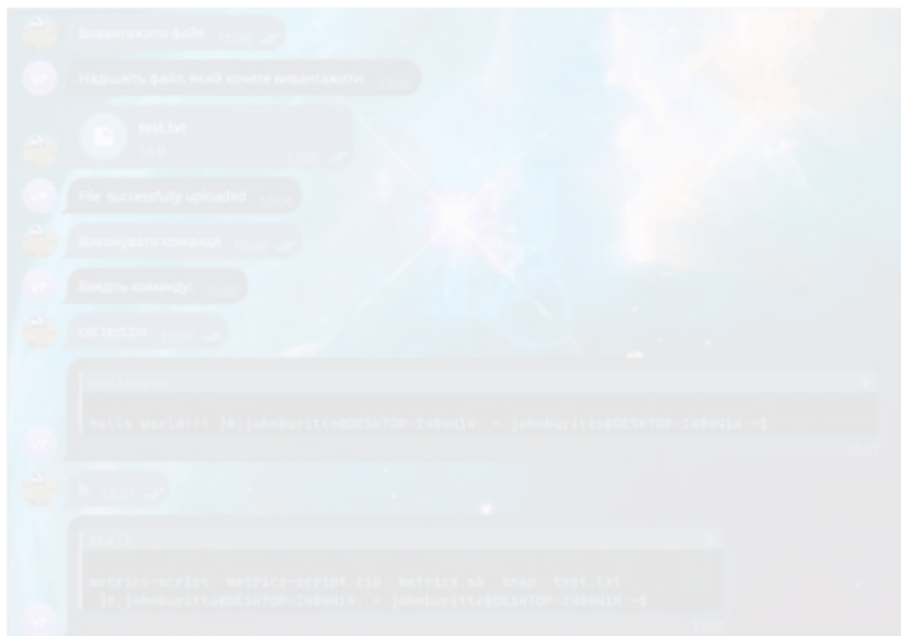


Рисунок 3.23 – Вивантаження файлу на віртуальну машину

При виборі користувачем опції «Метрики», система перевіряє, чи на віртуальній машині є клієнт, який буде збирати метрики. Якщо він присутній, то система виконає скрипт по збиранню метрик та отримає їх у вигляді об'єкта від машини. Якщо ж клієнта не буде, то система завантажить його на віртуальну машину разом із скриптом, який буде виконувати збір метрик, після чого збере та отримає їх у вигляді об'єкта. Сам же клієнт збору метрик є Python-скриптом, який зчитує всю необхідну інформацію із системних файлів, наприклад, інформацію про навантаження процесора із файлу */proc/stat*. Клієнт збирає метрики про процесор, диски та їх навантаженість, пам'ять, мережеві інтерфейси та кількість отриманих/надісланих байт. Після

отримання метрик система формує графіки на основі даних із об'єкта метрик (рис. 3.24): графік навантаження процесора (рис. 3.25), графік навантаження дисків, графік пам'яті, графік кількості читань/записів на носій пам'яті та кількість надісланих/отриманих бітів пам'яті для кожного з мережевих інтерфейсів.

