МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ ЕЛЕКТРОНІКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК

Кваліфікаційна робота бакалавра

ВЕБ-ДОДАТОК ДЛЯ АВТОМАТИЧНОГО НАЛАШТУВАННЯ ВІРТУАЛЬНОЇ ЛОКАЛЬНОЇ МЕРЕЖІ НА ОСНОВІ ТЕХНОЛОГІЇ ETHERNET

Здобувач освіти гр. ІН – 82 Олександр ДОЦЕНКО

Науковий керівник,
кандидат фізико-математичних наук,
старший викладач Дмитро ВЕЛИКОДНИЙ

Завідувач кафедри
доктор технічних наук, професор Анатолій ДОВБИШ

СУМИ 2022

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ ЕЛЕКТРОНІКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК

Затверджую_____
Зав. кафедрою Довбиш А.С.
"_____" _____2022 р.

ЗАВДАННЯ

до кваліфікаційної роботи

здобувача вищої освіти четвертого курсу, групи ІН-82 спеціальності «122 – Комп'ютерні науки» денної форми навчання Доценка Олександра Романовича.

Тема: «ВЕБ-ДОДАТОК ДЛЯ АВТОМАТИЧНОГО НАЛАШТУВАННЯ ВІРТУАЛЬНОЇ ЛОКАЛЬНОЇ МЕРЕЖІ НА ОСНОВІ ТЕХНОЛОГІЇ ETHERNET»

Затверджена наказом по СумДУ
№_____ від _____2022 р.

Зміст пояснювальної записки: 1) літературний огляд за обраною тематикою роботи; 2) постановка завдання для розробки; 3) вибір оптимальних інструментів для розробки додатку; 4) практична реалізація; 5) перевірка на коректність.

Дата видачі завдання «_____»_____ 2022 р.
Керівник роботи _____ Дмитро Великодний
Завдання прийняв до виконання _____ Олександр Доценко

РЕФЕРАТ

Записка: 56 сторінок, 16 рисунків, 1додаток, 15 джерел.

Об’єкт дослідження – способи створення веб-додатку, їхній функціонал та взаємодії з другими програмами.

Мета роботи – створити програму для автоматичного налаштування локальних віртуальних мереж.

Методи дослідження – засоби обробки даних, підходи створення веб-додатків.

Результати – Веб-програма для автоматичного налаштування Vlan у симуляторі.

ВЕБ-ДОДАТОК, МАРШРУТИЗАТОР, КОМУТАТОР, VLAN, DHCP, CISCO PACKET TRACER, JAVASCRIPT

ЗМІСТ

ВСТУП

Розробка мережі проходила в 60-х роках минулого століття для міністерства оборони США. Вже тоді люди розуміли, що ця технологія допоможе внести великий вклад у розвиток нашої планети та допомогти багатьом людям із кожного куточка землі. Щоб розвивати її глобально, для неї було створено велику кількість інновацій, а разом із цим випускалися у великих обсягах книжки з описом та налаштуванням цих методик. Звичайно, через деякий час ці розробки застарівали, як для своїх років, і залишились лише у спогадах. Але з кожними новими роками вона зазнає дедалі більші зміни, які роблять її легкою та швидкою у роботі. Зараз всі знають цю мережу, яка називається «Інтернет».

На сьогоднішній день дуже важко представити собі життя без інтернету. За допомоги оновлень мережі ним можуть користуватися від малого до великого. По всьому світу компанії платять великі гроші за для безперебійного, надійного, швидкого та безпечного підключення, щоб мати зв’язок зі своїми офісами, які знаходяться в інших містах, а деякі, навіть, на інших континентах. У наш час це настільки важлива тема, що перебій в системі на декілька хвилин, може позбавити господаря сотень мільйонів доларів або іміджу бренду, який створювався роками.

РОЗДІЛ 1

Мережа Ethernet та технології Dynamic Host Configuration Protocol та Local Area Network.

1.1 Мережа Ethernet

Ethernet – сімейство технологій пакетної передачі даних групи 802.3 між комп'ютерними та промисловими мережами. Саме таке визначення дає інститут інженерів електротехніки та електроніки (IEEE) цій інновації. У 80-х роках XX століття це відкриття посилило вплив мережевих з'єднань, а разом із цим генерацію нових ідей та методів, на формування всеохоплюючого інтернету.

Початок цієї технології розпочинається в далекому 1972 році, коли інженерам із компанії Xerox дали завдання, розробити метод, який дозволяє робочій станції Xerox Alto підключатися до лазерного принтеру, котрий знаходився іще в розробці. По задумці, цей спосіб мав би давати доступ сотні комп'ютерам підключатися одночасно та передавати данні з гарною швидкістю до принтеру. Серед розробників був Роберт Меткалф і Девід Боггс [1]. В часи розробки Меткалф скористався своїм раніше накопиченим досвідом при створенні ALOHAnet (перша мережа передачі даних, яка використовувала в якості середи - бездротову технологію та була введена в експлуатацію вченими Гавайського університету). 22 Травня 1973 року, Роберт Меткалф склав записку з доповіддю до глави PARC про потенціальну технологію. Законне право на цю методику йому надали дещо пізніше. У 1976 році Меткалф та Боггс створюють брошуру назва якої – «Ethernet: Distributed Packet-Switching For Local Computer Networks». У 1979 Роберт засновує свою компанію 3Com та починає розвивати локальні мережі разом з комп'ютерами. Невдовзі він домовляється з Xerox, Intel та DEC працювати разом і створити стандарт для Ethernet-у. 30 Вересня 1980 року виходить перший стандарт, в якому було описано протокол передачі даних на швидкості в 10Мб/с, котрий допоміг цьому методу прибрати із ринку такі відомі на той час технології, як ARCNET та Token ring. А вже у 1983-му інститут

інженерів електротехніки та електроніки відредагував його і зробив стандартом IEEE 802.3, після чого наступні 37 років він розглядувався та оновлювався.

Основою для користування ALOHAnet було використання спільної середи для всіх клієнтських передач даних. Для цього було придумано протокол, суть якого можна описати коротко: «слухай перед відправкою» - пізніше це назвали Carrier Sense Multiple Access with collision detection (CSMA/CD) [1]. Щоб не відбувалося колізій вигадали забезпечення використання каналу для одного відправника. Спосіб, котрий допомагає визначити чи вільне середовище – прослуховування основної несучої частоти. В цілому, метод, який застосовує CSMA/CD доступу до мережі цілком зрозумілий та продуманий. А за допомоги MAC-адреси було зрозуміло напрямок кадру від відправника до отримувача. MAC (media access control address) – унікальний ідентифікатор, призначений мережевому адаптеру, який використовується в стандарті IEEE 802 (при виготовлені стандарту IEEE 802.3 було передбачено, що кожна мережева карта, як і вбудований мережевий інтерфейс, повинні мати ідентифікаційний, шестибайтний номер при розробці цих компонентів) насамперед Ethernet, Wi-Fi і Bluetooth. Він офіційно називається «Ідентифікатор типу EUI-48». Як видно з назви, адреса має довжину 48 біт, тобто. 6 байт. Загальноприйнятого стандарту написання адрес не існує. Зазвичай він записується у вигляді шести шістнадцяткових цифр, розділених двокрапками: 11:AA:BB:CC:12:13, хоча деякі виробники пристроїв вважають за краще записувати його як 11-AA-BB-CC-12-13 або навіть 11aa.vvcc.1213. Робота CSMA/CD полягає в трьох періодах: передачі, конкуренції, простою. В періоді передачі, якщо нема несучої частоти, комп’ютер починає відправляти дані по такій схемі:

- Преамбула. Вона допомагає синхронізуватися між відправником та отримувачем і виділити кадр. Довжина цієї частина всього 8 байт.
- Кадр. Після закінчення преамбули починається передача самого кадру. Всі комп’ютери, котрі підключені до цього розділювального середовища починають читати інформацію та записувати його до буферу. Всі вони

читають перші 6 байт кадру до яких записано MAC-адресу отримувача. У кого вона збіглася, той починає записувати повністю фрейм в свій буфер, у кого ні – перестають читати це повідомлення та видаляють його. Також є спеціальний режим мережевого адаптеру – нерозбірливий. Він приймає всі повідомлення незалежно від MAC-адресу.

- Міжкадровий інтервал. Після відправки основної частини комп'ютер відправник тримає інтервал 9,6 мкс, для того, щоб запобігти монопольного захвату каналу та привести мережеві адаптери до вихідного стану.

Період витримки завершився, де наступним ділом можуть починати передавати свої дані інші комп'ютери. Якщо таких декілька, то починається період конкуренції. В цей час відбувається колізія, отже вони мають зробити паузу. Пауза робиться за допомогою формули:

$$L * 512 \text{ бітових інтервалів}$$

Бітовий інтервал – це час між двома появами двох послідовних біт даних (в класичному Ethernet це 0,1 мкс), а L – випадкове число із діапазону [0, 2^N-1], де N – номер спроби. Ця формула гарно працює коли в мережі мало комп'ютерів, які рідко передають данні. Однак, якщо ситуація протилежна, то зростає число спроб передачі, інтервал, із якого вибирається L та тривалість пауз, експонентно збільшується затримка.

Саме ці відкриття стануть невід'ємною частиною раннього Ethernet та дадуть життя такій мережі, як Wi-Fi та багато іншим. Спочатку ця інновація працювала на коаксіальних кабелях до кінців яких потрібно було встановлювати «т-конектори» та ретельно доглядати за ними [1]. Через деякий час підключення почало з'являтися на неекранованій витій парі (також відома під назвою патч-корд). Плюсів використання витої пари порівняно з коаксіальним кабелем багато:

1. Можливість працювати в дуплексному режимі (одночасно відбувається прийом та передача даних по каналу);
2. Вартість витої пари дещо менша ніж його суперника;

3. Хоча коаксіальний кабель і може передавати дані на більші відстані, але із-за діелектричного ізолятора, що огинає мідну серцевину, він є більш складним в установці та обслуговуванні;
4. Більш гнучкіший із-за відсутності мідного сердечника;
5. Коаксіальний кабель використовує топологію «спільна шина», а вита пара – «зірка» тому, якщо виникне проблема з дротом, то обірветься лише два елемента мережі, коли як в підключені по спільній шині виникне несправність по всій системі;
6. Використання диференційного сигналу дає змогу патч-корду більш стійкіше витримувати перешкоди [2];

Підводячи підсумок по вищесказаному, ця мережа стала кращою за Token Ring завдяки використанню неекранованої витої пари та дешевизни інтерфейсної плати. Через це, людям, котрим потрібна була мережа дома або в офісі вибирали розробку Роберта Меткалфа.

Зараз це є самий популярний спосіб для створення дротових комп'ютерних мереж. В моделі взаємодії відкритих систем OSI дана технологія знаходиться на фізичному (коаксіальний дріт, вита пара, оптоволокно) та канальному рівнях (способи доступу та протоколи, однакові для середи передачі інформації). В кінці 90-х років XX століття він становиться домінуючим методом в локальних мережах. Саме в 1995 році на світ з'явився стандарт Fast Ethernet для каналів, котрі потребують більш високі швидкості передачі даних. Це відкриття допомогло збільшити обмеження швидкості с 10 Мбіт/с до 100 Мбіт/с з мінімальними змінами існуючого кабелю [2]. Він надає змогу більш високій пропускній швидкості для світлин, відео, мультимедіа, графіки, а також для знаходження та виправлення похибок. Існує 3 типи Fast Ethernet: 100BASE-TX для користування с дротом UTP рівня 5-го (саме він є найпопулярнішою версією) ; 100BASE-FX для використання оптоволоконного дроту; 100BASE-T4 в якому є два додаткових дроти з використанням UTP 3-го рівня. Далі створюють Gigabit Ethernet, який досягає швидкості в 1000 Мбіт/с

(1 Гб/с) [1]. Цей спосіб є модернізацією вищесказаного Fast Ethernet. Але є відмінності між ними: 1. В цьому типі мережі працюють всі 4 пари дротів у витій парі. Кожен із них вносить вклад у швидкість та передачу даних. 2. У нього є допоміжна підтримка повно дуплексного режиму на рівні MAC та швидкості передачі інформації.

Ethernet описує, як формувати та передавати пакети даних до других мережевих пристроїв локальної сітки, щоб вони могли приймати, ідентифікувати, розпаковувати інформацію. Більшість цих задач відбувається на канальному рівні моделі OSI. Там, для передачі даних, використовуються кадри. Розглянемо передачу формату кадру Ethernet II, використання якого відбувається частіше ніж інші два стандарти. Цей формат складається із трьох частин: заголовку, даних, кінцевика. В заголовку знаходяться адреси відправника та отримувача. Поле під назвою «Тип» містить в собі код протоколу, по якому отримані дані. Кінцевик використовується для перевірки правильності доставки інформації. При отриманні кадру, одержувач просто рахує контрольну суму та звіряється з тою сумою, що лежить в кінці кадру. Якщо все вірно, то починається зчитування даних, якщо ні – то він відкидається і комп’ютер відправника ніяк про це не повідомляється. В полі «Дані» знаходиться інформація від протоколу верхнього рівня. У нього є обмеження по довжині: мінімальна довжина повинна мати 46 байт (це викликано особливостями технології Ethernet виявленням колізій). Максимальна довжина – 1500 байт. Це число вибрали розробники, тому що в 70-х роках воно було доволі великим об’ємом.

