Міністерство освіти і науки України
Національний університет «Києво-Могилянська академія»
Кафедра мережних технологій факультету інформатики

Система керування розподілом адресного простору (IPAM) мережі підприємства

Текстова частина до кваліфікаційної роботи
за спеціальністю «Комп'ютерні науки» 122

Керівник курсової роботи
старший викладач, кандидат тех. наук
Черкасов Д.І. _____
(підпис)
«____» _____ 2024р.

Виконав студент КН-4
Махаммедов Ж.Ж.
«____» _____ 2024 р.

Київ – 2024

Міністерство освіти і науки України

Національний університет «Києво-Могилянська Академія»

Кафедра мережних технологій факультету інформатики

ЗАТВЕРДЖУЮ

Зав. кафедри мережних технологій,

проф., доктор фіз.-мат. наук

_____ Малашонок Г.І.

(підпис)

«____» _____ 2024 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

На кваліфікаційну роботу

студенту 4 року навчання БП «Комп'ютерні науки»

Махаммедову Жану Жановичу

на тему:

Система керування розподілом адресного простору (IPAM) мережі підприємства

Зміст ТЧ до кваліфікаційної роботи:

1. Індивідуальне завдання
2. Вступ
3. Опис основних технологій і понять предметної області
4. Огляд критеріїв можливих рішень
5. Створення власного застосунку
6. Висновки
7. Список використаних джерел
8. Додаток А

Дата видачі: «____» _____2023 р.

Керівник: к. т. н. Черкасов Д.І. _____(підпис)

Завдання отримав _____(підпис)

Календарний план

№	Назва етапу кваліфікаційної роботи	Термін виконання етапу	Примітка
1.	Отримання завдання на кваліфікаційну роботу	28 жовтня 2023	
2.	Пошук літератури	13 лютого 2024	
3.	Ознайомлення з існуючими рішеннями	20 березня 2024	
4.	Написання перших двох розділів	5 квітня 2024	
5.	Розробка власного застосунку	12 травня 2024	
6.	Написання третього розділу	17 травня 2024	
7.	Корегування роботи	20 травня 2024	
8.	Створення презентації	21 травня 2024	
9.	Подання роботи на кафедру для перевірки на плагіат	23 травня 2024	
10.	Захист кваліфікаційної роботи	Кінець травня 2024	

Студент Махаммедов Ж.Ж

Керівник Черкасов Д.І.

«_____» _____

Зміст

Перелік використаних скорочень

API – Application Programming Interface

ARP – Address Resolution Protocol

DDI – DNS, DHCP, and IPAM

DHCP – Dynamic Host Configuration Protocol

DNS – Domain Name System

HTTP – HyperText Transfer Protocol

ICMP – Internet Control Message Protocol

IoT – Internet of Things

IP – Internet Protocol

IPAM – IP address management

JSON – JavaScript Object Notation

MVP – Minimum Viable Product

ORM – Object-Relational Mapping

SNMP – Simple Network Management Protocol

URL – Uniform Resource Locator

VLAN – Virtual Local Area Network

Анотація

У цій роботі було проведено аналіз предметної області, що включає розгляд основних компонентів мережевої інфраструктури. Це дало можливість зрозуміти вимоги до системи та визначити критерії для оцінки наявних рішень управління адресним простором, таких як ціна, рівень автоматизації, інтеграції, доступності та безпеки.

В практичній частині представлено розробку системи керування розподілом адресного простору (IPAM) для мережі підприємства.

Вступ

У сучасному світі, де технології розвиваються все швидше [1] мережеві структури стали необхідною складовою функціонування будь-якого підприємства. Зростання обсягів даних, розвиток хмарних технологій та віддалених робочих місць стимулюють ріст потужностей мережевої інфраструктури як приватних підприємств, так і державних установ, через що ростуть і видатки на мережеву інфраструктуру [2]. У цьому контексті, ефективне керування розподілом адресного простору стає важливою задачею адміністрування мереж підприємств.

Системи керування розподілом адресного простору (IPAM) є інструментом для забезпечення ефективного використання IP-адрес та керування й аналізу мережевих ресурсів. Вони дозволяють будувати структури мереж, підмереж і окремих пристроїв, у поєднанні з DHCP-сервером, вони можуть допомогти автоматизувати процеси призначення та управління IP-адресами, забезпечуючи стабільну та оптимальну роботу мережі.

Основні задачі системи IPAM:

1. Централізоване управління процесом призначення IP-адрес
2. Відстеження доступних та використаних IP-адрес
3. Автоматизація процесів пов’язаних з контролем IP-адрес
4. Забезпечення цілісності мережі

Більш просунуті системи IPAM можуть також керувати серверами DNS та DHCP і використовувати ICMP, SNMP протоколи для збирання даних в мережі та їх подальшої обробки[3]. Все це спрямовано на забезпечення ефективного та спрощеного управління адресним простором мережі, що є критичним для забезпечення надійності та продуктивності мережевої інфраструктури підприємства.

Хоча існує дуже багато різних програмних рішень, більшість побудовані за трирівневою архітектурою, тобто застосунок розділений на три окремих компоненти:

1. Рівень представлення
2. Рівень застосунку
3. Рівень даних[4]

Ця робота спрямована на дослідження роботи інтернет мереж, аналізу критеріїв можливих рішень у цій сфері, дослідження способів автоматизації систем IPAM, а також на розробку та реалізацію власної системи керування розподілом адресного простору для мережі підприємства.

Розділ 1. Опис основних технологій і понять предметної області

1.1 Internet Protocol

Internet Protocol (IP) – є протоколом третього мережного рівня у мережевій архітектурі TCP/IP [5]. Він є одним з основних блоків мережі інтернет, оскільки відповідає за адресацію та маршрутизацію даних у мережі.

Важливою складовою цього протоколу є IP-адреси – унікальні ідентифікатори пристроїв у мережі, вони й використовуються для адресації пакетів даних та їх маршрутизації в мережі. Кожен пакет має заголовок, який містить інформацію про IP-адресу джерела та призначення. Маршрутизатори в мережі використовують цю інформацію для визначення шляху доставлення пакета до призначення.

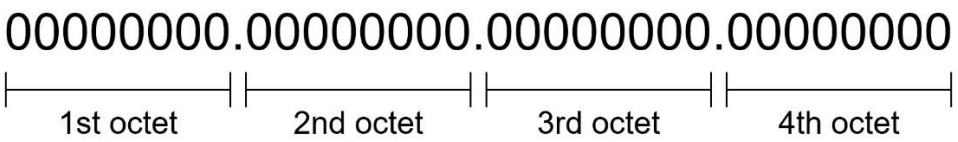

Рис. 1.1 Структура IPV4-адреси

Зараз існує дві версії протоколу IP:

1. IPV4 – оригінальна версія протоколу, розроблена 1981 року IETF [6]
2. IPV6 – друга версія, вперше описана у 1998 році.

IPV6 має достатньо багато відмінностей від свого попередника, але найбільша різниця – це розмір IP-адреси, оскільки в той час, як адреса IPV4 має розмір 32 біти, адреса IPV6 має розмір 128 біт [7]. Проблема в тому, що кількість вільних IPV4-адрес закінчилась достатньо давно, тому IPV6 має розв'язувати цю проблему, але повного переходу на IPV6 так і не відбулось, на цей час використовуються четвертий і шостий протоколи одночасно, що може створювати деякі проблеми при роботі систем IPAM. За даними «W3Techs» станом на перше травня 2024 року лише 24.9 відсотки сайтів використовують IPV6 [8], в той час, як за даними

компанії «Cloudflare» станом на жовтень 2023 року близько 36 відсотків всього трафіку – це IPV6 [9].

А це означає, що до повноцінного переходу на IPV6 ще далеко, а тому можна стверджувати, що система IPAM має повноцінно підтримувати обидві версії протоколу.

1.2 Domain Name System

Domain Name System (DNS) – це розподілена ієрархічна система, яка використовується для перетворення доменних імен в IP-адреси та навпаки. Вона дозволяє людям використовувати зрозумілі доменні імена для доступу до ресурсів в інтернеті замість користування числовими IP-адресами. Процес знаходження IP-адреси містить декілька запитів і окремих серверів, але для нашої задачі це не так важливо. Більш важливими для нас є записи, які зберігають DNS, а саме наступні три типи:

1. A record – він зберігає IPV4 адресу домену
2. AAAA record – він зберігає IPV6 адресу домену [10]
3. PTR record – він зберігає домен IP-адреси [11].

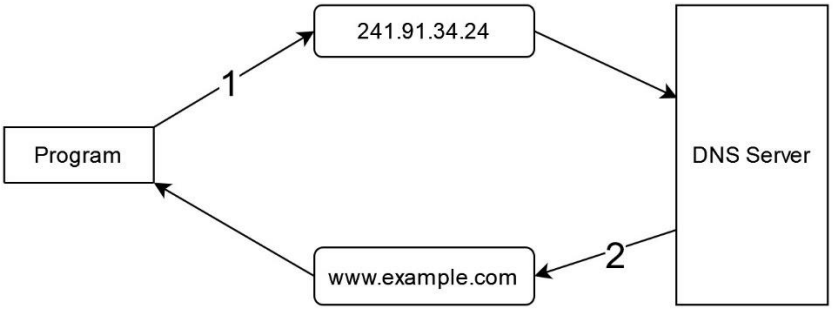

Рис. 1.2 Схема зворотного DNS запиту

При інтеграції з DNS, система IPAM може автоматично надсилати запит на створення цих записів до DNS сервера при виділенні нової IP-адреси, а також

забезпечувати синхронізацію записів при змінах в системі IPAM. Іншим плюсом такої взаємодії може бути об’єднання двох систем в один контрольний інтерфейс, що може сильно пришвидшити та спростити роботу мережного адміністратора.

Проте, треба зазначити, що існує декілька протоколів DNS, такі як RFC 1035, DoH і DoT – вони дозволяють робити запити для знаходження доменного імені, але проблема в тому, що універсального API для керування сервером DNS не існує, тому для реалізації такої інтеграції треба знати який конкретно сервер використовується або створювати достатньо складну систему взаємодії.

1.3 Dynamic Host Configuration Protocol

Dynamic Host Configuration Protocol (DHCP) – є протоколом мережевого рівня, призначеним для автоматичного призначення IP-адрес та іншої конфігураційної інформації клієнтам в комп'ютерних мережах. Він дозволяє забезпечити автоматизоване налаштування мережевих параметрів пристроїв без необхідності втручання людини [12].

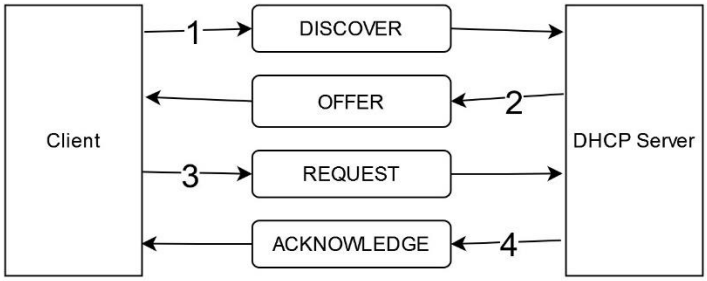

Рис 1.3 Схема процесу роботи DHCP

Є три типи призначення IP адрес за допомогою DHCP:

1. Статичне – в цьому випадку мережний адміністратор налаштовує таблицю MAC-адрес і кожній відповідає IP-адреса.

2. Динамічне – тут адміністратором задається діапазон IP-адрес і при надходженні запиту зі сторони клієнта сервер виділяє йому IP-адресу, але в цьому випадку він виділяє цю адресу на певний час.
3. Автоматичне – в останній варіації ситуація схожа на динамічне виділення, але тут адреса виділяється назавжди [13].

Інтеграція DHCP з IPAM має ті самі проблеми, що і з DNS.

1.4 DDI

DDI (DNS, DHCP, and IPAM) – це назва системи, яка охоплює інтеграцію всіх трьох технологій. Повноцінна автоматизація можлива лише у цьому випадку.