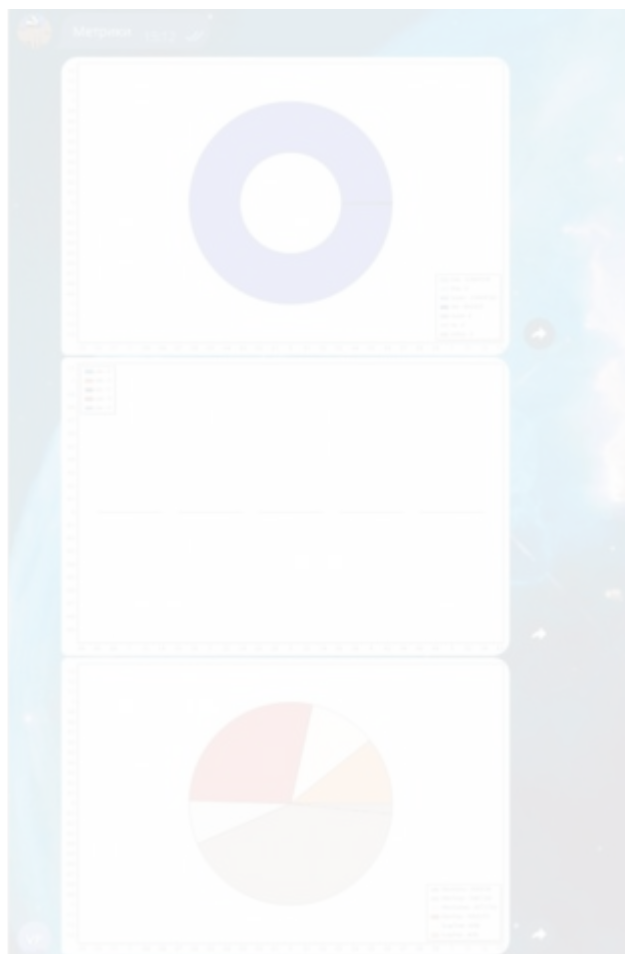


Рисунок 3.24 – Метрики

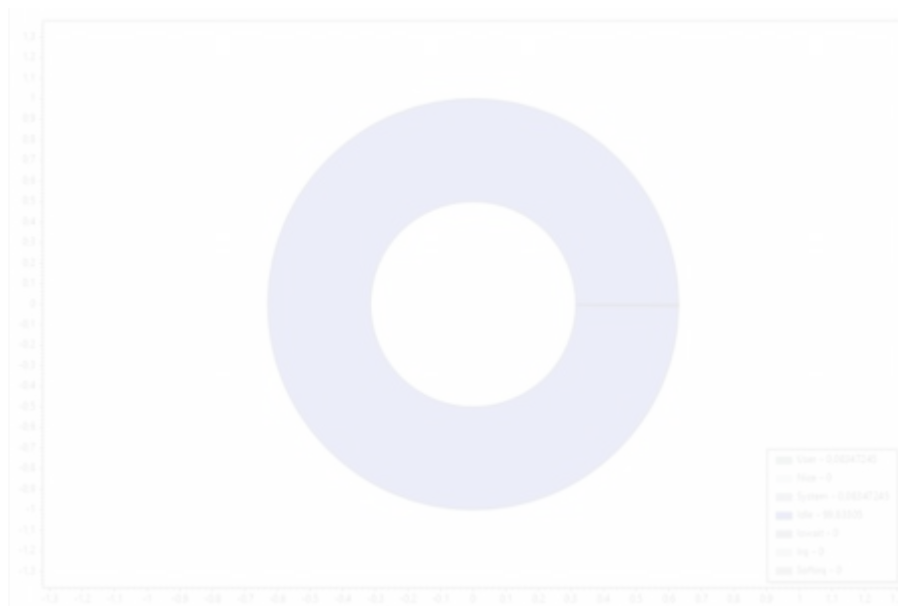


Рисунок 3.25 – Графік завантаженості процесора

При виборі користувачем опції «Оновити дані про віртуальну машину» (рис. 3.26), система запитує у користувача такі ж дані, як і при додаванні віртуальної машини (рис. 3.15). Після отримання інформації система зберігає її в базі даних та виводить користувачу повідомлення про успішне оновлення. Також система оновлює сесійну інформацію про віртуальну машину, з якою в даний момент взаємодіє користувач. Він може відмінити дію, натиснувши на кнопку «✕ Відмінити» (рис. 3.8).

Рисунок 3.26 – Оновлення віртуальної машини

При виборі користувачем опції «Видалити віртуальну машину» (рис. 3.27), система запитає, чи користувач впевнений, що хоче видалити машину (рис. 3.28). Якщо відповідь буде ствердною, то система видалить дані про машину, з якою відбувається взаємодія, з бази даних та з сесії, після чого користувача поверне до вибору віртуальної машини для взаємодії (рис. 3.13). Якщо ж ні, то користувача поверне до дій, які він може виконувати із віртуальною машиною (рис. 3.17).

Рисунок 3.27 – Запитання, чи користувач впевнений

Рисунок 3.28 – Видалення віртуальної машини

При виборі користувачем опції «Змінити мову», система просить обрати необхідну мову зі списку, що підтримує система (рис. 3.29). Після вибору необхідної мови інтерфейс системи перекладається на обрану мову, а користувачу надсилається повідомлення про успішну зміну мови (рис. 3.30). Також в базі даних у користувача оновлюється мова інтерфейсу на обрану.

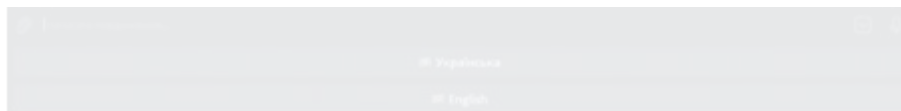


Рисунок 3.29 – Мови, що підтримує система

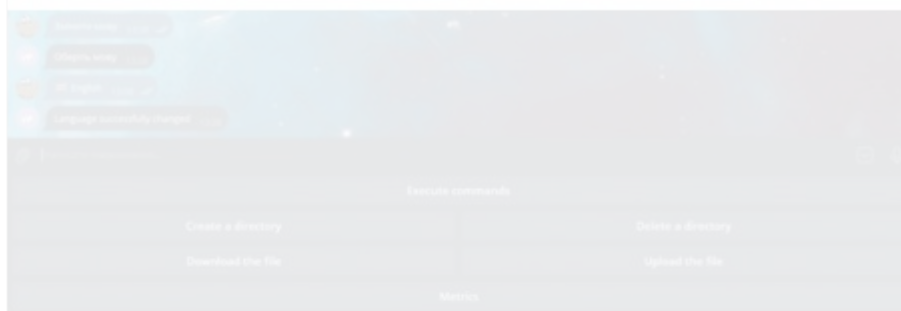


Рисунок 3.30 – Зміна мови


При виборі користувачем опції « Вийти із акаунта», система видаляє всі закешовані дані про сесію користувача, виводить повідомлення про вихід із системи (рис. 3.31) та повертає до початкових опцій взаємодії (рис. 3.6).



Рисунок 3.31 – Вихід із системи

Якщо користувач пробує виконати команду, для якої необхідно бути авторизованим, але він не авторизувався, то система просить його спочатку авторизуватися (рис. 3.32).



Рисунок 3.32 – Спроба виконати команди не авторизувавшись

Також система надає змогу виконати запит до ChatGPT, з метою уточнення питань щодо якоїсь команди. Для цього необхідно надіслати боту команду `/ask {zanim}` (рис. 3.33).

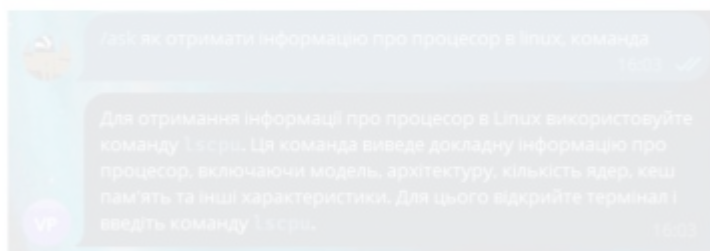


Рисунок 3.33 – Спілкування із ChatGPT

Для відповідної обробки повідомлень від користувача були використані класи-обробники `*Command.cs`, що реалізують патерн програмування `Command`. Відповідність обробників командам наведено в табл. 3.1.