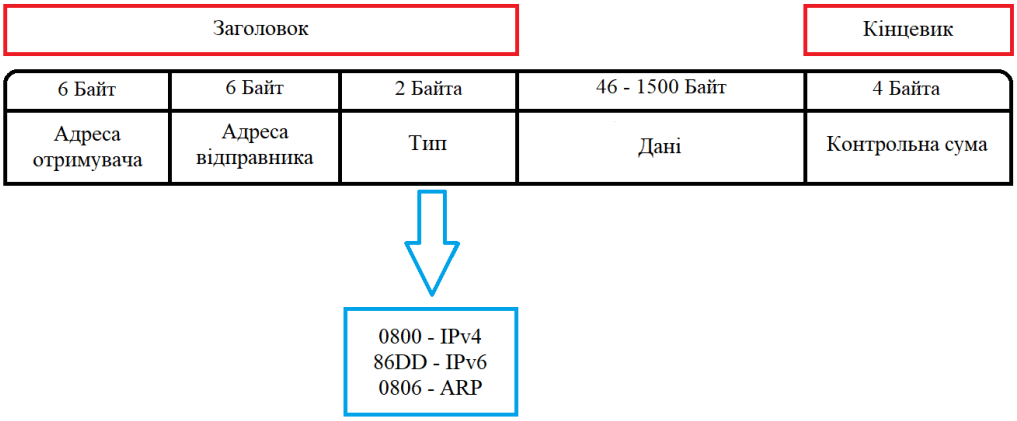

Рисунок 1.1 - Формат кадру Ethernet II

Зараз Ethernet популярний із-за забезпечення гарного балансу швидкості, вартості та легкої установки. Ці якості разом із можливістю підтримки всіх відомих протоколів та з широким розповсюдженням на комп'ютерному ринку роблять цю технологію ідеальною мережевою методикою для більшості сьогоденних користувачів.

1.2 Dynamic Host Configuration Protocol

DHCP (Dynamic Host Configuration Protocol) – це протокол управління мережею, котрий використовується в мережах TCP/IP на прикладному рівні, працює за моделлю «клієнт-сервер», де DHCP-сервер задає кожному пристрою рухливу IP-адресу та інші параметри налаштування, щоб інші пристрої могли зв'язатися з ним. Адміністратор може задати діапазон адрес, які роздаються сервером для пристроїв, що сильно полегшує роботу та мінімізує похибки у заданні IP-шників, із-за уникнення ручної роботи.

Свою історію він розпочинає в жовтні 1993 році. Тоді було прийнято стандарт протоколу. Одна з перших реалізацій протоколу IP-адреси з'явилася більше 30 років тому і отримала назву RARP (Reverse Address Resolution Protocol). Щоб спростити роботу, це виглядає так: клієнт запитує широкомовну мережеву адресу, сервер отримує її, знаходить у своїй базі даних прив'язку MAC-адреси та IP клієнта – і надсилає адресу у відповідь.

DNS розшифровується, як Domain Name System. Це глобально-розподілене сховище ключів і цінностей. Сервери в усьому світі можуть надавати користувачам значення ключа, і якщо вони не знають ключа, вони звернуться до іншого сервера. DNS заснований на ідеї ієрархії імен і зон. Кожен сервер, відповідальний за ім'я, може передати відповідальність за наступну частину домену іншому серверу, організації або особі, що дозволяє покласти відповідальність за релевантність інформації на іншу організацію (особу), відповідальну тільки за «свої» доменне ім'я на деяких серверах.

Система доменних імен, ще відомий як DNS, і протокол динамічної конфігурації хосту, знайомий нам як DHCP, є двома важливими методами TCP/IP в мережі. DNS робить перетворення імен хостів в IP адреси, в той час як DHCP призначає унікальні, динамічні адреси, маски підмережі та шлюзи за замовчуванням у працюючих комп’ютерів в конкретній серверній сітці. Завдяки рухливій адресації обчислювальна машина маже мати різні IP при кожному підключенню до мережі, до якої він належить. Це все робиться без єдиної допомоги адміністратора [4]. За допомоги цього способу, кожний комп’ютер, підключений до сітки, автоматично отримує свій унікальний IP. В більшості провідних Ethernet-маршрутизаторах та бездротових точок доступу використовуються DHCP-сервери, тому що вони в рази полегшують працю [3].

Цей протокол може надавати IP трьома шляхами:

- Ручний розподіл налаштовується за допомогою мережевого адміністратора, який зіставляє апаратну адресу (в Ethernet мережах - це MAC-адреса) кожного клієнтського комп’ютеру певний IP. Насправді, цей метод призначення адреси відрізняється від ручного налаштування кожного пристрою тим, що інформація про адресу записується централізовано (на сервері), що полегшує її подальшу зміну.
- Автоматичне розподілення дає кожному комп’ютеру довільне та вільне IP на постійне користування із певного діапазону заданого адміністратором.

- Динамічний розподіл. Цей метод подібний автоматичному окрім того, що адреса видається не на постійну основу, а на деякий час (так звана оренда адреси). Як тільки термін заданого часу минув IP вважається знову вільним і клієнт повинен знову запросити новий. Він також може відмовитися від отриманого коду або йому може дістатися попередній IP.

Деякі операції DHCP можуть автоматично оновлювати записи DNS, що відповідають клієнтським комп'ютерам, коли призначаються нові адреси. Це робиться за допомогою протоколу оновлення DNS. Крім IP-адрес, DHCP може інформувати клієнтів про додаткові параметри, необхідні для нормальної роботи мережі. Ці параметри називаються параметрами DHCP. Деякі продавці програмного забезпечення самі задають параметри. IP-адреса маршрутизатора, адреса серверів DNS, ім'я домену DNS, маска підмережі – це одні з найчастіше використовуваних опцій.

Робота цього протоколу може показатися важкою та не з першого разу зрозумілою, але це не так. Як і було сказано раніше, цей метод працює по моделі «клієнт-сервер», де клієнтом виступає обчислювальна машина, котра отримує IP автоматично, а сервером – комп'ютер, який забезпечує призначення адрес та веде таблицю, в якій записані всі номери, щоб уникнути повторення. У мережі сервер DHCP контролює пул з IP-адресами, а також даними DNS, інформацією про шлюз за замовчуванням, маршрутами та іншою інформацією, яка використовується для налаштування клієнтських мереж. Коли новий комп'ютер підключається, то у нього немає жодної інформації про мережу до якої він під'єднаний [3]. Він, першим ділом, намагається знайти DHCP-сервер, посилаючи повідомлення «DISCOVER» на широкомовний MAC-адрес (255.255.255.255). Всі пристрої в даному діапазоні отримують цей запит і сервер також. Потім, як сервер отримує повідомлення клієнта, він визначає необхідну конфігурацію клієнта на основі вказаних налаштувань адміністратора мережі. Після першої звістки, клієнт отримує і відповідь листа від серверу з назвою «OFFER», де знаходиться сам IP. Він пропонує взяти саме ці значення клієнту.

Якщо клієнт погодився на дану інформацію, то він висилає назад цю адресу у повідомленні з назвою «REQUEST». На наступному кроці сервер відсилає звістку підтвердження «ACK», що даний IP тепер належить приладу з певним MAC адресом [5]. Всі вони працюють по моделі запит-відповідь. Найпростіше всього запам'ятати процес отримування IP по протоколу DHCP можна по першим буквам кожного повідомлення – DORA.

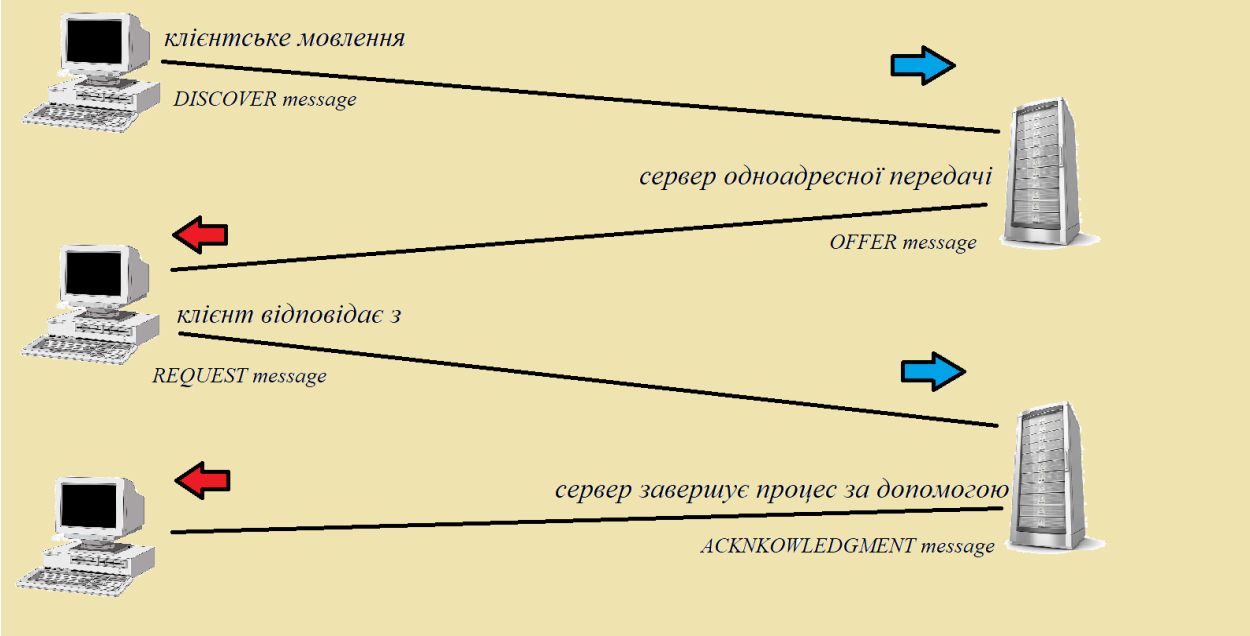

Рисунок 1.2 - Схема отримання IP-адреси

Якщо використання цього протоколу призначається для задання IP-адрес у власній мережі, то необхідно продумати схему таким чином, щоб сервер мав доступ до кожного клієнту. Бо коли новий пристрій щойно підключили до сітки, то він ще не знає де знаходиться сервер і посилає запит «DHCPDISCOVER» на широкомовну адресу. Його отримують всі учасники, в тому числі і сервер. Але, якщо він відділений від нього одним чи декількома комутаторами? Відбуваються ті самі дії, які були в першій ситуації, окрім того, що повідомлення не відправилось до другої підмережі. Запит зупинився на комутаторі, так як він не передає широкомовний трафік. Із-за цього клієнт не зможе стати частиною сітки без власного IP. Це може вирішитися за допомогою DHCP Relay. Ця спеціальна

конфігурація маршрутизатора, гарантує передачу широкомовного трафіку до іншої підмережі, але не всієї, а тільки тієї, що відноситься до протоколу. І тільки в цьому випадку повідомлення з IP надійде до запитувача.

Надання адреси приладу не відається на постійну основу, а лише на деякий час: година, день, 3 дні тощо. Розподіл часу оренди є питанням політики мережі, і різні типи пристроїв можуть мати різні часові інтервали. Після закінчення «lease time» IP звільнюється і потрапляє в пул серверу, де може потрапити до наступного користувача. Але, якщо потрібно залишити саме цю адресу, то можна скористатися одним способом: продовження оренди [4]. Зазвичай клієнт починає це робити після того, як пройде половина зарезервованого часу. В цьому випадку використовується спрощена процедура отримання. Так, як клієнт вже має та знає свою адресу, то він просто відсилає на сервер повідомлення «REQUEST» разом із IP, а у відповідь, якщо все гаразд, отримує «ACK».

На додаток до повідомлень, котрі необхідні для отримання IP-адреси клієнта, DHCP надає додаткові повідомлення для виконання інших завдань. Два інших типи повідомлень у визначенні DHCP використовуються клієнтами: повідомлення «DHCPINFORM» і параметр «DHCPRELEASE».

«DHCPINFORM». Повідомлення «DHCPOFFER» складається з кількох полів параметрів у структурі пакета. Однак сервер рідко використовує все це і не надає жодного значення. Певним програмам може знадобитися деяка інформація, щоб правильно налаштувати пристрій у мережі. Якщо ця важлива інформація відсутня в повідомленні пропозиції DHCP, він може надіслати деталі запиту «Informer». Якщо ця інформація доступна, вона буде надіслана сервером у вигляді іншого повідомлення «порада» із заповненими обов’язковими полями. Прикладом використання «INFORM» є те, що браузери часто використовують це повідомлення, як спосіб отримати веб-проксі шляхом автоматичного виявлення його.

«DHCPRELEASE». Повідомлення про звільнення — це коли клієнт розриває договір оренди з IP-адресою до зазначеної дати закінчення терміну дії. Цей тип повідомлень не є необхідним для роботи протоколу, оскільки звичайні

договори оренди зазвичай припиняються достроково, коли користувач вимикає пристрій. Не існує жодної процедури затримки процесу вимкнення, щоб дозволити клієнту DHCP надіслати повідомлення про звільнення. У цьому випадку IP-адреса залишатиметься виділеною цьому клієнту до кінця його оренди, навіть якщо пристрій протягом цього часу не буде активним. DHCP обробляє лише призначення доступних IP-адрес. Він не підтримує зв'язок з вузлами мережі. Він передбачає, що IP-адреса використовується протягом усього терміну оренди, тому не перевіряє, що вона, призначена пристроєм, все ще активна в мережі. Менеджер протоколу не перепризначує адреси, якщо змінюється конфігурація мережі. Ці обмеження пов’язані з тим, що DHCP зазвичай реалізується як частина набору протоколів мережевої адресації, які називаються DDI.