Очікується, що ринок DDI матиме сукупний середньорічний темп зростання з 2019 по 2029 у 11.40 відсотків [14]. Це показує наскільки важлива автоматизація у нашій швидко зростаючій мережі. Ріст зумовлено переходом все більшої частини послуг у цифрову форму, збільшенням кількості IoT пристроїв та переходом на IPV6.

1.5 ARP

Address Resolution Protocol (ARP) – це мережевий протокол, який використовується для відображення відношення IP-адреси до фізичної адреси у локальній мережі. ARP є критично важливим компонентом у роботі мереж, який забезпечує правильне доставляння пакетів даних між пристроями у локальній мережі.

Основні функції ARP:

a) Визначення MAC-адреси за IP-адресою: Коли пристрій у локальній мережі хоче надіслати дані іншому пристрою, він повинен знати MAC-адресу пристрою призначення. ARP запити використовуються для отримання цієї інформації.
b) Зберігання записів відношень IP-MAC: Після отримання MAC-адреси для конкретної IP-адреси, ARP зберігає цю відповідність у локальній ARP-таблиці, що дозволяє швидко знайти MAC-адресу для майбутніх запитів.

Робота ARP полягає у тому, що коли пристрій А хоче зв’язатися з пристроєм Б, знаючи лише його IP-адресу, він відправляє широкомовний ARP-запит локальною мережею. Цей запит містить IP-адресу пристрою Б і запитує, хто володіє цією IP-адресою. Всі пристрої у локальній мережі отримують цей запит, але лише пристрій з відповідною IP-адресою відповідає на нього. Він відправляє ARP-відповідь, що містить його MAC-адресу. Таким способом, ми можемо дізнатися чи є такий пристрій в локальній мережі[15].

Важливо зазначити, що цей протокол працює лише з IPV4, а в IPV6 ARP було замінено на Neighbor Discovery (ND) [16].

1.6 ICMP

Internet Control Message Protocol (ICMP) – це мережевий протокол, який використовується для обміну повідомленнями про стан мережі між пристроями. ICMP не використовується для передачі даних додатків, а забезпечує діагностичні та контрольні функції, які допомагають визначити стан мережі та розв’язувати проблеми з її функціонуванням.

Основні функції ICMP:

a) Діагностика мережевих з'єднань: ICMP дозволяє пристроям у мережі надсилати та отримувати повідомлення, що допомагають діагностувати проблеми з мережевими з'єднаннями. Основними інструментами є ping, який надсилає echo запит на пристрій, після чого він надсилає відповідь та traceroute, який дозволяє дізнатись шлях пакета до пристрою.
b) Повідомлення про помилки: ICMP використовується для передачі повідомлень про помилки, які виникають під час доставлення IP-пакетів. Це включає такі помилки, як недосяжність мережі або вузла, перевищення часу життя пакета (TTL) та інші.
c) Безпека: ICMP може використовуватись задля виявлення неавторизованого трафіку в мережі та пропускати лише правильні пакети [17].

Деякі типи ICMP-повідомлень:

- Echo та Echo Reply (типи 8 і 0): Використовуються для тестування доступності вузлів у мережі (команда ping). Пристрій відправляє запит Echo, а у відповідь отримує Echo Reply.
- Destination Unreachable (тип 3): Вказує на те, що пункт призначення недосяжний. Це повідомлення може містити різні коди для більш точного визначення причини, наприклад, мережа недосяжна, пристрій недосяжний, порт недосяжний [18].
- Time Exceeded (тип 11): Вказує на те, що час життя пакета вичерпався під час транзиту. Використовується, наприклад, у команді traceroute для визначення маршруту до пункту призначення. [19]

Процес роботи ICMP:

1) Відправлення запиту: Пристрій відправляє ICMP-запит до іншого вузла у мережі. Нехай це буде запит Echo для перевірки доступності пристрою.

2) Обробка запиту: Пристрій-отримувач обробляє запит та генерує відповідне повідомлення – Echo Reply, яке вказує на успішне доставляння запиту. Після цього повідомлення надсилається до відправника.

ICMP є важливим інструментом для діагностики та управління мережами. Він дозволить системі IPAM знаходити стан пристроїв у мережі.

1.7 Висновки до розділу 1

Наразі ми знаходимось у перехідному періоді з IPV4 до IPV6 і системи IPAM стають все більш потрібними. Залежно від складності інфраструктури, підприємству може вистачити простенької системи IPAM, навіть електронної таблиці в деяких випадках, особливо якщо сервер DHCP відсутній або працює в статичному режимі. З іншого боку, підприємства зі складною інфраструктурою потребують більш інтегровані системи IPAM, бо без автоматизації підтримка функціонування мереж неймовірно ускладнюється, а ймовірність помилок зростає, такі системи потребують великої кількості технологій, таких як: DNS, DHCP, ICMP. DDI, своєю чергою допомагає забезпечити постійність і узгодженість мереж.

Розділ 2. Огляд критеріїв можливих рішень

2.1 За ціною

При оцінці різних рішень у сфері керування розподілом адресного простору мережі підприємства одним із важливих критеріїв є їхня вартість. На ринку існує широкий спектр рішень, від безкоштовного програмного забезпечення з відкритим кодом і вільними ліцензіями та до комерційних продуктів з високими цінами ліцензій та підтримки.

Безкоштовні IPAM зазвичай забезпечують базовий функціонал для керування IP-адресами, можуть бути менш надійними й мати обмежену підтримку: як технічну, так і в сенсі розробки, оскільки такі рішення найчастіше підтримуються приватними особами та немає гарантій, що вони продовжуватимуть розробку й оновлення таких програм. Те саме стосується і проєктів, які підтримуються різного виду некомерційними фондами, але, з іншого боку, часто є можливість модифікувати таке програмне забезпечення під себе і вести його підтримку всередині підприємства, що в деяких випадках є цілком придатним рішенням. Серед безплатних рішень можна виділити «phpIPAM» з його активними оновленнями та відносно багатою функціональністю [20].

Комерційні рішення натомість мають розширений функціонал та гарантовану підтримку від виробників. Також часто при придбанні таких продуктів, розробник надає тренування для персоналу. Проте, їхня вартість може бути значною, особливо для невеликих підприємств зі складною мережевою інфраструктурою, яка має багато підмереж і пристроїв, оскільки багато компаній утворюють ціну пропорційно до розміру мережі. При цьому, витрати на ліцензії, підтримку та реалізацію можуть перевищити бюджетні можливості деяких підприємств. Так, наприклад, станом на квітень дві тисячі двадцять четвертого

року компанія «Infoblox» займає 43.2% ринку IPAM і річна ціна на ліцензію може складати від десяти до двадцяти тисяч доларів США на рік[21].

2.2 За рівнем автоматизації

При виборі системи керування розподілом адресного простору важливо враховувати її рівень автоматизації, оскільки це має прямий вплив на ефективність, продуктивність та цілісність управління мережею. Можна розділити системи IPAM на три категорії за рівнем автоматизації.

На найнижчому рівні автоматизації знаходяться базові рішення, які надають базовий функціонал, такий як призначення та відстеження IP-адрес, але вимагають ручного втручання адміністраторів для всього спектру завдань. Ці рішення зазвичай використовують спрощені інтерфейси користувача або командний рядок для керування програмою, вони не можуть відстежувати зміни в мережі та проводити аналіз цілісності у своїх даних. Такі рішення слушні для компаній, у яких мережева інфраструктура не є пріоритетом або вона достатньо проста і здебільшого статична.

На середньому рівні автоматизації знаходяться рішення, які надають розширений функціонал та підтримують автоматизацію деяких завдань. Ці системи можуть мати можливості автоматичного виявлення пристроїв у мережі, конфліктів адрес, забезпечувати цілісність структури мережі у своїх сховищах даних та інші функції, які полегшують рутинні адміністративні завдання та запобігають конфліктам. Такі рішення добре підійдуть компаніям, у яких середній розмір мережної інфраструктури, яка не потребує швидкого збільшення.

На високому рівні автоматизації знаходяться передові рішення DDI, які мають можливість автоматизації складних мережевих завдань. Ці системи можуть мати автоматичне масштабування, управління політиками безпеки, прогнозування

потреб у ресурсах та інші розумні функції, які можуть бути дуже корисними для підприємств з великими мережами, особливо якщо є потреба у швидкому розвитку потужностей.

2.3 За рівнем інтеграції

При виборі системи керування розподілом адресного простору важливо також враховувати рівень її інтеграції з наявними та майбутніми елементами мережевої інфраструктури.

Деякі IPAM-системи можуть пропонувати базову інтеграцію з іншими системами, такими як системи DNS, DHCP, системи моніторингу мережі або системи інвентаризації пристроїв. Це може включати передачу даних від таких систем для відображення у системі моніторингу або частковий чи повний контроль цих систем через IPAM.

На вищому рівні інтеграції знаходяться рішення, які підтримують широкий спектр інтеграції з іншими системами, такими як системи автоматизації мережевих процесів, системи управління конфігурацією, системи контролю доступу та інші. Ці системи можуть використовувати стандартні протоколи або спеціальні API для забезпечення обміну даними та співпраці з іншими платформами.

Важливо враховувати сумісність та можливість і рівні інтеграції між різними системами. Наприклад, якщо компанія використовує «AWS» для потреб мережевої інфраструктури, то, можливо, є сенс користуватися «Amazon VPC IP Address Manager» [22] Інтегрована IPAM-система може спростити управління мережею, забезпечуючи однорідну та централізовану платформу для керування IP-адресами та іншими мережевими ресурсами. Використання DDI може знизити витрати на обслуговування та забезпечити більш ефективне та стабільне використання мережевих ресурсів.

2.4 За доступністю та безпекою

При виборі системи керування розподілом адресного простору важливими критеріями є її доступність та безпека. Ці аспекти визначаються як технічними можливостями системи, так і стратегіями захисту, що застосовуються в організації загалом, оскільки дуже багато кібератак здійснюються за допомогою соціальної інженерії, як приклад можна навести дані однієї з найбільших американських телекомунікаційних компаній «Verizon», у їхньому звіті «2024 Data Breach Investigations Report» зазначено, що 68% зламів трапляються через людський фактор [23]. Тому дуже важливо періодично проводити тренінг персоналу.

З погляду безпеки, система IPAM повинна мати вбудовані механізми захисту даних та захисту від зовнішніх загроз. Це може включати шифрування даних, механізми аутентифікації та авторизації користувачів, механізми запобігання атак DNS та DHCP, аудит доступу до системи, а також захист від зловмисних і ненавмисних атак. Дуже важливим моментом, який впливає на безпеку IPAM, є аналіз журналу подій, що дозволяє виявляти та відстежувати спроби несанкціонованого доступу та інші потенційні загрози для мережі.

З погляду доступності, система IPAM в ідеалі повинна мати високий рівень відмовостійкості, щоб забезпечити безперервність роботи мережі. Це означає, що вона містить механізми резервування, реплікації даних та моніторингу стану серверів для запобігання відмов та забезпечення швидкого відновлення у разі аварій.

2.5 За on-premises та cloud моделями

Вибір між локальними та хмарними рішеннями IPAM є важливим аспектом при впровадженні системи у мережі підприємства.

On-premises рішення передбачають встановлення програмного забезпечення та зберігання даних на власних серверах підприємства. Це традиційний підхід, який надає організаціям повний контроль над їхніми даними та інфраструктурою. Серед переваг можна виділити такі пункти:

- Контроль і безпека – організації мають повний контроль над своїми даними, що дозволяє забезпечити високий рівень безпеки та відповідності внутрішнім політикам та регуляторним вимогам.
- Продуктивність – відсутність залежності від інтернет-з'єднання забезпечує стабільну роботу і швидкий доступ до даних.

Серед недоліків можна виділити:

- Високі початкові витрати – інвестиції в апаратне забезпечення, програмне забезпечення та людино-години можуть з’їсти немалий шматок бюджету.
- Витрати на обслуговування – постійні витрати на підтримку та оновлення інфраструктури, з урахуванням заробітної плати персоналу.
- Ускладнена масштабованість – масштабування інфраструктури може бути складним та витратним процесом.

Хмарні рішення передбачають використання інфраструктури та послуг, що надаються сторонніми постачальниками хмарних сервісів. Дані та додатки зберігаються в хмарі, і доступ до них здійснюється через інтернет.

