UNIVERSITY AMERICAN COLLEGE SKOPJE

SCHOOL OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY

Course

Internet Programming

Title of the project
(Web app for VLSM)

Student: Ivan Markovski
ID: 4286

Instructor Adrijan Bozinovski, *PhD*
Assistant: Bojan Bogdanovski, MSc

Academic year 2020/2021

Table of Contents

1. Outline 3
3. Overview of the application 13
4. Documentation of the application – mini manual 19
References 23

1. Outline

Application that I created is intended to calculate the IP addresses for creating the network from the given IP address space.
So far application can be used to calculate IP addresses for CIDR values from /24 to /32. In the future version's application will be upgraded to calculate the IP address range for CIDR /1 to /32.

Introduction to VLSM

The exponential growth of the Internet in the last 30 years exposed shortcomings in the original IP protocol design. As the internet began to rapidly expand from its initial military network research status into commercial prominence, the demand for IP addresses (particularly in the class B space) skyrocketed.

Experts started to worry about the long-term scaling properties of classes A, B, and C IP address scheme, and began considering ways to modify IP assignment policy and routing protocols to accommodate the growth. This led to the establishment of the Routing and Addressing (ROAD) group by the Internet Engineering Task Force (IETF) in the early 1990s to work out ways of restructuring the IP address space to increase its lifespan. The group according to IETF RFC 4632 identified three major problems:

Exhaustion of the Class B network address space

1. Growth in internet routers' routing tables beyond the capacity of current hardware and software.
2. Eventual exhaustion of the 32-bit IPv4 network address space

As a short to mid-term measure, the ROAD group proposed a solution to allow the use of "classless" IP assignment systems to slow the growth of global routing tables and to reduce the rate of consumption of IPv4 address space. This eventually gave birth to what we now know as Classless Inter-Domain Routing (CIDR), and Variable Length Subnet Mask (VLSM), which allows greater flexibility in the creation of sub-networks,

overcoming the strict rules of the A, B, and C classes. In this guide, we're going to help you understand the concept of VLSM, and show you how to implement VLSM Subnetting.

VLSM Fundamentals

In order to fully grasp the concept of VLSM, we first need to understand the term subnet mask, subnetting, and Supernetting.

Subnet Mask

Subnet masks are used by a computer to determine if any computer is on the same network or on a different network. An IPv4 subnet mask is a 32-bit sequence of ones (1) followed by a block of zeros (0). The ones designate the network prefix, while the trailing block of zeros designates the host identifier. In shorthand, we use /24, which simply means that a subnet mask has 24 ones, and the rest are zeros.

	Binary Notation	Decimal Notation
IP address	11000000.00000000.00000010.10000010	192.0.2.130
Subnet mask	11111111.11111111.11111111.00000000	255.255.255.0

Table 1 IP address and subnet mask in binary and decimal format

Subnetting

As the name implies, subnetting is the process of dividing a single large network into multiple small networks known as subnets. The primary purpose of subnetting is to help relieve network congestion and improve efficiency in the utilization of the relatively small network address space available especially in IPv4.

Supernetting

Supernetting is the direct opposite of subnetting, in which multiple networks are combined into a single large network known as supernets. Supernetting provides route updates in the most efficient way possible by advertising many routes in one advertisement instead of individually.

The main objective of supernetting is to simplify or summarize network routing decisions to minimize processing overhead when matching routes, and storage space of route information on routing tables. A routing table is a summary of all known networks. Routers share routing tables to find the new path and locate the best path for the destination. Without Supernetting, the router will share all routes from routing tables as they are. With Supernetting, it will summarize them before sharing, which significantly reduces the size of routing updates.

There are two approaches to subnetting an IP address for a network: Fixed length subnet mask (FLSM) and variable-length subnet mask (VLSM). In FLSM subnetting, all subnets are of equal size with an equal number of host identifiers. You use the same subnet mask for each subnet, and all the subnets have the same number of addresses in them. It tends to be the most wasteful because it uses more IP addresses than are necessary.

VLSM is a subnet design strategy that allows all subnet masks to have variable sizes. In VLSM subnetting, network administrators can divide an IP address space into subnets of different sizes, and allocate it according to the individual need on a network. This type of subnetting makes more efficient use of a given IP address range. VLSM is the defacto standard for how every network is designed today. Table 2.0 below is a summary of the differences between FLSM and VLSM Subnetting. VLSM is supported by the following protocols: Open Shortest Path First (OSPF), Enhanced Interior Gateway Router Protocol (EIGRP), Border Gateway Protocol (BGP), Routing Information Protocol (RIP) version 2 and 3, and Intermediate System-to-Intermediate System (IS-IS). You need to configure your router for VLSM with one of those protocols.

