Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет *Інфокомунікаціїі*
(повна назва)

Кафедра *Інформаційно-мережної інженерії*
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

Рівень вищої освіти *другий (магістерський)*

Розробка додатку для моніторингу офісних пристроїв в локальній мережі з використанням хмарних технологій
(тема)

Виконав:
студент 2 курсу, групи *ІМІм-22-3*
Кулініч А.О.

Спеціальності *172 Телекомунікації та радіотехніка*
(код і повна назва спеціальності)
Тип програми *Освітньо-наукова*
(освітньо-професійна або освітньо-наукова)
Освітня програма *Інформаційно-мережна інженерія*
(повна назва освітньої програми)
Керівник *проф, д.т.н. Безрук В.М.*
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ *Безрук В.М.*
(підпис) (прізвище, ініціали)

2024 р.

Не містить відомостей, заборонених до відкритого публікування

Студент _____ *Кулініч А.О*
(підпис) (прізвище та ініціали)

Керівник _____ *Безрук В.М.*
(підпис) (прізвище та ініціали)

Харківський національний університет радіоелектроніки

Факультет *Інфокомунікацій*
(повна назва)

Кафедра *Інформаційно-мережної інженерії*
(повна назва)

Рівень вищої освіти *другий (магістерський)*

Спеціальність *172 Телекомунікації та радіотехніка*
(код і повна назва)

Тип програми *Освітньо-наукова*
(освітньо-професійна або освітньо-наукова)

Освітня програма *Інформаційно-мережна інженерія*
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри ІМІ_____
(підпис)
“____” _____2024 року

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Студентові *Кулінічу Антону Олександровичу*
(прізвище, ім’я, по батькові)

1. Тема роботи *Розробка додатку для моніторингу офісних пристроїв в локальній мережі з використанням хмарних технологій*
затверджені наказом університету від *18 березня 2024* року № *232 Ст*
2. Термін подання студентом роботи до екзаменаційної комісії *15 червня 2024 р.*
3. Вихідні дані до роботи
Проаналізувати методи підключення локальної мережі до віртуальної мережі, дослідити важливість комфорту в офісному середовищі. Дослідити методи підключення додатку з хмари до локальної мережі. Розробити додаток, який керує периферійними пристроями в локальній мережі через комп’ютер адміністратора, отримує інформацію про кліматичну техніку та зберігає її в хмарному сервісі.

4. Перелік питань, що потрібно опрацювати в роботі
Вступ.
1. Теоретичні аспекти розробки додатку.
2. Розробка хмарної інфраструктури.
3. Розробка веб додатку для моніторингу.
Висновки.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) *Слайди у форматі Power Point (назва, мета і задачі роботи, основні результати роботи, висновки)*

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів атестаційної роботи	Строк виконання етапів роботи	Примітка
1	*Ознайомлення із завданням. Уточнення ТЗ*	*18.03.24*	*виконано*
2	*Підбір літератури за темою роботи*	*19.03 - 02.04.24*	*виконано*
3	*Виконання розділу 1*	*03.04 - 19.04.24*	*виконано*
4	*Виконання розділу 2*	*20.04 – 12.05.24*	*виконано*
5	*Виконання розділу 3*	*13.05 – 02.06.24*	*виконано*
6	*Оформлення пояснювальної записки*	*03.06 - 07.06.24*	*виконано*
7	*Оформлення презентаційного матеріалу,*		
	підготовка до захисту у ЕК	*08.06 - 14.06.24*	*виконано*

Дата видачі завдання *18.03.2024 р.*

Студент _____ *Кулініч А.О.*
(підпис) (прізвище та ініціали)

Керівник роботи _____ *Безрук В.М.*
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка: 76 с., 37 рис., 1 табл., 8 джерел, 2 додатки.

Об'єкт дослідження – взаємодія хмарних технологій з локальною мережею.

Мета роботи – дослідити методи підключення хмарних додатків до локальної мережі.

Результати – в роботі виконано аналіз методів підключення локальної мережі до віртуальної мережі, аналіз важливості комфорту в офісному середовищі. Детально розглянуто методи підключення додатку з хмари до локальної мережі. В якості вирішення запропоновано створити VPN з'єднання між двома віртуальними мережами, одна з яких імітує роботу локальної. Розроблено додаток, який керує периферійними пристроями в локальній мережі через комп'ютер адміністратора, та отримує інформацію про кліматичну техніку, яку в свою чергу зберігає в хмарному сервісі.

МЕРЕЖА, ДОДАТОК, АДМІНІСТРАТОР, ХМАРНА ТЕХНОЛОГІЯ, ЛОКАЛЬНА МЕРЕЖА, ВІРТУАЛЬНА МЕРЕЖА.

THE ABSTRACT

Explanatory note: 76 p., 37 fig., 1 tabl., 8 sources, 2 app.

The object of study is is the interaction of cloud technologies with a local network.

The purpose of the work is investigate methods of connecting cloud applications to a local network.

Results - Results - the work includes an analysis of methods of connecting a local network to a virtual network, an analysis of the importance of comfort in an office environment. The methods of connecting the application from the cloud to the local network are considered in detail. As a solution, it is proposed to create a VPN connection between two virtual networks, one of which simulates a local network. An application has been developed that controls peripheral devices in the local network through the administrator's computer, and receives information about climate technology, which in turn is stored in a cloud service.

NETWORK, APPLICATION, ADMINISTRATOR, CLOUD TECHNOLOGIES, VIRTUAL NETWORKS.

ЗМІСТ

С.

ПЕРЕЛІК СКОРОЧЕНЬ

API (Application Programming Interface) – інтерфейс програмування за стосунків;

CIM (Common Information Model) - загальна інформаційна модель;

IaaS (Infrastructure as a Service) – інфраструктура як послуга;

PaaS (Platform as a Service) – це модель, коли розробники отримують доступ до платформи для розробки та розгортання додатків через Інтернет;

PAYG (Pay-as-you-go) – оплата по мірі споживання;

SaaS (Software as a Service) – програмне забезпечення як послуга;

VNet (Virtual Network) – віртуальна мережа;

VPN (Virtual Private Network) – віртульна приватна мережа;

WBEM (Web-Based Enterprise Management) – керування підприємством через інтернет;

WMI (Windows Management Instrumentation) – інструмент керування Windows.

ВСТУП

В сучасному цифровому світі офісні пристрої стають не лише невід'ємною частиною робочого середовища, але й ключовим елементом ефективного управління та моніторингу. З урахуванням постійно зростаючих потреб у забезпеченні безпеки, продуктивності та оптимізації ресурсів у сфері офісної діяльності, розробка та впровадження спеціалізованого додатку для моніторингу офісних пристроїв у локальній мережі є актуальною та важливою задачею.

У цьому контексті пропонується розглянути роботу, яка спрямована на розробку інноваційного додатку, здатного забезпечити комплексний моніторинг та управління офісними пристроями через локальну мережу, з використанням сучасних хмарних технологій. Цей підхід відкриває широкі можливості для вдосконалення процесів обслуговування, підвищення надійності та безпеки даних, а також забезпечення швидкого доступу до інформації про стан пристроїв та їх функціональні можливості.

Хмарні провайдери пропонують високий рівень безпеки, включаючи шифрування даних, автентифікацію користувачів та регулярні оновлення безпеки. Це допомагає захистити дані та знизити ризики несанкціонованого доступу, що є критично важливим для бізнесу. Не менш важливою є доступність додатку для користувачів, хмарні сервіси дозволяють швидко розгортати нові версії програмного забезпечення та оновлення. Це означає, що нові функції та покращення будуть доступні користувачам набагато швидше.

Враховуючи необхідність постійного моніторингу та реагування на поточні потреби офісного середовища, дана кваліфікаційна робота пропонує високоефективне рішення, спрямоване на покращення управління та експлуатації офісних пристроїв з використанням передових технологій хмарного зберігання та обробки даних.

1 ТЕОРЕТИЧНІ АСПЕКТИ РОЗРОБКИ ДОДАТКУ

1.1 Опис важливості комфорту робочого середовища

Офісна техніка, така як пристрої для роботи з документами та кліматична техніка, відіграють надзвичайно важливу роль у забезпеченні ефективної роботи офісу.

Принтери є важливим елементом для друкування документів, звітів, листів, аналізів та інших матеріалів, які можуть бути необхідні для роботи офісу. Багатофункціональні принтери дозволяють копіювати та сканувати документи, що спрощує процес обробки та обміну інформацією в офісі. Можливість швидкого та якісного друку дозволяє співробітникам зосередитися на своїй основній роботі, не витрачаючи час на пошук зовнішніх друкованих послуг. Використання внутрішнього принтера дозволяє зменшити час на отримання друкованих матеріалів, а також зменшити витрати на зовнішні друкарні.

Правильно налаштована кліматична техніка (кондиціонери, обігрівачі, вентиляційні системи) забезпечує комфортні умови для роботи, що сприяє збереженню здоров'я та підвищенню продуктивності персоналу. Відповідний рівень вологості є важливим для збереження якості матеріалів та обладнання, що знаходиться в офісі. Кліматична техніка допомагає підтримувати оптимальний рівень вологості, запобігаючи пошкодженню документів, меблів та іншого обладнання.

Кліматична техніка також може допомогти захистити офісне обладнання від негативного впливу зовнішніх факторів, таких як екстремальні температури та вологість.

В цілому, офісна техніка, така як принтери та кліматична техніка, не лише сприяє підвищенню продуктивності та ефективності роботи офісу, але й забезпечує комфортні та безпечні умови для працівників. Їх правильне

використання і обслуговування є важливими для успішного функціонування будь-якого бізнесу.

Мікроклімат виробничого середовища істотно впливають на стан організму працівників, їх працездатність протягом робочого дня. Показники температури, відносної вологості, швидкості руху повітря, теплового випромінювання поверхонь, що нагріваються, характеризують клімат внутрішнього середовища виробничих приміщень. У процесі праці людина перебуває в тепловій взаємодії з виробничим середовищем. с

Вологість повітря має величезний вплив на самопочуття та працездатність. Через високу вологість зменшується віддача тепла шляхом випаровування. Зниження вологості посилює процес тепловіддачі. Однак навіть низька вологість викликає пересихання слизової оболонки дихальних шляхів. Відповідно до нового дослідження, опублікованого в журналі “Building and Environment” командою дослідників з Японії комфорт офісного середовища має значні економічні вигоди, пов’язані з ефективністю роботи та презентабельністю, використовуючи дані 1644 працівників 29 офісних будівель у Токіо, Японія [1].

Комфортне робоче середовище є важливим фактором для привабливості компанії для потенційних працівників. Бізнеси, які надають увагу клімату в офісі, зазвичай привертають та утримують талановитих співробітників. Комфортні умови праці сприяють підвищенню продуктивності. Працівники, які відчувають себе комфортно, швидше зосереджуються на своїй роботі та більш ефективно виконують завдання. Оптимізований клімат дозволяє ефективніше використовувати енергію та зменшити витрати на опалення або кондиціювання повітря. Це може значно зменшити витрати на утримання офісу для бізнесу [1].

1.2 Переваги хмарних технологій у розробці та експлуатації систем моніторингу

Хмарні обчислення передбачають використання віддалених серверів в Інтернеті для зберігання таких ресурсів, як програмне забезпечення, дані, сховище, мережі та електронні листи. Шифруванням даних, обробкою та керуванням займається хмарний провайдер, який також здійснює аналітику даних і звітність.

Хмарні обчислення мають необмежені можливості зберігання та обчислювальну потужність, що дозволяє малим підприємствам конкурувати на одному рівні з великими компаніями. Загалом, вони використовується за потреби за моделлю PAYG; користувачі платять лише за те, що вони використовують, і можуть зменшувати або збільшувати відповідно до потреб ринку та клієнтів.

Хмарні платформи надають можливість зберігати величезний обсяг даних в онлайн-сховищах, доступ до яких можна отримати з будь-якого місця з підключенням до Інтернету. Це дозволяє адміністраторам моніторингу отримувати доступ до даних навіть здалеку.

На рис. 1.1 зображені переваги хмарних технологій.

Хмарні системи легко масштабуються залежно від потреб користувача. Це означає, що систему можна розширити або зменшити, не витрачаючи час і ресурси на розгортання нового обладнання або інфраструктури.

За допомогою хмарних технологій можна швидко налаштовувати та адаптувати системи моніторингу відповідно до змінних вимог та умов. Наприклад, можна легко додавати нові сенсори або моніторингові точки без необхідності значних витрат часу і ресурсів. Багато хмарних платформ надають інструменти для автоматизації процесів моніторингу. Це може включати автоматичне оповіщення про події, автоматичне масштабування ресурсів або автоматичне виявлення аномальних змін у даних.

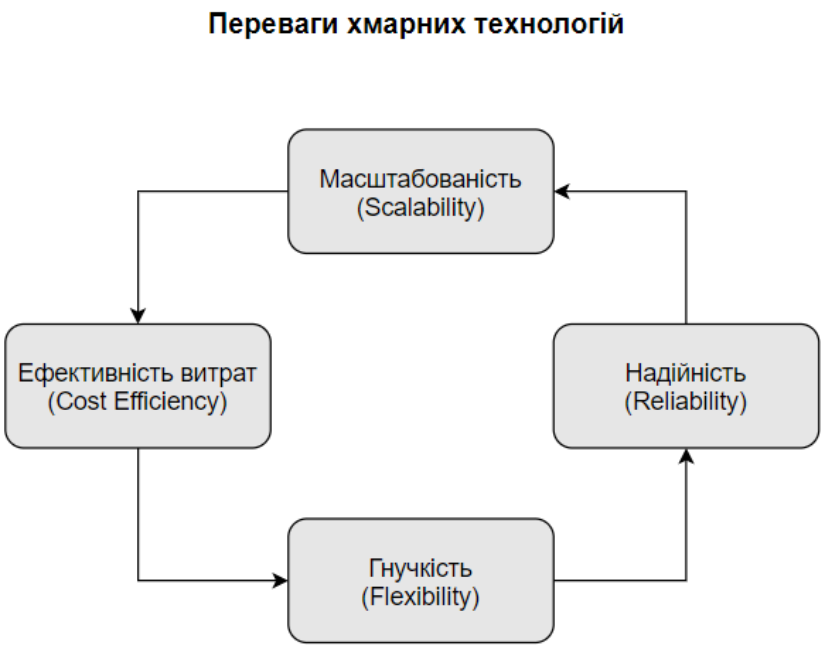

Рисунок 1.1 – Переваги хмарних технологій

Великі хмарні платформи зазвичай мають високий рівень захисту даних, включаючи резервне копіювання, шифрування та інші заходи безпеки. Це дозволяє забезпечити безпеку даних моніторингу від несанкціонованого доступу та втрат.

Використання хмарних технологій дозволяє уникнути значних витрат на придбання, обслуговування та підтримку власної інфраструктури. Користувачі можуть платити лише за фактично використані ресурси, що знижує витрати.

Хмарні технології забезпечують можливість спільного доступу до даних між різними користувачами та відділами, що сприяє співпраці та координації роботи.

Загалом, хмарні технології роблять системи моніторингу більш доступними, гнучкими, ефективними та безпечними, що дозволяє підприємствам ефективно використовувати дані для прийняття рішень [2].

Для досягнення поставленої мети буде використовуватись хмарна платформа від компанії Microsoft – Azure. Microsoft Azure, раніше відома як Windows Azure, є публічною платформою хмарних обчислень Microsoft. Вона

надає широкий спектр хмарних послуг, включаючи обчислення, аналітику, зберігання даних та мережеві послуги. Користувачі можуть вибирати з цих служб для розробки та масштабування нових програм або запуску існуючих програм у загальнодоступній хмарі.

Платформа Azure має на меті допомогти компаніям впоратися з проблемами та досягти організаційних цілей. Вона пропонує інструменти, які підтримують усі галузі, включаючи електронну комерцію, фінанси та різноманітні компанії зі списку Fortune 500, і сумісний із технологіями з відкритим кодом. Це дає користувачам гнучкість у використанні своїх бажаних інструментів і технологій. Крім того, Azure пропонує чотири різні форми хмарних обчислень: інфраструктура як послуга (IaaS), платформа як послуга (PaaS), програмне забезпечення як послуга (SaaS) і безсерверні функції.

Корпорація Майкрософт стягує плату за Azure на основі оплати за використання (PAYG), тобто передплатники щомісяця отримують рахунок лише за певні ресурси та послуги, якими вони користуються. Така модель допомагає бізнесу не витрачати зайві ресурси [3].

Як тільки клієнти зареєструються на Azure, вони отримають доступ до всіх послуг, які містяться на порталі цієї платформи. Користувачі можуть використовувати ці служби для створення хмарних ресурсів, таких як віртуальні машини, бази даних, віртуальні мережі та ін. Тоді ресурси та служби можна зібрати в робочі середовища, які використовуються для розміщення робочих навантажень і зберігання даних.

1.3 Методи збору, аналізу та візуалізації даних про стан пристроїв в мережі

Для отримання інформації від офісних пристроїв таких як принтери, сканери і т.д буде використовуватися інструмент керування Windows (WMI) – це інфраструктура для керування даними й операціями в операційних системах Windows. WMI повністю підтримується Microsoft. Остання версія

адміністративних сценаріїв і керування доступна через інфраструктуру керування Windows. WMI надає користувачам інформацію про стан локальних або віддалених комп'ютерних систем.