Мінуси в захисті протоколу. DHCP використовує метод User Datagram Protocol. Це система без з’єднання, тому шифрування не включено. Так як майже всі типи повідомлень у протоколі призначені для трансляції в будь-якому мережевому режимі, зловмисники можуть отримати великий контроль над мережевими операціями та завдати руйнівної шкоди, звертаючись до мережі та прослуховуючи трансляції DHCP. Ось чому цей протокол рідко реалізується окремо. Під час призначення IP-адрес необхідно враховувати багато питань координації. DNS-сервери також повинні посилатися на ці адреси. Зловмисник може вставити в мережу віртуальний підроблений сервер DNS або DHCP. Безпека мережі та дійсність адреси здійснюється диспетчером IP-адрес. Це ключовий елемент набору DDI. DDI — це абревіатура для інтеграції DNS, DHCP та IPAM (Керування IP-адресою) в уніфіковану службу або рішення. Він являється основою для базових мережевих послуг, які дозволяють будь-який зв'язок через мережу на основі IP. Існують притаманні ризики, якщо DNS, DHCP та IPAM керуються окремо. Однак за допомогою централізованого рішення адміністратори мережі можуть отримати видимість і контроль над своєю мережею з однієї платформи. Зокрема, він надає адміністраторам:

- Можливість автоматизувати завдання технічного обслуговування
- Краще розуміння потенційних конфліктів під час оновлення даних служби
- Докладніше тексту про аудит та звітність
- Підвищити ефективність
- Підвищення безпеки, стійкості та підтримки [5].

Оскільки це рішення є інтегрованим, багато завдань спрощуються для мережевої команди. Між записом і фактичним використанням IP-адреси немає розриву, натомість запис оновлюється в режимі реального часу. Наприклад, ось деякі речі, які користувач може робити без проблем:

- Додавати, видаляти та оновлювати записи хоста. Цільові призначення IP оновлюються автоматично.
- Вибіркове розгортання мережевого обладнання.
- Додавання, видалення та оновлення записів IP-адрес. Записи DNS можна додавати, оновлювати або видаляти автоматично.
- Оновіть, додайте або видаліть діапазони DHCP, переконайтеся, що статичні пристрої ще не знаходяться в зоні дії.

1.3 Virtual Local Area Network

Комп'ютерні мережі можуть бути вибудовані як у локальні (LAN) так і в глобальні (WAN). Робочі станції, сервери, комутатори, мости, концентратори – це все мережеві пристрої, котрі підключені один до одного в одній сітці в певному місці часто називають локальною і також вважають широкомовним доменом. VLAN дозволяють кільком мережам працювати майже, як локальна мережа. Одним з найкорисніших елементів VLAN є те, що він усуває затримку мережі, що економить ресурси сітки та підвищує ефективність її. Крім того, VLAN призначені для забезпечення сегментації та підтримки таких питань, як безпека, керування нею та масштабованість. Контролювання трафіку також є однією із її функцій.

Віртуальна локальна мережа (VLAN) — це логічна локальна мережа (або LAN), яка розширює межі однієї традиційної локальної до групи сегментів локальної сітки відповідно до певної конфігурації. Метод має ID. Йому призначено 12 біт, тобто теги можуть бути пронумеровані від 0 до 4095. Технологія знаходиться на канальному рівні в моделі взаємодії відкритих систем. VLAN є логічним об'єктом, тому її створення та налаштування повністю виконуються програмно. Інакше кажучи, VLAN — це логічне угруповання робочих станцій, серверів і мережевих пристроїв, які, знаходяться в одній локальній мережі, незважаючи на їх географічний розподіл [6]. Він дозволяє мережі комп'ютерів і користувачів взаємодіяти в імітованому середовищі, наче вони існували в одній локальній сітці та спільно використовували один широкомовний та багатоадресний домен. Так само вони реалізовані для забезпечення масштабованості, безпеки та простоти керування мережею, а також для швидкої адаптації до мінливих вимог мережі та мобільних серверних робочих станцій і вузлів. Розширені комутатори дозволяють використовувати функціональні можливості віртуальної локальної мережі та їх застосування. Метою впровадження VLAN є покращення продуктивності сітки або використання відповідних функцій безпеки.

Основні прерогативи впровадження VLAN:

- Дозволяє мережевим адміністраторам застосовувати додатковий захист мережевого зв'язку;
- Розширити та переміщувати мережеве обладнання;
- Розділити один такий домен на кілька: широкомовний трафік, що належить одному домену, не проходить через інший домен. Це зменшує навантаження на мережеве обладнання.
- Захищають мережу від перешкод. Порти-маяки зможуть ігнорувати та вирізати кадри з інших VLAN незалежно від вихідного IP-адресу.

- Забезпечує гнучкість, оскільки адміністратори можуть налаштовувати мережі в централізованому середовищі, а пристрої можуть бути розташовані в різних географічних місцях;
- Групувати комп’ютери, які належать до однієї підмережі та застосовувати політику для всієї групи;
- Підвищення продуктивності, зменшуючи затримку та перевантаження трафіку в мережі та мережевих пристроях [6].

Віртуальні локальні мережі також мають деякі недоліки та обмеження, котрі наведені нижче:

- Високий ризик вірусів, оскільки одна заражена система може поширювати віруси по всій логічній мережі;
- Обмеження пристроїв у великих мережах, оскільки для керування робочим навантаженням можуть знадобитися додаткові маршрутизатори;
- Більш ефективний, ніж WAN, для контролю затримок, але не такий ефективний, як LAN.

ВЛМ – це не перевага робочих станцій, це прерогатива комутаторів [6]. Порти таких пристроїв вказують, у якій віртуальній мережі вони знаходяться: весь трафік, що проходить через цей порт, буде позначено тегами VLAN. Завдяки цьому він зможе продовжити роботу через інші інтерфейси свічів, що працюють під цією відміткою. Однак інші порти не приймуть цей трафік. Це створить окрему підмережу, яка не взаємодіє з іншими підмережами без використання комутаторів або маршрутизаторів.

Native VLAN — це параметр для кожного порту, який визначає номер ВЛМ, який отримують усі пакети без тегів.

Щоб мати можливість надсилати та отримувати трафік, що належить до різних підмереж, портів, його необхідно перевести в стан ретрансляції або тегу. У першому випадку з’єднувач передає трафік до всіх VLAN, тоді як у другому випадку він обробляє трафік лише в деяких ВЛМ. Варто зазначити, що для різних пристроїв можуть знадобитися різні налаштування. Так, деякі пристрої

вимагають фізичного інтерфейсу для вказівки стану певного порту. Для інших пристроїв може знадобитися призначити роз’єм, який належить до певної ВЛМ. Якщо адміністратор за бажанням хоче, щоб кілька VLAN проходили через порт, потрібно вказати позначений інтерфейс у кожній. Наприклад, на комутаторі Cisco потрібно вказати дві речі:

1. access port – порт, який входить до VLAN і передає нетегований трафік. Відповідно до специфікації cisco, порт доступу може належати тільки до однієї VLAN, за замовчуванням це перша (нетегована) ВЛМ. Будь-який кадр, що проходить через порт доступу, ідентифікується номером, що належить до нього.
2. trunk port – порт, який передає позначений трафік до однієї або кількох мереж технології. З іншого боку, цей порт не змінює мітку, а лише пропускає кадри з мітками, дозволеними портом [7].

Для передачі трафіку через порт кількох VLAN, сам він перемикається в режим магістралі.

Режим інтерфейсу за замовчуванням залежить від моделі комутатора. Чотири режими наведені нижче.

Auto - порт перебуває в автоматичному режимі і стане магістральною лише в тому випадку, якщо порт на іншому кінці перебуває у включеному або desirable режимі. Тобто, якщо обидва порти знаходяться в автоматичному режимі, магістраль використовуватися не буде.

Desirable - порт перебуває в режимі "готовий до входу в магістраль"; кадри порту DTP періодично передаються іншій стороні, запитуючи віддалений порт увійти в стан магістралі (стан магістралі встановлюється, якщо порт на іншій стороні увімкнено, desirable, або автоматичний режим).

Trunk - порт завжди знаходиться в стані магістралі, навіть якщо інший кінець порту не підтримує цей режим.

Nonegotiate - порт готується до переходу в режим магістралі, але не передає кадри DTP для іншого кінцевого порту. Цей режим використовується для

запобігання конфліктам з іншими пристроями, які не є від компанії Cisco. У цьому випадку комутатор на іншому кінці необхідно вручну налаштувати на використання trunk.

Магія віртуальних локальних мереж — це заголовок Ethernet. Якщо кадр був отриманий від іншого комутатора, цей комутатор вставив тег ВЛМ (хоча кадр надходить від мережевого пристрою, такого як комп'ютер і у кадрі не буде цього тегу). Сам цей тег вставляється у кадр Ethernet перед полем «Тип» і важить 4 байти. В цьому заголовку присутні дані фрагменти інформації:

1. 2-байтовий ідентифікатор протоколу тегу (TPID), який буде встановлено на 0x8100, щоб вказати, що цей кадр містить інформацію про теги 802.1Q або 802.1p.
2. 2 байти інформації керування тегами (TCI), включаючи наступне:
    - 3-значна точка пріоритету користувача (PCP), яка встановлює значення пріоритету від 0 до 7, яке можна використовувати для забезпечення пріоритетного трафіку якості обслуговування (QoS).
    - 1-бітовий канонічний індикатор (CFI), який є частиною сумісності між Ethernet та іншими мережевими структурами, такими як Token Ring. Для Ethernet це значення також буде встановлено на нуль.
    - 12-розрядний ідентифікатор VLAN (VID), який ідентифікує VLAN, до якої належить кадр.

8 Байт	6 Байт	6 Байт	8 Байт	2 Байта	46 - 1500 Байта	8 Байт	12 Байт
Преамбула	Адрес отримувача	Адрес відправника	802.1Q Тег	Тип	Дані	Контрольна сума	Міжкадровий розрив

Рисунок 1.3 - Стандарт IEEE 802.1Q

Під час тегування VLAN у кадрі може виникнути неприємна помилка. Максимальний розмір кадру Ethernet для IEEE 802.3 становить 1518 байт. Якщо корисне навантаження або частина даних містить повні 1500 байт даних і

додатковий 4-байтовий заголовок у кадрі, розмір кадру становитиме 1522 байти. У відповідь на цю ситуацію IEEE випустила новий стандарт Ethernet (IEEE 802.3ac) у 1998 році, збільшивши максимальний розмір кадру Ethernet до 1522 байт. Якщо у вас є старий комутатор, який не підтримує більший розмір кадру IEEE 802.3ac, ваш комутатор може відхилити ці непідтримувані кадри повідомлень або повідомити, що вони завеликі.

Переваги мережі Vlan.

Підвищена безпека: використання мережі VLAN підвищує безпеку, зменшуючи внутрішні та зовнішні загрози. При розділенні користувачів це покращує її та конфіденційність, гарантуючи, що користувачі мають доступ лише до мережі, яка підпадає під їхню відповідальність. Зовнішні загрози також зведені до мінімуму. Якщо зовнішні зловмисники отримають доступ до VLAN, вони залишаться в цій мережі, поза контролем елементами керування, які повинні бути утримані окремо від інших.

Покращена якість обслуговування: мережі VLAN керують трафіком ефективніше, що призводить до підвищення продуктивності для кінцевих користувачів. У вас буде менше проблем із затримкою мережі та вища надійність для критичних програм. ВЛМ також спрощують визначення пріоритетів трафіку, гарантуючи, що важливі дані програми продовжують надходити, навіть з менш пріоритетним трафіком, таким як перегляд веб-сторінок.

Легко усунути неполадки. Усунення несправностей мережі стає легшим і швидшим, коли різні групи користувачів сегментовані та ізольовані один від одного. Якщо користувач знає, що скарга надходить лише від певної кількості користувачів, він може швидко звузити пошук, щоб знайти проблему.

Полегшене керування для адміністраторів мережі. Однією з найкращих особливостей віртуалізації є те, що вона полегшує керування. Логічно групуючи користувачів в одну віртуальну мережу, клієнт можете легко налаштовувати та контролювати політики на рівні групи [6].

1.4 Постановка задачі

Після перегляду та вивчення даної інформації можна описати поступові дії для виконання практичної частини бакалаврської роботи.

1. Зібрати топологію у симуляторі Cisco Packet Tracer і зробити налаштування даної схеми.
2. Провести перевірку у правильності підключення приладів, написанні команд, кінцевого результату. Переконатися, що прилади працюють як потрібно і передають інформацію між собою.
3. Змоделювати веб-додаток для подальшого автоматичного налаштування практичної частини. Основна задача програми - бути легкою у використанні, з приємним дизайном і коректними запитами на введення даних.
4. В симуляторі перевірити та переконатися у даних командах, які згенерував додаток, на правильність налаштування.

РОЗДІЛ 2

Налаштування VLAN та DHCP у мережі Ethernet

Оскільки складність мережевих систем продовжує розвиватися, появи нових навчальних програм та освітніх засобів для полегшення навчання та дослідження мережевих технологій прогресує також. Мережа Cisco спрямована на забезпечення інноваційних навчальних програм і навчальні засоби, які допомагають учням зрозуміти складність інформаційно-комунікаційних технологій (ІКТ). Всередині ця структура є програмним забезпеченням Cisco Packet Tracer Learning створено, щоб допомогти отримати практичні знання, навички використання мережевих технологій в умовах, що швидко змінюються. Студенти, які шукають навички ІКТ, тепер можуть скористатися перевагами доступності програми навчання онлайн та нові можливості для соціального навчання, співпраці та змагань.