Серед переваг можна виділити:

- Низькі початкові витрати – відсутність необхідності в інвестиціях у власну апаратну інфраструктуру. Платежі здійснюються за підпискою або за фактом використання ресурсів.
- Масштабованість – легке масштабування ресурсів через зміну потреб організації.

- Доступність – доступ до системи можна отримати з будь-якої точки світу, що спрощує управління та підвищує мобільність співробітників.
- Підтримка та оновлення – відповідальність за підтримку та оновлення інфраструктури лежить на постачальнику хмарних послуг, що зменшує навантаження на внутрішній персонал.

Серед недоліків можна виділити:

- Безпека – зберігання даних у хмарі може викликати занепокоєння щодо безпеки та конфіденційності даних, особливо для організацій з високими вимогами до безпеки.
- Залежність від інтернет-з'єднання: Продуктивність системи залежить від якості інтернет-з'єднання. Перебої в з'єднанні з хмарою можуть вплинути на доступність системи.
- Контроль – менший рівень контролю над інфраструктурою та даними, що може бути критично важливим для деяких організацій.

2.6 Короткий огляд існуючих систем

Infoblox є одним із найпоширеніших комерційних рішень DDI, пропонує комплексний набір функцій для автоматизації та централізації управління мережевими адресами. Цей продукт має високий рівень автоматизації процесів управління IP-адресами та DNS й DHCP сервісами, підтримує великі корпоративні мережі, має вбудовані функції безпеки, такі як захист від атак на DNS й DHCP та механізми аутентифікації, а також потужні інструменти для моніторингу та аналітики.

SolarWinds IP Address Manager є ще одним популярним комерційним рішенням для управління IP-адресами. Цей інструмент інтегрується з іншими продуктами SolarWinds, що забезпечує додаткові можливості для моніторингу та

управління мережевою інфраструктурою. Забезпечує автоматизацію багатьох рутинних завдань, має інтуїтивний інтерфейс, але може мати більш обмежену функціональність у порівнянні з дорожчими продуктами.

BlueCat Integrity є потужним рішенням для управління IP-адресами, DNS та DHCP сервісами. Цей продукт орієнтований на великі підприємства та забезпечує високий рівень автоматизації та безпеки.

Cygna Labs є провайдером рішень DDI, а також забезпечує аналітику та безпеку мережевої інфраструктури. Cygna Labs пропонують два продукти: Diamond IP, який був придбаний у British Telecom у 2022 році [24] та VitalQIP, який був розроблений Nokia, але був проданий Cygna Labs у 2023 році [25]. Cisco використовують Diamond IP для демонстрації інтеграції IPAM з Cisco Prime Network Registrar [26], з чого можна зробити висновок, що вони мають хороший рівень інтеграції.

phpIPAM є безкоштовним та відкритим рішенням для управління IP-адресами, надає широкий набір функцій для управління адресним простором та легко розширюється завдяки своїй модульній архітектурі. Може бути хорошим рішенням для невеликих підприємств, але відсутність офіційної підтримки та обмеження по масштабованості та надійності може бути недостатнім для великих корпоративних мереж.

Такі сервіси як AWS та Microsoft Azure пропонують власні рішення для управління IP-адресами у своїх хмарних середовищах: AWS IP Address Manager та Azure IPAM відповідно.

2.7 Висновки до розділу 2

Отже, вибір системи IPAM за ціною має вирішуватися з урахуванням конкретних потреб, бюджетних обмежень та стратегічних цілей організації. Важливо збалансувати вартість рішення з його функціональністю, оскільки при нескладній побудові мережі можливо слід придивитися до простіших рішень з більш обмеженими можливостями, а не «стріляти гарматою по горобцях». Важливо також враховувати, який рівень автоматизації та інтеграції найбільше відповідає потребам мережі та забезпечить оптимальну ефективність її управління. В цьому питанні один з ключових факторів – це наявна та запланована в майбутньому інфраструктура, оскільки деякі програмні продукти будуть мати вельми обмежену функціональність або не працюватимуть взагалі з наявними елементами мережі, використання рішень від одного виробника зазвичай гарантують сумісність систем. Вибір системи IPAM з погляду доступності та безпеки повинен бути здійснений з урахуванням важливості рівня відмовостійкості та чутливості даних.

Розділ 3. Створення власного застосунку

3.1 Архітектура застосунку

Було вирішено розробити вебзастосунок, оскільки такий формат має цілий ряд переваг:

1. Клієнт-серверна архітектура – тут клієнтська частина являє собою графічний інтерфейс, який відображається у браузері, а серверна частина оброблює запити, які надходять від клієнта, зберігає та проводить обробку даних, надсилає відповіді клієнту.
2. Незалежність від платформи – через те, що запити надходять від браузерів, гарантується незалежність платформи для клієнтів, оскільки браузери доступні на всіх головних операційних системах, як настільних, так і мобільних пристроїв.
3. Легкість розробки користувацького інтерфейсу – через те, що у вебдодатках користувацький інтерфейс, по суті, є сторінками веб-браузера, а вони складаються з текстових файлів у форматі HTML, то розробка графічного інтерфейсу спрощується та пришвидшується.

Отже, структура вебзастосунку буде включати клієнтську та серверну частини. Клієнтська частина буде представлена вебінтерфейсом, який взаємодіє з сервером через браузер. Це може бути статична вебсторінка, яка відображає інформацію про IP-адреси. Серверна частина включає в себе програмну логіку для обробки запитів вебінтерфейсу, взаємодії з базою даних для збереження та витягування інформації про IP-адреси, а також забезпечення безпеки та аутентифікації користувачів. Застосунок матиме рівноцінну підтримку IPV4 та IPV6.

У вебзастосунку для керування розподілом адресного простору база даних відіграє ключову роль у збереженні та організації інформації про IP-адреси, пристрої та інші мережеві ресурси. Добре спроєктована база даних дозволяє

ефективно здійснювати операції з керування адресним простором та забезпечує швидкий доступ до необхідної інформації. Використання реляційної бази даних забезпечує надійне зберігання та швидкий доступ до інформації. Також можуть використовуватися інші типи баз даних, залежно від конкретних потреб та обмежень проєкту.

Також важливим компонентом системи буде черга завдань – модуль, який буде виконувати періодичні завдання, такі як запити до DNS-сервера та відстеження стану вузлів.

Для отримання доменних імен буде створено емулятор, який міститиме API, роблячи запити з IP-адресами до якого, можна буде отримати доменне ім’я.

Для відстеження стану вузлів буде використана функція, яка робитиме ICMP запити.

Отже, структура застосунку буде мати наступний вигляд:

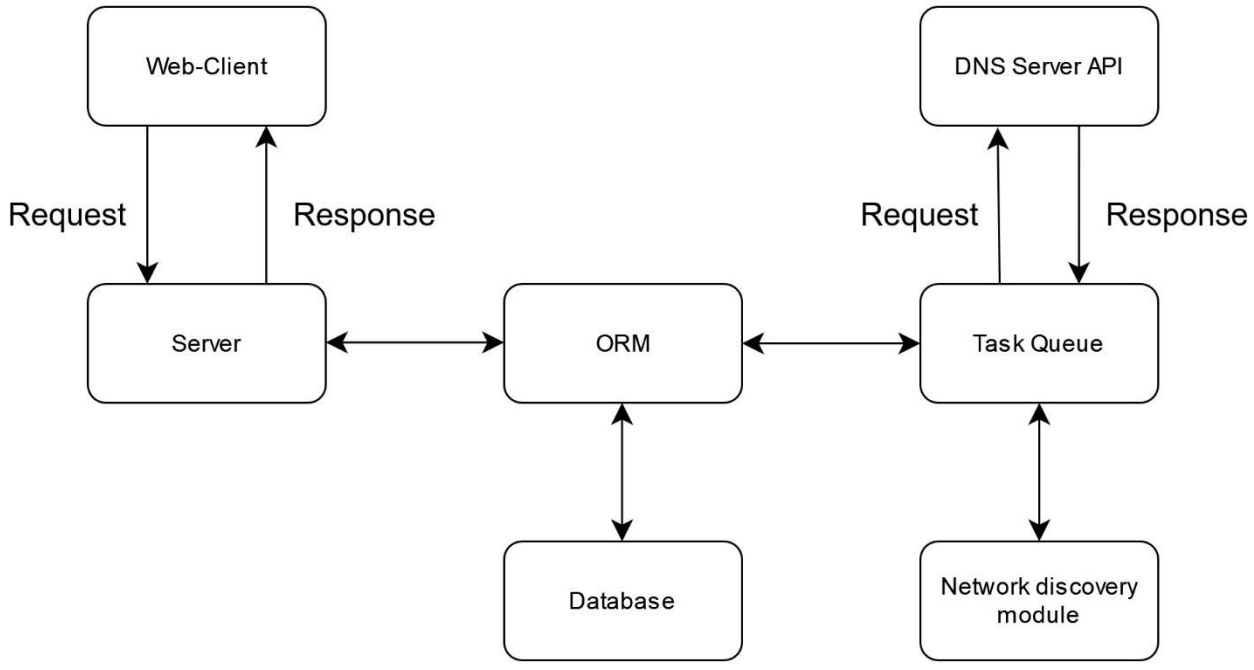

Рис.3.1 Модель структури застосунку

3.2 Інструменти, обрані для реалізації застосунку

У процесі розробки вебзастосунку для керування розподілом адресного простору були обрані різні актуальні інструменти, що надають можливості для зручного та ефективного створення і підтримки програмного забезпечення. Основні інструменти, використані в розробці, включають:

- Python – був обраний як основна мова програмування для розробки вебзастосунку IPAM. Python – це високорівнева, інтерпретована мова програмування, відома своєю простотою, читабельністю коду та широким спектром наявних бібліотек, що сприяє швидкій розробці та підтримці проєкту. Ця мова є однією з найпопулярніших мов програмування і станом на дві тисячі двадцять третій рік була другою за популярністю мовою на гітхабі, поступаючись лише JavaScript[27]. Завдяки тому, що це інтерпретована мова, код написаний на Python можна запускати у будь-якому середовищі, де встановлено Python.
- Django – це вебфреймворк, побудований на мові Python, який надає зручний інструментарій для розробки вебзастосунків. Використання Django дозволяє прискорити розробку завдяки вбудованим функціям для роботи з базами даних, обробки URL-адрес, автентифікації користувачів [28]. Django-проєкт складається з основних файлів конфігурації проєкту та додатків.
- SQLite – було обрано як базу даних для вебзастосунку. SQLite є простою, реляційною базою даних, що працює локально на рівні файлу. Він ідеально підходить для розробки вебзастосунків невеликих та середніх розмірів. Пізніше можна легко замінити базу даних на MySQL або PostgreSQL
- virtualenv – це інструмент для створення ізольованих середовищ Python [29]. Використання Virtualenv дозволяє ізолювати залежності та бібліотеки для конкретного проєкту, що полегшує управління середовищами та уникнення

конфліктів між версіями пакетів. Використання цього інструменту дозволяє забезпечити чистоту та стабільність середовища розробки.

- Nmap (Network Mapper) – це відкрите програмне забезпечення для аналізу мережі та сканування портів. Воно дозволяє користувачам виявляти робочі пристрої у мережі, визначати відкриті порти та типи сервісів, які працюють на цих портах. Nmap надає розширені можливості для збору інформації про мережеві пристрої та ідентифікації потенційних проблем безпеки [30].
- Бібліотека netaddr – бібліотека netaddr надає засоби для роботи з мережевими адресами та підмережами у Python. Вона дозволяє легко виконувати операції над IP-адресами, перевіряти їх належність до певних підмереж, перевіряти валідність IP та MAC адрес.
- Бібліотека Bootstrap 5 – це набір інструментів для полегшення та пришвидшення розробки фронтенду, надає широкий набір готових компонентів, які можна використовувати для швидкої побудови гарних вебсторінок. Bootstrap пропонує гнучку сітку та компоненти, які автоматично адаптуються до різних розмірів екранів, забезпечуючи коректне відображення вебзастосунку на різних пристроях, від настільних комп'ютерів до смартфонів.
- Бібліотека Huey – це проста та зручна бібліотека для планування асинхронних, паралельних та періодичних завдань у Python. Huey дозволяє розподілити завдання між кількома потоками, що дозволяє підвищити продуктивність та швидкість виконання задач. В цьому випадку використовується для виконання періодичних запитів.