Метою WMI є допомога адміністраторам керувати різними операційними середовищами Windows, включаючи віддалені системи. Одна велика перевага WMI полягає в тому, що вона зменшує обслуговування та витрати на керування компонентами корпоративної мережі. WMI – це реалізація Microsoft ініціативи Web-Based Enterprise Management (WBEM) для підтримуваних платформ Windows. WBEM – це загальна ініціатива з розробки стандартів інфраструктури керування для доступу та поєднання інформації з різних апаратних і програмних систем керування в IT-середовищі підприємства.

WBEM побудовано на схемі загальної інформаційної моделі (CIM), стандарті комп’ютерної промисловості для визначення характеристик пристроїв і програм. CIM дозволяє системним адміністраторам і програмам керування контролювати пристрої та програми від багатьох виробників або джерел.

Для кліматичної техніки: використання сенсорів для збору даних про температуру, вологість, рівень CO2 тощо. Інтеграція зі системами управління кліматом для отримання інформації про статус та параметри роботи пристроїв.

Аналіз роботи пристроїв може включати безліч сценаріїв:

– для пристроїв роботи з документами – частота використання пристроїв та їх функцій для виявлення неефективності використання або можливостей для оптимізації;

– для кліматичної техніки – моніторинг змін температури, вологості та інших параметрів для виявлення аномалій та можливих проблем з обслуговуванням.

Візуалізація даних може відбуватися відбуватися в окремому додатку в якому використовуються бібліотеки для роботи з графіками або через Grafana. Додаток для моніторингу включатиме в себе потрібні дані для побудови графіків

Для зберігання великих обсягів даних про стан пристроїв буде використовуватися Azure Storage. Azure Storage пропонує високодоступне, масштабоване, довговічне та безпечне сховище для різноманітних об’єктів даних у хмарі. Об’єкти даних Azure Storage доступні з будь-якої точки світу через HTTP або HTTPS через REST API. Платформа Azure Storage включає в себе багато різних служб для зберігання даних. В додатку буде використовуватися Azure Tables - сховище NoSQL для безсхемного зберігання структурованих даних [4].

1.4 Методи доступу до локальної мережі через хмару

В нашому випадку буде розроблено додаток, який буде включати в себе бекенд частину – API. Додаток буде знаходитися в хмарі в сервісі PaSS від Azure – App Service. Але йому також потрібно мати доступ до локальної мережі підприємства, для досягнення цієї мети є декілька способів.

1. Hybrid Connection. Azure App Service має функціональність, відому як Hybrid Connections. Ми можемо налаштувати Hybrid Connection Manager на своєму локальному сервері, а потім на Azure Portal налаштувати це з'єднання з нашим App Service. Це дозволить додатку взаємодіяти з ресурсами в локальній мережі через це з'єднання [5]. На рис. 1.2 представлена схема підключення додатку з хмари до локальної мережі.

2. Virtual Network Integration (рис. 1.3). Якщо локальна мережа використовує віртуальні мережі Azure (Azure Virtual Network), можна налаштувати інтеграцію App Service з цією віртуальною мережею. Це дозволить додатку з'єднуватися з ресурсами у локальній мережі через віртуальний приватний тунель [6].

3. VPN або ExpressRoute: Якщо у підприємства є VPN або ExpressRoute з'єднання між локальною мережею і Azure, можна сконфігурувати Azure App Service, щоб він був доступний через це з'єднання [7]. На рис. 1.4 наведений приклад підключення додатку до локальної мережі через ExpressRoute.

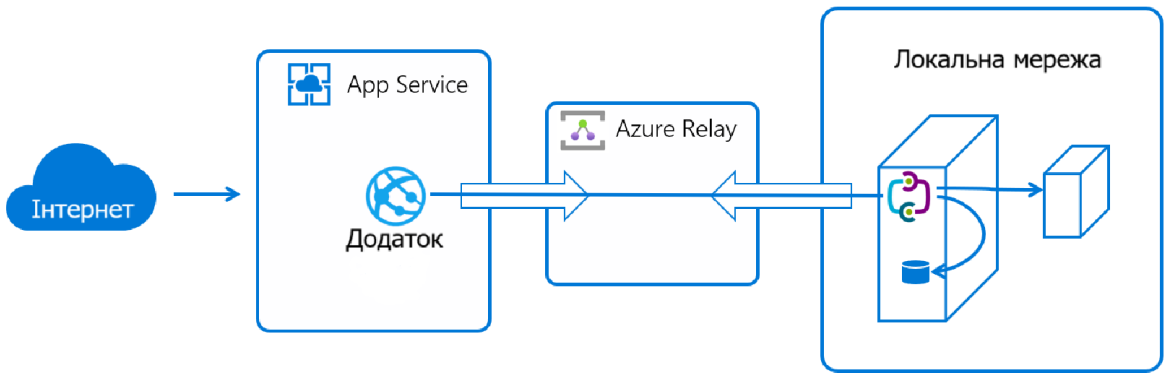

Рисунок 1. 2 – Схема підключення додатку з хмари до локальної мережі використовуючи Hybrid Connections

Рисунок 1.3 – Схема підключення додатку з хмари до локальної мережі з використанням Virtual Network Integration

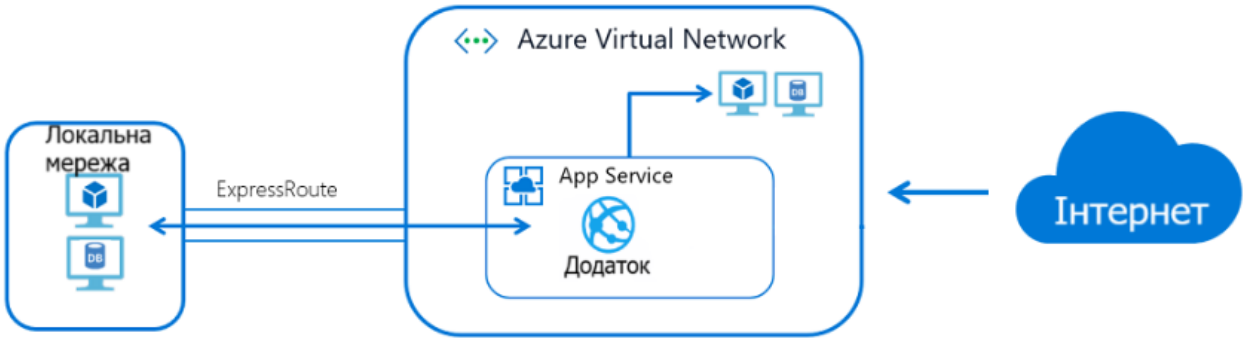

Рисунок 1.4 – Схема підключення додатку з хмари до локальної мережі з використанням ExpressRoute

Для досягнення мети буде використовуватися дві віртуальні мережі, одна з них буде імітувати локальну. Тому буде використовуватися наступний підхід через VPN між двома віртуальними мережами. В імітованій мережі буде знаходитись віртуальна машина, яка в свою чергу буде імітувати головний комп'ютер адміністратора в мережі (рис. 1.5).

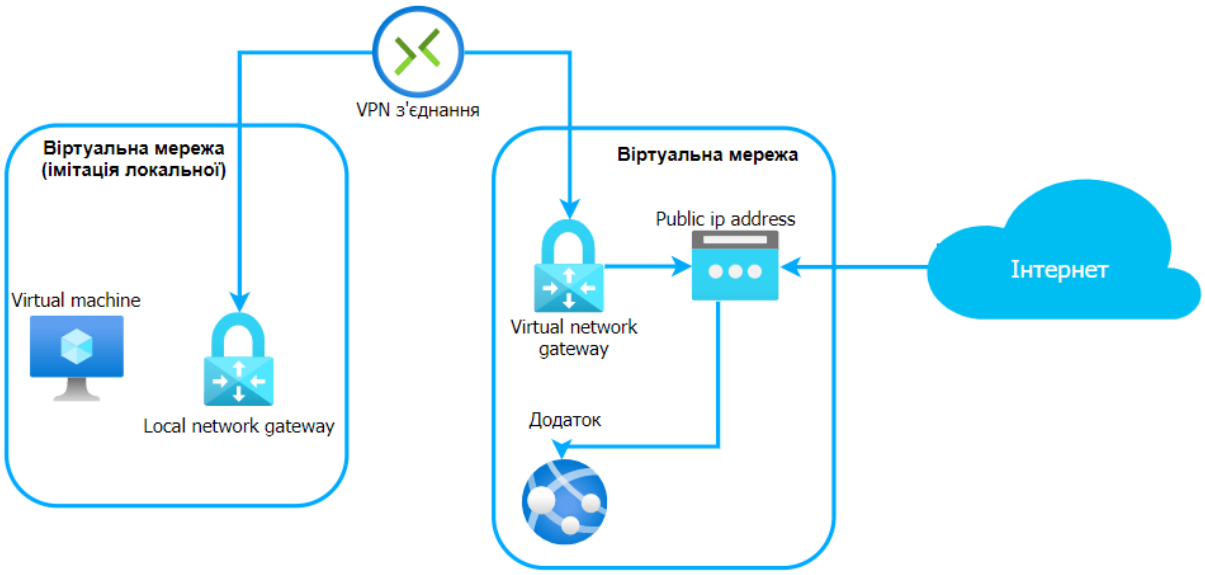

Рисунок 1.5 – Схема підключення з використанням імітації локальної мережі віртуальною мережею

2 СТВОРЕННЯ ХМАРНОЇ ІНФРАСТРУКТУРИ

2.1 Створення віртуальної мережі та APP Service для хостингу додатку

Віртуальна мережа Azure (VNet) – це служба, яка є основним стандартним блоком для приватної мережі в Azure. Екземпляр служби (віртуальна мережа) дозволяє багатьом типам ресурсів Azure безпечно взаємодіяти один з одним, Інтернетом та локальними мережами. До цих ресурсів Azure також належать віртуальні машини. Віртуальна мережа схожа на традиційну мережу, яка могла б працювати у власному центрі обробки даних. Але це дає додаткові переваги інфраструктури Azure, такі як масштабування, доступність та ізоляція.

За замовчуванням усі віртуальні мережі можуть обмінюватися вихідними даними через Інтернет. Ви також можете використовувати загальнодоступну IP-адресу, шлюз NAT або загальнодоступну підсистему балансування навантаження для керування вихідними підключеннями. Ви можете обмінюватися даними з ресурсом, призначивши загальнодоступну IP-адресу або загальнодоступну підсистему балансування навантаження.

При використанні лише внутрішньої підсистеми балансування навантаження рівня "Стандартний" вихідні підключення недоступні, доки ви не визначите, як вихідні підключення повинні працювати з загальнодоступною IP-адресою рівня екземпляра або загальнодоступною підсистемою балансування навантаження.

Для фільтрації мережного трафіку між підмережами можна використовувати один або два з таких параметрів.

Групи безпеки мережі та групи безпеки програм можуть містити декілька правил безпеки для вхідного та вихідного трафіку. Ці правила дозволяють фільтрувати трафік у ресурси та з них за вихідною та цільовою IP-адресою, портом та протоколом.

Для створення VNet у порталі Azure необхідно виконати наступні кроки:

1) перейти до Azure Portal,
2) обрати "Create a resource",
3) вибрати «Networking» > «Virtual Network» (рис. 2.1).

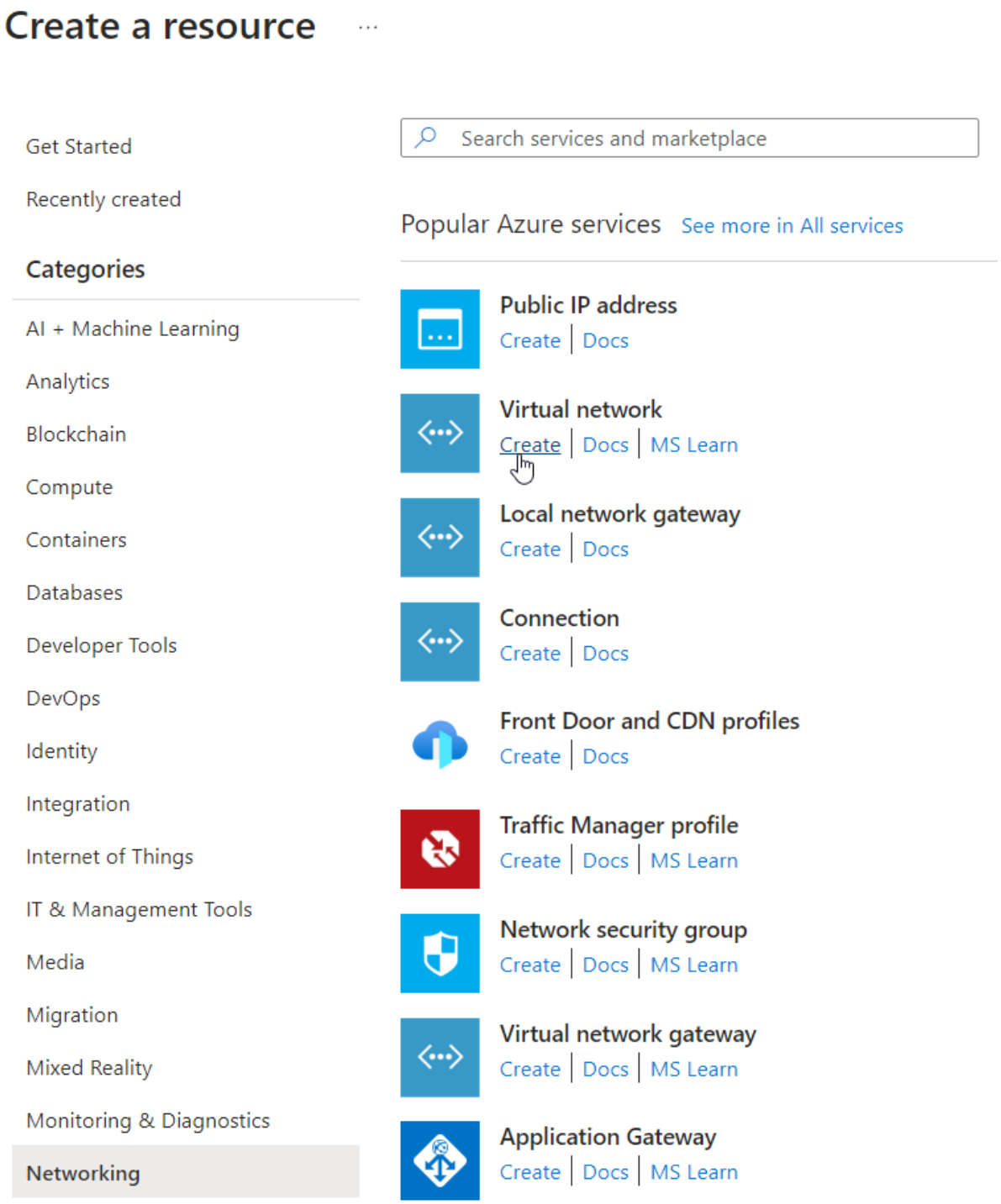

Рисунок 2.1 – Вікно створення ресурсів Azure

Заповнити поля (рис. 2.2):

1) Subscription: підписка, в рамках якої буде створено VNet,
2) Resource group: група ресурсів для організації ваших ресурсів,
3) Name: ім'я віртуальної мережі,
4) Location: регіон, в якому буде розміщений VNet.

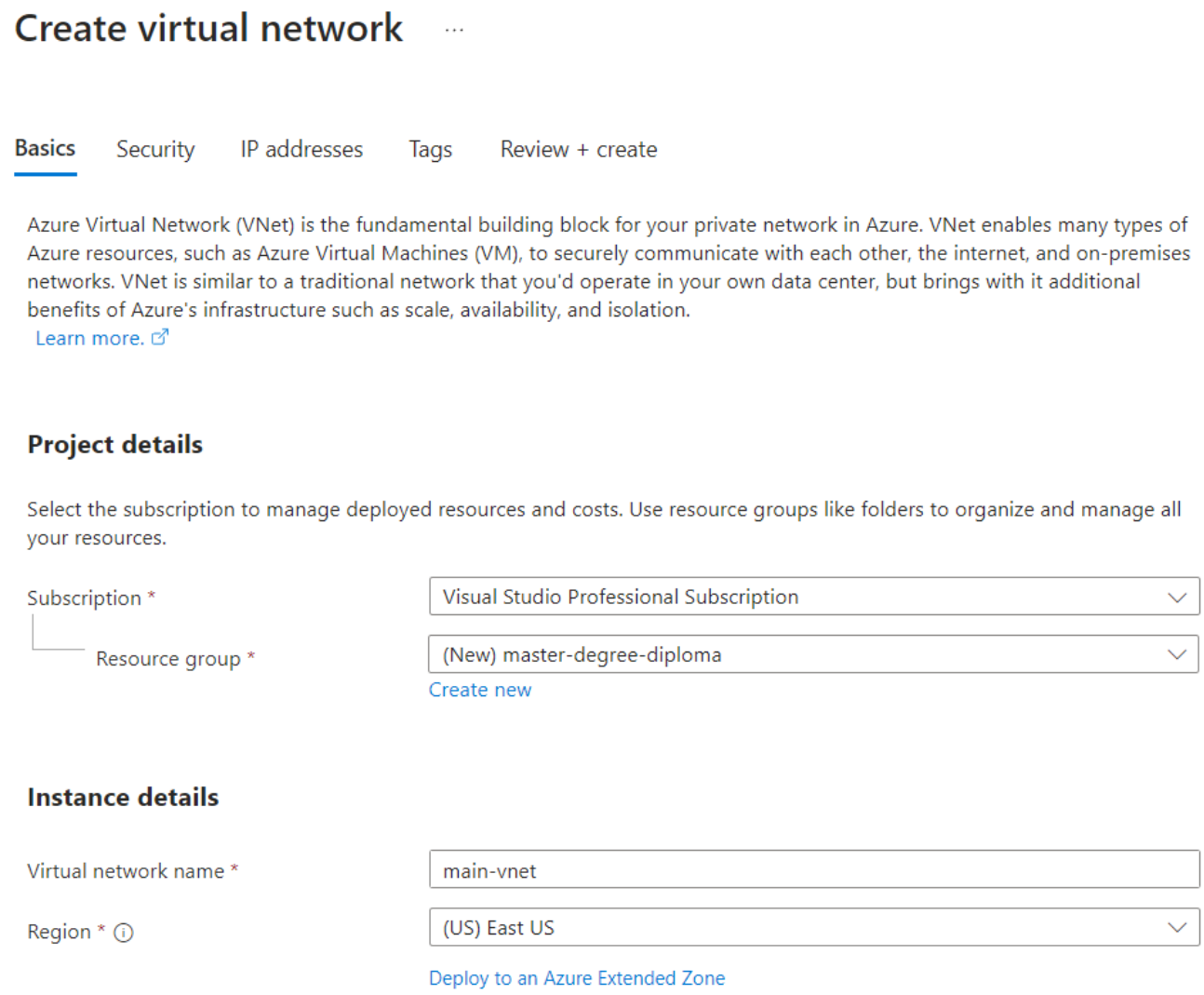

Рисунок 2.2 – Створення віртуальної мережі на порталі Azure

Для налаштувати підмережі необхідно вказати ім'я та діапазон IP-адрес для кожної під мережі (рис. 2.3).