Команда	Клас
/start	StartCommand.cs
Реєстрація	RegisterCommand.cs
Вхід в систему	AuthCommand.cs
Видалити акаунт	DeleteAccountCommand.cs
Вибір віртуальної машини	ChooseVirtualMachineCommand.cs
Додання віртуальної машини	AddVirtualMachineCommand.cs

Виконання команд	ExecuteSSHCommandsCommand.cs
Створення директорії	CreateDirectoryCommand.cs
Видалення директорії	DeleteDirectoryCommand.cs
Завантаження файлу	GetFileFromVirtualMachineCommand.cs
Вивантаження файлу	UploadFileToVirtualMachineCommand.cs
Метрики	GetMetricsCommand.cs
Оновити віртуальну машину	UpdateVirtualMachineCommand.cs
Видалити віртуальну машину	DeleteVirtualMachineCommand.cs
Зміна мови	ChangeLanguageCommand.cs
Вийти із системи	ExitCommand.cs
Запитати в ChatGPT	GetOpenAIResponseMessage.cs

Таблиця 3.1 – Відповідність команд обробникам

3.3. Тестування системи

Мануальне тестування інформаційної системи віддаленої взаємодії з віртуальними машинами включало проведення різних тестів для перевірки функціональності та якості системи. Основною метою таких тестів було переконатися, що всі функції системи працюють правильно і задовольняють вимоги користувачів. Нижче наведено декілька прикладів тестів, які можуть бути проведені:

1. Тестування авторизації та реєстрації:

- Перевірка правильності збереження даних в БД.
- Перевірка правильності хешування паролів.
- Перевірка відповідності даних, що ввів користувач, даним із БД.

2. Тестування додання віртуальної машини:

- Перевірка правильності збереження даних в БД.
- Перевірка правильності шифрування паролів.

3. Тестування під'єднання до віртуальної машини:

- Перевірка надійності SSH- та SFTP-підключень.
- Перевірка правильності виконання команд та отримання очікуваного результату.
- Перевірка створення/видалення директорій.
- Перевірка завантаження/вивантаження файлів.

82

4. Тестування метрик:

- Перевірка правильності отримання метрик.
- Перевірка переведення даних із класу метрик в графіки.

5. Тестування локалізації:

- Перевірка зміни мови.
- Перевірка виявлення багів при авторизації.
- Перевірка виявлення багів при виході із системи.
- Перевірка зміни інтерфейсу в БД.

Також було проведено Unit-тестування всіх сервісів системи. Для кожного з методів були протестовані всі можливі варіанти виконання. Під час тестування було виявлено декілька не критичних багів, які пізніше було виправлено. З більш детальними результатами та самими тестами можна ознайомитися на GitHub-репозиторії проекту. Також за допомогою GitHub Action було налаштовано автоматичне виконання тестів під час кожного коміту. Завдяки цьому в GitHub-репозиторію проекту зберігається історія з усіма проходженнями тестів. Приклад виконання тестів наведено на рис. 3.34.

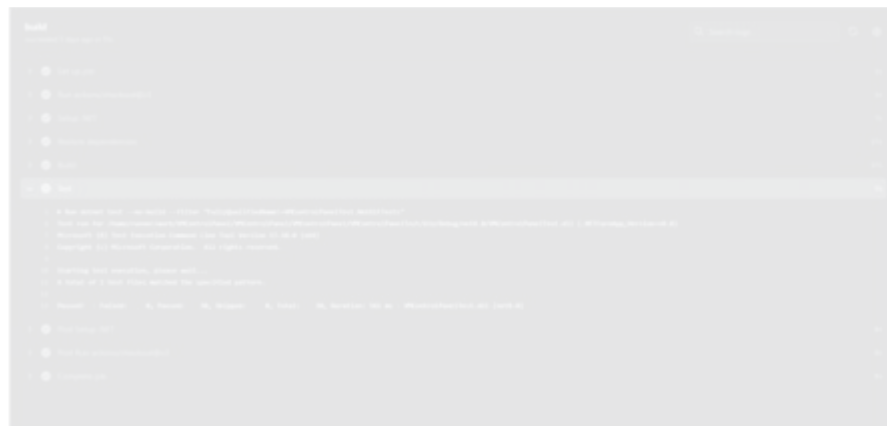


Рисунок 3.34 – Автоматичне виконання тестів

Висновки до розділу 3

У даному розділі було наведено порядок налаштування системи та визначено, що розроблений програмний код не потребує спеціалізованого апаратного забезпечення, додаткових налаштувань та засобів для розгортання, крім стандартних.

Реалізований додаток містить у своєму складі інтерфейс користувача, який надає наступний функціонал: вхід в систему, реєстрація в системі, прив'язка віртуальних машин, адміністрування акаунтів та прив'язаних до них машин, вибір віртуальних машин для взаємодії, віддалене виконання команд, захищене виконання операцій над папками/файлами, отримання первинних метрик для відображення картини стану віртуальної машини, зміна мови системи, вихід із системи.

Було проведено мануальне та unit-тестування, знайдена низка багів, які в подальшому були виправлені. Потім систему було протестовано ще раз, і більше подібних багів не виникало.

ВИСНОВКИ

Під час виконання випускної роботи бакалавра було спроектовано та реалізовано систему віддаленої взаємодії з віртуальними машинами. Для цього було вирішено наступні завдання:

1. Проведено аналіз аналогічних розробок Virtualmin, Proxmox VE, Webmin, cPanel, Grafana, OpenStack Horizon, Virtkick та визначено, що основним функціоналом є розгортання віртуальних машин, виконання команд, отримання інформації про стан, операції над папками/файлами. Однак доцільно реалізувати для системи можливість створення декількох акаунтів з метою розділення потреб, прив'язки необмеженої кількості машин, можливість прив'язати будь-які машини, на яких налаштований SSH, відображення метрик у вигляді графіків, зміна локалізації.

2. Для реалізації було вибрано наступні засоби: мови програмування C# та Python; ORM-бібліотеку EntityFramework Core; бібліотеку для SSH-та SFTP-з'єднання SSH.NET; бібліотеку Telegram.Bot для інтеграції в проєкт Telegram Bot Api; бази даних Microsoft SQL Server та Redis; середовище контейнеризації Docker; середовища розробки Visual Studio Code 1.89.1 та Visual Studio 2022 17.9.6.