Симулятор дозволяє студентам створювати практично необмежену мережу декількох пристроїв, які в свою чергу, стимулюють практику, тестування та усунення несправностей. Навчальне середовище, котре засноване на моделюванні, допомагає студентам розвивати навички XXI століття, такі як прийняття рішень, творче та критичне мислення та навички вирішення проблем. У ньому є відладчик, який дозволяє візуалізувати проходження пакетів по мережі крок за кроком - функція, якою не може похвалитися жоден інший інструмент. Студенти можуть створювати, налаштовувати та розвивати самостійно або у співпраці з іншими учнями використання віртуальних пристроїв та імітованих підключених мереж. У цьому симуляторі можна непогано вивчити ефективне, інтерактивне середовище для роботи в сітці.

Cisco Packet Tracer має дві робочі області - логічну та фізичну. Логічні робочі області дозволяють користувачам за допомогою розміщення, підключення та створення логічних мережевих топологій вибудовувати свою віртуальну мережу. Фізична робоча область забезпечує графічний, фізичний

вимір логічної мережі даючи відчуття масштабу та розташування мережевого обладнання, наприклад маршрутизатори, комутатори та хости виглядали б, як у справжньому житті. Цей зовнішній вигляд також надає географічне представлення мережі, включаючи кілька міст, будівель та шаф для електропроводки. Найголовніше, Packet Tracer допомагає студентам та викладачам створювати свій власний віртуальний світ для досліджень, експериментів та пояснення мережі концепцій та технологій.

Чому варто вибрати Cisco Packet Tracer замість більш сучасного GNS3? Немає сумніву, що GNS3 набагато кращий. Але краще він у вивченні функцій та до підготовки будь-яких іспитів. Якщо ж людина тільки починає знайомитись з роботою мережі або хоче удосконалити свій початковий рівень знань, то краще почати з симулятора від компанії Cisco. Він дуже легкий у вивченні та дозволяє покращити базове розуміння функціональностей мережевих пристроїв (VLAN, IP-адресація, маршрутизація тощо). Це добре для швидкого початку роботи.

Схема буде складатися із таких пристроїв: 4-х персональних комп’ютерів (PC1, PC2, PC3, PC4), серверу (Server 0), комутатору (Switch3) та роутеру (Router4). Підключимо всі ці прилади до одного свічу по топології «Зірка» та почнемо налаштування саме з нього.

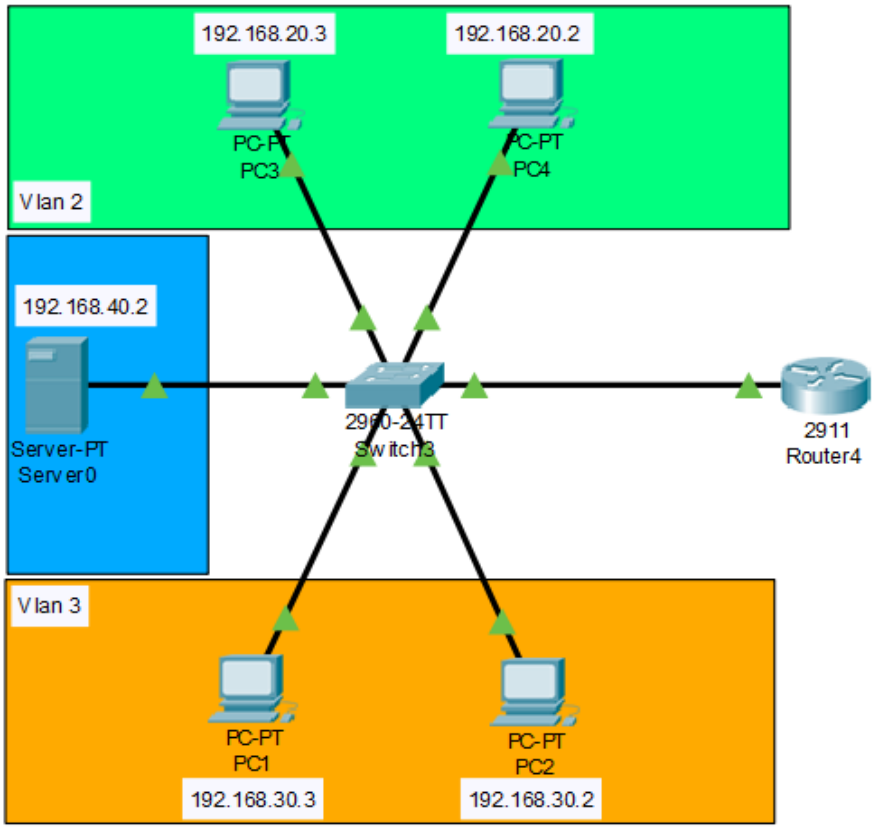

Рисунок 2.1 - Топологія налаштованої мережі

Ці 4 ПК я розділю на дві віртуальні мережі, тобто Vlan 2 та Vlan 3. Користувачі із першої області ввімкнені до 2 та 3 порту, а із другої – до 4 і 5. Під час знаходження інформації, читання та виконання практики я дізнався, що за правилами гарного тону потрібно виділяти сервери в окремий сегмент відмінний від сегментів користувачів. Отже сервер становиться 4-ю областю віртуальної локальної мережі і підключений до 6-го порту комутатора. В command line interface заходимо в режим глобального конфігурування та створимо Vlan-и, про які було описано раніше. Їх вийшло 3 і кожен я назвав своїм ім’ям («second», «third», «dhcp» - воно належить серверу).

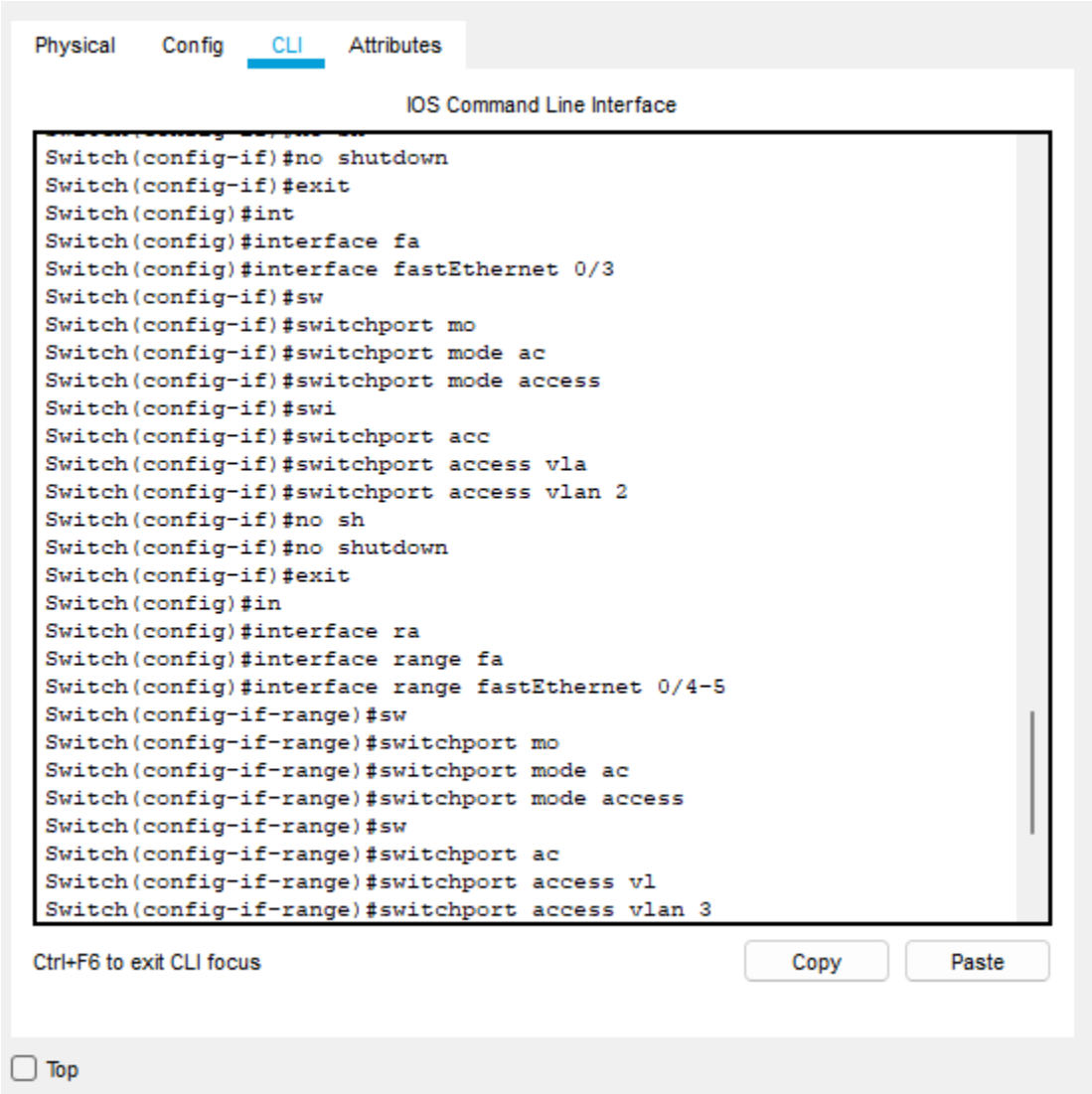

Рисунок 2.2 - створення Vlan для кожного сегменту мережі

Після цього, початок налаштування портів у свічі. Потрібно задати 2-у ВЛМ на порти fastEthernet 0/2 і 0/3. За допомоги команди «range» це можна налаштувати відразу на два. Повністю вона має такий вигляд «interface range fastEthernet 0/2-3». Аналогічні дії робимо разом з 4, 5 і 6 підключенням. Тепер потрібно протягнути ці пристрої до роутеру, тому що саме там вони будуть рухатися. Порт fastEthernet 0/1 потрібно поставити в режим «trunk» для того, щоб він завжди знаходився в режимі магістралі та передавав трафік декільком мережам. В команді «switchport trunk allowed vlan 2,3,4» ми позначаємо, що три ВЛМ можуть проходити через маршрутизатор. Після налаштувань потрібно зберегти дані та вписати команду «show run», щоб переконатися в правильності

задання інформації. Там ми можемо утвердитися, що дані пристрої знаходяться у своїх vlan та перевірити їхні режими.

```
!
spanning-tree mode pvst
spanning-tree extend system-id
!
interface FastEthernet0/1
 switchport trunk allowed vlan 2-4
 switchport mode trunk
!
interface FastEthernet0/2
 switchport access vlan 2
 switchport mode access
!
interface FastEthernet0/3
 switchport access vlan 2
 switchport mode access
!
interface FastEthernet0/4
 switchport access vlan 3
 switchport mode access
!
interface FastEthernet0/5
 switchport access vlan 3
 switchport mode access
!
interface FastEthernet0/6
 switchport access vlan 4
 switchport mode access
!
interface FastEthernet0/7
!
```

Рисунок 2.3 - Перевірка портів завдяки команді «show run»

Тепер налаштуємо інтерфейс на маршрутизаторі. Спочатку треба включити фізичний interface командою «no shutdown». В рядку «interface gigabitEthernet 0/0.2» сказано про створення субінтерфейсу для 2-го влану. «encapsulation dot1Q 2» - вказуємо, що він буде приймати інформацію від 2-го vlan. І наступною командою («ip address 192.168.20.1 255.255.255.0») задаємо IP-адресу та 24 бітну маску для даного субінтерфейсу. Такі дії робимо для 3 та 4 ВЛМ.

В налагодженні серверу необхідно задати статичний IP, маску та шлюз за замовчуванням (це природня адреса, котра знаходиться в роутері). Для цього нам потрібно натиснути на самий сервер та піти по такому шляху: «Desktop – IP Configuration – IP Address».

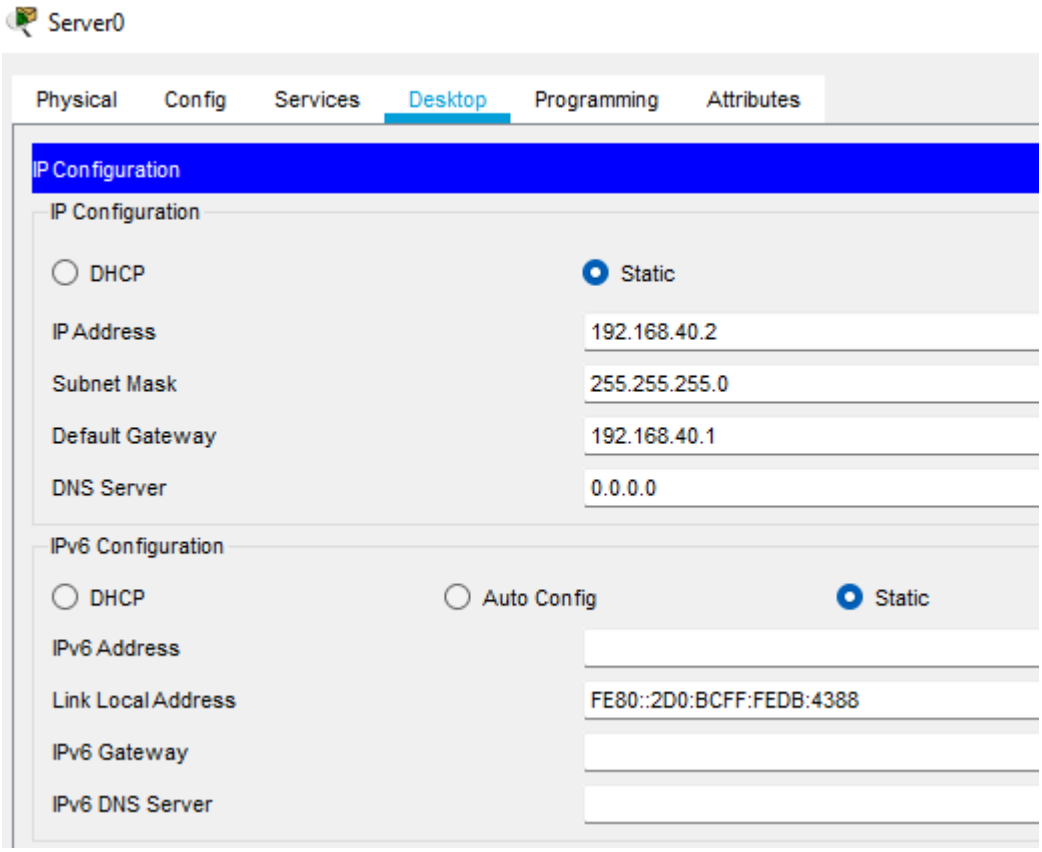

Рисунок 2.4 - Задання статичної IP-адреси серверу

У вкладці «Config - DHCP» можна побачити вже створений стандартний пул. Але його не достатньо, щоб зробити автоматичну видачу адресів. Перший басейн створимо для Vlan-у другої мережі та назвемо «MainDHCPV2». Так як на роутері була вписана адреса 192.168.20.1 на другу ВЛМ (вона буде рахуватися, як і дефолтний шлюз), то початок IP-адресів буде починатися з нуля (192.168.20.0). Після введення даних потрібно ввімкнути та додати пул до списку інших щоб сервер почав роздавати IP відповідним сегментам. Аналогічно робимо для 3-ї ВЛМ.