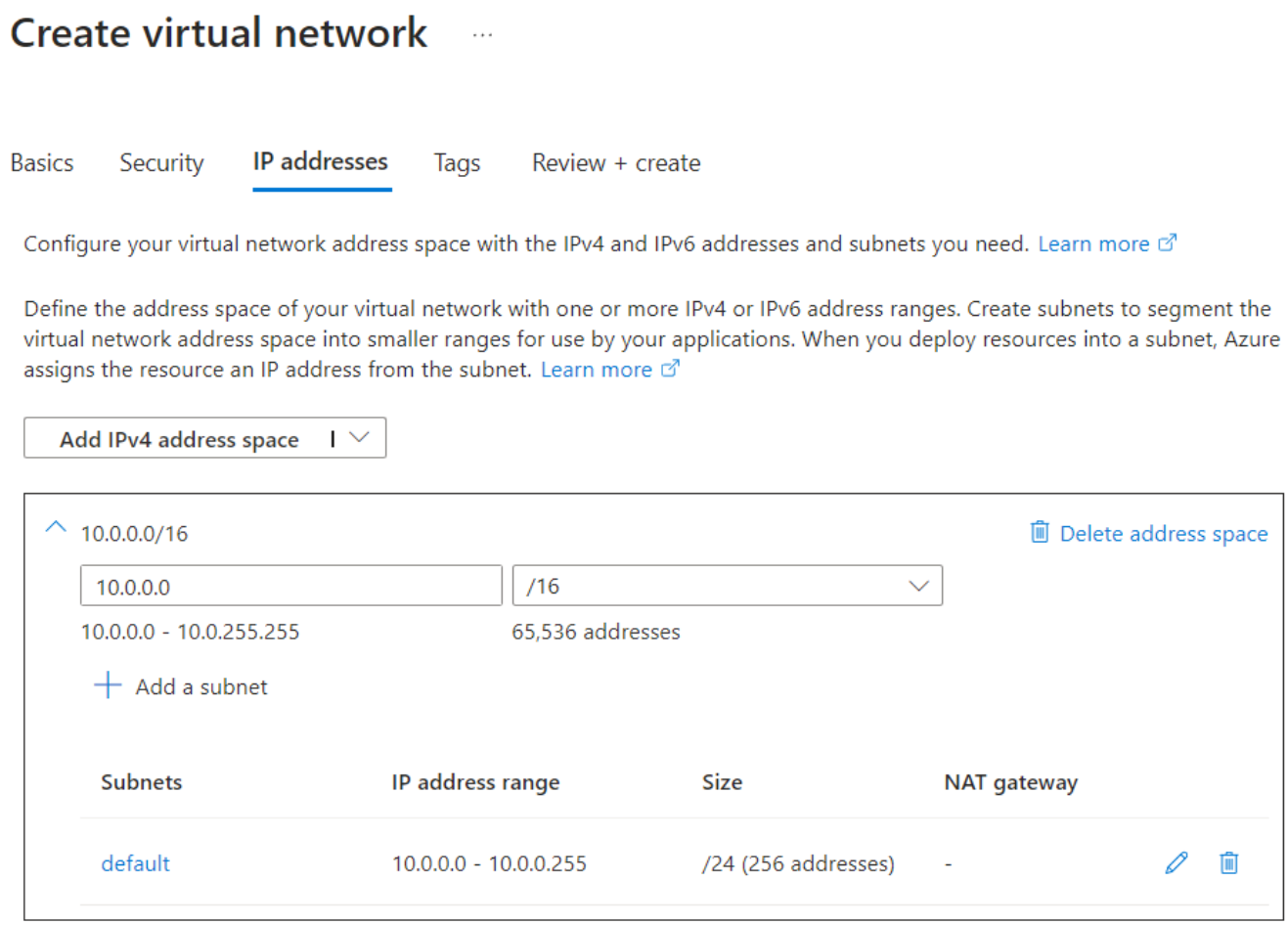

Рисунок 2.3 – Діапазон IP-адрес віртуальної мережі

Для того, щоб ця віртуальна мережа доступ мала доступ до інтернету, потрібно створити загальнодоступну IP-адресу (Public IP address) (рис. 2.4). Загальнодоступні IP-адреси дозволяють інтернет-ресурсам обмінюватися вхідними даними з ресурсами Azure. Загальнодоступні IP-адреси дозволяють ресурсам Azure спілкуватися з Інтернетом і загальнодоступними службами Azure. Ресурс без загальнодоступної IP-адреси може обмінюватися вихідними даними. Azure динамічно призначає доступну IP-адресу, яка не призначена для ресурсу.

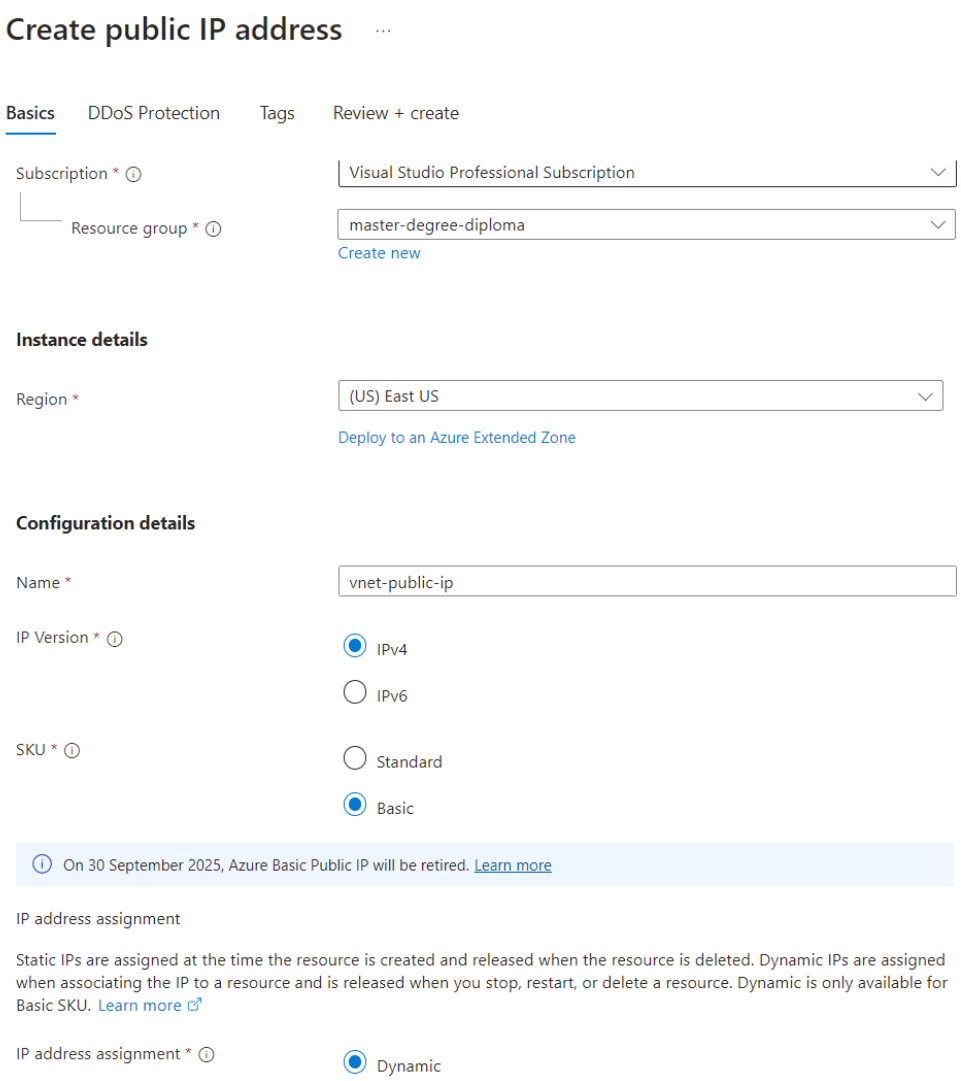

Рисунок 2.4 – Створення загальнодоступної IP-адреси

Для налаштування VPN з’єднання між двома мережами потрібно створити VPN шлюз (Gateway) . VPN-шлюз Azure – це служба, яку можна використовувати для надсилання зашифрованого трафіку між віртуальною мережею Azure і локальними розташуваннями через загальнодоступний Інтернет. Використовується шлюз VPN також для надсилання зашифрованого трафіку між віртуальними мережами Azure через мережу Microsoft, що і потрібно в нашому випадку. До одного шлюзу VPN можна створити кілька підключень. Коли ви створюєте кілька з’єднань, усі тунелі VPN спільно використовують доступну пропускну здатність шлюзу, яка залежить від обраної ліцензії та покоління. В даному випадку буде використовуватись перше

покоління (Generation 1) та тип ліцензії VpnGw1. В табл. 2.1 наведено характеристики різних ліцензій для першого покоління.

Таблиця 2.1 - Характеристики різних ліцензій для першого покоління

Покоління	Ліцензія	Кількість тунелів між мережами	Пропускна здатність	Максимальна к-ть віртуальних машин в мережі
Generation1	Basic	Max. 10	100 Mbps	200
Generation1	VpnGw1	Max. 30	650 Mbps	450
Generation1	VpnGw2	Max. 30	1 Gbps	1300
Generation1	VpnGw3	Max. 30	1.25 Gbps	4000
Generation1	VpnGw1AZ	Max. 30	650 Mbps	1000
Generation1	VpnGw2AZ	Max. 30	1 Gbps	2000
Generation1	VpnGw3AZ	Max. 30	1.25 Gbps	5000

Перед створенням VPN шлюзу потрібно створити ще одну підмережу в віртуальній мережі, яка була створена раніше, для нього (рис. 2.5).

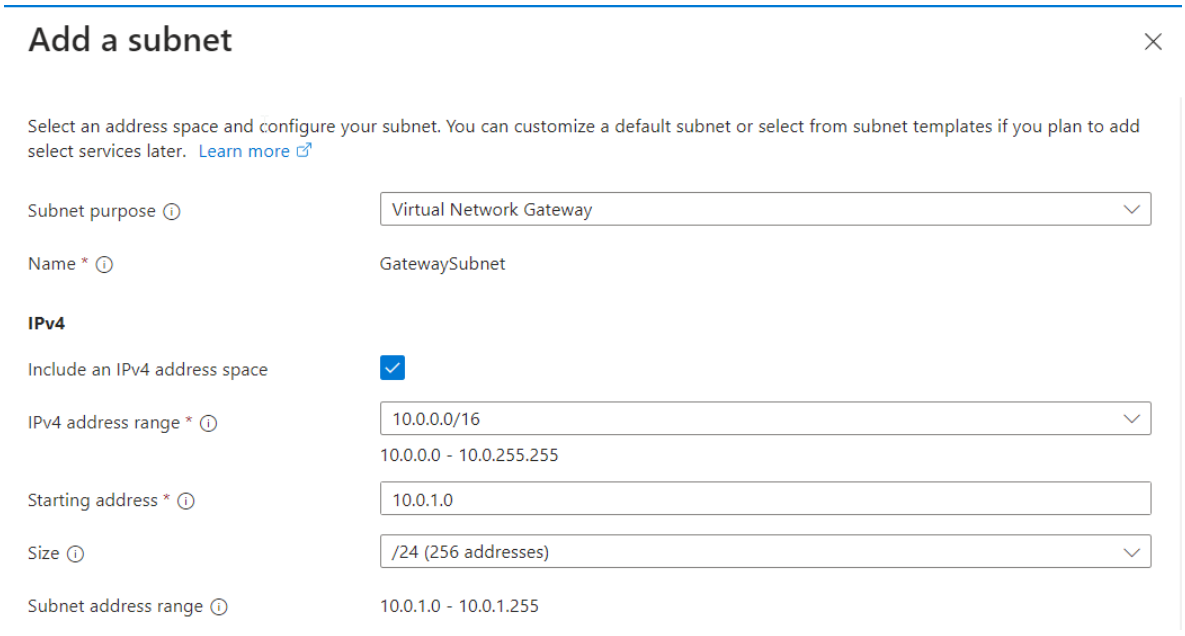

Рисунок 2.5 – Створення підмережі для VPN шлюзу

Тепер можна перейти до створення самого шлюзу (рис. 2.6). Потрібно ввести ім’я для шлюзу, обрати регіон. Вказати який тип шлюзу та обрати тип ліцензії та покоління. Наступним кроком є вибір віртуальної мережі, яка була створена раніше, підмережа підставляється автоматично. Останнім кроком є вибір загальнодоступної IP-адреси, яка в свою чергу теж була створена раніше.

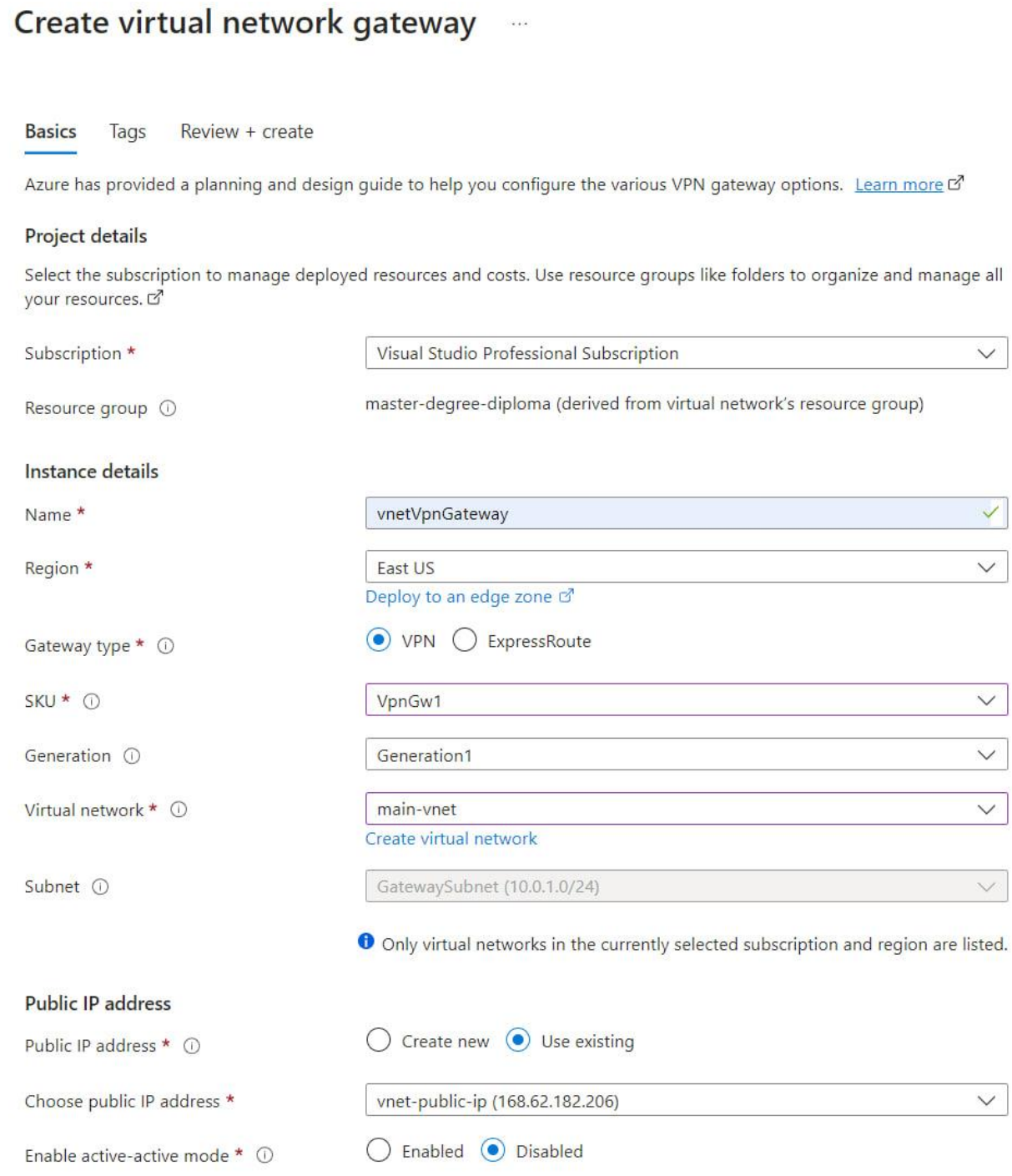

Рисунок 2.6 – Створення VPN шлюзу для віртуальної мережі

Для хостингу додатку створимо ресурс App Services (Служба додатків). Служба додатків Azure – це служба на основі HTTP для розміщення веб-додатків, REST API і мобільних серверів. Ви можете розробляти на своїй улюбленій мові, будь то .NET, .NET Core, Java, Node.js, PHP і Python. Програми запускаються та легко масштабуються в середовищах Windows і Linux.