Використання цих інструментів дозволить спростити розробку та підтримку застосунку, забезпечуючи швидкий розвиток та ефективну роботу програми, а також дозволить використовувати її незалежно від операційної системи. Актуальні версії операційних систем Windows та Linux зможуть запускати серверну частину,

оскільки всі використані програми та бібліотеки є незалежними від операційних систем.

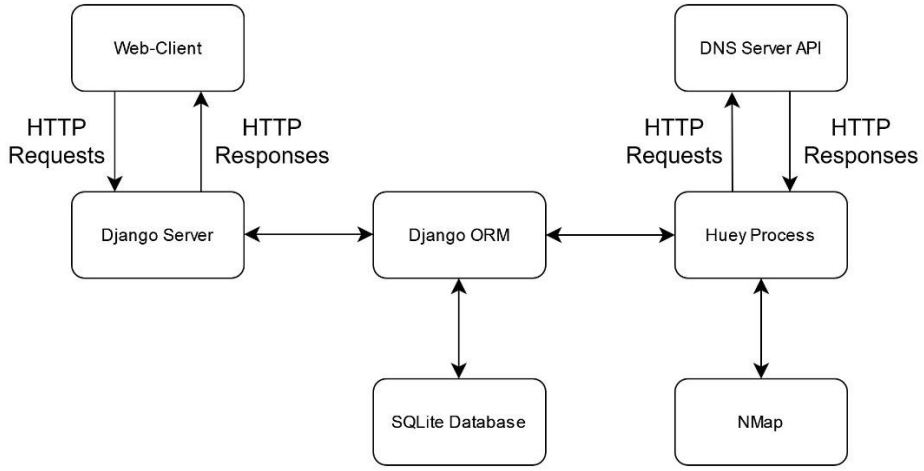

Рис. 3.2 Модель структури застосунку з урахуванням використаних інструментів

3.3 Опис основних модулів та їх функцій

1) urls.py – елемент у структурі Django-проєкту, який забезпечує маршрутизацію запитів до відповідних обробників. Завдяки йому, можна легко управляти шляхами та забезпечувати гнучкість і масштабованість вебдодатку. Чітка організація маршрутів у urls.py спрощує розширення функціональності та підтримку коду в майбутньому.

```
from django.contrib import admin
from django.urls import path, include

urlpatterns = [
    path('', include('dashboard.urls')),
    path('admin/', admin.site.urls),
]
```

Рис. 3.3 urls.py рівня проєкту

```
urlpatterns = [
    path('', views.main, name='main'),
    path('dashboardv4/', views.dashboardv4, name='dashboardv4'),
    path('dashboardv4/entrydetailsv4/<int:id>', views.entrydetailsv4, name='entrydetailsv4'),
    path('dashboardv6/', views.dashboardv6, name='dashboardv6'),
    path('dashboardv6/entrydetailsv6/<int:id>', views.entrydetailsv6, name='entrydetailsv6'),
    path('subnetsv4/', views.subnetsv4, name='subnetsv4'),
    path('subnetsv6/', views.subnetsv6, name='subnetsv6'),
    path('subnetsv4/subnetdetailsv4/<int:id>', views.subnetdetailsv4, name='subnetdetailsv4'),
    path('subnetsv6/subnetdetailsv6/<int:id>', views.subnetdetailsv6, name='subnetdetailsv6'),
]
```

Рис. 3.4 urls.py рівня додатку

Можна побачити (рис. 3.4), що елемент path складається зі шляху, функції, яка відповідає за обробку запитів та назви, яку можна використовувати замість повного шляху.

2) admin.py – компонент Django-проєкту, який визначає, як моделі додатка відображаються та управляються в адміністративній панелі Django. Панель адміністрування надає зручний інтерфейс для додавання, редагування та видалення записів бази даних. Для того, щоб таблиця бази даних з'явилася в адміністративній панелі, її необхідно зареєструвати у файлі admin.py, тут також можна налаштувати, які поля зможе коригувати адміністратор, за якими полями можна буде здійснювати пошук та які фільтри будуть наявні. Грамотне налаштування адміністративної панелі значно спрощує розробку та підтримку вебдодатків, забезпечуючи зручний інтерфейс для взаємодії з базою даних.

```
class EntryAdmin(admin.ModelAdmin):
    list_display = ("id", "ipaddr", "subnetip", "name", "location", "description", "active")
    readonly_fields = ("date_changed", "active")
    list_filter = ("subnetip", "location")
    search_fields = ("ipaddr", "subnetip", "name", "location", "description",)


admin.site.register(EntryV4, EntryAdmin)
admin.site.register(EntryV6, EntryAdmin)
```

Рис. 3.5 Приклад налаштування таблиць у admin.py

В прикладі (рис. 3.5) можна побачити, які поля будуть відображені у панелі адміністрації, які поля не можна редагувати вручну, які фільтри доступні та за якими полями буде відбуватися пошук.

3) models.py – компонент Django-проєкту, який визначає структуру бази даних за допомогою моделей. Моделі в Django описують поля та поведінку збережених даних, забезпечуючи ORM функціональність, яка дозволяє взаємодіяти з базою даних як з об'єктами Python. Також тут відбувається валідація вхідних даних та створення сигналів ORM.

```python
class EntryV4(models.Model):
    subnetip = models.CharField(max_length=18, verbose_name="SubNet IP Address")
    ipaddr = models.GenericIPAddressField(protocol="IPv4", unique=True, verbose_name="IPv4 Address")
    macaddr = models.CharField(max_length=17, default='-', verbose_name="MAC Address")
    vlan = models.CharField(max_length=16, default='-', verbose_name="VLAN")
    name = models.CharField(max_length=64, verbose_name="Device Name")
    location = models.CharField(max_length=64, verbose_name="Location")
    date_changed = models.DateField(max_length=16, auto_now=True, verbose_name="Changed on")
    url = models.CharField(max_length=128, default='-', verbose_name="URL")
    description = models.CharField(max_length=256, verbose_name="Description")
    active = models.BooleanField(default=False, verbose_name="Is Online")
```

Рис. 3.6 Приклад моделі у models.py

Ось (рис. 3.6) приклад моделі в Django, тут можна побачити назви полів, їх типи та параметри – такі, як максимальна довжина, значення за замовчуванням, унікальність, автоматичне заповнення та описова назва.

```python
def clean(self):
    if str(self.macaddr) != '-' and not valid_mac(str(self.macaddr)):
        raise ValidationError("Incorrect Mac Address")

    subnets = SubnetV4.objects.filter(subnetip=str(self.subnetip)).values()
    if len(subnets) == 0:
        raise ValidationError("This Subnet does not exist")

    for subnet in subnets:
        if IPAddress(str(self.ipaddr)) not in IPNetwork(f'{self.subnetip}/{subnet.get("subnetmaskcidr")}'):
            raise ValidationError("Address Not In Subnet!")
```

Рис. 3.7 Приклад валідації у models.py

Тут (рис. 3.7) можна побачити приклад валідації, а саме: валідація формату MAC-адреси, якщо вона вказана, валідація існування підмережі, якій має належати ця адреса й валідація належності IP-адреси до підмережі.

```
@receiver(post_delete, sender="dashboard.SubnetV6")
def delete_all_entries(sender, instance, using, **kwargs):
    EntryV6.objects.filter(subnetip=instance.subnetip).all().delete()
```

Рис. 3.8 Приклад сигналу post_delete у models.py

У цьому прикладі (рис. 3.8) наведено приклад сигналу post_delete, як зрозуміло із назви, він викликається після видалення моделі, в такому випадку після видалення підмережі IPV6 буде видалено всі адреси цієї підмережі.