3. Було проаналізовано функціональні та нефункціональні вимоги до системи віддаленої взаємодії з віртуальними машинами, для чого побудовано діаграму варіантів використання та на її основі розроблено діаграми класів для аналізу структури системи.

4. Спроектовано базу даних, що забезпечує безпечний та колективний доступ до інформації системи контроль панелі. База даних складається з 2 колекцій: Users та VirtualMachines.

5. Розроблено алгоритми функціонування системи, визначено порядок взаємодії класів під час виконання програмного коду та реалізовано додаток.

Реалізовано інтерфейс користувача, що надає наступний функціонал: вхід в систему, реєстрація в системі, прив'язка віртуальних машин, вибір віртуальних машин для взаємодії, віддалене виконання команд, захищене виконання операцій над папками/файлами, отримання первинних метрик для відображення картини стану віртуальної машини, зміна мови системи, вихід із системи. Реалізовано дані команди за допомогою обробників повідомлень бота:

- StartCommand.cs;
- RegisterCommand.cs;
- AuthCommand.cs;
- DeleteAccountCommand.cs;
- ChooseVirtualMachineCommand.cs;
- AddVirtualMachineCommand.cs;
- ExecuteSSHCommandsCommand.cs;
- CreateDirectoryCommand.cs;
- DeleteDirectoryCommand.cs;
- GetFileFromVirtualMachineCommand.cs;
- UploadFileToVirtualMachineCommand.cs;
- GetMetricsCommand.cs;
- UpdateVirtualMachineCommand.cs;
- DeleteVirtualMachineCommand.cs;
- ChangeLanguageCommand.cs;
- ExitCommand.cs;
- GetOpenAIResponseMessage.cs.

Реалізований програмний комплекс готовий до промислового використання. Може бути застосований для зручного менеджменту, керування та взаємодії із віртуальними машинами користувачів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ylonen; C. Lonvick (January 2006). The Secure Shell (SSH) Protocol Architecture
2. Amies, A; Wu, C F; Wang, G C; Criveti, M (2012). «Networking on the cloud»
3. Tatu Ylönen. "SSH Port"
4. "The What's, How's and Why's of SFTP"
5. Galbraith, Joseph; Saarenmaa, Oskari (18 July 2006). "SSH File Transfer Protocol"
6. "OpenBSD manual page for the "sftp" command: "See Also" section". OpenBSD.org. Retrieved 2018-02-04
7. "The Object-Relational Impedance Mismatch". AgileData.org. Retrieved 2022-12-12.
8. "Announcing C# 12". Retrieved November 18, 2023.
9. <https://sshnet.github.io/SSH.NET/>
10. <https://tegrambots.github.io/book/>
11. "Introduction to Redis". Redis is written in ANSI C and works in most POSIX systems like Linux, *BSD, OS X without external dependencies.
12. What's new in SQL Server 2022 (16.x)
13. "What is a Container?". docker.com. Docker, Inc. Retrieved May 13, 2019.

ДОДАТКИ

Додаток А. Код програми

Схожість

Джерела з Інтернету

1

2	https://learn.ztu.edu.ua/pluginfile.php/22940/mod_resource/content/1/%D0%B1%D0%B0%D0%BA%D0%B0%D0%BB%D0%B0...	4.09%
---	---	-------

Джерела з Бібліотеки

2

1	Осіпов	ID файлу: 1015255623	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University	11%
---	--------	----------------------	---	-----

3	Дипломна_Семотюк П.В	ID файлу: 1011480396	Навчальний заклад: Yuriy Fedkovych Chernivtsi National Univ...	2.32%
---	----------------------	----------------------	--	-------

Цитати

Цитати

8

- 1 The scenarios of the software complex, the general structure of the software complex, the description of the system database (DBMS Microsoft SQL Server), the description of the interaction algorithms of individual system modules, the description of the object structure of the system (class diagram), the work algorithms of the main modules implemented using the programming language are given C# and Python.
- 2 Presentation Tier (**Шар презентації**): Цей рівень відповідає за відображення та взаємодію з користувачем.
- 3 Business Logic Tier (**Шар бізнес-логіки**): Цей рівень містить бізнес-логіку програми.
- 4 Data Access Tier (**Шар доступу до даних**): Цей рівень забезпечує доступ до даних та їх управління.
- 5 Infrastructure Tier (**Інфраструктурний шар**): Цей рівень надає загальні служби та інфраструктуру для функціонування програми.
- 6 Visual Studio Code (VS Code) – це безкоштовний та потужний текстовий редактор, розроблений компанією Microsoft.
- 7 Configurations { public class UserConfiguration : IEntityTypeConfiguration<User> { public void Configure(EntityTypeBuilder<User> builder) { builder.
- 8 Configurations { public class VirtualMachineConfiguration : IEntityTypeConfiguration<VirtualMachine> { public void Configure(EntityTypeBuilder<VirtualMachine> builder) { builder.