Служба додатків додає можливості Microsoft Azure до вашої програми, зокрема безпеку, балансування навантаження, автомасштабування та автоматизоване керування. Крім того, ви можете скористатися його можливостями DevOps, такими як безперервне розгортання з Azure DevOps, GitHub, Docker Hub та інших джерел.

З App Service ви платите за обчислювальні ресурси Azure, які використовуєте. Обчислювальні ресурси, які ви використовуєте, визначаються планом служби додатків, на якому ви запускаєте свої програми.

Для створення переходимо на портал там обираємо App Services. Після натискання на кнопку “Create” обираємо “Web App” (рис. 2.7).

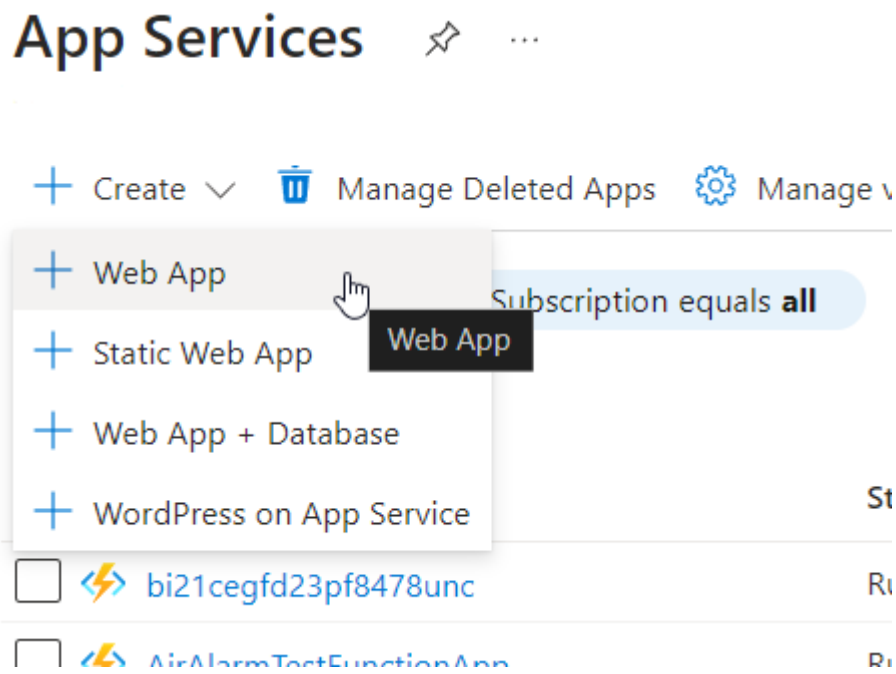

Рисунок 2.7 – Створення App Services Web App для хостингу додотку

Оскільки додаток буде розроблятися на платформі .NET то вказуємо середовище виконання (Runtime) .NET 8 та операційну систему Windows (рис. 2.8).

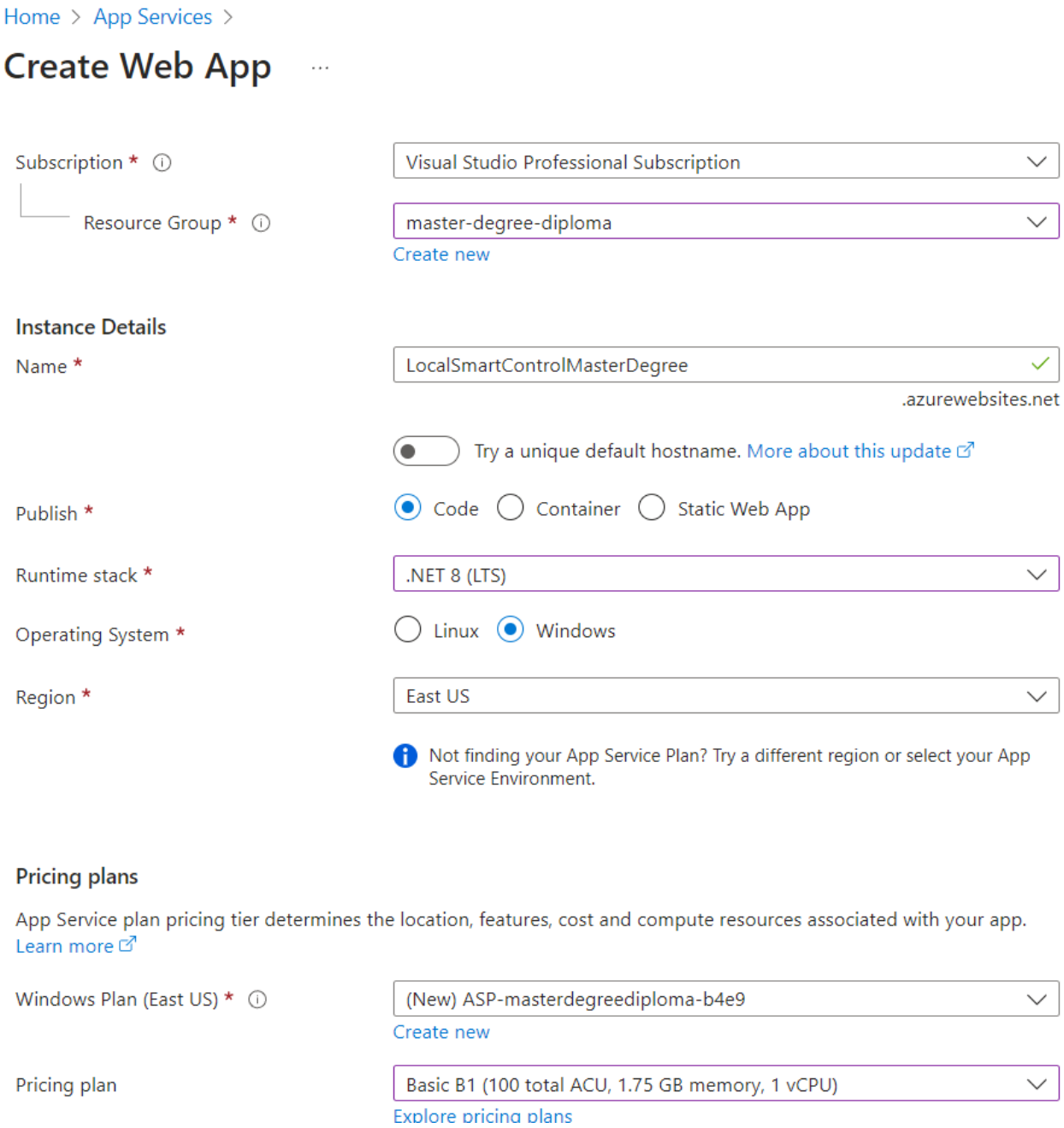

Рисунок 2.8 – Налаштування параметрів перед створенням Web App

Наступним кроком після створення App Services Web App є інтеграція з віртуальною мережею (рис. 2.9). Завдяки інтеграції з віртуальної мережі Azure з

App Service можливий доступ до приватних ресурсів із програми у віртуальній мережі.

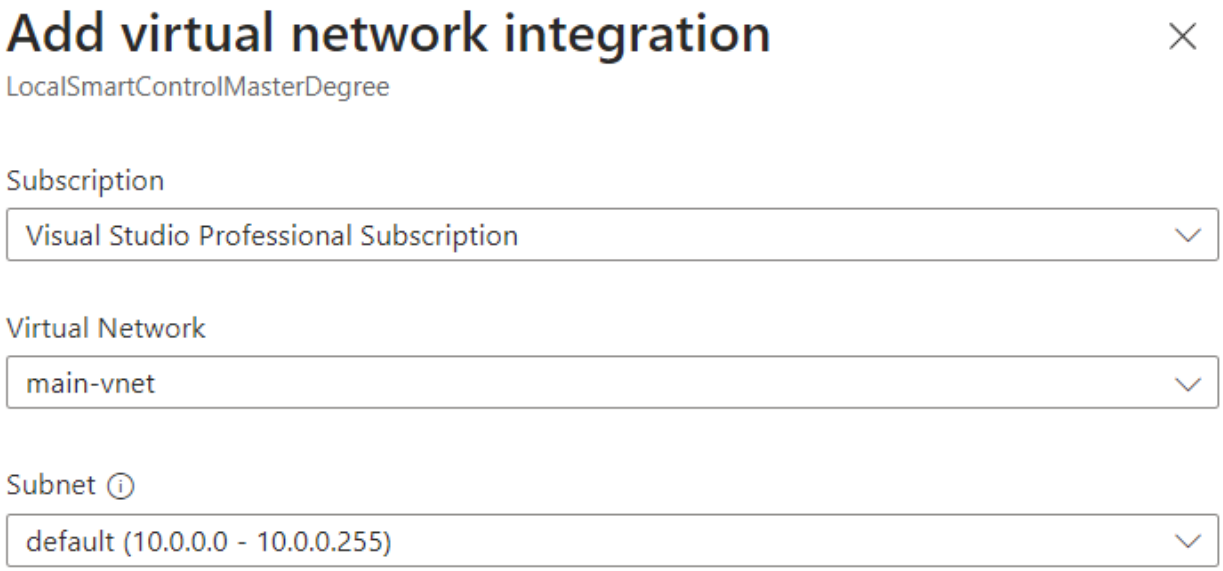

Рисунок 2.9 – Створення інтеграції між додатком та віртуальною мережею

Для створення інтеграції переходимо до вкладки “Networking” на порталі App Service. У розділі “Outbound traffic configuration” трафіку обираємо “Virtual network integration” в полі “Virtual Network” обираємо “main-vnet”.

2.2 Створення імітації локальної мережі з комп’ютером адміністратора

Оскільки в рамках дослідження буде використовуватись віртуальна мережа з імітацією локальної. То процес створення цієї мережі буде майже ідентичний до створення попередньої. Але ще потрібно створити підмережу для віртуальної машини, яка буде в ролі комп’ютера адміністратора (рис. 2.10).

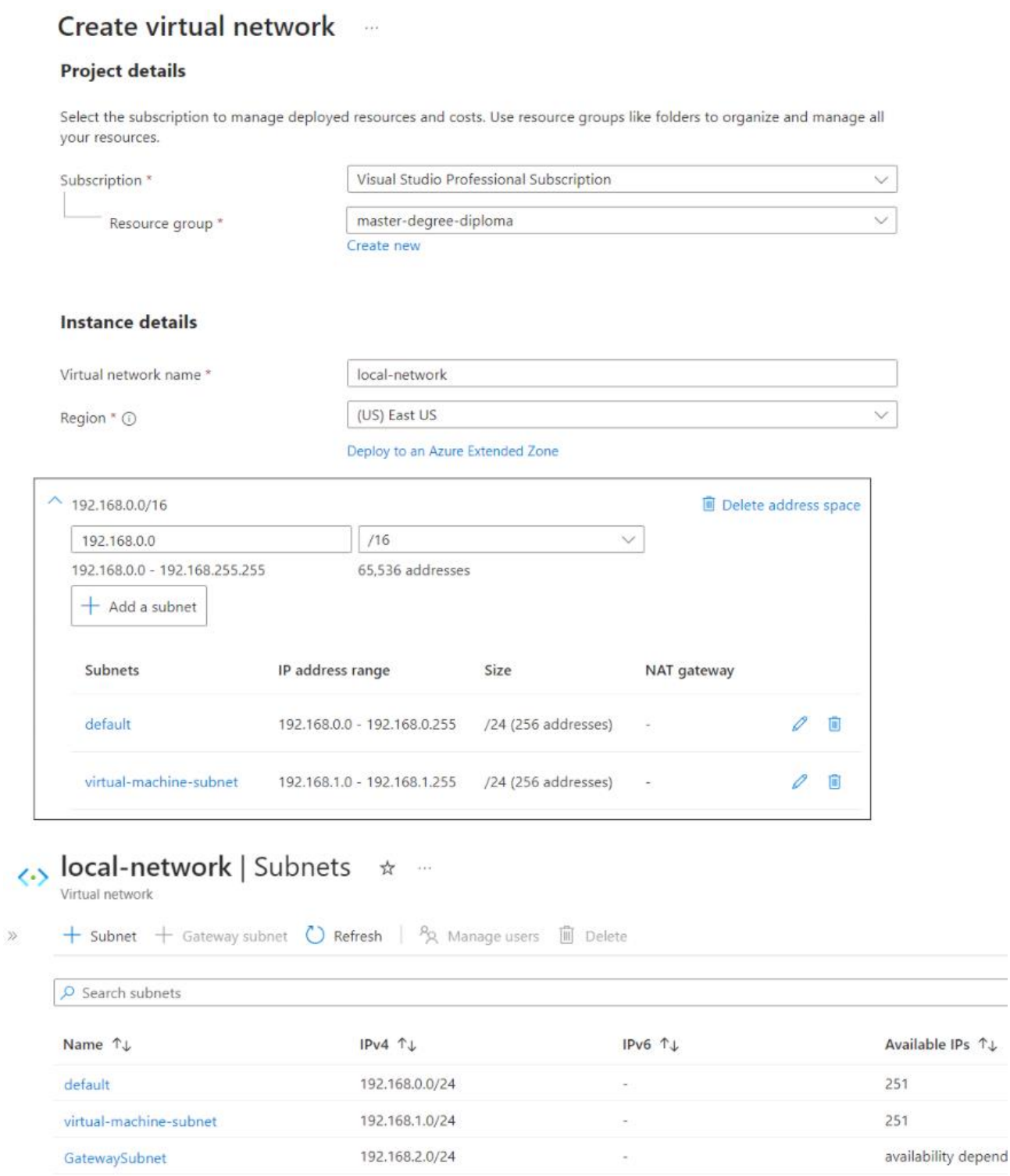

Рисунок 2.10 – Створення “локальної” віртуальної мережі з підмережою для віртуальної машини

Наступним кроком створюємо загальнодоступну IP-адресу для подальшого створення VPN з'єднання між двома віртуальними мережами (рис. 2.11).

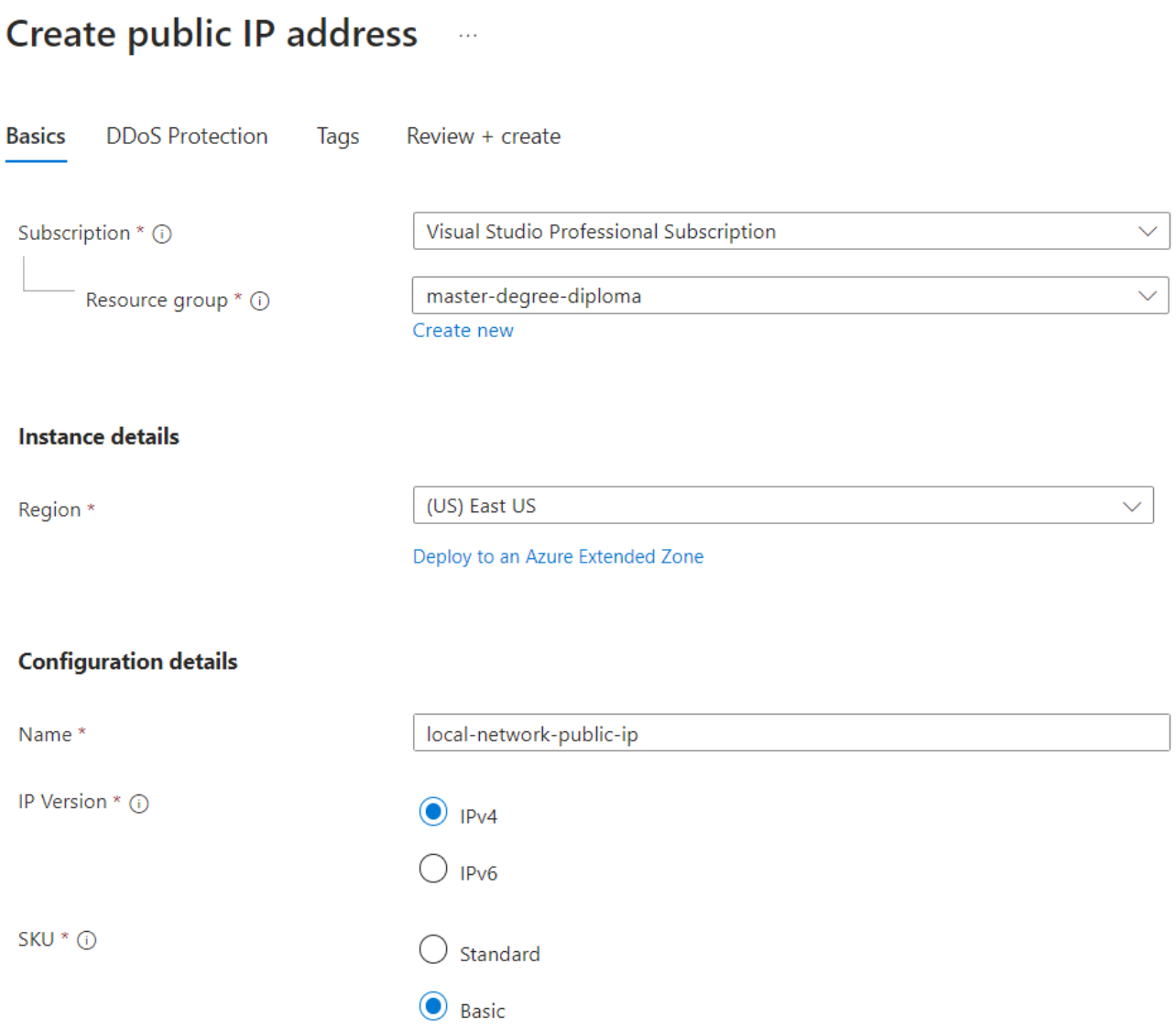

Рисунок 2.11 – Створення загальнодоступної IP-адреси "local-network-public-ip"

За аналогією з попередньою мережею створюємо VPN шлюз (рис. 2.12).