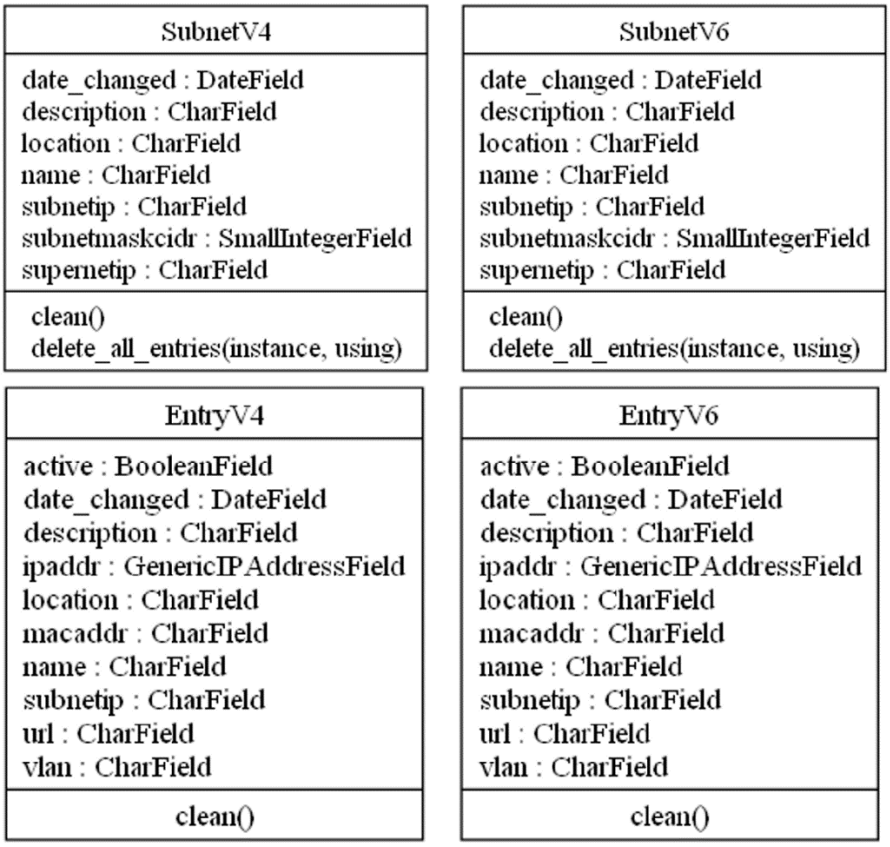

Рис. 3.9 Модель класів у models.py

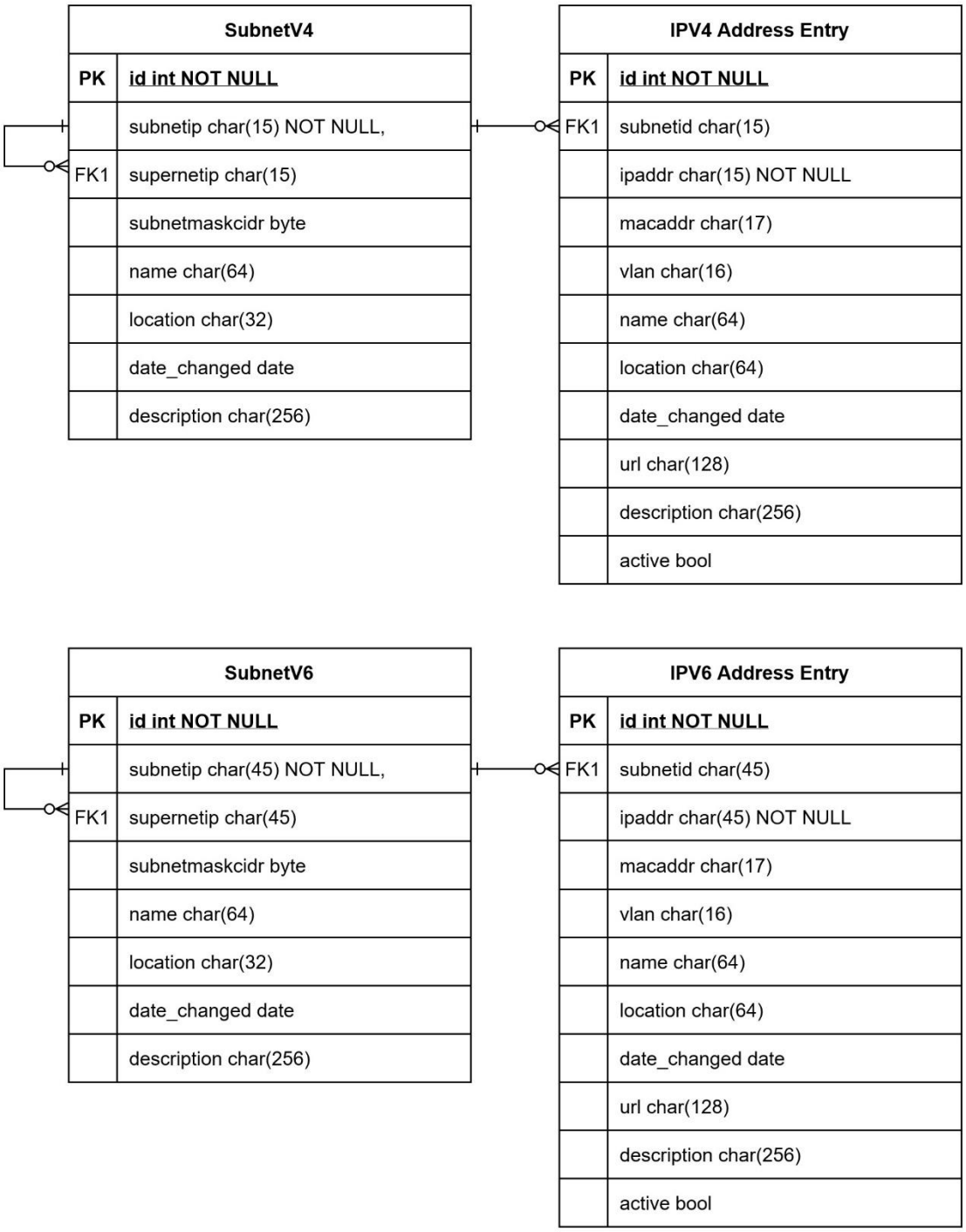

Рис. 3.10 ER діаграма класів у models.py

4) views.py – компонент у структурі Django-проєкту, який визначає логіку обробки запитів і відповіді на них. У цьому файлі визначаються функції або класи, які відповідають за обробку HTTP-запитів від користувачів, генерацію відповідних відповідей та заповнення HTML-шаблонів.

```python
def entrydetailsv6(request, id):
    myentry = EntryV6.objects.get(id=id)
    template = loader.get_template('entrydetailsv6.html')
    context = {
        'myentry': myentry
    }
    return HttpResponse(template.render(context, request))
```

Рис. 3.11 Приклад обробки запиту і відправлення відповіді у views.py

У наведеному прикладі (рис. 3.11) можна побачити обробку запиту за шляхом «entrydetailsv6»: функція отримує HTTP-запит та id з URL, після чого функція робить запит в ORM з цим id, результат запиту завантажує у HTML файл та відправляє результат клієнту.

5) tasks.py – модуль, який відповідає за виконання періодичних завдань, у цьому файлі визначаються функції та час їх виконання.

```python
@db_periodic_task(crontab(minute='*'))
def check_if_online_ipv6():
    ipv6s = EntryV6.objects.all()
    nm = nmap.PortScanner()
    for entry in ipv6s:
        nm.scan(hosts=entry.ipaddr, arguments='-sn -disable-arp-ping')
        x = EntryV6.objects.filter(ipaddr=entry.ipaddr).first()
        if len(nm.all_hosts()) != 0:
            x.active = "True"
        else:
            x.active = "False"
        x.save(update_fields=["active"])
```

Рис. 3.12 Приклад періодичної задачі перевірка роботи пристрою у tasks.py

No.	Time	Source	Destination	Protocol	Length	Info
1949	4.069865	ASUSTekCOMPU_1f:d2:…	Broadcast	ARP	42	Who has 192.168.0.26? Tell 192.168.0.143
1950	4.114878	b2:c0:09:6f:dc:ff	ASUSTekCOMPU_1f:d2:…	ARP	60	192.168.0.26 is at b2:c0:09:6f:dc:ff

Рис. 3.13 ARP-запит та відповідь

No.	Time	Source	Destination	Protocol	Length	Info
79	3.202968	192.168.0.143	192.168.0.26	ICMP	42	Echo (ping) request id=0x4137, seq=0/0, ttl=37 (reply in 95)

Рис. 3.14 ICMP запит

На рисунку 3.12 наведено приклад функції, яка робить запити по IPV6-адресам, які містяться у базі даних та робить ICMP Echo-запити (рис. *3.*14) до них, оскільки серед аргументів є «-disable-arp-ping», інакше б спрацьовував ARP-запит (рис. *3.*13)[31], у разі відповіді у поле «active» записується «True», інакше – «False». Дана функція спрацьовує через хвилину. Така сама функція є і для IPV4 і при запуску Huey із двома і більше працівниками вони будуть відпрацьовувати паралельно.

```python
@db_periodic_task(crontab(hour='*'))
def make_dns_requests_ipv4():
    ipv4s = EntryV4.objects.all()
    for entry in ipv4s:
        payload = {'ip': entry.ipaddr}
        try:
            response = requests.get( url: f'http://{dns_ip_address}/dns4', params=payload)
            x = EntryV4.objects.filter(ipaddr=entry.ipaddr).first()
            if response.status_code == 200:
                x.url = response.json()["url"]
            else:
                x.url = '-'
            x.save(update_fields=["url"])
        except requests.exceptions.ConnectionError:
            print("Could not establish connection with a dns server")
```

Рис. 3.15 Приклад періодичної задачі звернення до DNS API

На рисунку 3.15 наведено приклад зворотного запиту DNS до DNS API (Додаток А). Якщо такий запис є, то з відповіді у форматі JSON дістається URL і додається в базу даних. Ця функція спрацьовує раз на годину.

3.4 Інтерфейс користувача та його функціональність

Інтерфейс користувача є критично важливою частиною у будь-якому вебдодатку, оскільки він визначає, як користувачі взаємодіють з системою. У цьому підрозділі розглянуто основні аспекти дизайну та реалізації інтерфейсу користувача для вебдодатка IPAM. Для розробки інтерфейсу користувача

використовується підхід, орієнтований на зручність і зрозумілість. Основні принципи дизайну включають:

- Простота і мінімалізм – інтерфейс повинен бути інтуїтивно зрозумілим і не перевантаженим зайвою інформацією.
- Консистентність – використання єдиних стилів та елементів управління на всіх сторінках додатка.
- Швидкодія – користувач не повинен чекати пів хвилини на завантаження сторінки.
- Адаптивність – забезпечення коректного відображення та функціональності на пристроях різного розміру та формату.

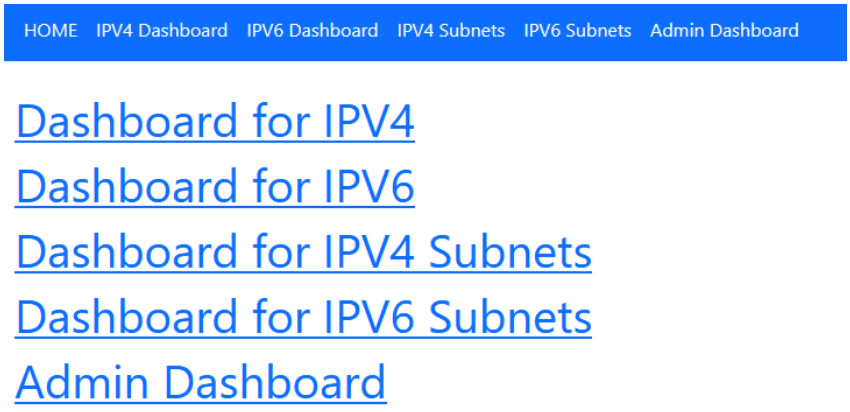

Рис. 3.16 Вигляд головної сторінки

На головній сторінці (рис. 3.16) є посилання на список усіх IPV4 адрес, IPV6 адрес, IPV4 підмереж, IPV6 підмереж, а також посилання на сторінку адміністрування. Зверху можна побачити панель навігації, вона містить ті самі посилання.

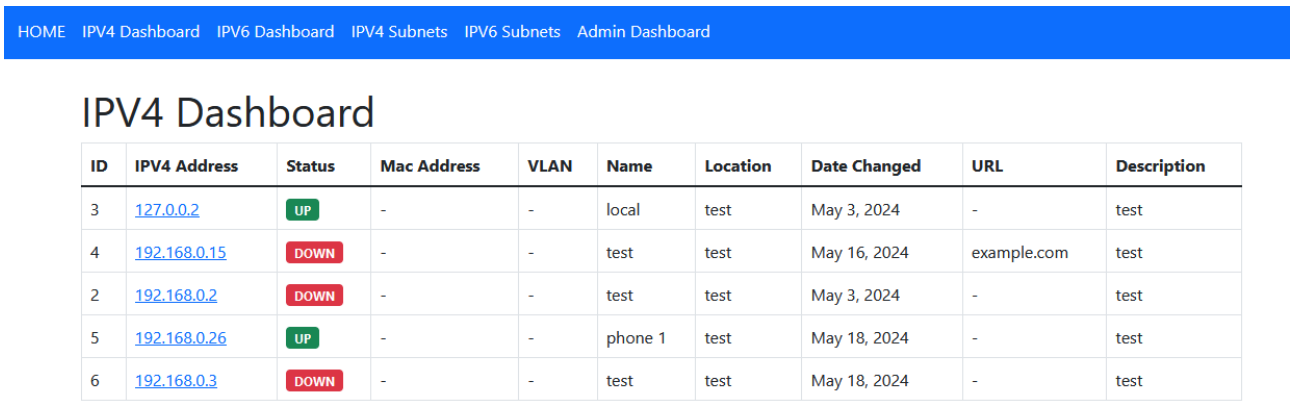

Рис. 3.17 Вигляд списку IPV4 адрес

На сторінці перегляду всіх IPV4 адрес (рис. 3.17) знаходиться таблиця, яка складається з ID, IP-адреси, показника чи є зв'язок з пристроєм, фізичної адреси, VLAN, назви, локації, дати останньої зміни, URL та опису.

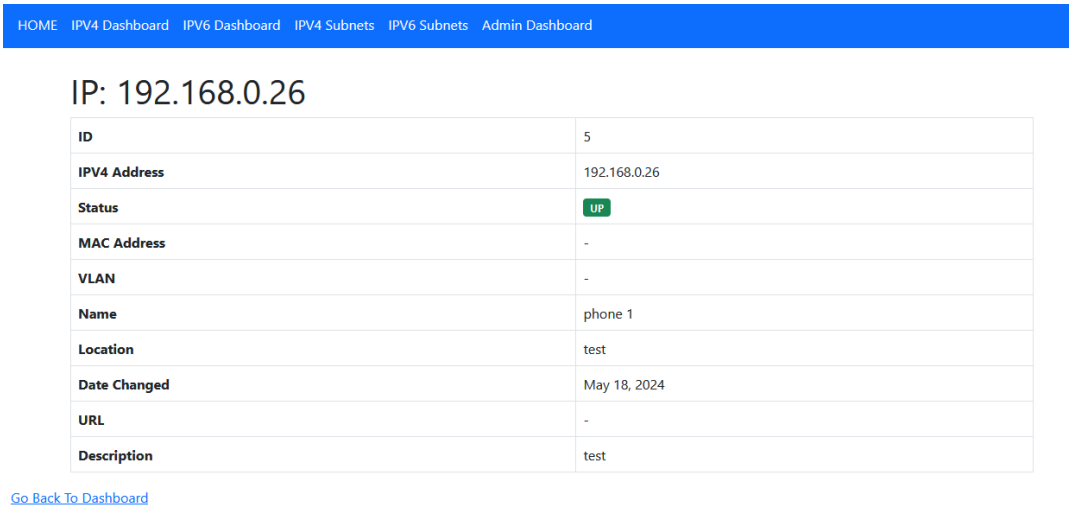

Рис. 3.18 Вигляд сторінки про деталі IPV4 адреси

Якщо натиснути на IP-адресу, то відкриється сторінка (рис. 3.18) з даними саме про конкретну IP адресу, на цю мить ніякої додаткової інформації тут немає.