Вилучення

Вилучення

303

https://dspace.nuft.edu.ua/server/api/core/bitstreams/b20108f3-1ed1-4a6b-84b1-759c266fd601/content	1.16%
https://ela.kpi.ua/bitstream/123456789/47794/1/Modeliuvannia.pdf	0.72%
https://ua-referat.com/uploaded/poyasnyvalena-zapiska-do-diplomnogo-proektu-na-temu-rozrobka/index1.html	0.7%
http://ir.nusta.edu.ua/bitstream/123456789/7355/1/122_%d0%a8%d1%83%d0%bf%d0%b8%d0%ba_2020.pdf	0.69%
https://ela.kpi.ua/bitstream/123456789/58965/1/Bitlian_bakalavr.pdf	0.68%
https://ir.library.knu.ua/server/api/core/bitstreams/19a28d06-028c-45ae-a38b-7f37af2f9c16/content	0.65%
https://dspace.nuft.edu.ua/jspui/bitstream/123456789/33599/1/122_Rolyk%20Vadym%20Mykolaiovych.pdf	0.63%
https://ir.nmu.org.ua/bitstream/handle/123456789/165928/121%d0%9c-22-2_%d0%9b%d0%b5%d0%b2%d0%b4%d0%b8%cd	8 джерел 0.6%
http://bmc.fbmi.kpi.ua/uploads/diplom/bosenko-natal%D1%96ya-volodimir%D1%96vna.pdf	0.55%
https://ela.kpi.ua/handle/123456789/43965	2 джерел 0.54%
http://biblio.umsf.dp.ua/xmlui/bitstream/handle/123456789/6382/2023_Kozhemiaka_122_Bak.pdf?isAllowed=y&sequence=1	0.53%
https://dspace.nau.edu.ua/bitstream/NAU/54150/1/%d0%9a%d0%b2%d0%b0%d0%bb%d1%96%d1%84%d1%96%d0%ba%d0%b0...	0.46%
https://er.nau.edu.ua/bitstream/NAU/50631/1/%d0%a4%d0%9a%d0%9a%d0%9f%d0%86_2020_123_%d0%86%d0%b2%d0%b0%d...	0.46%
https://dspace.nuft.edu.ua/server/api/core/bitstreams/917940b1-36d7-4a67-bb4d-4f9469903f59/content	8 джерел 0.41%
https://dspace.nuft.edu.ua/jspui/bitstream/123456789/33629/1/122_Fen%20Khrystyna%20Romanivna.pdf	20 джерел 0.41%
https://ela.kpi.ua/bitstream/123456789/55945/1/Kuznietsov_bakalavr.pdf	0.4%
http://dspace.nuft.edu.ua/jspui/bitstream/123456789/33606/1/122_Slietsov%20Artem%20Ruslanovych.pdf	0.39%
http://lpc-dspace.org.ua/bitstream/123456789/395/1/visnik_krnu_2_2022%20%281%29.pdf	0.35%
https://ela.kpi.ua/bitstream/123456789/40933/1/Tarassenko_bakalavr.pdf	5 джерел 0.35%
http://elartu.tntu.edu.ua/bitstream/lib/30658/4/dyplom_Hilyta_M_2020.pdf	0.32%

https://ela.kpi.ua/bitstream/123456789/58181/1/Shmatko_bakalavr.pdf	0.2%
https://ir.library.knu.ua/server/api/core/bitstreams/df71a595-1099-4093-99b0-96800370794d/content	0.29%
https://ir.library.knu.ua/server/api/core/bitstreams/0df1f485-eeac-485f-ae85-1ed7fef0b21a/content	20 джерел 0.28%
https://ela.kpi.ua/bitstream/123456789/44142/1/Sergeev_bakalavr.pdf	0.28%
http://eprints.library.odeku.edu.ua/id/eprint/7086/1/Osadcha_Rozrobka_samopysnogo_CMS_dviguna_B_2020.pdf	0.27%
https://theses.oa.edu.ua/DATA/2520/%D0%94%D0%B5%D0%BC%D0%B8%D0%B4%D1%8E%D0%BA_%D0%B4%D1%80%D1%83%...	0.26%
https://wiki2.org/en/Sftp:	10 джерел 0.23%
https://ela.kpi.ua/bitstream/123456789/52181/1/Hrybniak_bakalavr.pdf	6 джерел 0.15%
https://uk.unionpedia.org/i/java	0.23%
https://en.m.wikipedia.org/wiki/SSH	0.23%
https://ela.kpi.ua/bitstream/123456789/36233/1/Dmytruk_bakalavr.pdf	8 джерел 0.23%
https://uk.wikipedia.org/wiki?curid=4020765	0.21%
https://ela.kpi.ua/bitstream/123456789/38195/1/Storozhyk_magistr.pdf	0.2%
https://www.wikizero.com/m/Redis	29 джерел 0.19%
https://ela.kpi.ua/bitstream/123456789/57136/1/Kolodko_bakalavr.pdf	0.18%
http://www.macoratti.net/21/05/ef_fluentapi2.htm	25 джерел 0.18%
https://journals.nupp.edu.ua/sunz/issue/download/115/63	0.17%
https://conf.ztu.edu.ua/wp-content/uploads/2019/02/tezy-dopovidej-ktipr-2018.pdf	3 джерела 0.17%
http://elar.khmnu.edu.ua/jspui/bitstream/123456789/13801/1/%D0%9a%D0%b2%D0%a0-%D0%91%D0%b0%D0%bb%D0%b	2 джерела 0.16%
https://krs.chmnu.edu.ua/jspui/bitstream/123456789/2975/1/%D0%A1%D1%82%D1%80%D0%B8%D0%B6%D0%B0%D0%BA%20...	0.15%
https://ela.kpi.ua/handle/123456789/44195	3 джерела 0.15%
https://ela.kpi.ua/bitstream/123456789/38176/1/Sieviertsev_bakalavr.pdf	0.12%