Віртуальні машини Azure – це один із кількох типів масштабованих обчислювальних ресурсів на вимогу, які пропонує Azure. Як правило, ви вибираєте віртуальну машину, коли вам потрібен більший контроль над обчислювальним середовищем, ніж пропонують інші варіанти.

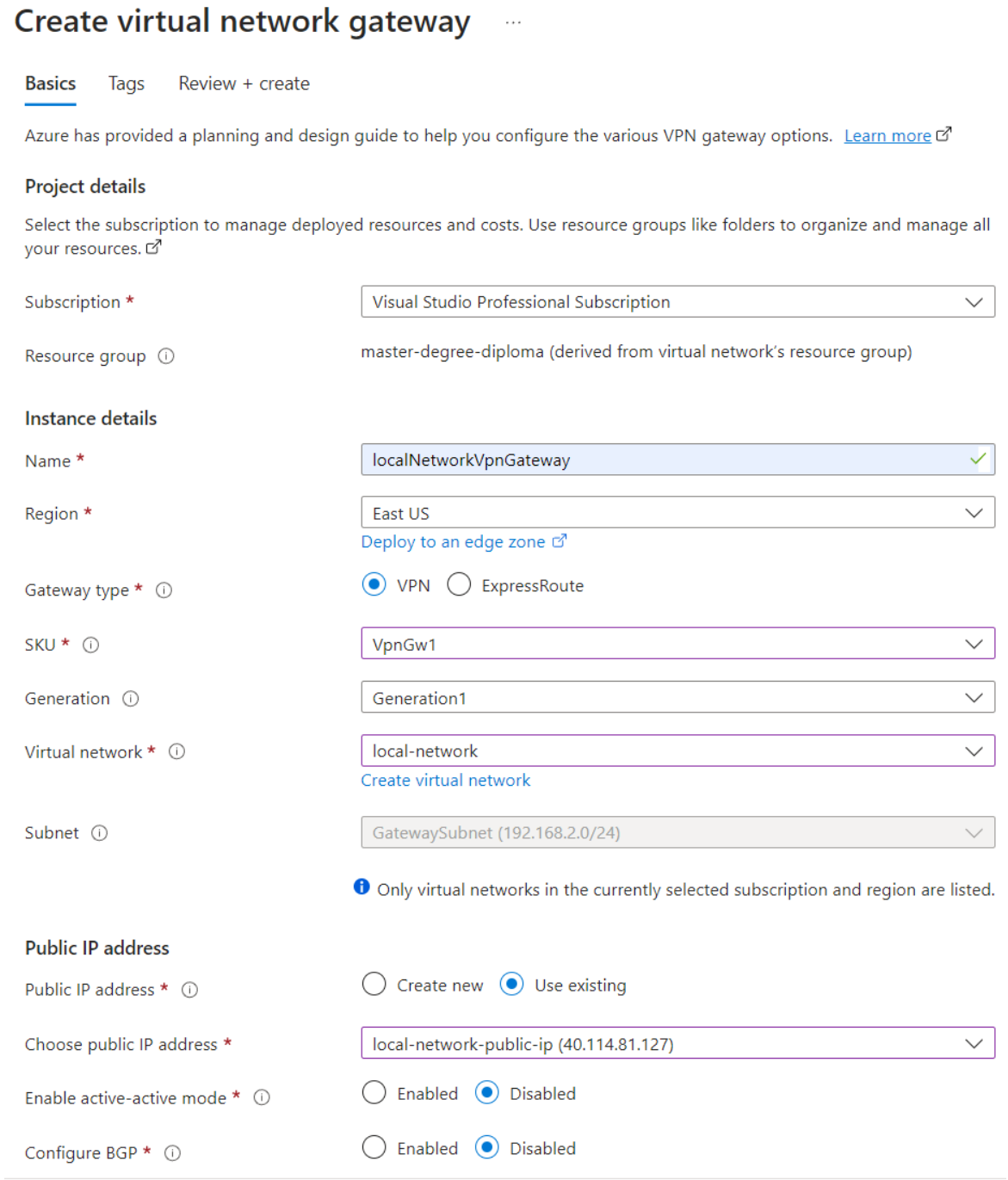

Рисунок 2.12 – Створення VPN шлюзу для “локальної” мережі

Мережевий інтерфейс дозволяє віртуальній машині Azure спілкуватися з Інтернетом, Azure і локальними ресурсами. Під час створення віртуальної машини за допомогою порталу Azure портал створює для неї один мережевий інтерфейс із параметрами за замовчуванням.

Для створення віртуальної машини на порталі знаходимо “Virtual Machines” та створюємо з потрібними характеристиками (рис. 2.13).

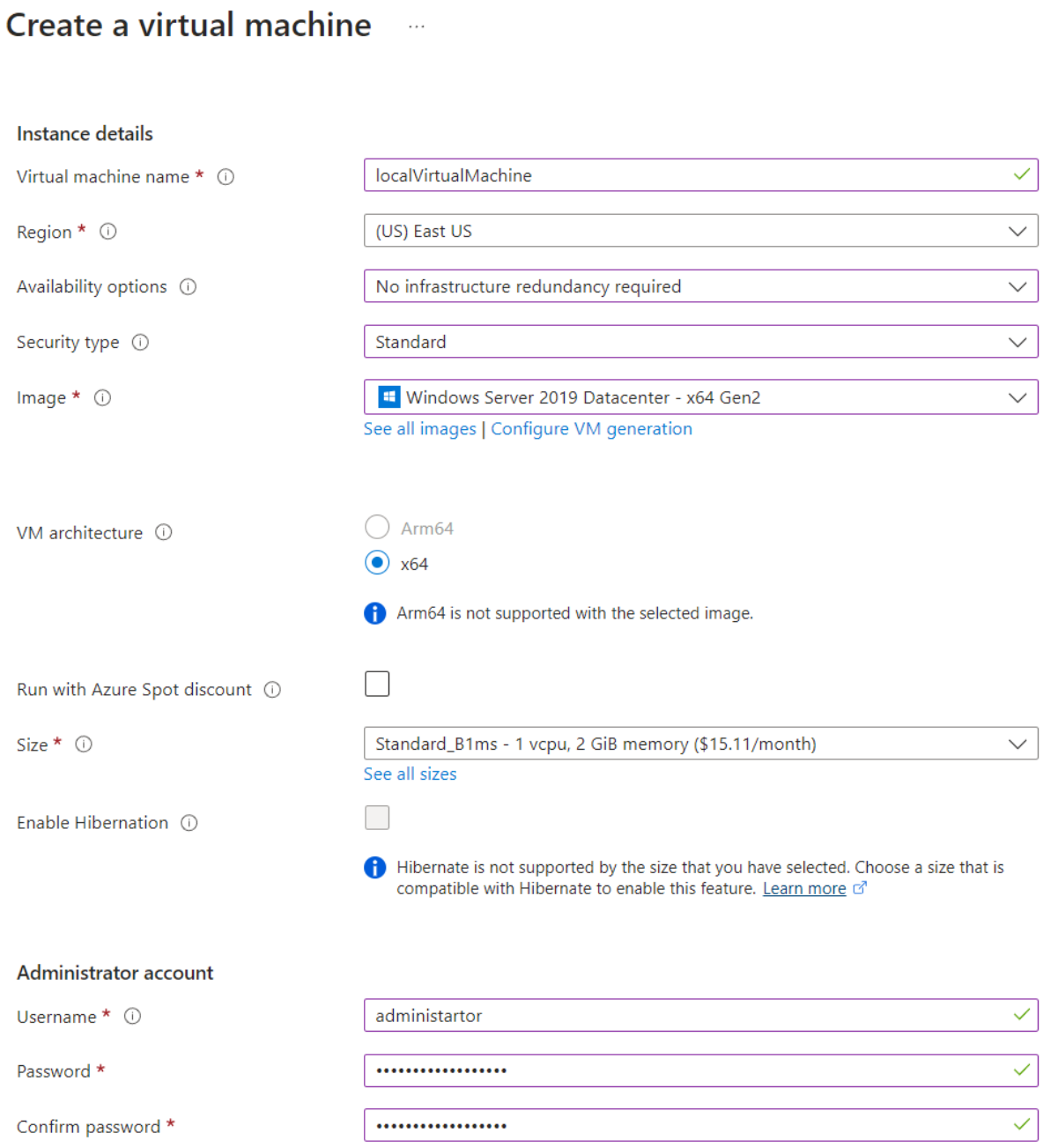

Рисунок 2.13 – Створення віртуальної машини

В налаштування мережі для віртуальної мережі вказуємо раніше створену мережу та підмережу для неї (рис. 2.14).

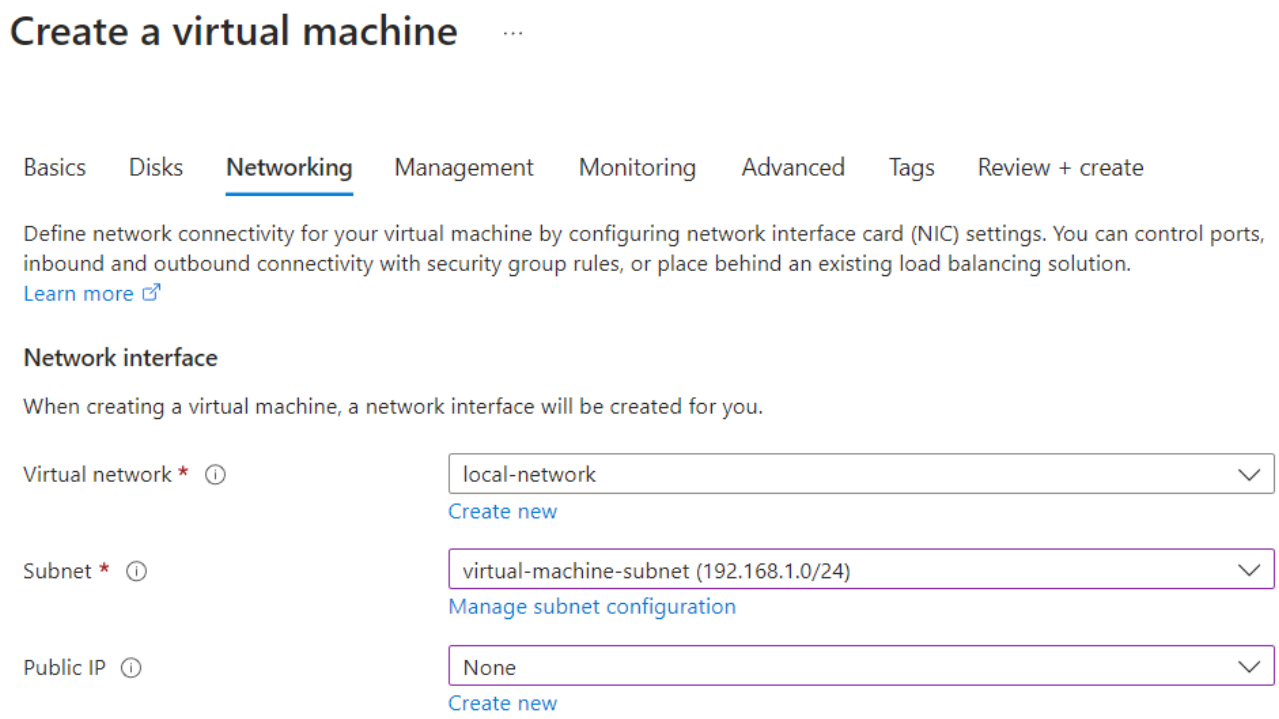

Рисунок 2.14 – Мережеві налаштування для віртуальної машини

2.3 Створення VPN з’єднання між мережами

Наступним етапом буде створення VPN з’єднання між двома раніше створеними мережами. VPN, що розшифровується як віртуальна приватна мережа, встановлює цифрове з’єднання між вашою мережею, обладнанням і т.д. з віддаленим сервером, що належить постачальнику VPN, створюючи тунель «точка-точка», який шифрує ваші особисті дані, маскує вашу IP-адресу. Це гарантує конфіденційність, захист і безпеку вашої роботи в Інтернеті. За самим визначенням VPN-з’єднання це: Virtual (Віртуальна), тому що в процесі підключення не задіяні фізичні кабелі. Private (Приватна), тому що через це з’єднання ніхто інший не зможе побачити ваші дані чи дії веб-переглядача. Network (Мережа), оскільки кілька пристроїв – ваш комп’ютер і сервер VPN – працюють разом, щоб підтримувати встановлений зв’язок. Ця технологія дозволяє користувачам у двох окремих мережах, як у межах організації, так і поза нею, ділитися ресурсами один з одним, обмежуючи при цьому повний

доступ до всіх своїх ресурсів, гарантуючи, що спілкування всередині компанії залишається максимально приватним і безпечним.

Для налаштування VPN з'єднання переходимо на портал Azure та в пошуку вводимо Connection. Натискаємо "Create" та обираємо "Connection-Type" Vnet-to-Vnet. Обираємо регіон та вводимо ім'я для з'єднання (рис. 2.15).

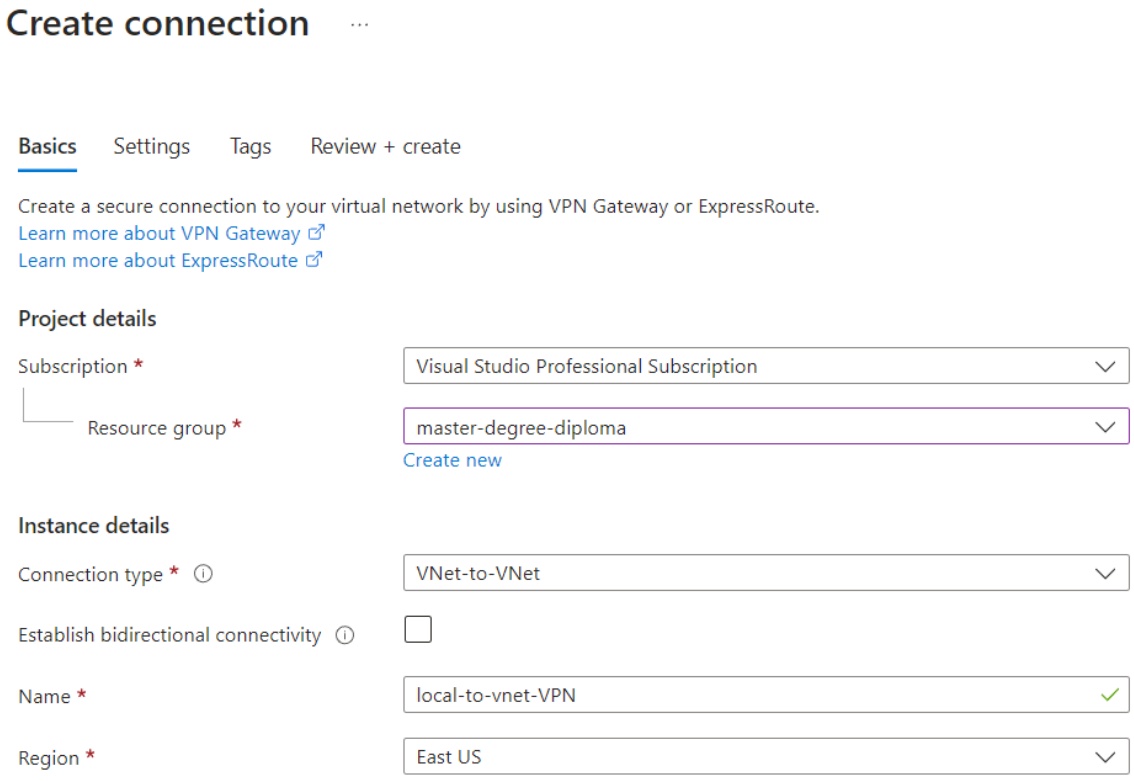

Рисунок 2.15 – Створення VPN з'єднання між мережами

Створюємо спільний ключ (SharedKey) – суміш літер і цифр, які використовуються для встановлення шифрування підключення. Та обираємо дві мережі між якими буде з'єднання (рис. 2.16).