Зі списком та детальною сторінкою IPV6-адрес ситуація аналогічна.

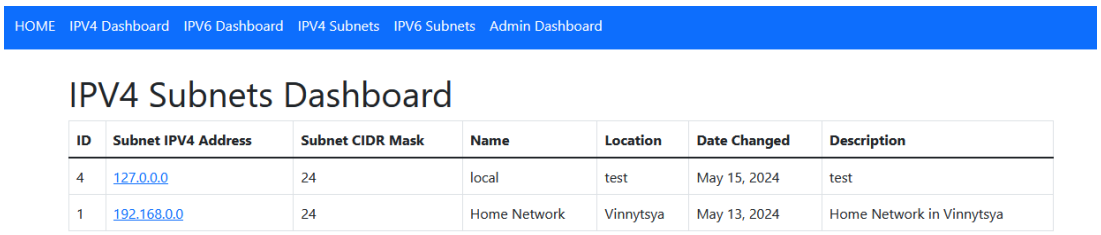

Рис. 3.19 Вигляд сторінки списку IPV4-підмереж

На сторінці списку IPV4-підмереж (рис. 3.19) показуються всі підмережі, надмережа у яких не вказана. У таблиці знаходиться наступна інформація: IP-адреса підмережі, маска підмережі у форматі CIDR[32], назва, локація, дата останньої зміни та опис.

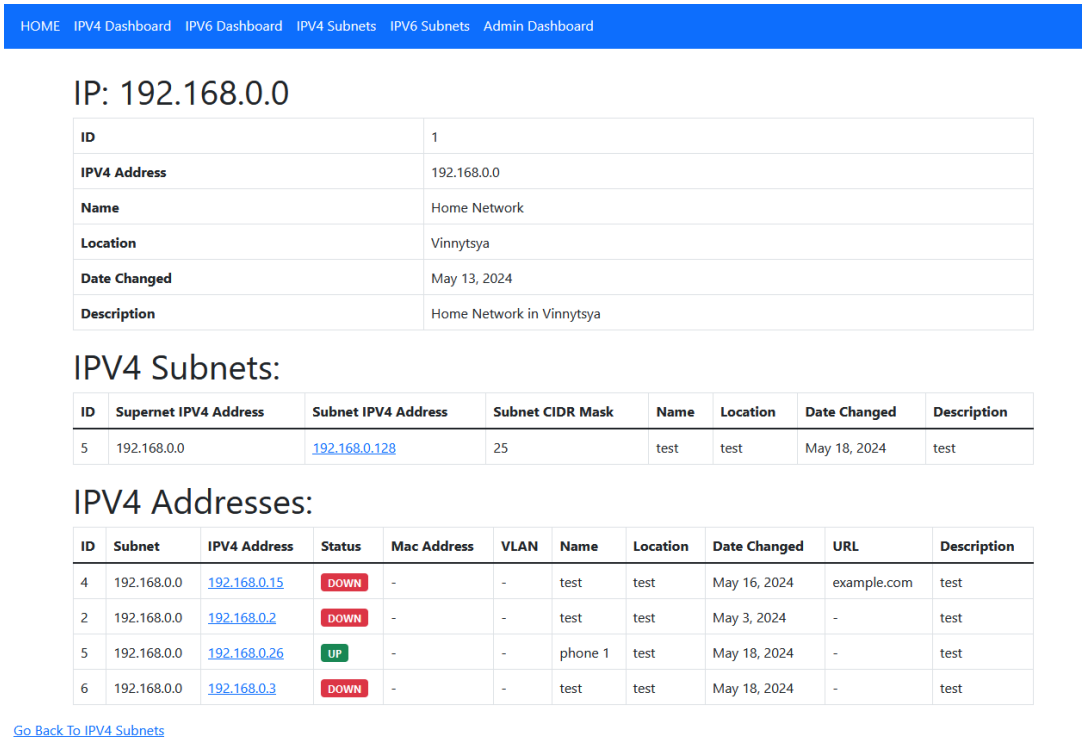

Рис. 3.20 Вигляд сторінки з деталями підмережі

На сторінці з деталями підмережі (рис. 3.20) видно інформацію про цю підмережу, список підмереж у цій підмережі та список IP-адрес у цій підмережі.

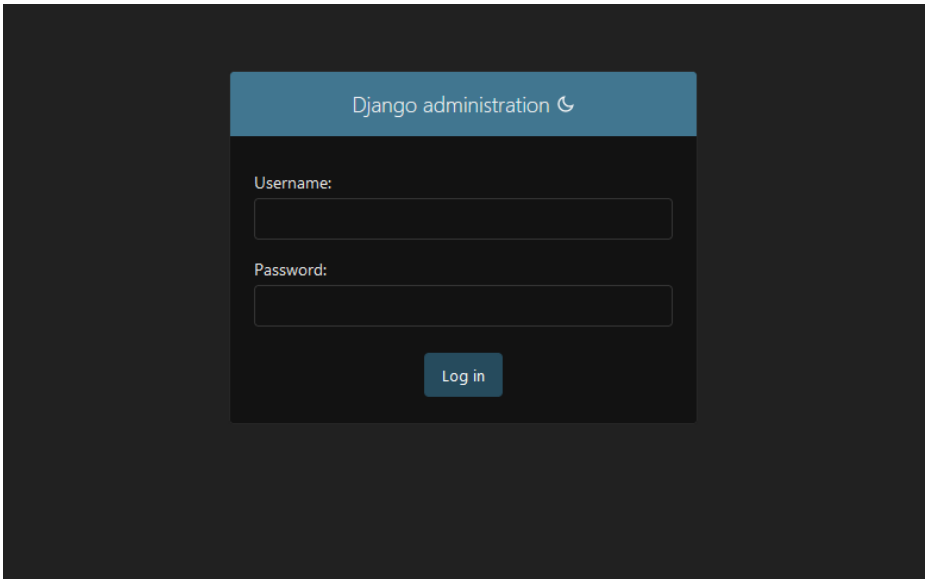

Рис. 3.21 Форма входу в панель адміністрації

Якщо спробувати зайти в панель адміністрації, то при умові, що клієнт ще не увійшов у свій обліковий запис, йому буде показана така форма входу (рис. 3.21), тут треба ввести свої логін та пароль.

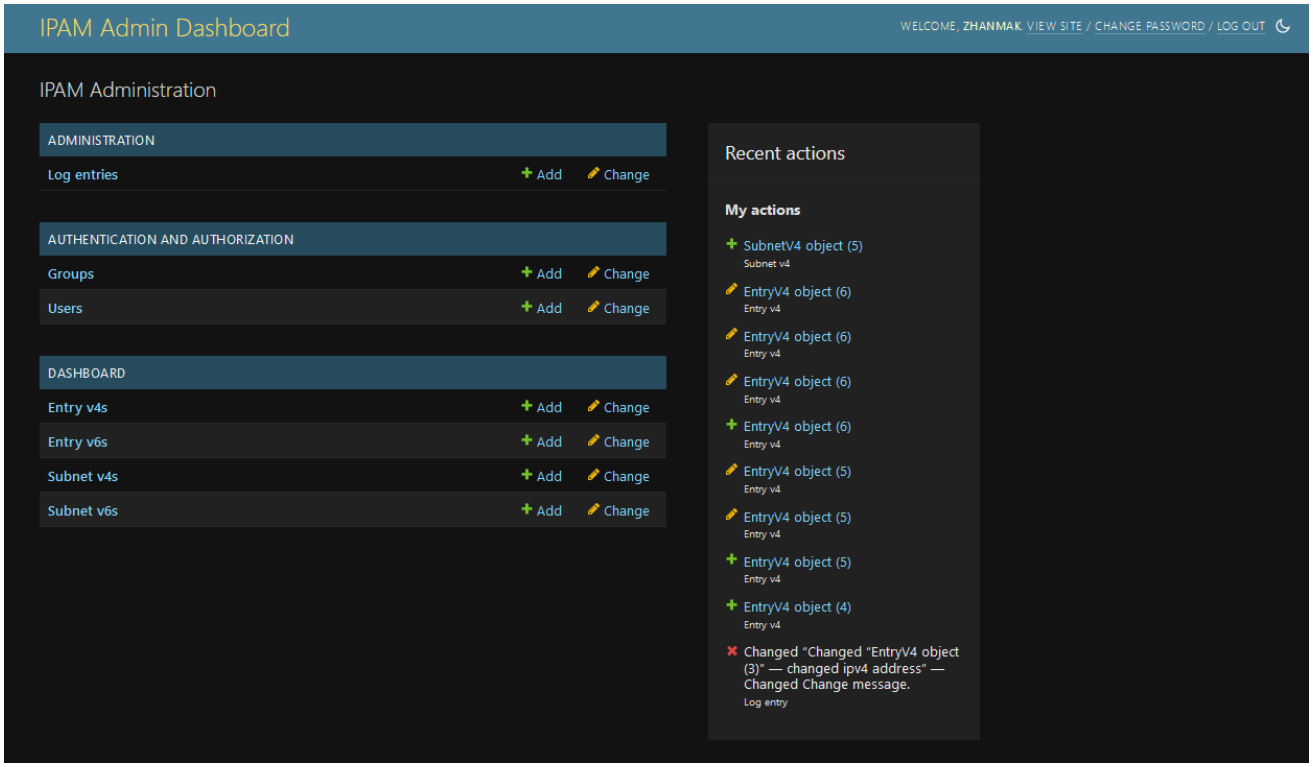

Рис. 3.22 Головна сторінка адміністрації

На головній сторінці адміністрації (рис. 3.22) можна перейти на сайт, змінити пароль, вийти зі свого облікового запису, поміняти тему на темну, світлу й тему браузера. Також можна перейти до журналу змін, налаштувань групових політик, налаштувань користувачів та налаштувань таблиць IPAM.

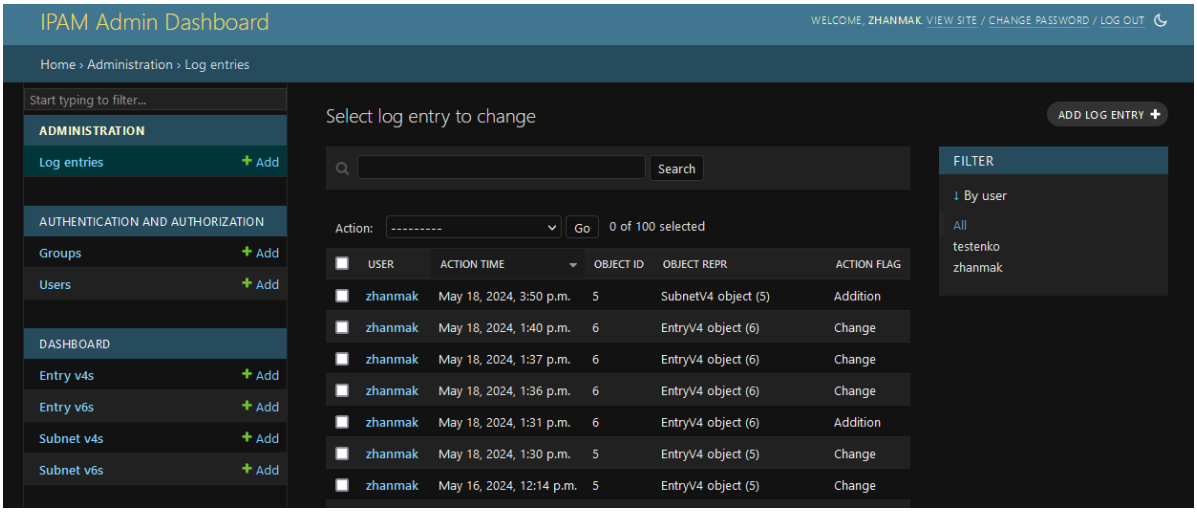

Рис. 3.23 Сторінка списку змін

На сторінці змін (рис. 3.23) можна переглядати хто, що і коли зробив, є можливість пошуку та фільтрації за користувачами.