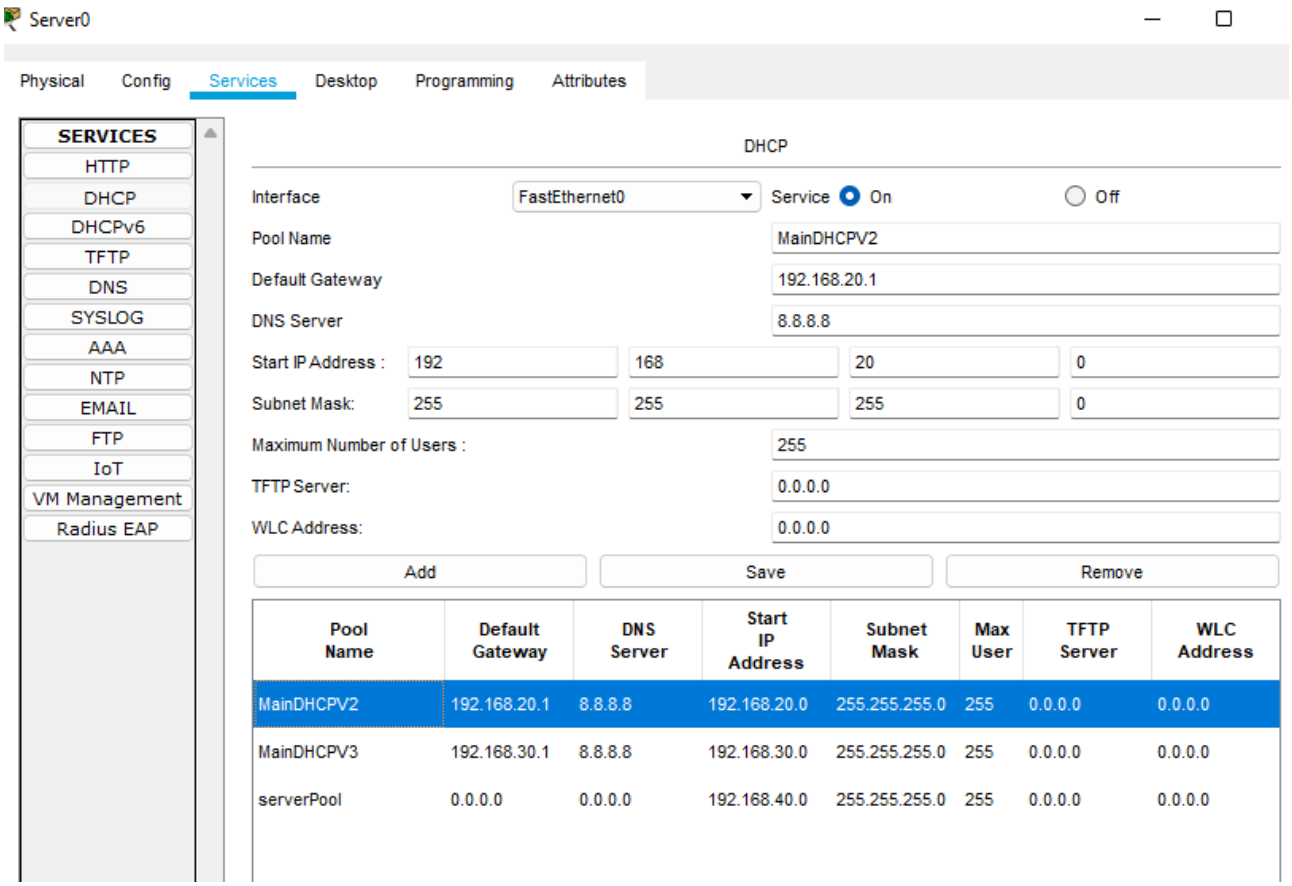

Рисунок 2.5 - Створення DHCP-пулів для різних віртуальних мереж

Так як, SERVER0 знаходиться в окремій області від комп’ютерів, то широкомовний запит від клієнтів, при пошуку DHCP-серверу не дійде до 4 мережі. Необхідно переадресувати запити від Vlan2 та Vlan3 до Vlan 4. Саме для таких проблем було створено таку функцію, як «DHCP Relay». Налаштування відбувається на роутері на основі команди «ip helper-address». Це допоможе кожному субінтерфейсу переадресувати DHCP-запит до існуючого серверу. Після всіх дій можна попробувати задати адреси комп’ютерам за допомоги протоколу. Всього-навсього треба клацнути на комп’ютер зайти до вкладки «Desktop» натиснути на «IP Configuration» та перевести його із статичного до DHCP.

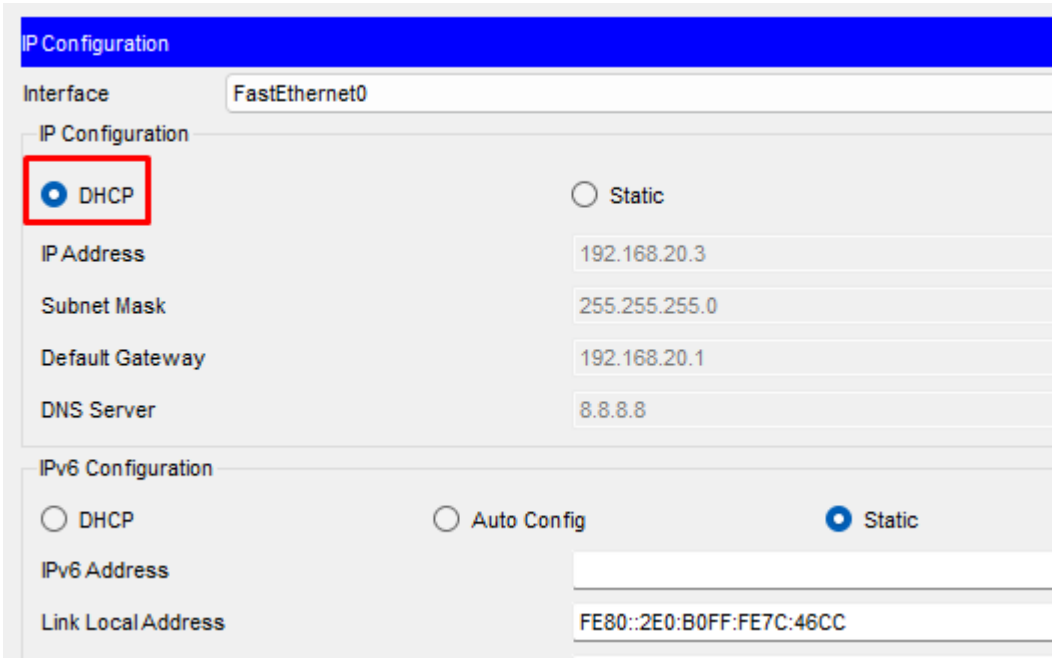

Рисунок 2.6 - автоматичне отримання IP-адреси по DHCP-протоколу

Якщо все зроблено вірно, то клієнт отримає адресу задану в межах пулу для свого сигменту та напис «DHCP request successful». Такі дії потрібно зробити на кожному PC. Для перевірки можна відправити листа до іншого ПК.

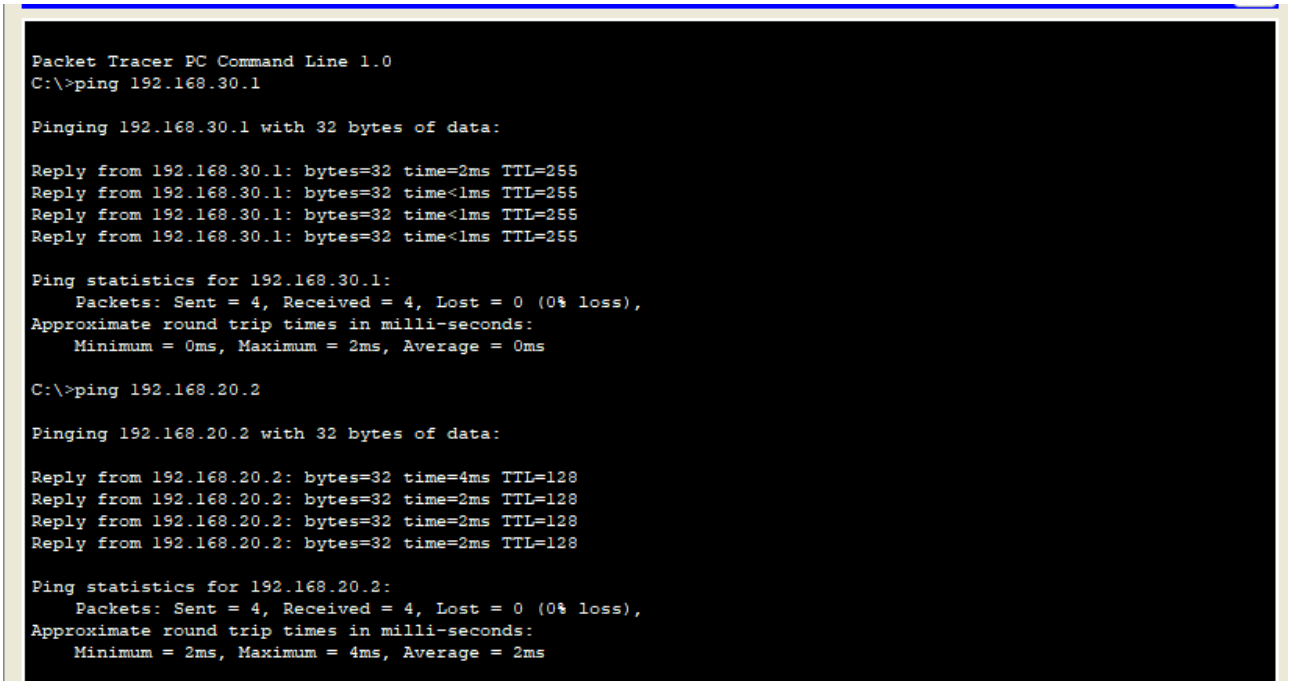

Рисунок 2.7 - Перевірка ping серед комп’ютерів даної мережі

Як бачимо інформація дійшла до отримувача і це означає, що всі дії зроблені правильні, мережа налаштована і готова до використання.

РОЗДІЛ 3

Створення та описання функціоналу веб-додатку

3.1 Описання робочих інструментів для поставленої задачі

Для виконання даного завдання, було вирішено використовувати такі програми: JavaScript, HTML та CSS. Вивчення цих додатків відбувалося в університеті і було зрозуміло, що вони допоможуть зробити у повному обсязі весь поставлений функціонал для автоматичного налаштування Vlan і DHCP. Далі докладний опис про кожен із додатків.

JavaScript (зазвичай просто JS) — це неважка, інтерпретована або JIT-компільована об'єктно-орієнтована мова з першокласними функціями. На даний момент широке застосування вона знайшла у постанові сценаріїв веб-сторінок, але вона також використовується в інших програмних продуктах, таких як Apache CouchDB або node.js. З її використанням можна створювати анімовану 2D та 3D-графіку, а разом з цим ігри, програми з базами даних веб-сторінки тощо. Сама розробка являється прототипно-орієнтованою, багатопарадигмальною мовою з динамічною типізацією, яка підтримує об’єктно-орієнтований, імперативний та декларативний (наприклад, функціональне програмування) стилі програмування [10]. JavaScript сам по собі дуже компактний і в той же момент дуже гнучкий. Розробники написали велику кількість інструментів, поверх основної мови, які можуть легко розблокувати велику кількість додаткових функцій. До таких відносяться:

- Застосування сторонніх фреймворків та бібліотек до HTML, дозволяє не тільки спростити, але і прискорити створення сайтів і програм.
- API, котрі вбудовані в браузер, забезпечують такі функції, як динамічне створення HTML і стилізації CSS, захоплення відеопотоків і керування ними, праця з веб-камеро. користувача або створення тривимірної графіки та аудіо-зразків.

- Сторонні API дозволяють розробникам реалізувати на своєму веб-сайті функції інших розробників, наприклад таких як Facebook або Twitter.