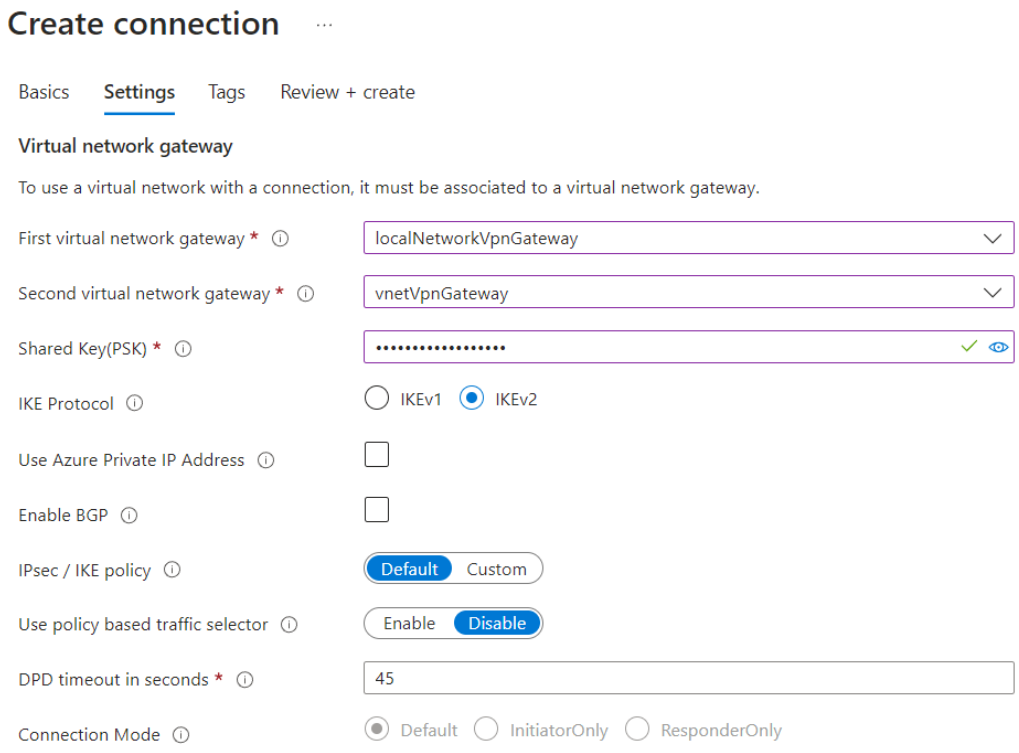

Рисунок 2.16 – Налаштування VPN з’єднання

2.4 Шаблон для створення хмарної інфраструктури мовою Bicep

Для того, щоб спростити процес розгортки всієї інфраструктури буде створено шаблон для розгортання мовою Bicep. Bicep – це предметно-орієнтована мова (DSL), яка використовує декларативний синтаксис для розгортання ресурсів Azure. У файлі Bicep необхідно визначити інфраструктуру, яку потрібно розгорнути в Azure. Цей файл буде використовуватись протягом усього життєвого циклу розробки для багаторазового розгортання інфраструктури. Ресурси розгортаються узгоджено.

Bicep забезпечує короткий синтаксис, надійну безпеку типів та підтримку повторного використання коду. Bicep надає найкращі у своєму класі можливості для розробки рішень Azure типу інфраструктура як код.

Для початку створимо змінну для визначення регіону в якому будуть знаходитися ресурси:

```
param location string = resourceGroup().location
```

Далі створимо зміні, які міститимуть назви Vpn шлюзів, назви мереж, їхні діапазони IP-адрес та підмереж для віртуальної мережні:

```
param primaryVnetName string = 'primaryVnet'
param primaryVnetAddressPrefix string = '10.0.0.0/16'
param primaryGatewaySubnetName string = 'GatewaySubnet'
param primaryGatewaySubnetPrefix string = '10.0.0.0/24'
param primaryGatewayName string = 'primaryVpnGateway'
param primaryGatewaySku string = 'VpnGw1'
param primaryPublicIpName string = 'primaryVpnGatewayPublicIp'
```

Наступні змінні будуть містити в собі цю саму інформацію, але для “локальної” мережі:

```
param onPremiseVnetName string = 'onPremiseVnet'
param onPremiseVnetAddressPrefix string = '192.168.0.0/16'
param onPremiseGatewaySubnetName string = 'GatewaySubnet'
param onPremiseGatewaySubnetPrefix string = '192.168.0.0/24'
param onPremiseSubnetName string = 'onPremiseSubnet' // підмережа для VM
param onPremiseSubnetPrefix string = '192.168.1.0/24'
param onPremiseGatewayName string = 'onPremiseVpnGateway'
param onPremiseGatewaySku string = 'VpnGw1'
param onPremisePublicIpName string = 'onPremiseVpnGatewayPublicIp'
```

Для налаштування VPN з’єднання будуть створені наступні змінні:

```
param connectionName string = 'localToVnetConnection'
param sharedKey string = 'shared-key'
```

Віртаульна машина:

```
param vmName string = 'onPremiseVM'
param vmSize string = 'Standard_B1s'
param adminUsername string = 'azureuser'
param adminPassword string = 'P@ssw0rd1234'
param vmImagePublisher string = 'MicrosoftWindowsServer'
param vmImageOffer string = 'WindowsServer'
param vmImageSku string = '2019-Datacenter'
param vmImageVersion string = 'latest'
```

Створення віртуальної мережі, загальнодоступної IP-адреси та VPN шлюзу:

```
resource primaryVnet 'Microsoft.Network/virtualNetworks@2021-05-01' = {
  name: primaryVnetName
  location: location
  properties: {
    addressSpace: {
      addressPrefixes: [
        primaryVnetAddressPrefix
      ]
    }
    subnets: [
      {
        name: primaryGatewaySubnetName
        properties: {
          addressPrefix: primaryGatewaySubnetPrefix
        }
      }
    ]
  }
}
resource primaryPublicIp 'Microsoft.Network/publicIPAddresses@2021-05-01' = {
```

```
  name: primaryPublicIpName
  location: location
  sku: {
    name: 'Basic'
  }
  properties: {
    publicIPAllocationMethod: 'Dynamic'
  }
}
resource primaryVpnGateway 'Microsoft.Network/virtualNetworkGateways@2021-05-01' =
{
  name: primaryGatewayName
  location: location
  properties: {
    ipConfigurations: [
      {
        name: 'vnetGatewayConfig'
        properties: {
          publicIPAddress: {
            id: primaryPublicIp.id
          }
          privateIPAllocationMethod: 'Dynamic'
          subnet: {
            id: primaryVnet.properties.subnets[0].id
          }
        }
      }
    ]
    gatewayType: 'Vpn'
    vpnType: 'RouteBased'
    enableBgp: false
    sku: {
      name: primaryGatewaySku
```

```
    }
  }
  dependsOn: [
    primaryVnet
    primaryPublicIp
  ]
}
```

Створення “локальної мережі”, загальнодоступної IP-адреси та VPN шлюзу:

```
resource onPremiseVnet 'Microsoft.Network/virtualNetworks@2021-05-01' = {
  name: onPremiseVnetName
  location: location
  properties: {
    addressSpace: {
      addressPrefixes: [
        onPremiseVnetAddressPrefix
      ]
    }
    subnets: [
      {
        name: onPremiseGatewaySubnetName
        properties: {
          addressPrefix: onPremiseGatewaySubnetPrefix
        }
      }
      {
        name: onPremiseSubnetName
        properties: {
          addressPrefix: onPremiseSubnetPrefix
        }
```

```
      }
    ]
  }
}

resource onPremisePublicIp 'Microsoft.Network/publicIPAddresses@2021-05-01' = {
  name: onPremisePublicIpName
  location: location
  sku: {
    name: 'Basic'
  }
  properties: {
    publicIPAllocationMethod: 'Dynamic'
  }
}

resource onPremiseVpnGateway 'Microsoft.Network/virtualNetworkGateways@2021-05-01'
= {
  name: onPremiseGatewayName
  location: location
  properties: {
    ipConfigurations: [
      {
        name: 'vnetGatewayConfig'
        properties: {
          publicIPAddress: {
            id: onPremisePublicIp.id
          }
          privateIPAllocationMethod: 'Dynamic'
          subnet: {
            id: onPremiseVnet.properties.subnets[0].id
          }
        }
```

```
      }
    ]
    gatewayType: 'Vpn'
    vpnType: 'RouteBased'
    enableBgp: false
    sku: {
      name: onPremiseGatewaySku
    }
  }
  dependsOn: [
    onPremiseVnet
    onPremisePublicIp
  ]
}
```

Створення VPN з’єднання:

```
resource vpnConnection 'Microsoft.Network/connections@2021-05-01' = {
  name: connectionName
  location: location
  properties: {
    connectionType: 'Vnet2Vnet'
    virtualNetworkGateway1: {
      id: primaryVpnGateway.id
    }
    virtualNetworkGateway2: {
      id: onPremiseVpnGateway.id
    }
    sharedKey: sharedKey
  }
  dependsOn: [
    primaryVpnGateway
    onPremiseVpnGateway
```

```
  ]
}
```

Створення віртуальної машини та мережевого інтерфейсу:

```
resource onPremiseVmNic 'Microsoft.Network/networkInterfaces@2021-05-01' = {
  name: '${vmName}Nic'
  location: location
  properties: {
    ipConfigurations: [
      {
        name: 'ipconfig1'
        properties: {
          subnet: {
            id: onPremiseVnet.properties.subnets[1].id // Підмережа для VM
          }
          privateIPAllocationMethod: 'Dynamic'
        }
      }
    ]
  }
  dependsOn: [
    onPremiseVnet
  ]
}

resource onPremiseVm 'Microsoft.Compute/virtualMachines@2021-03-01' = {
  name: vmName
  location: location
  properties: {
    hardwareProfile: {
      vmSize: vmSize
```

```
    }
    osProfile: {
      computerName: vmName
      adminUsername: adminUsername
      adminPassword: adminPassword
    }
    storageProfile: {
      imageReference: {
        publisher: vmImagePublisher
        offer: vmImageOffer
        sku: vmImageSku
        version: vmImageVersion
      }
    }
    networkProfile: {
      networkInterfaces: [
        {
          id: onPremiseVmNic.id
        }
      ]
    }
  }
  dependsOn: [
    onPremiseVmNic
  ]
}
```

Створення App Services для хостингу додатку:

```
resource appServicePlan 'Microsoft.Web/serverfarms@2022-03-01' = {
  name: 'appServicePlanName'
  location: location
```

```
  sku: {
    name: 'F1' // Free tier
    tier: 'Free'
    size: 'F1'
    capacity: 1
  }
  properties: {
    reserved: true
  }
}

resource webApp 'Microsoft.Web/sites@2022-03-01' = {
  name: 'AppName'
  location: location
  properties: {
    serverFarmId: appServicePlan.id
    siteConfig: {
      appSettings: [
        {
          name: 'WEBSITE_RUN_FROM_PACKAGE'
          value: '1'
        }
        {
          name: 'DOTNET_VERSION'
          value: 'v8.0'
        }
      ]
      virtualNetworkSubnetId: onPremiseVnet.id
    }
  }
}
```

Візуалізація інфраструктури, яка буде створена за допомогою шаблона зображена на рис. 2.17.

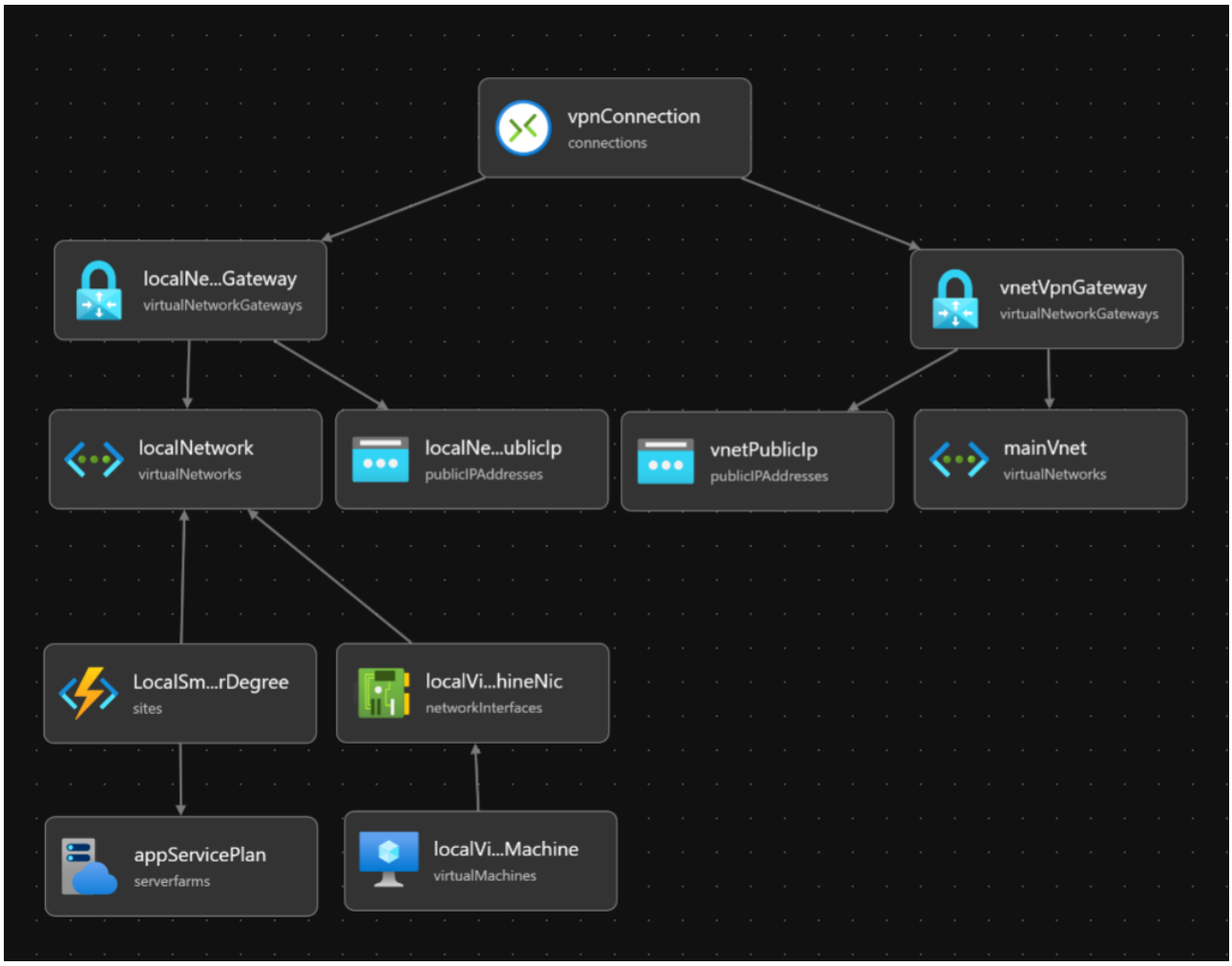

Рисунок 2.17 – Візуалізація інфраструктури, яка буде створена за допомогою шаблона

Таким чином даний шаблон може перевикористовуватися безліч разів з різними параметрами ресурсів, що дозволяє прискорити процес створення інфраструктури. Також завдяки візуалізації видно, які ресурси взаємодіють між собою.

3 РОЗРОБКА ВЕБ ДОДАТКУ ДЛЯ МОНІТОРИНГУ

3.1 Створення та підготовка веб додатку на платформі .NET

NET – це безкоштовна платформа розробника з відкритим вихідним кодом для створення різноманітних програм. Він може запускати програми, написані кількома мовами, найпопулярнішою з яких є C#. Він покладається на високопродуктивне середовище виконання, яке використовується у виробництві багатьма високомасштабними програмами. Платформа .NET була розроблена для забезпечення продуктивності, продуктивності, безпеки та надійності. Він забезпечує автоматичне керування пам’яттю через збирач сміття (Garbage Collector). Він безпечний для типів і пам’яті завдяки використанню GC і строгих мовних компіляторів. Він пропонує паралельність через примітиви async/await і Task. Він включає в себе великий набір бібліотек, які мають широку функціональність і оптимізовані для продуктивності в багатьох операційних системах і архітектурах мікросхем.

Веб додаток буде розроблятись за допомогою фреймворка .NET Web API. .NET Web API – це фреймворк для створення веб-служб HTTP у .NET (рис. 3.1). Він дозволяє розробникам будувати та підтримувати RESTful веб-сервіси, які можуть обробляти запити з різних клієнтів, включаючи веб-браузери, мобільні додатки та інші сервери. .NET Web API повністю підтримує HTTP-протокол, включаючи методи GET, POST, PUT, DELETE та інші. Це дозволяє легко створювати RESTful служби, що є стандартом для інтеграції веб-додатків.

REST (Representational State Transfer) – це архітектурний стиль для розробки веб-додатків, що ґрунтується на наборі принципів та обмежень. Він використовує стандартні HTTP методи, такі як GET, POST, PUT та DELETE, для взаємодії з ресурсами. У REST, кожен ресурс ідентифікується за допомогою унікального URI. RESTful служби повинні бути безстандартними,

тобто не зберігати стан між запитами, і використовувати HTTP статусні коди для інформування клієнтів про результати запитів. Цей підхід робить веб-служби ефективними, масштабованими та надійними.