[illegible]

https://ela.kpi.ua/handle/123456789/23912	2 джерела	0.07%
https://ela.kpi.ua/bitstream/123456789/30673/1/Demydov_magistr.pdf	2 джерела	0.07%
https://ela.kpi.ua/bitstream/123456789/35463/1/Lytvynenko_bakalavr.pdf		0.07%
https://er.nau.edu.ua/bitstream/NAU/62166/1/%d0%a4%d0%9a%d0%9d%d0%a2_2023_122_%d0%9b%d1%83%d1%86%d1%8c%d...		0.07%
https://aistis.knu.ua/wp-content/uploads/2021/09/AISTIS_2023.pdf		0.07%
http://ekhsuir.kspu.edu/bitstream/handle/123456789/18220/_%d0%a8%d1%82%d0%b0%d0%bd%d0%b3_%d0%9d_.pdf?isAllowed...		0.07%
https://dspace.nuft.edu.ua/jspui/bitstream/123456789/40634/1/181_Guzar_Yuliya_Muhailivna_67_10229.pdf		0.07%
https://ela.kpi.ua/handle/123456789/43652	21 джерело	0.07%
https://dspace.nuft.edu.ua/jspui/bitstream/123456789/35853/1/181_Vasyliiev%20Denys%20Viacheslavovych.pdf		0.07%
https://er.nau.edu.ua/bitstream/NAU/63370/1/%d0%a4%d0%9a%d0%9a%d0%9f%d0%86_2024_121_%d0%a1%d0%b2%d0%b8%d...		0.07%
https://ir.nmu.org.ua/xmlui/bitstream/handle/123456789/164862/23_06_12_121-19-2_%d0%9d%d0%b0%d0%b7%d0%b0%d1%80%...		0.07%
https://ela.kpi.ua/handle/123456789/42540	2 джерела	0.07%
http://ir.znau.edu.ua/bitstream/123456789/11859/1/Klymenko%20YO%20KR%20122%202021.pdf		0.07%

Вилучення по Бібліотеці акаунту

523

Палійчук_Дипломна_443_група	ID файлу: 1015163876	Навчальний заклад: Yuriy Fedkovych Chernivtsi National U...	1.74%
Tomashchuk_doc (2)	ID файлу: 1011478898	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University	1.95%
бакалаврат_Баранов	ID файлу: 1011492811	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University	1.82%
Бакалаврська робота Мороз	ID файлу: 1011473213	Навчальний заклад: Yuriy Fedkovych Chernivtsi National Uni...	1.82%
Платаш	ID файлу: 1011480397	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University	1.58%
Фісяк Владислав антиплагіат4	ID файлу: 1011482871	Навчальний заклад: Yuriy Fedkovych Chernivtsi National U...	1.34%
Diploma - Дронь (1)	ID файлу: 1009711510	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University	1.33%
diploma-Datsko	ID файлу: 1015255617	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University	0.98%

Редчук	ID файлу: 1011475871	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University	2 Джерело	1.3%
Diploma_Bidiuk	ID файлу: 1015196878	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University	11 Джерело	1.28%
бакалаврат_Гладчук_В_Юдокінця	ID файлу: 1011490610	Навчальний заклад: Yuriy Fedkovych Chernivtsi Nation...		1.28%
Дипломна робота Богусевич - Цього разу вже точно	ID файлу: 1011478880	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University	20 Джерело	1.2%
Сагайдак1	ID файлу: 1011480398	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University		1.2%
Дипломна_робота_Ворощук_БЕЗДОДАТКІВ	ID файлу: 1015196868	Навчальний заклад: Yuriy Fedkovych Cherni...		0.85%
чудієвич2	ID файлу: 1011478897	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University	3 Джерело	1.14%
diploma-Viglush	ID файлу: 1015226896	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University		1.05%
Пояснювальна записка Халус Іван.docx	ID файлу: 1015693253	Навчальний заклад: Yuriy Fedkovych Chernivtsi N...		1.07%
Антиплагіат_документація_Кисорець	ID файлу: 1015247288	Навчальний заклад: Yuriy Fedkovych Chernivtsi Na...		1.07%
iftc_2022_098.pdf	ID файлу: EF-100000246600	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University		1.05%
записка_Соколюк_final_no_additional_pdf	ID файлу: 1011459281	Навчальний заклад: Yuriy Fedkovych Chernivtsi N...		1.03%
Лучик_А	ID файлу: 1011473218	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University		1.03%
Середенко Богдан дипломна (1)	ID файлу: 1011468177	Навчальний заклад: Yuriy Fedkovych Chernivtsi National...		0.99%
Бідяк	ID файлу: 1015196872	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University		0.89%
Ярема Вадим 2022	ID файлу: 1011475799	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University		0.96%
Студентська робота	ID файлу: 1015200625	Навчальний заклад: Lviv Polytechnic National University	12 Джерело	0.94%
Бален_бакалаврат (2)	ID файлу: 1011488709	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University		0.91%
Халус Іван	ID файлу: 1011480400	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University	8 Джерело	0.74%
Горецький_Євген	ID файлу: 1011434868	Навчальний заклад: Yuriy Fedkovych Chernivtsi National Univers	2 Джерело	0.69%
Студентська робота	ID файлу: 1011348889	Навчальний заклад: National Aviation University	7 Джерело	0.67%
Студентська робота	ID файлу: 1015227477	Навчальний заклад: National Technical University of Ukraine	5 Джерело	0.68%