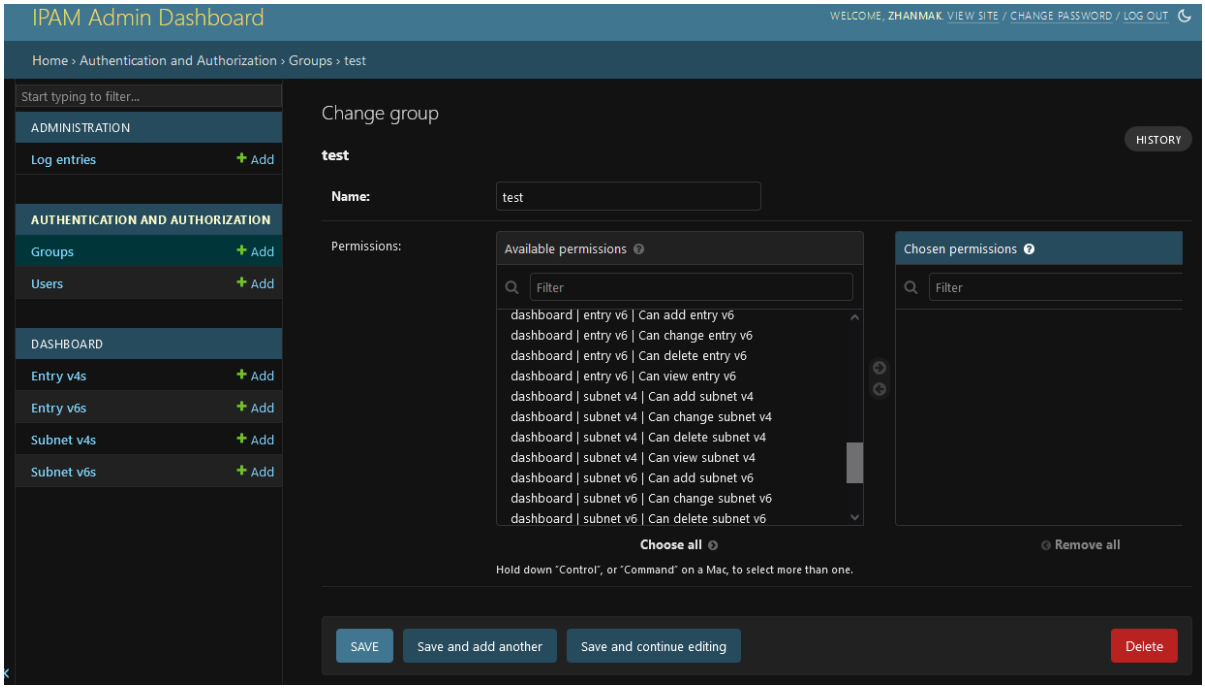

Рис. 3.24 Сторінка зміни групових політик

На сторінці зміни групових політик (рис. 3.24) можна встановлювати які права будуть мати користувачі цієї групи, а саме: право перегляду записів конкретної моделі, право додавати, редагувати та видаляти записи моделі.

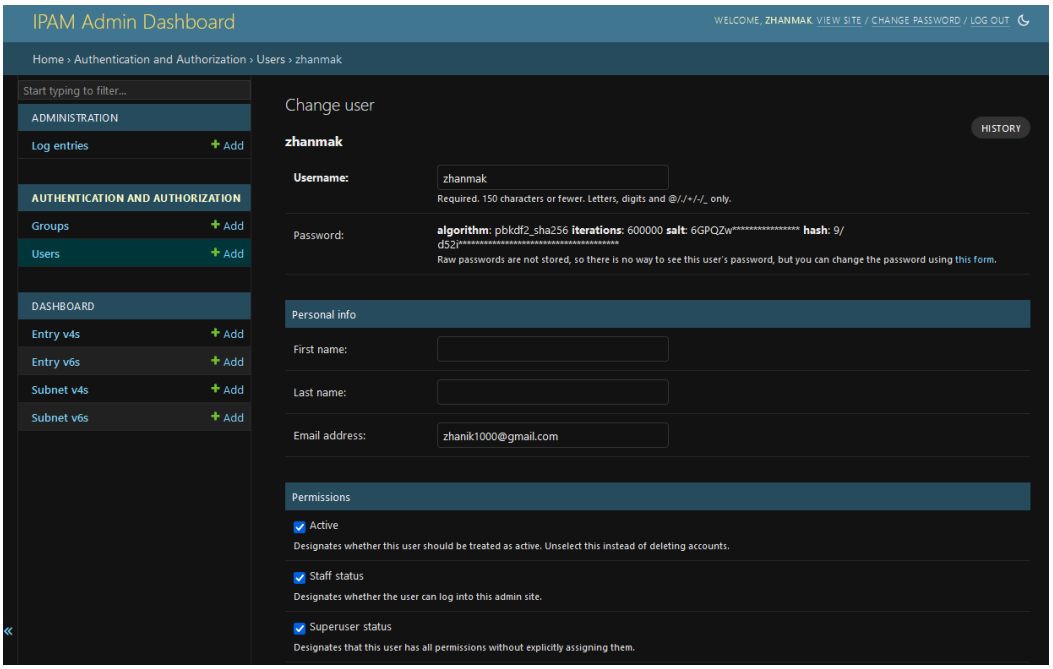

Рис. 3.25 Сторінка зміни користувачів

На сторінці зміни налаштувань користувачів (рис. 3.25) можна змінити логін, пароль, електронну пошту, ім’я, прізвище, дозволити користувачу входити в адміністративну панель, давати статус суперюзера, давати дозволи на взаємодію з моделями.

На сторінці зі списком користувачів можна здійснювати пошук і фільтрувати користувачів за статусом робітника, суперкористувача, дійсністю облікового запиту та групами.

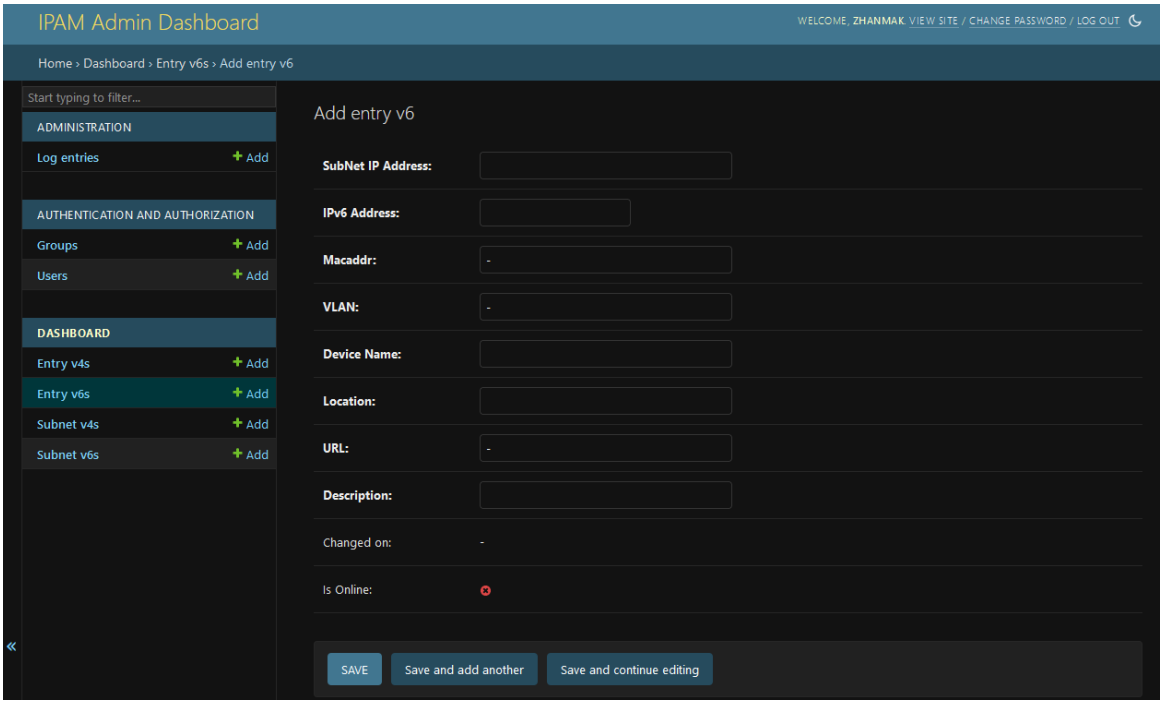

Рис. 3.26 Сторінка зміни запису IPV6-адреси

На сторінках зміни IP-адреси кінцевого пристрою (рис. 3.26) можна створювати, редагувати та видаляти записи.

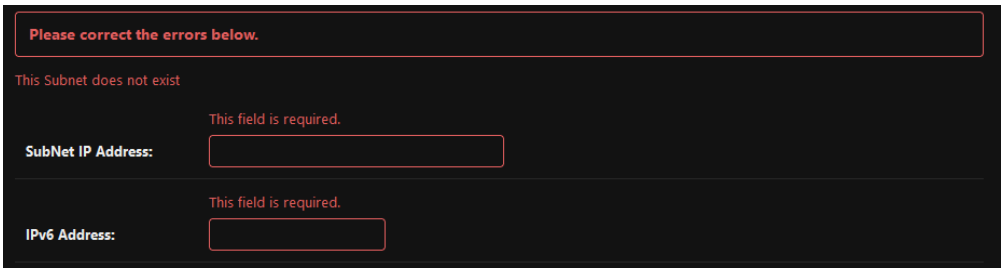

Рис. 3.27 Повідомлення про помилку

При невірно введених даних, форма підсвітить поля з помилками та висвітить першу помилку, в цьому випадку (рис. 3.27), що немає підмережі з такою адресою.

На сторінці зі списком IP-адрес можна здійснювати пошук та фільтрувати за підмережею та локацією.

На сторінках зі списком підмереж можна робити пошук та фільтрувати за мережами й локацією.

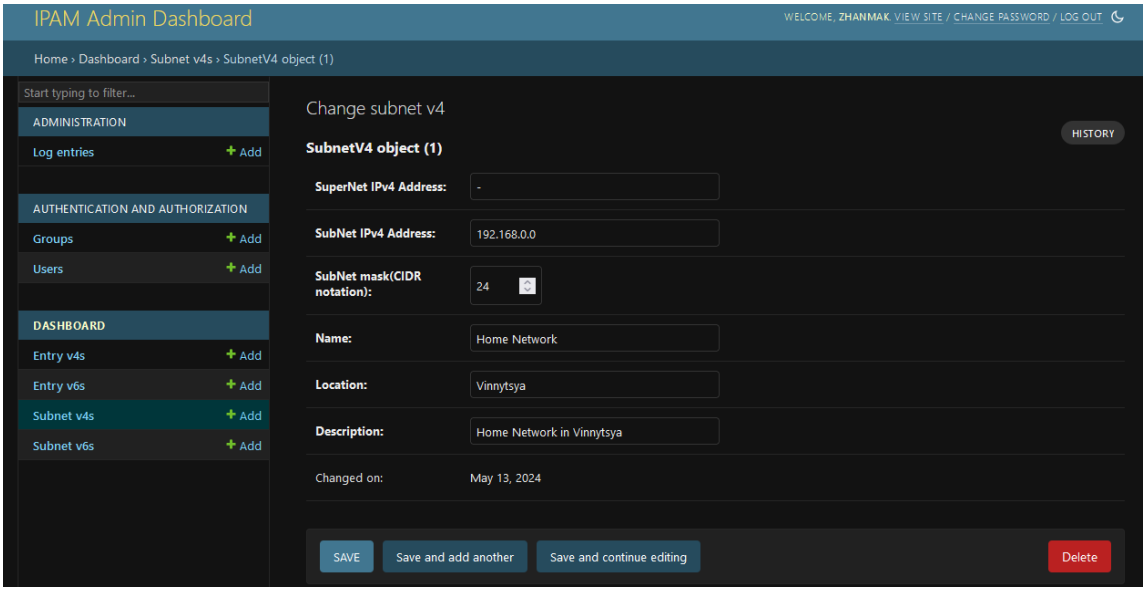

Рис. 3.28 Сторінка зміни запису IPV4-підмережі

На сторінках зміни підмереж (рис. 3.28) можна створювати, редагувати та видаляти записи. Валідація також присутня.