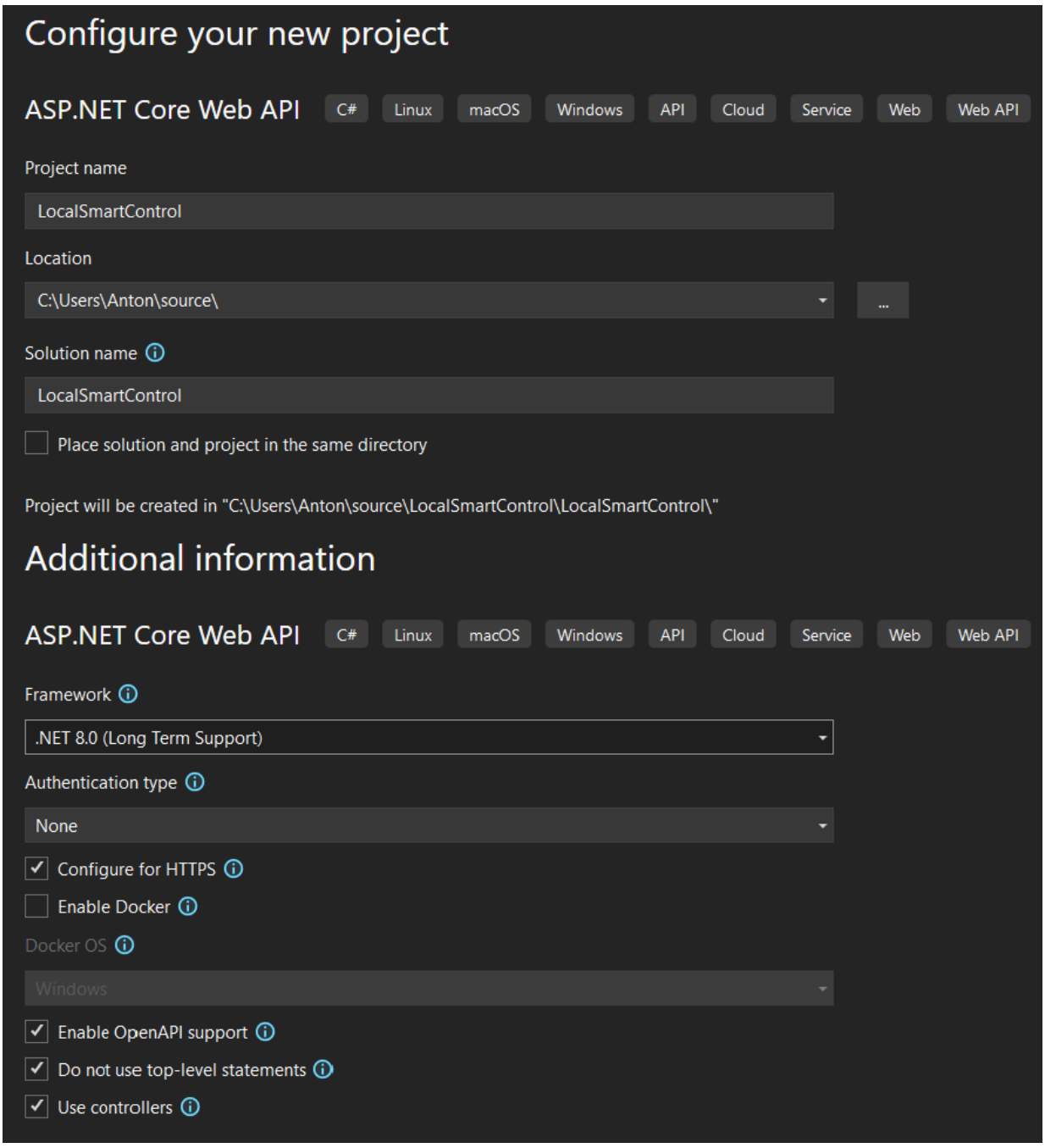

Рисунок 3.1 – Вікно створення проекту додатка з використанням .NET Web API

3.2 Розробка контролеру для роботи з периферійними пристроями

У мові програмування C# контролер – це клас, який відповідає за обробку HTTP-запитів та управління взаємодією між клієнтами та сервером. Контролер приймає запити, виконує відповідні дії та повертає відповіді. Використання контролерів робить процес розробки ендпоінтів більш організованим та ефективним

Для того, щоб отримати дані інформацію про периферійні пристрої буде використовуватися інструмент керування Windows (WMI). WMI використовує стандартні мови і протоколи для обміну даними, такі як WQL (WMI Query Language), який схожий на SQL.

Для взаємодії з принтерами використовується клас Win32_Printer у Windows Management Instrumentation (WMI) представляє принтер, встановлений на комп'ютері. Цей клас дозволяє отримувати інформацію про принтери та виконувати різні операції з ними. Win32_Printer є частиною простору імен Root\CIMv2 і надає великий обсяг інформації про налаштування та статус принтерів. Win32_Printer містить багато властивостей, які описують принтер:

1) Name: ім'я принтера,
2) Status: поточний статус принтера (наприклад, "OK", "Error", "Degraded"),
3) Default: показує, чи є цей принтер принтером за замовчуванням,
4) DriverName: назва драйвера принтера,
5) PortName: ім'я порту, до якого підключений принтер,
6) PrintProcessor: ім'я процесора друку, який використовується принтером,
7) Location: фізичне місце розташування принтера,
8) Shared: вказує, чи є принтер загальнодоступним (спільним),
9) ShareName: ім'я, під яким принтер спільно використовується,
10) Network: вказує, чи є принтер мережевим,

11) JobCount: кількість завдань у черзі друку,

12) PrinterStatus: більш детальний статус принтера (наприклад, "Printing", "Idle", "Offline").

Клас також має кілька методів, які дозволяють керувати принтером:

1) Pause: зупиняє друк на принтері,

2) Resume: відновлює друк на принтері,

3) CancelAllJobs: скасовує всі завдання друку,

4) SetDefaultPrinter: робить цей принтер принтером за замовчуванням.

Створимо декілька методів в контролері для того, щоб отримувати дані про периферійні пристрої, які підключені до комп’ютера адміністратора, в нашому випадку віртуальної машини (рис. 3.2).

```
public class PeripheralsController : ControllerBase
    public IActionResult GetPeripheralsController()
        try
        {
            // конфігурація підключення до комп'ютера адміністратора
            string remoteMachineIp = Settings.RemotePcIp;
            ConnectionOptions options = new ConnectionOptions
            {
                Username = Settings.RemotePcUser,
                Password = Settings.RemotePcPassword,
                Impersonation = ImpersonationLevel.Impersonate,
                Authentication = AuthenticationLevel.PacketPrivacy,
                EnablePrivileges = true
            };
            ManagementScope scope = new ManagementScope($"\\\\{remoteMachineIp}\\root\\cimv2", options);
            scope.Connect();

            // запит на отримання списку принтерів
            ObjectQuery query = new ObjectQuery("SELECT * FROM Win32_Printer");
            ManagementObjectSearcher searcher = new ManagementObjectSearcher(scope, query);

            ManagementObjectCollection printers = searcher.Get();
            var printerDevices = new List<PrinterDevice>();

            // додаємо принтери в список
            foreach (ManagementObject printer in printers)
            {
                var printerDevice = new PrinterDevice();

                printerDevice.Name = printer["Name"] != null ? printer["Name"].ToString() : string.Empty;
                printerDevice.Status = printer["PrinterStatus"] != null ? printer["PrinterStatus"].ToString() : string.Empty;
                printerDevice.Default = printer["Default"] != null ? printer["Default"].ToString() : string.Empty;
                printerDevice.Network = printer["Network"] != null ? printer["Network"].ToString() : string.Empty;
                printerDevice.Location = printer["Location"] != null ? printer["Location"].ToString() : string.Empty;
                printerDevice.IsShared = printer["Shared"] != null ? printer["Shared"].ToString() : string.Empty;
                printerDevice.JobCount = printer["JobCount"] != null ? printer["JobCount"].ToString() : string.Empty;
                printerDevices.Add(printerDevice);
            }

            return Ok(printerDevices);
        }
```

Рисунок 3.2 – Метод для отримання основної інформації про принтери в мережі, які підключені до комп’ютера адміністратора

Далі створимо метод, щоб керувати друком (рис. 3.3).

```
[HttpPost]
[Route("print-file")]
0 references
public IActionResult Print(string deviceName, string filePath)
{
    Підключення до віртуальної машини

    // Обираємо потрібний принтер, до якого потрібно надіслати документ
    ManagementObject targetPrinter = null;
    foreach (ManagementObject printer in printers)
    {
        if (printer["Name"].ToString() == deviceName)
        {
            targetPrinter = printer;
            break;
        }
    }

    if (targetPrinter != null)
    {
        ManagementBaseObject printMethod = targetPrinter.GetMethodParameters("PrintFile");

        // Встановлює шлях до файлу
        printMethod["FileName"] = filePath;
        printMethod["JobName"] = "PrintFile";

        // Виклик методу друку
        targetPrinter.InvokeMethod("PrintFile", printMethod, null);
    }
    else
    {
        return StatusCode(500);
    }

    return Ok();
}
```

Рисунок 3.3 – Метод для запуску друку

Наступний метод буде відповідати за скасування друку (рис. 3.4).

```
[HttpPost]
[Route("stop-printing")]
0 references
public IActionResult StopPrinting(string deviceName)
{
    Підключення до віртуальної машини

    ManagementObject targetPrinter = null;
    foreach (ManagementObject printer in printers)
    {
        if (printer["Name"].ToString() == deviceName)
        {
            targetPrinter = printer;
            break;
        }
    }

    if (targetPrinter != null)
    {
        // Зупинка всіх завдань друку
        targetPrinter.InvokeMethod("CancelAllJobs", null);
    }
    else
    {
        return StatusCode(500);
    }

    return Ok();
}
```

Рисунок 3.4 – Метод для скасування друку

3.3 Розробка методів для роботи з кліматичною технікою та збереженням інформації про них Azure Storage Table

Для роботи з кліматичною технікою буде створено окремий контролер, який міститиме в собі методи для отримання даних від кліматичної техніки. Мається на увазі, що дані буде відправляти кліматичний контролер, наприклад: Ecobee Smart Thermostat - популярний розумний термостат, який також має можливість відправки даних на сервери для моніторингу та керування.

Для отримання даних потрібна модель даних, вона буде містити в собі назву пристрою, його ідентифікатор, тип пристрою, місцерозташування та дані про виміри (рис. 3.5).

```
public class ClimateRequestModel
{
    0 references
    public string DeviceName { get; set; }
    2 references
    public Guid Deviceid { get; set; }
    1 reference
    public string? DeviceType { get; set; }
    1 reference
    public string? Location { get; set; }
    1 reference
    public Dictionary<string, string> Data { get; set; }
}
```

Рисунок 3.5 – Модель відправки кліматичних даних

Дані отримані від кліматичної техніки будуть зберігатися в Azure Table Storage. Для цього потрібно перейти на портал Azure та створити ресурс Azure Storage Account – це служба, якою керує Microsoft, що надає хмарне сховище, яке є високодоступним, безпечним, довговічним, масштабованим і резервованим. Після заповнення полів з назвами групи ресурсів та назви самого ресурсу потрібно зробити мережеві налаштування, щоб доступ був тільки з віртуальної мережі де знаходиться додаток (рис. 3.6).

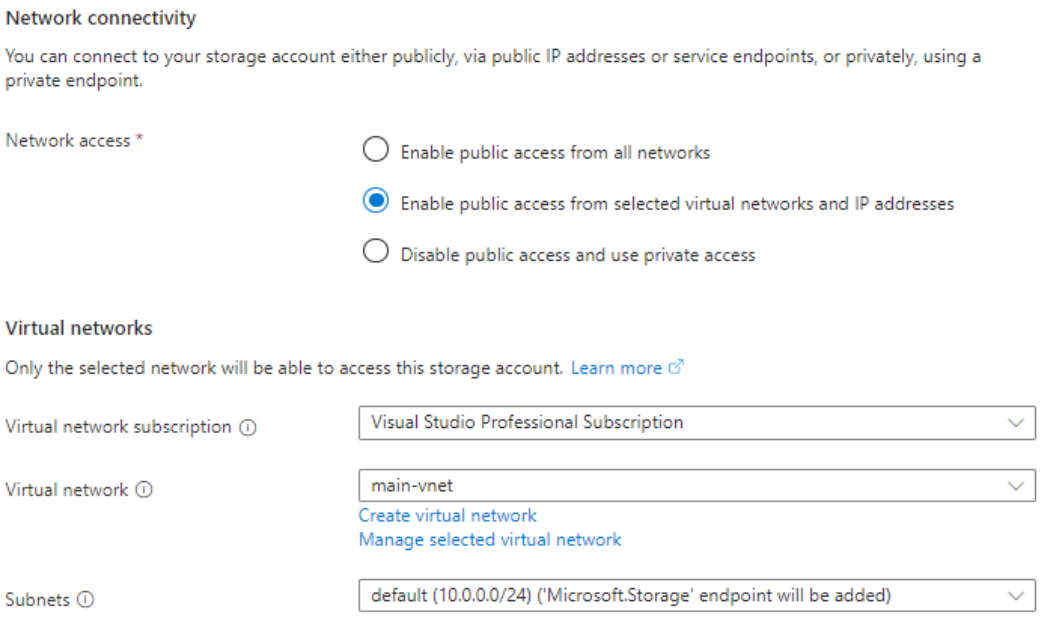

Рисунок 3.6 – Мережеві налаштування для Storage Account

Після створення Storage Account створимо в ньому таблицю для зберігання записів про стан та моніторинг клімату (рис. 3.7).

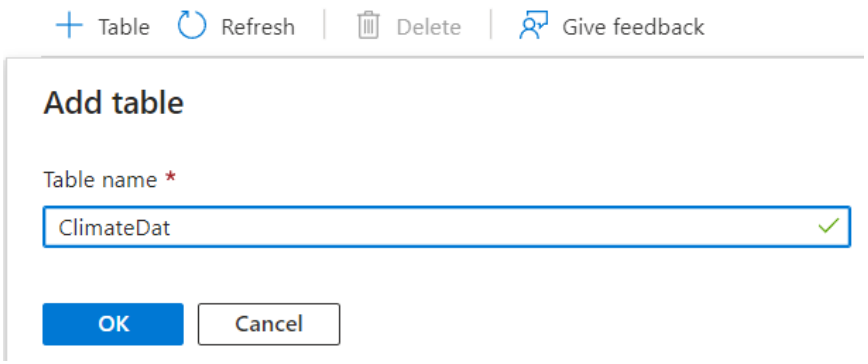

Рисунок 3.7 – Створення таблиці для збереження даних про клімат

Наступним кроком є створення сервісу для додавання записів до таблиці та їх отримання. Для того, щоб додаток зміг отримати доступ до таблиці йому потрібно мати ключі доступу та рядок підключення, які можна дізнатися в налаштуваннях Storage Account у розділі Access Keys (рис. 3.8).

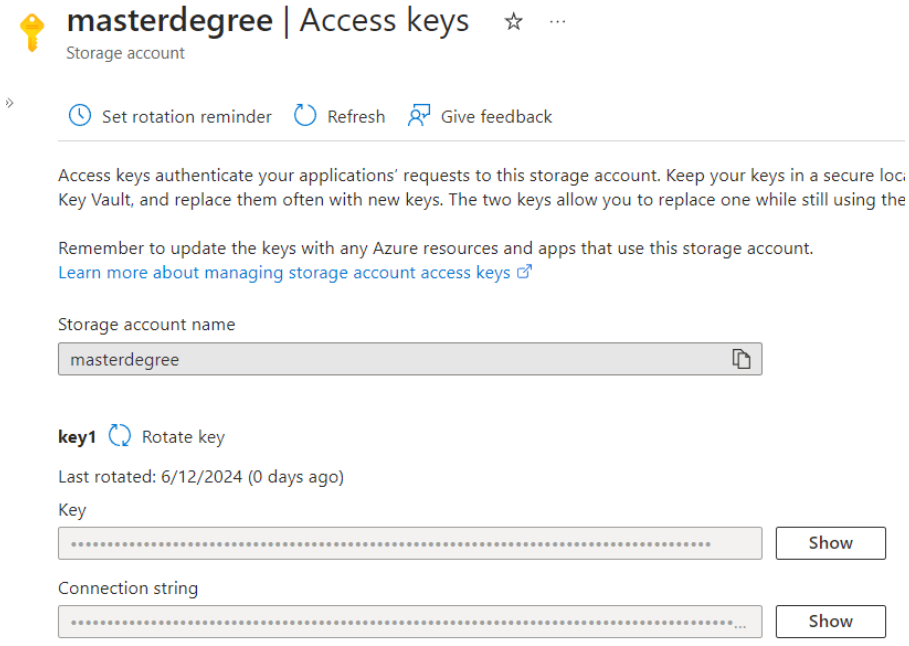

Рисунок 3.8 – Ключ доступу та рядок підключення до Storage Account

Наступний код відповідає за запис та отримування даних з Table Storage (рис. 3.9).

```
public class TableStorageService : ITableStorageService
    // Отримання запису по id пристрою
    2 references
    public ClimateResponseModel GetData(Guid id)
    {
        TableClient table = GetTableClient();
        var response = table.Query<ClimateTableEntity>(filter: $"RowKey eq '{id.ToString()}'");
        var entity = response.FirstOrDefault();
        return new ClimateResponseModel()...;
    }

    // Отримання всі записів з усіх пристроїв
    2 references
    public List<ClimateResponseModel> GetData()
    {
        TableClient table = GetTableClient();
        var response = table.Query<ClimateTableEntity>().ToList();
        var dataList = new List<ClimateResponseModel>();

        foreach (var entity in response)...
        return dataList;
    }

    // Запис даних про пристрій
    2 references
    public async Task SendData(ClimateRequestModel requestModel)
    {
        TableClient table = GetTableClient();
        await table.CreateIfNotExistsAsync();

        var tableEntity = new ClimateTableEntity()
        {
            RowKey = requestModel.Deviceid.ToString(),
            PartitionKey = requestModel.Deviceid.ToString(),
            Timestamp = DateTime.UtcNow,
            DeviceType = requestModel.DeviceType,
            Location = requestModel.Location,
            Data = JsonConvert.SerializeObject(requestModel.Data),
            DeviceName = requestModel.DeviceName
        };

        await table.AddEntityAsync(tableEntity);
    }
```

Рисунок 3.9 – Сервіс для роботи з Table Storage

Додаємо цей сервіс до контролеру Climate та отримуємо наступні ендпоніти з якими можна працювати (рис. 3.10):

1) GET /api/Climate/{deviceId} – отримує записи по конкретному кліматичному пристрою з Table Storage;

2) GET /api/Climate – отримує записи з усіх кліматичних пристроїв з Table Storage;

3) POST /api/Climate – запис даних з кліматичного пристрою до Table Storage;

4) GET /api/Peripherals – отримати записи про всі переферійні пристрої підключені до комп'ютера адміністратора;

5) POST /api/Peripherals/print-file/{deviceName}/{filePath} – відправити запит на друк файл конкретним пристроєм;

6) POST /api/Peripherals/stop-printing/{deviceName} – зупинити друк для конкретного пристрою.