Студентська робота	ID файлу: 1016089117	Навчальний заклад: Cherkasy State Technological University	6 Джерело	0.68%
Макидон	ID файлу: 1015211264	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University	5 Джерело	0.56%
Студентська робота	ID файлу: 1011392022	Навчальний заклад: National Aviation University	8 Джерело	0.66%
Студентська робота	ID файлу: 1015183286	Навчальний заклад: National Aviation University		0.66%
Студентська робота	ID файлу: 1008168023	Навчальний заклад: Taras Shevchenko National University of	19 Джерело	0.66%
Студентська робота	ID файлу: 1008420743	Навчальний заклад: State University Kyiv National Economic Universit...		0.64%
Студентська робота	ID файлу: 1005713409	Навчальний заклад: Zaporizhzhya National University	24 Джерело	0.65%
math_2019_008.pdf	ID файлу: EF-100000098401	Навчальний заклад: Yuriy Fedkovych Chernivtsi National Un	10 Джерело	0.64%
Мойсей Дипломна 2023_removed	ID файлу: 1015217460	Навчальний заклад: Yuriy Fedkovych Chernivtsi National U..		0.38%
Бакалаврат Андрух Сергій 2022	ID файлу: 1011468891	Навчальний заклад: Yuriy Fedkovych Chernivtsi National U..		0.54%
Андрійчук	ID файлу: 1011475791	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University		0.58%
diplomaN4P1-Kalenyk	ID файлу: 1011505805	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University		0.57%
Студентська робота	ID файлу: 1008408920	Навчальний заклад: National University of Water Management and Na...		0.54%
Диплом_Антиплагіат_Фуштей	ID файлу: 1016151933	Навчальний заклад: Yuriy Fedkovych Chernivtsi National Un...		0.54%
Попик	ID файлу: 1011467841	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University	2 Джерело	0.46%
Студентська робота	ID файлу: 1008332418	Навчальний заклад: National Technical University of Ukraine	4 Джерело	0.54%
Студентська робота	ID файлу: 1007938778	Навчальний заклад: National Technical University of Ukraine "Kyiv Po...		0.51%
Документація_Олейнич_без_додатків	ID файлу: 1015226894	Навчальний заклад: Yuriy Fedkovych Chernivtsi Na...		0.44%
comp_2012_026.pdf	ID файлу: EF-12588	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University	3 Джерело	0.48%
Диплом Митроняк	ID файлу: 1015196867	Навчальний заклад: Yuriy Fedkovych Chernivtsi National Univer	2 Джерело	0.49%
дипломМагістр1 - антиплагіат	ID файлу: 1013082908	Навчальний заклад: Yuriy Fedkovych Chernivtsi N	2 Джерело	0.49%
Матій_антиплагіат	ID файлу: 1016147677	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University		0.49%

Студентська робота	ID файлу: 1008295753	Навчальний заклад: National Technical University of Ukraine "Kyiv Po...	0.44%
Студентська робота	ID файлу: 1005759467	Навчальний заклад: National Aviation University	12 Джерело 0.46%
Студентська робота	ID файлу: 5907719	Навчальний заклад: National Technical University of Ukraine "Ky	13 Джерело 0.45%
Середенко Богдан диплома	ID файлу: 1015685768	Навчальний заклад: Yuriy Fedkovych Chernivtsi National Un...	0.44%
Студентська робота	ID файлу: 1005729057	Навчальний заклад: National Aviation University	0.44%
Дипломна_робота_Боднарюка_Івана	ID файлу: 1015234526	Навчальний заклад: Yuriy Fedkovych Chernivtsi National Un...	7 Джерело 0.43%
Магістрська без додатків - Єсипчук	ID файлу: 1009709001	Навчальний заклад: Yuriy Fedkovych Chernivtsi Nati...	0.42%
ДИПЛОМ_Городеньського_Станіслава	ID файлу: 1015226898	Навчальний заклад: Yuriy Fedkovych Chernivtsi N...	0.41%
Студентська робота	ID файлу: 1008374688	Навчальний заклад: National Technical University of Ukraine	3 Джерело 0.4%
Студентська робота	ID файлу: 1011407070	Навчальний заклад: Lviv Polytechnic National University	7 Джерело 0.39%
Студентська робота	ID файлу: 1005703075	Навчальний заклад: National Technical University of Ukraine	10 Джерело 0.39%
Мазурець	ID файлу: 1015188190	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University	0.39%
Студентська робота	ID файлу: 1015725501	Навчальний заклад: Poltava National Technical Yuri Kondratyuk Unive...	0.38%
антиплагіат Сагайдак (1)	ID файлу: 1015691123	Навчальний заклад: Yuriy Fedkovych Chernivtsi National Univers...	0.37%
Пояснювальна записка Костащук (1)	ID файлу: 1015223337	Навчальний заклад: Yuriy Fedkovych Cher	15 Джерело 0.36%
Пояснювальна записка Томащук Дмитро	ID файлу: 1015696407	Навчальний заклад: Yuriy Fedkovych Chernivtsi...	0.36%
Студентська робота	ID файлу: 1004100888	Навчальний заклад: National Technical University of Ukraine	8 Джерело 0.36%
Андруйчук Кваліфікаційна Робота	ID файлу: 1013074560	Навчальний заклад: Yuriy Fedkovych Chernivtsi Natio...	0.22%
iftc_2023_084.pdf	ID файлу: EF-100000249418	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University	0.22%
Магістерська_робота_Богусевич_Друк_Антиплагіат	ID файлу: 1015693274	Навчальний заклад: Yuriy Fedkovycs...	0.23%
Бабаєв А.І	ID файлу: 1016123852	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University	0.29%
iftc_2022_099.pdf	ID файлу: EF-100000246601	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University	0.28%

Студентська робота	ID файлу: 1008337544	Навчальний заклад: National Technical University of Ukraine	15 Джерело	0.28%
iftc_2023_089.pdf	ID файлу: EF-100000249423	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University		0.28%
Рудяга_записка (2)	ID файлу: 1011506016	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University		0.27%
Піцул Д.М. 643. Диплома робота	ID файлу: 1009698587	Навчальний заклад: Yuriy Fedkovych Chernivtsi	2 Джерело	0.27%
Студентська робота	ID файлу: 1011427794	Навчальний заклад: National University Ostroh Academy		0.27%
Студентська робота	ID файлу: 5988149	Навчальний заклад: National Technical University of Ukraine "Kyiv Polyte...		0.26%
дипломна_робота_Батуєв_Андрій_антиплагіат	ID файлу: 1011488716	Навчальний заклад: Yuriy Fedko	2 Джерело	0.25%
Студентська робота	ID файлу: 1015246700	Навчальний заклад: National Technical University of Ukraine	3 Джерело	0.13%
Студентська робота	ID файлу: 1010381961	Навчальний заклад: Lviv Polytechnic National University		0.23%
Студентська робота	ID файлу: 1008340927	Навчальний заклад: National Technical University of Ukraine "Kyiv Po...		0.21%
Студентська робота	ID файлу: 1001022452	Навчальний заклад: National Aviation University		0.22%
Студентська робота	ID файлу: 1000099749	Навчальний заклад: Lviv Polytechnic National University	14 Джерело	0.21%
Студентська робота	ID файлу: 1011417301	Навчальний заклад: National Technical University of Ukraine	2 Джерело	0.2%
Студентська робота	ID файлу: 1011490727	Навчальний заклад: National Technical University of Ukraine "Kyiv Po...		0.19%
Студентська робота	ID файлу: 1015705927	Навчальний заклад: Vasyl Stus Donetsk National University	3 Джерело	0.19%
Студентська робота	ID файлу: 1016107303	Навчальний заклад: Cherkasy State Technological University	6 Джерело	0.18%
Студентська робота	ID файлу: 1000045727	Навчальний заклад: National University of Life and Environmental Sci...		0.18%
Студентська робота	ID файлу: 1016142907	Навчальний заклад: Lviv Polytechnic National University	4 Джерело	0.18%
Гавалешко	ID файлу: 1015217462	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University		0.18%
Сербенюк	ID файлу: 1015226895	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University		0.16%
Ткачук Ніколау, Бакалаврська	ID файлу: 1016125080	Навчальний заклад: Yuriy Fedkovych Chernivtsi National U...		0.07%
ABDELRAZEK DIPLOME	ID файлу: 1011482584	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University		0.08%