Його сильні сторони полягають у тому, що він повністю інтегрований з HTML та CSS і підтримується абсолютно всіма відомими браузерами та його включено за замовченням. Хоч він і дозволяє робити програми на мобільних пристроях та серверах, але найпоширенішим він все-таки залишається для створення інтерфейсів у браузері.

HTML (Hypertext Markup Language) — це код, який використовується для створення та відображення веб-сторінок та їх вмісту. Наприклад, вміст може бути структурований у вигляді кількох абзаців, маркованих списків або за допомогою зображень і таблиць даних. Він складається з ряду елементів, які можна використовувати для вбудовування або обертання різних частин вмісту, щоб зробити його видимим або маніпулювати певним чином [11]. Закриваючі теги можуть зробити слово або зображення посиланням на щось, виділити шрифт курсивом, збільшити або зменшити шрифт тощо. Головними частинами розмітки є:

- Початковий тег складається з назви елемента, укладеного у відкриваючу та закриваючу кутові дужки. Саме він вказує, де починається або починає функціонувати елемент.
- Контент являє собою звичайний текст замкнутий у початковий та кінцевий тег.
- Кінцевий тег: це те саме, що і початковий тег, за винятком того, що він містить косу риску перед назвою елемента. Він вказує, де закінчується елемент. Їх відсутність є однією з найпоширеніших помилок новачків і може призвести до неприємних результатів.

Із цих трьох частин створюється елемент HTML. Також до них може бути добавлений атрибут. Він містить додаткову інформацію про деталі, які потрібно відобразити у фактичному вмісті. Слово «class», вписане перед властивістю, дозволяє дати компоненту ідентифікаційне ім’я, яке пізніше можна буде

використовувати для посилання на елемент, котре містить інформацію про стилі та інший вміст. Із цього витікає, що основне завдання HTML - інформувати браузер про смислове навантаження елементів, які знаходяться на веб-сторінці. Сам код дозволяє клієнту «розмітити» веб-сторінку, як одну область і повідомити браузеру, що це за деталь.

CSS (Cascading Style Sheets) або каскадні таблиці стилів — це мова, яка використовується для опису зовнішнього вигляду документів, написаних мовами розмітки. Часто використовується для опису веб-дизайну, написаного за допомогою розмітки вищесказаного HTML. Він використовується для визначення шрифтів і кольорів, положення окремих елементів та інших параметрів зовнішнього вигляду сторінки. CSS відокремлює опис зовнішнього вигляду від логічної структури веб-сторінки, реалізованої за допомогою HTML [12]. Цей поділ збільшує доступність документів і забезпечує більшу гнучкість для керування ними. CSS3 — остання версія стандарту, яка внесла багато нововведень у розробку веб-інтерфейсів. Існують також препроцесори CSS, які спрощують розробку візуальних утворень, додаючи до мови описи проектування можливостей мови програмування (наприклад змінні функції). Зовнішня таблиця стилів являє собою текстовий файл із розширенням .css, який містить набір елементів стилю CSS. Файл створюється в редакторі коду, наприклад на сторінці HTML. У файлі можуть бути лише стилі, без тегів коду розмітки. Існує три способи застосування таблиць стилів до документів HTML:

- Вибудовуваний. Цей метод дозволяє застосувати стилі до заданого тегу HTML.
- Впроваджений. Використання дозволяє повністю керувати стилями сторінок.
- Зв’язуваний або зовнішній. Пов’язані таблиці стилів дозволяють описувати стилі у зовнішньому файлі, на який можна посилатися, щоб керувати відображенням усіх сторінок вашого сайту.

Під час виконання завдання у симуляторі Cisco Packet Tracer я побачив, що створення віртуальних локальних мереж, опис портів та задання їм режиму (access або trunk), описання нових субінтерфейсів – всі команди є подібними окрім вказування IP-адрес, масок і до яких Vlan вони належать. Перша проблема виступає в правильності написання команд комутатору чи роутеру. Бо маленька похибка, яку не так вже і легко знайти, може зіпсувати всю схему та призвести до масштабних проблем. І це вже не говорячи про прикладне обладнання, де не відразу можна передивитися всі налаштування. Друга проблема є в одноманітності написання одних і тих же команд, які можна просто винести в одну із замінами деяких чисел. Третя складність ґрунтується на часі написання вказівок. Налаштування топології, котра складається із 10 комп'ютерів, буде вимагати більше часу ніж із 2-х пристроїв.

3.2 Описання програми та перевірка на правильність

Проаналізувавши команди, виписав позитивні і негативні збіги. На їхній основі будуть створенні вікна для вводу даних. Далі почалося продумування інтерфейсу для зручності, зрозумілості, компактності та коректності. Початкове вікно програми має три поля у яких проситься вказати кількість пристроїв, роутерів та віртуальних локальних мереж. Сам дизайн розроблявся під сучасний стиль. Потрібно було зробити її легкою у користуванні та не навантажувати різними непотрібними функціями. Дані вводились на основі зробленої раніше топології у симуляторі.

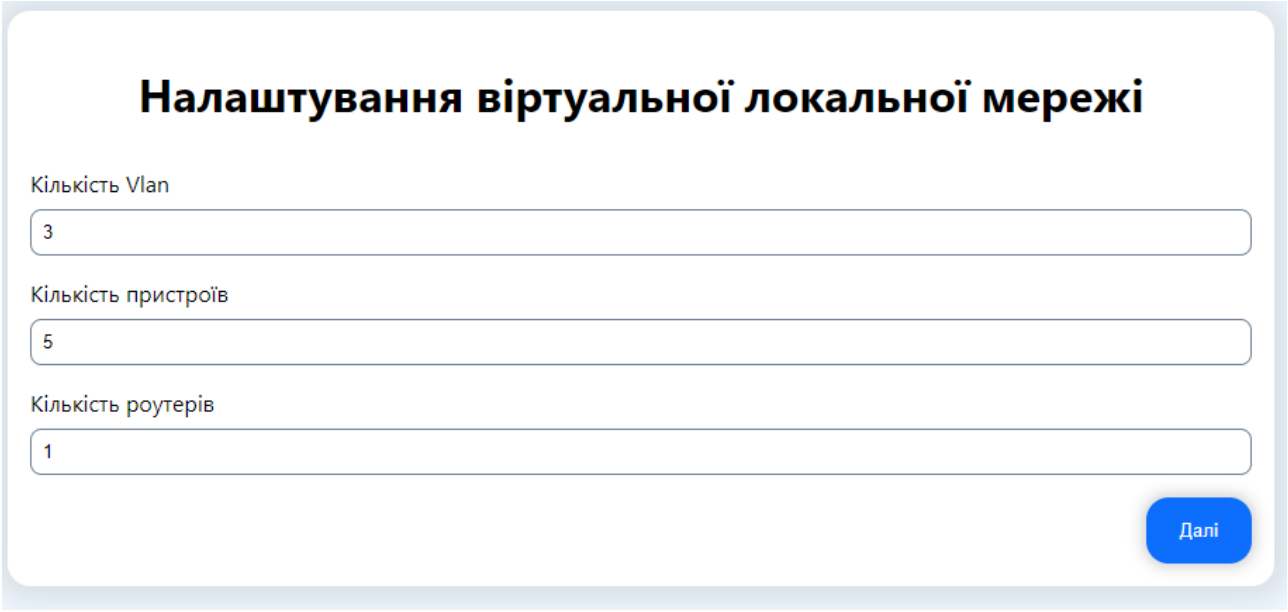

Рисунок 3.1 - Початкове вікно налаштувань

Згідно з тим, скільки було введено клієнтів, з’являється відповідна кількість вікон для налаштування кожного. Треба ввести порт, яким підключено сам комп’ютер та номер мережі до якого він належить. Це не займає багато часу, тим паче, що всю інформацію можна скопіювати і змінити значення на коректні. В даному випадку це п’ять пристроїв та три Vlan. Якщо, в ході роботи, необхідно ввести інші значення то завжди можна натиснути кнопку «Назад» і змінити їх. Всі введені раніше дані збережуться і масштабних змін не відбудиться. Але все-таки потрібно бути уважним з числами, які вписані в поля.

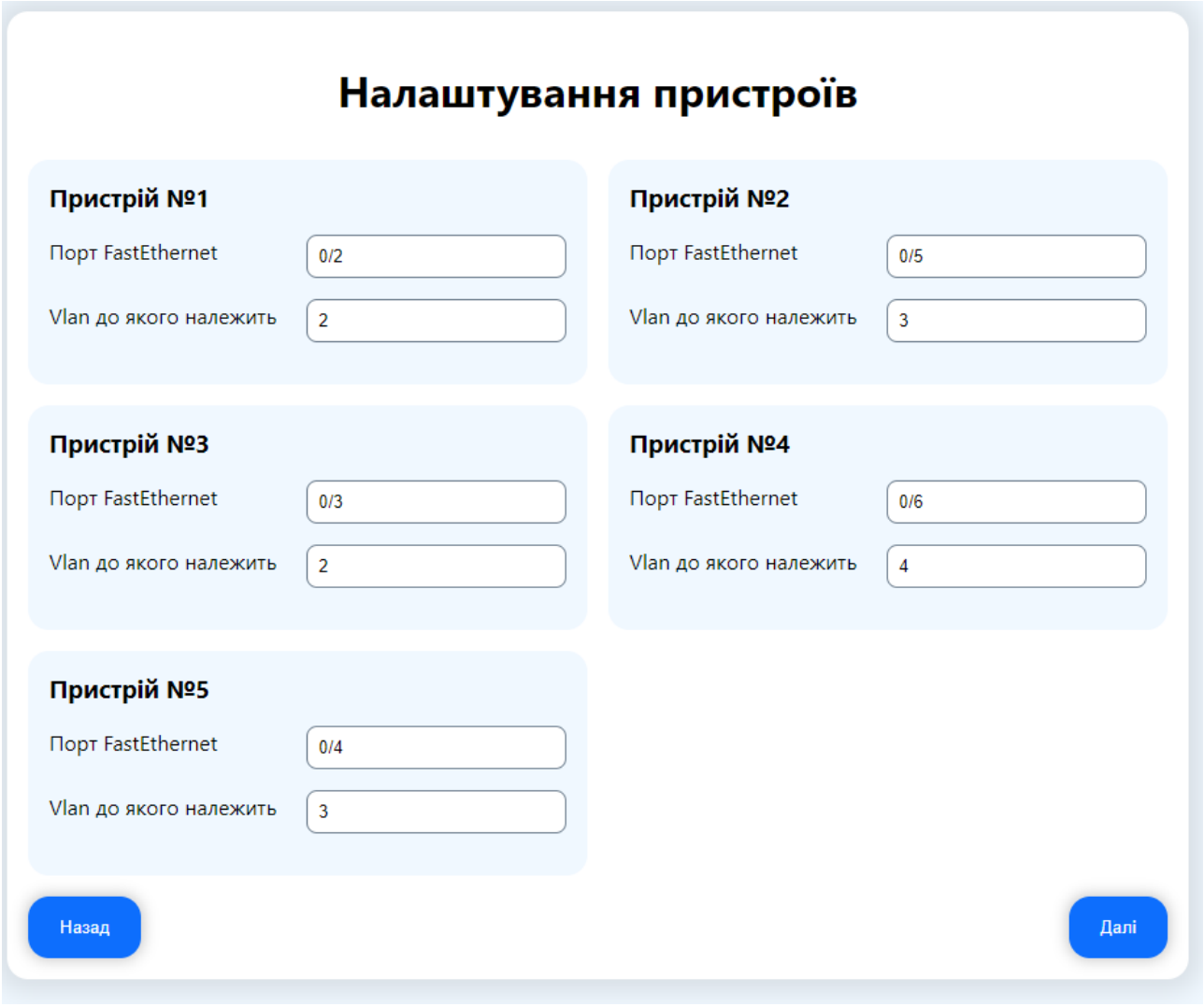

Рисунок 3.2 - Введення даних кожного пристрою

Окрім налаштування ВЛМ, програма зможе додатково прописати команди, які знадобляться до протоколу DHCP. Тому третя сторінка зроблена для підтримки і такої функції. Слід занести дані до налаштування роутеру про місце підключення та які саме мережі він контролює.

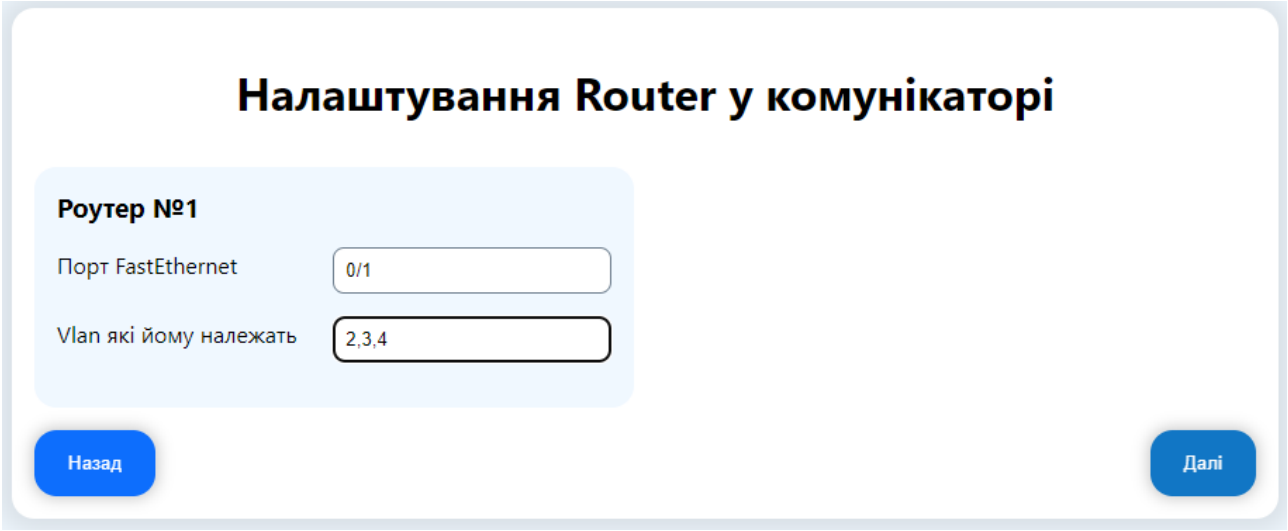

Рисунок 3.3 - Робота з Router для подальшого налаштування всієї мережі

Для подальшої роботи з протоколом необхідно вказати субінтерфейс, його ip-адресу та маску і трафік якого сегменту він буде приймати. Саме для цього було розроблене четверте вікно додатку і команди згенеровані у ньому будуть відноситися до маршрутизатору. До речі, в файлі відразу будуть занесені команди для запитів комп’ютерів до подальшого автоматичного налаштування ip-адрес. Це вікно допоможе за малу кількість часу налаштувати одну із головних пристроїв у даній топології. Головне бути пильним у введеній інформації.