Рисунок 3.10 – Список ендпоінтів додатку

3.4 Розгортання додатку в Azure App Services

Після розробки коду додаток потрібно розмістити в хмарі. Раніше вже була створена інфраструктура для цього. Для цього потрібно у Visual Studio

обрати Publish для додатку, та обрати публікацію в Azure. Після цього обрати раніше створений App Service (рис. 3.11).

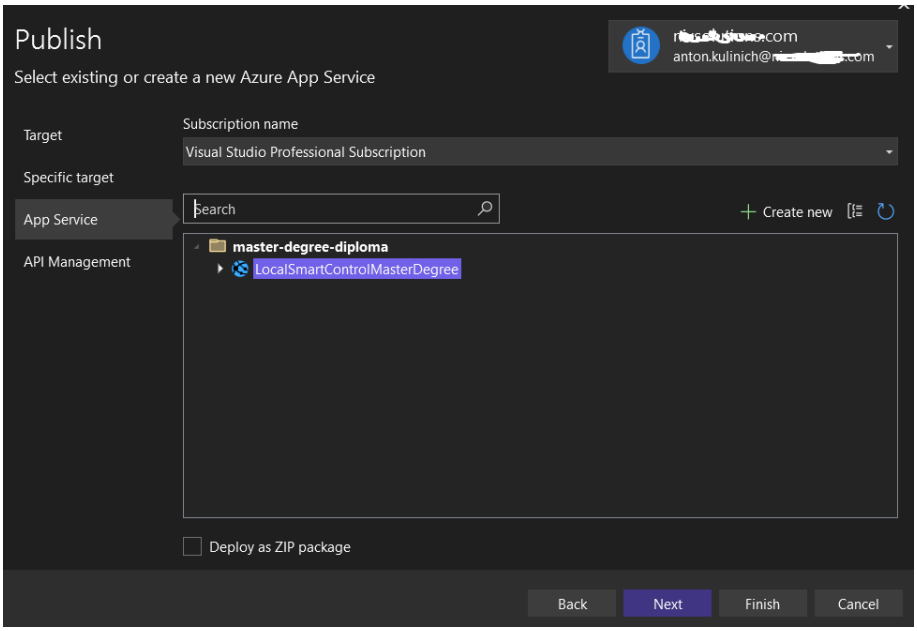

Рисунок 3.11– Вікно публікації додатка

Після налаштувань публікації буде створено профіль для публікації (рис. 3.12). У файлі містяться відомості про те, в якій групі ресурсів та підписці знаходиться додаток, а також його URL адреса.

```
<Project>
  <PropertyGroup>
    <WebPublishMethod>MSDeploy</WebPublishMethod>
    <ResourceId>/subscriptions/63fa26d0-2ecd-4585-9b1b-2a8103949f68/resourceGroups/master-degree-diploma/providers/Microsoft.W
    <ResourceGroup>master-degree-diploma</ResourceGroup>
    <ApiResourceId>/subscriptions/63fa26d0-2ecd-4585-9b1b-2a8103949f68/resourceGroups/master-degree-diploma/providers/Microsof
    <OpenApiDocumentName>v1</OpenApiDocumentName>
    <UpdateApiOnPublish>true</UpdateApiOnPublish>
    <PublishProvider>AzureWebSite</PublishProvider>
    <LastUsedBuildConfiguration>Release</LastUsedBuildConfiguration>
    <LastUsedPlatform>Any CPU</LastUsedPlatform>
    <SiteUrlToLaunchAfterPublish>https://localsmartcontrolmasterdegree.azurewebsites.net</SiteUrlToLaunchAfterPublish>
    <LaunchSiteAfterPublish>true</LaunchSiteAfterPublish>
    <ExcludeApp_Data>false</ExcludeApp_Data>
    <ProjectGuid>e56ed3ee-469b-4662-9fa6-c971e20f079c</ProjectGuid>
    <MSDeployServiceURL>localsmartcontrolmasterdegree.scm.azurewebsites.net:443</MSDeployServiceURL>
    <DeployIisAppPath>LocalSmartControlMasterDegree</DeployIisAppPath>
    <RemoteSitePhysicalPath />
    <SkipExtraFilesOnServer>true</SkipExtraFilesOnServer>
    <MSDeployPublishMethod>WMSVC</MSDeployPublishMethod>
    <EnableMSDeployBackup>true</EnableMSDeployBackup>
    <EnableMsDeployAppOffline>true</EnableMsDeployAppOffline>
    <UserName>REDACTED</UserName>
    <_SavePWD>true</_SavePWD>
    <_DestinationType>AzureWebSite</_DestinationType>
  </PropertyGroup>
</Project>
```

Рисунок 3.12 – Профіль для публікації додатка в Azure App Services

Після успішної публікації додатку переходимо по URL додатку, та перевіряємо чи все працює (рис. 3.13).

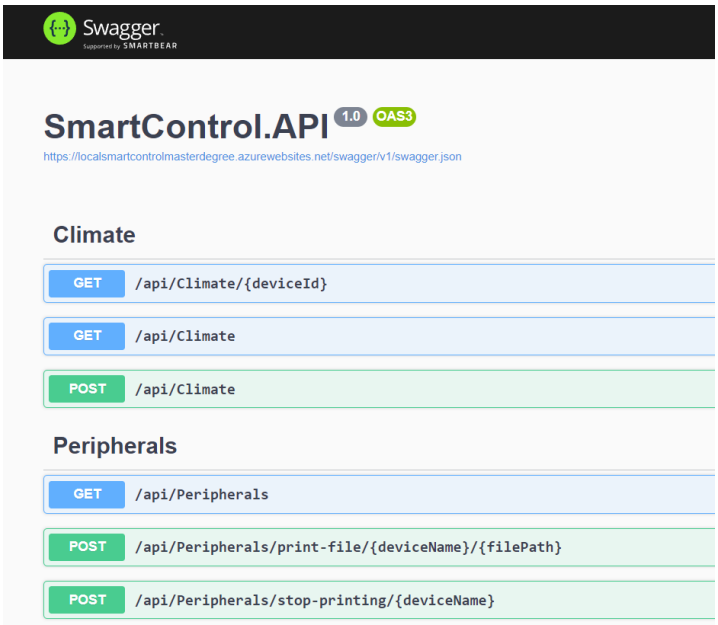

Рисунок 3.13 – Доступні ендпоінти в опублікованому додатку

Для тестування спробуємо відправити дані про виміри від кліматичного пристрою (рис. 3.14).

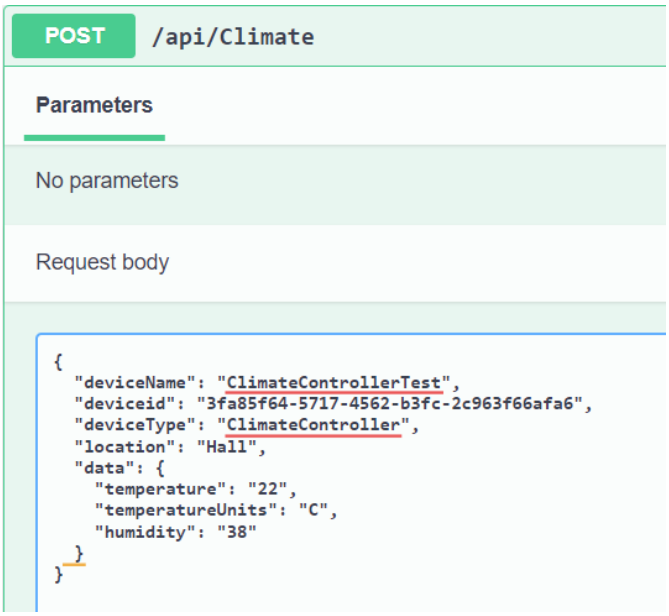

Рисунок 3.14 – Відправка тестових даних про виміри від кліматичної техніки

Перевіряємо чи дані були успішно за допомогою GET запита до серверу, успішно отримуємо дані про пристрій (рис. 3.15).

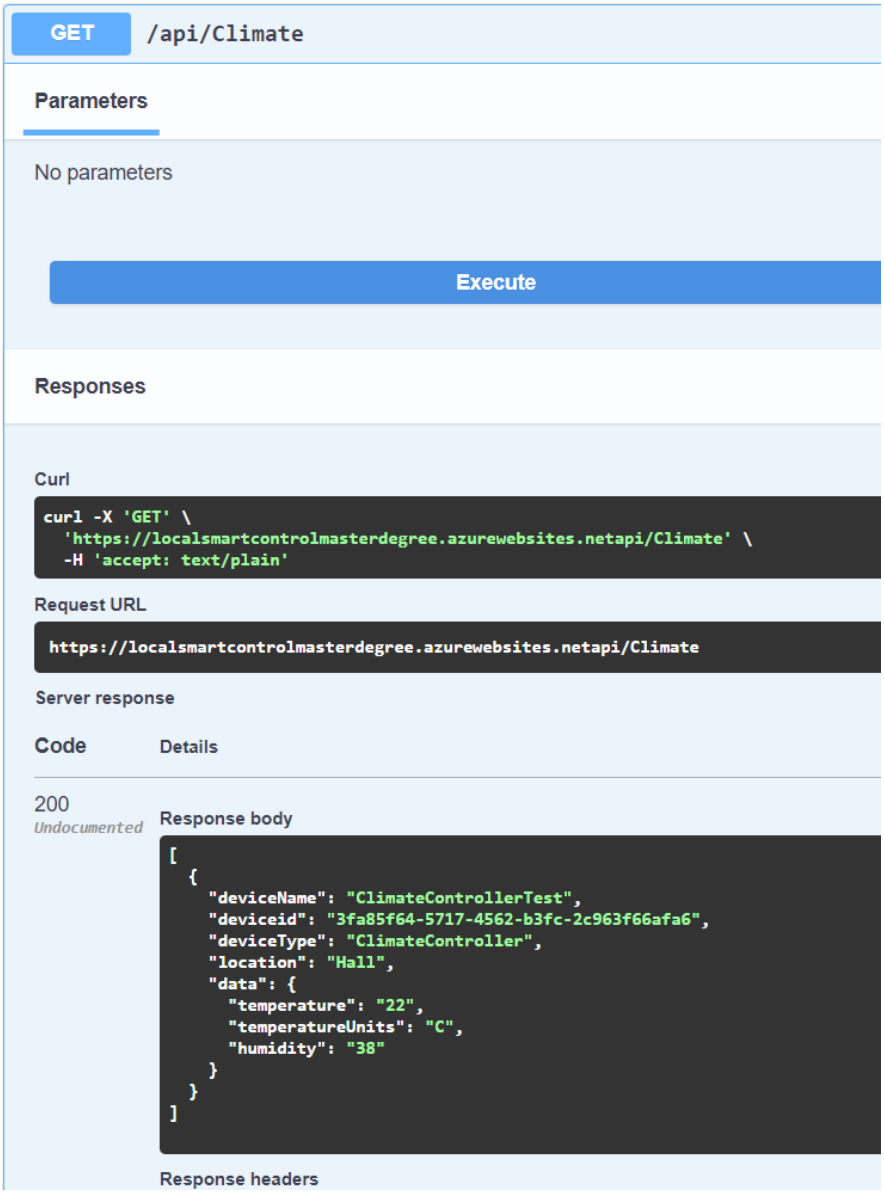

Рисунок 3.15 – Отримані дані про пристрої через запит до серверу

Таким чином, додаток може інтегруватися з Grafana, або іншими схожими сервісами. Це дозволяє забезпечити візуалізацію даних у реальному часі, що є надзвичайно корисним для моніторингу та аналізу продуктивності системи.

Окрім цього, при розробці фронтенд частини додатка можливо виводити ці дані на графіки безпосередньо на веб-інтерфейсі. Це забезпечує зручний доступ до інформації та полегшує її сприйняття. Відображення даних у графічному вигляді дозволяє користувачам швидко оцінювати ситуацію, порівнювати різні показники та аналізувати історичні дані. Для цього можна використовувати різні бібліотеки JavaScript, які забезпечують широкий спектр можливостей для створення інтерактивних та динамічних графіків..

ВИСНОВКИ

В кваліфікаційній роботі було проведено дослідження та розробка додатку для моніторингу офісних пристроїв в локальній мережі з використанням хмарних технологій. В результаті виявлено, що застосування цього додатку дозволить забезпечити ефективне управління пристроями, підвищити рівень безпеки та надійності даних, а також забезпечити швидкий та зручний доступ до інформації про їхній стан та функціональні можливості без фізичної присутності в офісі.

Отримані результати свідчать про актуальність та перспективність використання хмарних технологій у сфері моніторингу та управління офісними пристроями. Розроблений додаток може бути використаний як основа для подальших досліджень у цьому напрямі та для впровадження в реальних умовах офісного середовища з метою підвищення продуктивності та оптимізації бізнес-процесів.

В цілому, результати даної роботи відкривають нові можливості для покращення управління та моніторингу офісних пристроїв і сприяють розвитку сучасних технологій в офісному середовищі.

В першому розділі кваліфікаційної роботи була описана важливість комфортного офісного середовища для бізнесу, описані переваги хмарних технологій загалом та в контексті моніторинговим систем. Також було розглянуто різні варіанти підключення додатку з хмари до локальної мережі.

У другому розділі була розроблена інфраструктура для взаємодії локальної мережі, яка в свою чергу була імітована віртуальною мережею, з віртуальною мережею де знаходиться додаток. Було створено VPN з’єднання між двома віртуальними мережами. Створено віртуальну машину, яка виступає в ролі комп’ютера адміністратора до якого підключені офісні пристрої. Для додатку був підготовлений хмарний сервіс для хостингу. Останнім кроком було написання шаблону для розгортання раніше створеної інфраструктури, що

дозволяє перевикористовувати цей шаблон безліч разів, без втручання у графічний інтерфейс на порталі Azure.

Третій розділ присвячено написанню коду для додатку та створення сховища в хмарі. Було розроблено веб додаток на платформі .NET, який взаємодіє з комп’ютером адміністратора через інструмент керування Windows (WMI). Додаток представляє з себе API через який кліматична техніка може зберігати дані, а аналітики та адміністратори отримувати цю інформацію для розробки звітів та побудови графіків.

Деякі результати роботи було апробовано на XXVIII Міжнародному молодіжному форумі «Радіоелетроніка і молодь у XXI столітті» в рамках конференції «Перспективи розвитку інфокомунікацій та інформаційно-вимірювальних технологій» та опубліковано тези доповіді [8] за тематикою кваліфікаційної роботи.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Economic benefits of the effects of office environment on perceived work efficiency and presenteeism / [S. Arata, M. Sugiuch, T. Ikaga та ін.]. // Building and Environment. – 2023. – №243. – 110712.

2. Izraylevych I. Cloud vs On Premise Cost Comparison: A Comprehensive Guide [2024] [Електронний ресурс] / Igor Izraylevych // S-PRO. – 2024. – Режим доступу до ресурсу: https://s-pro.io/blog/cloud-computing-vs-on-premises-advantages-disadvantages-and-cost-comparison.

3. How Azure pay-as-you-go pricing works [Електронний ресурс] // Microsoft. – 2024. – Режим доступу до ресурсу: https://azure.microsoft.com/en-us/pricing/purchase-options/pay-as-you-go.

4. Introduction to Azure Storage [Електронний ресурс] // Microsoft. – 2023. – Режим доступу до ресурсу: https://learn.microsoft.com/en-us/azure/storage/common/storage-introduction.

5. Azure App Service Hybrid Connections [Електронний ресурс] // Microsoft. – 2024. – Режим доступу до ресурсу: https://learn.microsoft.com/en-us/azure/app-service/app-service-hybrid-connections.

6. Integrate your app with an Azure virtual network [Електронний ресурс] // Microsoft. – 2024. – Режим доступу до ресурсу: https://learn.microsoft.com/en-us/azure/app-service/overview-vnet-integration.

7. What is Azure ExpressRoute? [Електронний ресурс] // Microsoft. – 2024. – Режим доступу до ресурсу: https://learn.microsoft.com/en-us/azure/expressroute/expressroute-introduction.

8. Кулініч А.О. Розробка додатку для моніторингу офісних пристроїв в локальній мережі з використанням хмарних технологій / Науковий керівник – к.т.н., доц. Чеботарьова Д.В. // Тези доповідей 28-го Міжнародного молодіжного форуму «Радіоелектроніка та молодь у XXI столітті». Збірник матеріалів форуму. Т.4. – Харків: ХНУРЕ. – 2024. – С. 125 – 126.