diploma-Tserkovniuk	ID файлу: 1015285282	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University	0.08%
Документаціяяя Кібак. К В	ID файлу: 1016114040	Навчальний заклад: Yuriy Fedkovych Chernivtsi National Unive...	0.13%
Студентська робота	ID файлу: 1015202480	Навчальний заклад: National Technical University of Ukraine "Kyiv Po...	0.13%
iftc_2022_093.pdf	ID файлу: EF-100000246595	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University	0.13%
бакалаврат_MOSKALIUK_	ID файлу: 1011480394	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University	0.13%
Студентська робота	ID файлу: 1015633188	Навчальний заклад: National University Ostroh Academy	3 Джерело 0.07%
Огороднік_Диплом_2021	ID файлу: 1009710059	Навчальний заклад: Yuriy Fedkovych Chernivtsi National	3 Джерело 0.13%
скрипник 422	ID файлу: 1015232426	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University	2 Джерело 0.13%
Студентська робота	ID файлу: 1015715447	Навчальний заклад: Lutsk National Technical University	0.12%
Студентська робота	ID файлу: 1015756306	Навчальний заклад: National Technical University of Ukraine	2 Джерело 0.11%
Студентська робота	ID файлу: 1015236136	Навчальний заклад: Lviv Polytechnic National University	6 Джерело 0.08%
Студентська робота	ID файлу: 1016126676	Навчальний заклад: Lviv Polytechnic National University	0.1%
Студентська робота	ID файлу: 1013055082	Навчальний заклад: National Technical University of Ukraine	5 Джерело 0.09%
iftc_2021_085.pdf	ID файлу: EF-100000153601	Навчальний заклад: Yuriy Fedkovych Chernivtsi National Univ	11 Джерело 0.09%
Студентська робота	ID файлу: 1003941426	Навчальний заклад: National Aviation University	2 Джерело 0.09%
Студентська робота	ID файлу: 1015835101	Навчальний заклад: National Aviation University	2 Джерело 0.08%
Студентська робота	ID файлу: 1015283602	Навчальний заклад: Donetsk National Technical University	0.08%
Студентська робота	ID файлу: 1000097308	Навчальний заклад: National University of Water Managemen	41 Джерело 0.08%
Студентська робота	ID файлу: 1016116129	Навчальний заклад: Uzhhorod National University	3 Джерело 0.08%
Студентська робота	ID файлу: 1003694724	Навчальний заклад: Cherkasy State Technological University	15 Джерело 0.08%
Студентська робота	ID файлу: 1015284022	Навчальний заклад: National Technical University of Ukraine "Kyiv Po...	0.08%
iftc_2021_126.pdf	ID файлу: EF-100000153643	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University	0.08%

Студентська робота	ID файлу: 1015551991	Навчальний заклад: Zaporizhzhya National University		0.08%
Студентська робота	ID файлу: 1016095149	Навчальний заклад: Cherkasy State Technological University		0.08%
Студентська робота	ID файлу: 1008296747	Навчальний заклад: National Aviation University	28 Джерело	0.08%
Студентська робота	ID файлу: 1005754644	Навчальний заклад: National University of Water Management	2 Джерело	0.07%
Студентська робота	ID файлу: 1000037999	Навчальний заклад: National Technical University of Ukraine	3 Джерело	0.07%
Студентська робота	ID файлу: 1016116311	Навчальний заклад: National Technical University of Ukraine "Kyiv Po...		0.07%
Студентська робота	ID файлу: 1016095619	Навчальний заклад: Lviv Polytechnic National University		0.07%
Студентська робота	ID файлу: 1015726514	Навчальний заклад: Zhytomyr National Agroecological Unive	29 Джерело	0.07%
Студентська робота	ID файлу: 1015088946	Навчальний заклад: National Technical University of Ukraine "Kyiv Po...		0.07%
Студентська робота	ID файлу: 1000800949	Навчальний заклад: National Technical University of Ukraine	2 Джерело	0.07%
Студентська робота	ID файлу: 5439688	Навчальний заклад: National Technical University of Ukraine "Kyiv Polyte...		0.07%
inoz_2010_029.pdf	ID файлу: EF-9398	Навчальний заклад: Yuriy Fedkovych Chernivtsi National University	2 Джерело	0.07%
Студентська робота	ID файлу: 1014924640	Навчальний заклад: Lviv Polytechnic National University	3 Джерело	0.07%
Студентська робота	ID файлу: 1015652808	Навчальний заклад: Lviv Polytechnic National University	2 Джерело	0.07%
Студентська робота	ID файлу: 1015276281	Навчальний заклад: National Technical University of Ukraine "Kyiv Po...		0.07%