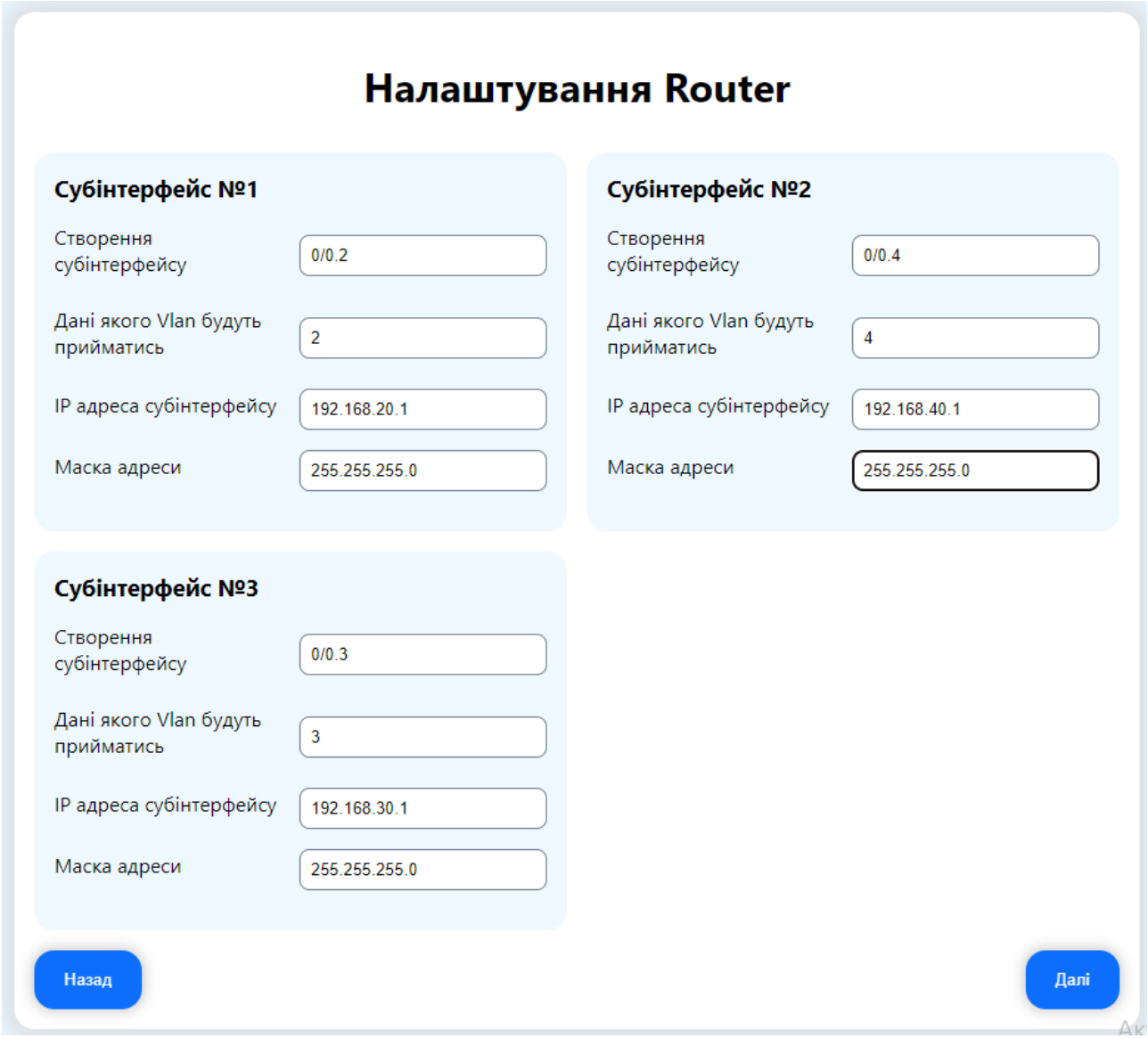

Рисунок 3.4 - Вікно налаштування підінтерфейсів

В кінці з’являється два віконця в котрих вказано які команди відносять до маршрутизатору, а які до комутатору. Під низом відразу знаходяться кнопки для копіювання всього коду, який потім можна вводити в CLI симулятору. На цьому автоматичне налагодження мережі закінчено. Залишилося тільки спробувати правильність роботи програми та переконатися в її коректності.

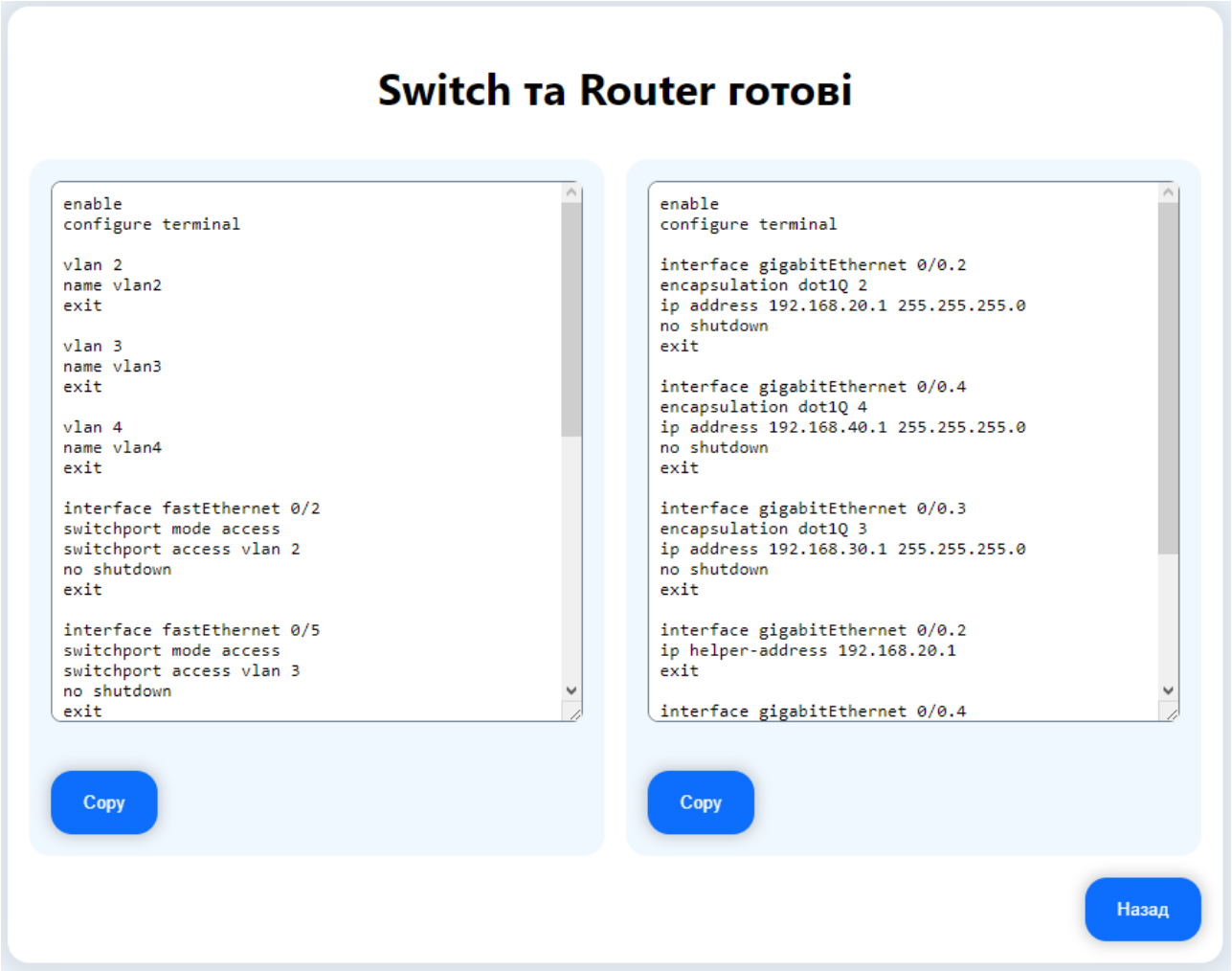

Рисунок 3.5 - Виконання запитів додатком

Для перевірки зроблена топологія однакова із основною, але тепер налаштування буде проводитись за допомогою створеної програми. Після введення даних отримуємо інформацію для роутера та свічу. Вводимо їх у симулятор, налаштовуємо пул для протоколу DHCP на сервері, автоматично підключаємо їх у комп’ютерах. Після всіх дій спробуємо передати листа із однієї локальної мережі до іншої та навпаки. Як показує симулятор, передача даних відбулася успішно.

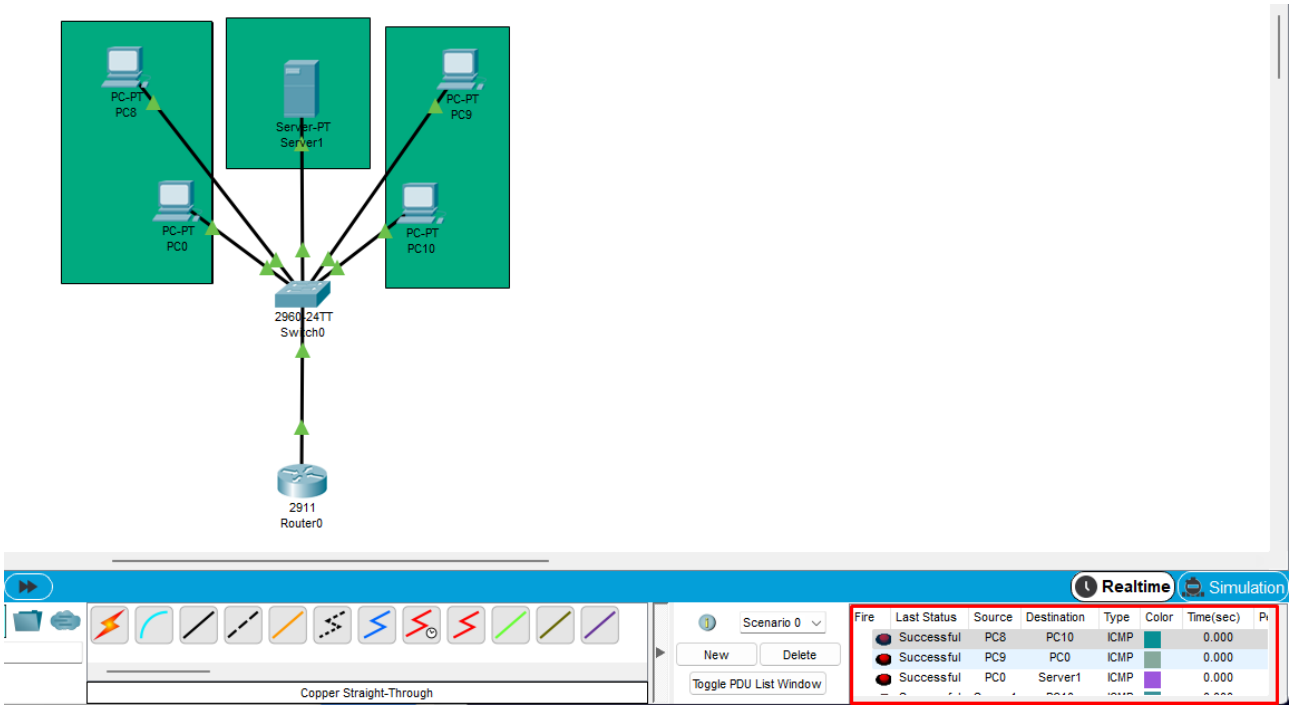

Рисунок 3.6 - Перевірка додатку на коректність налаштування

На далі веб-додаток зможе допомагати у генеруванні команд для налаштувань не тільки подібних топологій, а і мереж у яких використовуються дані технології. На мою думку, особливо корисною програма є для Vlan, так як всі рядки, які мені надав додаток можна відразу вводити у командний рядок симулятору і не переживати стосовно налаштування цього розділу. Також зменшився час у написанні і впевненість в коректності запитів.

ВИСНОВОК

Під час виконання даної роботи було переглянуто такі теми як: мережа Ethernet та її історія створення і розвитку, протокол управління мережею DHCP разом з цим прочитано про інші протоколи та функції, які використовуються на сьогоднішній день, використання симулятору Cisco Packet Tracer і нововведення у даній програмі, віртуальні локальні мережі, їх побудова та призначення у створенні комп'ютерних сіток. В практичній частині розроблено топологію із чотирьох комп'ютерів, одного роутеру, серверу та комутатору в котрій використовуються вищевикладені методи. Відкриті нові навички у створенні окремих сегментів у комп'ютерному моделюванні.