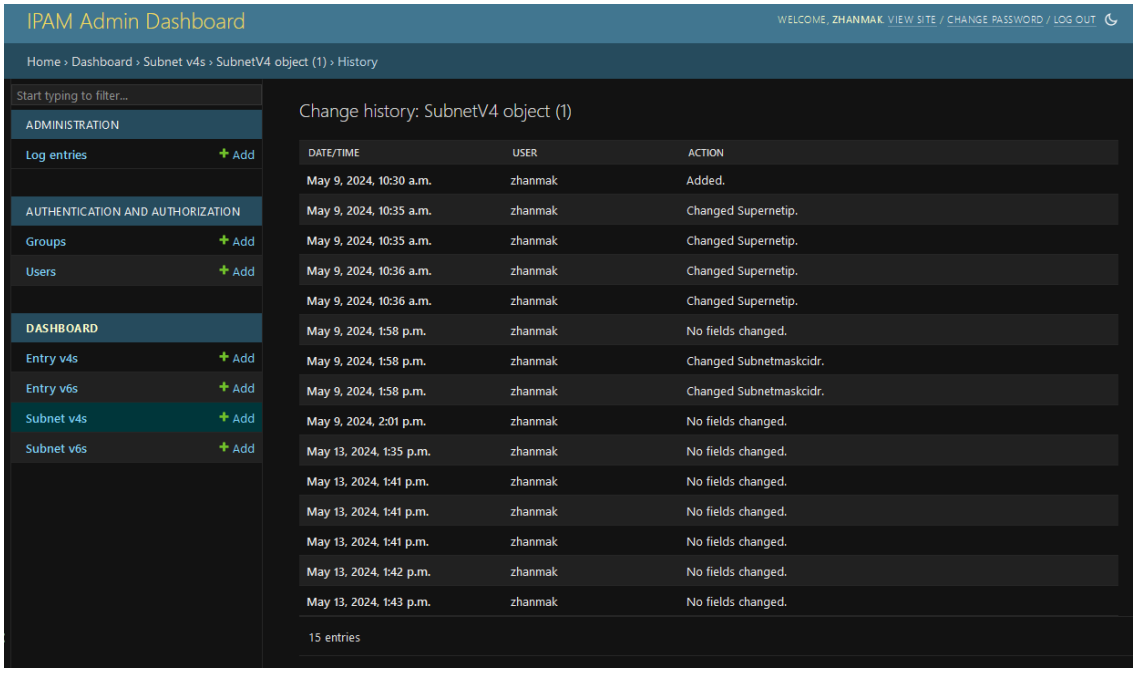

Рис. 3.29 Історія зміни запису

Для кожного запису в моделях є можливість подивитися історію змін (рис. 3.29).

3.5 Висновки до розділу 3

У розділі 3 було розглянуто створення власного застосунку для керування розподілом адресного простору у мережі підприємства. Основні аспекти цього розділу включали структуру застосунку, інструменти, використані для реалізації, дизайн інтерфейсу користувача та функціональність програми. Було обґрунтовано вибір вебзастосунку для реалізації системи IPAM, враховуючи його доступність, масштабованість та можливість централізованого управління, описано архітектуру застосунку, включаючи ключові компоненти, такі як models.py, views.py, urls.py, admin.py, та їхню роль у функціонуванні системи, описано інструменти, обрані для реалізації застосунку.

Висновок

У цій роботі було розроблено систему керування розподілом адресного простору для мережі підприємства. На початковому етапі було проведено аналіз предметної області, розглянуто основні компоненти мережевої інфраструктури, такі як IP-адреси, DNS, DHCP, ARP та ICMP. Це дозволило краще зрозуміти вимоги до системи та визначити критерії оцінки наявних рішень для управління адресним простором, зокрема за ціною, рівнем автоматизації, інтеграції, доступності та безпеки.

Після аналізу предметної області було розпочато розробку власного застосунку. Вибір вебзастосунку як оптимального рішення був обґрунтований його здатністю забезпечувати централізоване управління IP-адресами, легкістю розробки вебінтерфейсу та невибагливості до пристроїв користувачів. Для реалізації застосунку було використано сучасні інструменти й технології, такі як Python, Django, SQLite, Bootstrap 5, netaddr, virtualenv, nmap, huey та requests. Кожен з цих інструментів був обраний з урахуванням його переваг та можливостей, що дозволило створити надійну та ефективну систему, яка може запускатися на локальних пристроях з актуальними версіями операційних систем Linux та Windows, сервер також теоретично має можливість працювати й на MacOS.

Дизайн інтерфейсу користувача був розроблений з урахуванням принципів простоти, цілісності та адаптивності. Використання Bootstrap 5 дозволило створити зручний та інтуїтивно зрозумілий інтерфейс, який забезпечує позитивний досвід користувача. Основні сторінки та функції інтерфейсу, такі як перегляд, додавання, редагування та видалення IP-адрес, були детально описані та реалізовані з урахуванням потреб користувачів.

У процесі розробки програми було забезпечено можливість управління користувачами та їхніми правами доступу, що підвищує рівень безпеки системи.

Ведення аудиту активності користувачів дозволяє контролювати доступ до критично важливої інформації та забезпечувати надійний захист мережі підприємства.

Загалом, розроблений застосунок IPAM відповідає вимогам та завданням, поставленим на початку дослідження. Він є надійним, зручним та ефективним інструментом для управління адресним простором, має рівноцінну підтримку як IPV4, так й IPV6, що сприяє підвищенню продуктивності та безпеки мережі підприємства. Використання сучасних технологій та фреймворків забезпечує легкість масштабування системи та її адаптацію до змін у мережевій інфраструктурі, що робить її довготривало актуальною та ефективною.

Проте слід зазначити, що система реалізована у вигляді MVP і передбачає подальший розвиток шляхом додавання API, поглиблення рівня інтеграції та автоматизації, перенесення бази даних з SQLite на рішення, які більше придатні для виробничого середовища: PostgreSQL, MySQL, Oracle Database – вони підвищать продуктивність, безпеку та відмовостійкість, перехід з virtualenv на Docker чи подібну систему контейнеризації.

Список використаних джерел

1. Roser M. “Technology over the long run: zoom out to see how dramatically the world can change within a lifetime” [Електронний ресурс] – Режим доступу: https://ourworldindata.org/technology-long-run

2. Network Infrastructure - Worldwide [Електронний ресурс] – Режим доступу: https://www.statista.com/outlook/tmo/data-center/network-infrastructure/worldwide#revenue

3. How IPAM works [Електронний ресурс] – Режим доступу: https://documentation.solarwinds.com/en/success_center/ipam/content/ipam-how-does-ipam-work.htm

4. What is three-tier architecture? [Електронний ресурс] – Режим доступу: https://www.ibm.com/topics/three-tier-architecture

5. Internet Protocol [Електронний ресурс] – Режим доступу: https://www.ibm.com/docs/en/aix/7.3?topic=protocols-internet-protocol

6. What is IPv4? Everything you need to know [Електронний ресурс] – Режим доступу: https://www.cloudns.net/blog/what-is-ipv4-everything-you-need-to-know/

7. Internet Protocol, Version 6 (IPv6) Specification [Електронний ресурс] – Режим доступу: https://datatracker.ietf.org/doc/html/rfc2460

8. Usage statistics of IPv6 for websites [Електронний ресурс] – Режим доступу: https://w3techs.com/technologies/details/ce-ipv6

9. Using DNS to estimate the worldwide state of IPv6 adoption [Електронний ресурс] – Режим доступу: https://blog.cloudflare.com/ipv6-from-dns-pov

10. DNS records [Електронний ресурс] – Режим доступу: https://www.cloudflare.com/learning/dns/dns-records/

11. What is a DNS PTR record? [Електронний ресурс] – Режим доступу: https://www.cloudflare.com/learning/dns/dns-records/dns-ptr-record/

12. Dynamic Host Configuration Protocol (DHCP) [Електронний ресурс] – Режим доступу: https://learn.microsoft.com/en-us/windows-server/networking/technologies/dhcp/dhcp-top

13. G. Kessler, C. Monaghan “The Dynamic Host Configuration Protocol (DHCP) and Windows NT” Windows NT Magazine, May 1999 [Електронний ресурс] – Режим доступу: https://www.garykessler.net/library/dhcp.html

14. DDI Solution Market Size & Share Analysis - Growth Trends & Forecasts (2024 - 2029) [Електронний ресурс] – Режим доступу: https://www.mordorintelligence.com/industry-reports/ddi-dns-dhcp-and-ipam-solutions-market

15. Address Resolution Protocol [Електронний ресурс] – Режим доступу: https://www.ibm.com/docs/en/aix/7.3?topic=protocols-address-resolution-protocol

16. IPv4 ARP and IPv6 ND [Електронний ресурс] – Режим доступу: https://documentation.nokia.com/srlinux/23-10/books/interfaces/ipv4-arp-ipv6-nd.html

17. What is ICMP? [Електронний ресурс] – Режим доступу: https://aws.amazon.com/what-is/icmp/

18. ICMP type and code IDs [Електронний ресурс] – Режим доступу: https://www.ibm.com/docs/en/qsip/7.5?topic=applications-icmp-type-code-ids#c_DefAppCfg_guide_ICMP_intro__title__3

19. Internet Control Message Protocol (ICMP) Parameters [Електронний ресурс] – Режим доступу: https://www.iana.org/assignments/icmp-parameters/icmp-parameters.xhtml

20. phpIPAM [Електронний ресурс] – Режим доступу: https://phpipam.net/

21. Infoblox IPAM Reviews [Електронний ресурс] – Режим доступу: https://www.peerspot.com/products/infoblox-ipam-reviews#pricing

22. Amazon IP Address Manager – Related information [Електронний ресурс] – Режим доступу: https://docs.aws.amazon.com/vpc/latest/ipam/related-info.html

23. 2024 Data Breach Investigations Report [Електронний ресурс] – Режим доступу: https://www.verizon.com/business/resources/reports/dbir/#takeaways

24. Cygna Labs Completes Acquisition of Diamond IP, Third-Largest Global DDI Vendor [Електронний ресурс] – Режим доступу: https://cygnalabs.com/en/press/cygna-labs-completes-acquisition-of-diamond-ip/

25. VitalQIP [Електронний ресурс] – Режим доступу: https://www.nokia.com/networks/bss-oss/vitalqip-ip-address-management/

26. IPAM & DHCP: Cisco Prime Network Registrar & Cygna Labs IPControl [Електронний ресурс] – Режим доступу: https://www.cisco.com/c/en/us/products/cloud-systems-management/prime-network-registrar/pnr-ipam-wp.html

27. Octoverse: The state of open source and rise of AI in 2023 [Електронний ресурс] – Режим доступу: https://github.blog/2023-11-08-the-state-of-open-source-and-ai/

28. Django overview [Електронний ресурс] – Режим доступу: https://www.djangoproject.com/start/overview/

29. virtualenv [Електронний ресурс] – Режим доступу: https://virtualenv.pypa.io/en/latest/

30. nmap [Електронний ресурс] – Режим доступу: https://nmap.org/

31. Chapter 15. Nmap Reference Guide – Host Discovery [Електронний ресурс] – Режим доступу: https://nmap.org/book/man-host-discovery.html

32. CIDR Notation [Електронний ресурс] – Режим доступу: https://docs.digitalocean.com/glossary/cidr/

Додаток А

```
const express = require('express')

const app = express()

const port = 3000


const dns_data = [
	{
		ipv4: "192.168.0.15",
		ipv6: "a190:a6b7:c81f:c3ab:aec9:c10c:a027:a921",
		url: "example.com"
	},
]


app.get('/dns4', (req, res) => {
	const ip = req.query.ip;
	let found = false
	for (let i = 0; i < dns_data.length; i++){
		if (dns_data[i].ipv4 === ip){
			found = true
			res.send(JSON.stringify(dns_data[i]))
		}
```

```
        }
        if (!found){
                res.status(204).json({message: "No such ip here"});
        }
})

app.get('/dns6', (req, res) => {
        const ip = req.query.ip;
        let found = false
        for (let i = 0; i < dns_data.length; i++){
                if (dns_data[i].ipv6 === ip){
                        found = true
                        res.send(JSON.stringify(dns_data[i]))
                }
        }
        if (!found){
                res.status(204).json({message: "No such ip here"});
        }
})
```