Створено веб-додаток для автоматичного налаштування вищеперерахованих технологій у симуляторі. Удосконаленні знання з web розробки, які в подальшому знадобляться для інших проектів. Сама програма дала змогу вставляти перевірену інформацію у командний рядок із цим працювати швидше та безпечніше.

Дана робота допомогла зрозуміти, як використовуються Vlan-и та DHCP-протокол їхнє виникнення, формування, вклад у сучасну технологію інтернет та допомога в реальному світі. А створення додатку показало, що інформаційні технології зроблені для полегшення життя та роблять великий вклад до нашого майбутнього.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ethernet – 40 лет: от наброска на салфетке до гигабитных линий связи [Електронний ресурс] – Режим доступу: https://habr.com/ru/post/526670/
2. Технология Ethernet [Електронний ресурс] – Режим доступу: https://skomplekt.com/technology/ethernet.htm/
3. Протокол DHCP [Електронний ресурс] – Режим доступу: https://docs.microsoft.com/ru-ru/windows-server/networking/technologies/dhcp/dhcp-top
4. DHCP vs статический IP: какой из них лучше? [Електронний ресурс] – Режим доступу: https://community.fs.com/ru/blog/dhcp-vs-static-ip-differences.html
5. Руководство по DHCP для пользователей [Електронний ресурс] – Режим доступу: https://www.juniper.net/documentation/ru/ru/software/junos/dhcp/topics/topic-map/dhcp-overview.html
6. VLAN на пользователя: архитектура и альтернативы [Електронний ресурс] – Режим доступу: https://www.imena.ua/blog/vlan-na-pol'zovatelya-arhitektura-i-al'te/
7. Маршрутизация между vlans [Електронний ресурс] – Режим доступу: http://www.netconfig.org/routing/864/
8. CISCO CCNA Самостоятельная подготовка к экзамену [Електронний ресурс] – Режим доступу: http://ccnastepbystep.blogspot.com/2010/11/cisco-ccna-640-802-15-configuring-vtp.html
9. DHCP по VLAN на коммутаторах Cisco [Електронний ресурс] – Режим доступу: https://habr.com/ru/post/113431/
10. VLAN и настройка DHCP на активном сетевом оборудовании Cisco [Електронний ресурс] – Режим доступу: https://slipsoad.wordpress.com/cisco-ios-2/vlan-и-настройка-dhcp-на-активном-сетевом-обо/

11. Основы JavaScript [Електронний ресурс] – Режим доступу: https://developer.mozilla.org/ru/docs/Learn/Getting_started_with_the_web/JavaScript_basics
12. Что такое HTML и почему его должен знать каждый веб-разработчик [Електронний ресурс] – Режим доступу: https://skillbox.ru/media/code/chto_takoe_html/
13. Каскадные таблицы стилей [Електронний ресурс] – Режим доступу: https://iit-web-lectures.readthedocs.io/ru/latest/www/css.html#id1
14. CSS. Полный справочник. Визуальное форматирование веб-страниц. 4-е издание / Э. Мейер, Э. Уэйл, 2019, -1088 с.- (Диэлектрика)
15. JavaScript на примерах. Практика, практика и только практика / Никольский А. П., 2018, -272 с.- (Наука и техника)

ДОДАТОК

Switch

> enable

> configure terminal

> vlan 2

> name second

> exit

> vlan 3

> name third

> exit

> vlan 4

> name dhcp

> exit

> interface fastEthernet 0/2

> switchport mode access

> switchport access vlan 2

> no shutdown

> exit

> interface fastEthernet 0/3

> switchport mode access

> switchport access vlan 2

> no shutdown

> exit

> interface range fastEthernet 0/4-5

> switchport mode access

> switchport access vlan 3

> no shutdown

> exit

> interface fastEthernet 0/6

> switchport mode access

> switchport access vlan 4

> no shutdown

> exit

> interface fastEthernet 0/1

> switchport mode trunk

> switchport trunk allowed vlan 2,3,4

> no shutdown

> exit

> end

> write memory

Router

> enable

> configure terminal

> interface gigabitEthernet 0/0

> no shutdown

> exit

> interface gigabitEthernet 0/0.2

> encapsulation dot1Q 2

> ip address 192.168.20.1 255.255.255.0

> no shutdown

> exit

> interface gigabitEthernet 0/0.3

> encapsulation dot1Q 3

> ip address 192.168.30.1 255.255.255.0

> no shutdown

> exit

> interface gigabitEthernet 0/0.4

```
> encapsulation dot1Q 4
> ip address 192.168.40.1 255.255.255.0
> no shutdown
> end
> write memory
> configure terminal
> interface gigabitEthernet 0/0.2
> ip helper-address 192.168.40.2
> exit
> interface gigabitEthernet 0/0.3
> ip helper-address 192.168.40.2
> exit
> interface gigabitEthernet 0/0.2
> ip helper-address 192.168.40.2
> end
> write memory
```

JavaScript

```
import React, { useEffect, useState } from "react";
import styles from "./index.module.scss";
import Init from "../UI/Organisms/Init";
import Device from "../UI/Organisms/Device";
import Router from "../UI/Organisms/Router";
import SubRouter from "../UI/Organisms/SubRouter";
import Result from "../UI/Organisms/Result";

export default function App() {
   const defaultData = {
      init: {
         vlan: {
            label: "Кількість Vlan",
            value: "",
            count: 10,
         },
```

```
    device: {
        label: "Кількість пристроїв",
        value: "",
        count: 10,
    },
    router: {
        label: "Кількість роутерів",
        value: "",
        count: 5,
    },
},
device: {},
router: {},
vlan: {},
templates: {
    device: {
        ethernet: {
            label: "Порт FastEthernet",
            value: "",
        },
        vlan: {
            label: "Vlan до якого належить",
            value: "",
        },
    },
    router: {
        ethernet: {
            label: "Порт FastEthernet",
            value: "",
        },
        vlan: {
            label: "Vlan які йому належать",
            value: "",
        },
    },
    vlan: {
        face: {
            label: "Створення субінтерфейсу",
            value: "",
        },
```

```
            vlan: {
                label: "Дані якого Vlan будуть прийматись",
                value: "",
            },
            ip: {
                label: "IP адреса субінтерфейсу",
                value: "",
                placeholder: "xxx.xxx.xxx.xxx",
            },
            mask: {
                label: "Маска адреси",
                value: "",
                placeholder: "xxx.xxx.xxx.xxx",
            },
        },
    },
};

const [data, setData] = useState(defaultData);
const [tab, setTab] = useState(0);

const tabList = [Init, Device, Router, SubRouter, Result];

useEffect(() => {
    const keys = Object.keys(data.init);
    const newData = { ...data };

    for (const key of keys) {
        const items = {};
        for (let i = 0; i < Number(newData.init[key].value); i++) {
            items[i] = { ...newData.templates[key] };
        }
        newData[key] = items;
    }

    setData(newData);
    // eslint-disable-next-line react-hooks/exhaustive-deps
}, [data.init]);

return (
```

```
      <main className={styles.main}>
        <div className={styles.mainInner}>
          {tabList.map(
            (Component, index) =>
              tab === index && (
                <Component
                  key={index}
                  data={data}
                  setData={setData}
                  setTab={setTab}
                  index={index}
                  tabListCount={tabList.length}
                  onClic={() => setTab(index)}
                />
              )
          )}
        </div>
      </main>
    );
}

import React from "react";
import Nav from "../../Molecules/Nav";
import Input from "../../Atoms/Input";
import Item from "../../Molecules/Item";
import Items from "../../Molecules/Items";

export default function SubRouter({ data, setData, setTab, index, tabListCount }) {
    const pushData = (e) => {
      const [index, name] = e.target.name.split("-");

      const updateData = {
        ...data,
        vlan: {
          ...data.vlan,
          [index]: {
            ...data.vlan[index],
            [name]: {
              ...data.vlan[index][name],
              value: e.target.value,
```

```
            },
          },
        },
      };
      setData(updateData);
    };

    const next = () => {
      for (const [index, item] of Object.entries(data.vlan)) {
        for (const { label, value } of Object.values(item)) {
          if (value.length === 0)
            return alert(
              `Заповніть ${label} субінтерфейс  №${Number(index) + 1}.`
            );
        }
      }
      setTab(index + 1);
    };

    return (
      <div>
        <h1>Налаштування Router</h1>
        <Items>
          {Object.values(data.vlan).map((item, index) => (
            <Item key={index}>
              <h3>Субінтерфейс №{index + 1}</h3>
              {Object.entries(item).map(([key, { label, value, placeholder }]) => (
                <Input
                  key={key}
                  label={label}
                  name={`${index}-${key}`}
                  value={value}
                  placeholder={placeholder}
                  onChange={pushData}
                />
              ))}
            </Item>
          ))}
        </Items>
        <Nav
```

```
            index={index}
            setTab={setTab}
            tabListCount={tabListCount}
            next={next}
          />
        </div>
    );
}
import React from "react";
import styles from "./index.module.scss";
import Button from "../../Atoms/Button";

export default function AppNav({ index, setTab, tabListCount, next }) {
    return (
        <div className={styles.nav}>
            {index + 1 !== tabListCount && <Button onClick={next}>Далі</Button>}
            {index !== 0 && (
                <Button onClick={() => setTab(index - 1)}>Назад</Button>
            )}
        </div>
    );
}
import React from "react";
import Nav from "../../Molecules/Nav";
import Input from "../../Atoms/Input";
import Item from "../../Molecules/Item";
import Items from "../../Molecules/Items";

export default function Router({ data, setData, setTab, index, tabListCount }) {
    const pushData = (e) => {
        const [index, name] = e.target.name.split("-");

        const updateData = {
            ...data,
            router: {
                ...data.router,
                [index]: {
                    ...data.router[index],
                    [name]: {
                        ...data.router[index][name],
```

```
                value: e.target.value,
              },
            },
          },
        };
        setData(updateData);
};

const next = () => {
    for (const [index, item] of Object.entries(data.router)) {
        for (const { label, value } of Object.values(item)) {
            if (value.length === 0)
                return alert(
                    `Заповніть ${label} роутер  №${Number(index) + 1}.`
                );
        }
    }
    setTab(index + 1);
};

return (
    <div>
        <h1>Налаштування Router у комунікаторі</h1>
        <Items>
            {Object.values(data.router).map((item, index) => (
                <Item key={index}>
                    <h3>Роутер №{index + 1}</h3>
                    {Object.entries(item).map(([key, { label, value }]) => (
                        <Input
                            key={key}
                            label={label}
                            name={`${index}-${key}`}
                            value={value}
                            onChange={pushData}
                        />
                    ))}
                </Item>
            ))}
        </Items>
        <Nav
```

```
            index={index}
            setTab={setTab}
            tabListCount={tabListCount}
            next={next}
          />
        </div>
    );
}
import React, { useRef } from "react";
import styles from "./index.module.scss";
import Nav from "../../Molecules/Nav";
import Item from "../../Molecules/Item";
import Items from "../../Molecules/Items";
import Button from "../../Atoms/Button";

export default function Result({ data, setData, setTab, index, tabListCount }) {
    console.log(data);
    const refT1 = useRef();
    const refT2 = useRef();

    const computedSwitch = () => {
        let result = `enable\nconfigure terminal\n\n`;
        for (let i = 0; i < Number(data.init.vlan.value); i++) {
            result += `vlan ${i + 2}\nname vlan${i + 2}\nexit\n\n`;
        }
        result += Object.values(data.device)
            .map(({ ethernet, vlan }) => {
                return `interface fastEthernet ${ethernet.value}\nswitchport mode
access\nswitchport access vlan ${vlan.value}\nno shutdown\nexit\n\n`;
            })
            .join("");
        result += Object.values(data.router)
            .map(({ ethernet, vlan }) => {
                return `interface fastEthernet ${ethernet.value}\nswitchport mode
trunk\nswitchport trunk allowed vlan ${vlan.value}\nno shutdown\nexit\n\n`;
            })
            .join("");
        result += `end\nwrite memory`;
        return result;
    };
```

```
  const computedRouter = () => {
    let result = `enable\nconfigure terminal\n\n`;
    result += Object.values(data.vlan)
      .map(({ face, ip, mask, vlan }, index) => {
        return `interface gigabitEthernet ${face.value}\nencapsulation dot1Q ${vlan.value}\nip address ${ip.value} ${mask.value}\nno shutdown\nexit\n\n`;
      })
      .join("");
    result += Object.values(data.vlan)
      .map(({ face, ip, mask, vlan }) => {
        return `interface gigabitEthernet ${face.value}\nip helper-address ${ip.value}\nexit\n\n`;
      })
      .join("");
    result += `end\nwrite memory`;
    return result;
  };

  const copy = (ref) => {
    ref.select();
    document.execCommand("copy");
    alert("Copied the text: " + ref.value);
  };

  return (
    <div>
      <h1>Switch та Router готові</h1>
      <Items>
        <Item>
          <textarea
            ref={refT1}
            className={styles.textarea}
            defaultValue={computedSwitch()}
          ></textarea>
          <Button onClick={() => copy(refT1.current)}>Copy</Button>
        </Item>
        <Item>
          <textarea
            ref={refT2}
```

```
                    className={styles.textarea}
                    defaultValue={computedRouter()}
                  ></textarea>
                  <Button onClick={() => copy(refT2.current)}>Copy</Button>
                </Item>
              </Items>
              <Nav index={index} setTab={setTab} tabListCount={tabListCount} />
            </div>
        );
}
```