FLSM (Fixed Length Subnet Masks) Subnetting	VLSM (Variable Length Subnet Masks) Subnetting
Old-fashioned	Modern
Subnets are equal in size	Subnets are variable in size.
Subnets have an equal number of hosts	Subnets have a variable number of hosts
Supports both classful and classless routing protocols	Supports only classless routing protocols
Wastes more IP addresses	Wastes fewer IP addresses
Subnets use the same subnet mask	Subnets use different subnet masks
Simple configuration and administration	Complex configuration and administration

Table 2 Differences between FLSM and VLSM Subnetting

Implementing VLSM Subnetting

We will begin this section by attempting to solve a practical VLSM problem. Now, imagine you were recently hired as a Network Engineer. Using the VLSM technique, design an IP plan for the company with an IP range of 192.168.4.0/24. The company's network consists of three local area networks: LAN A, LAN B, and LAN C as shown in Figure 1 below. These three LANs are connected with three serial links: Link AB, Link BC, and Link AC.

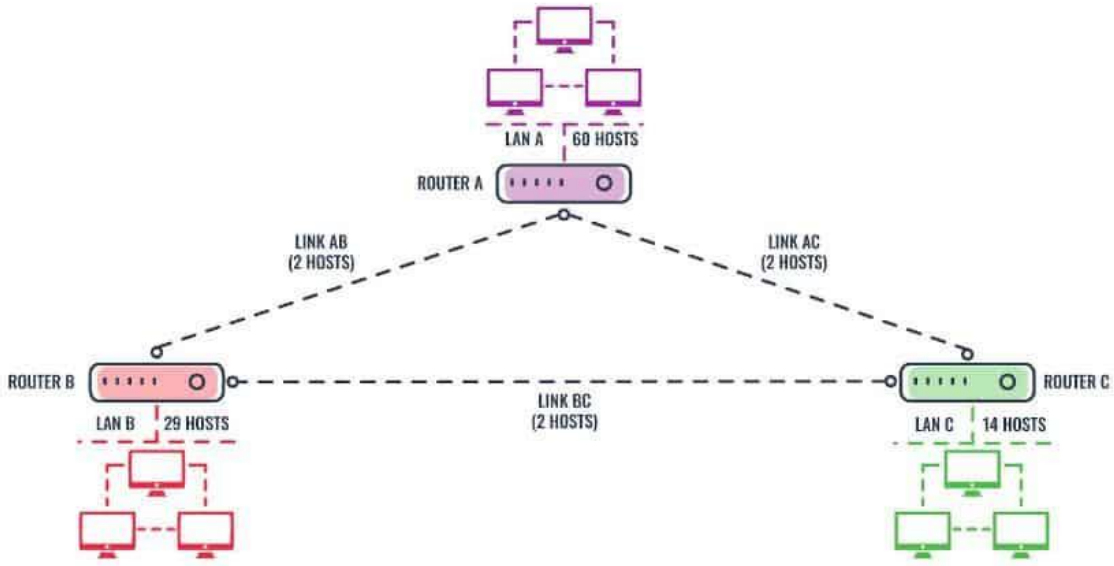


Figure 1 Network diagram

One of the easiest ways to solve VLSM problems is by using a subnetting chart like the one shown in Table 3 below. We will use this chart to tackle the above problem

Subnet	**1**	**2**	**4**	**8**	**16**	**32**	**64**	**128**	**256**
Hosts	256	128	64	32	16	8	4	2	1
SubnetMask	/24	/25	/26	/27	/28	/29	/30	/31	/32

Table 3 VLSM subnetting chart

As you can see from the diagram, we have six networks LAN A, LAN B, LAN C, and link A, link B and a link C. Links A, B, and C are also three separate networks and each requires two host identifiers. Thus, our task is to design an IP plan for each of the six networks according to their stipulated sizes using VLSM subnetting method. We need five steps to solve the problem:

Step 1: Arrange the networks from the largest to the smallest as shown in Table 4 below:

LAN Name	No of Host
LAN A	60
LAN B	29
LAN C	14
Link AB	2
Link AC	2
Link BC	2

Table 4 LAN arranged according to the number of hosts

Step 2: Implement VLSM subnetting for the largest network (LAN A)

The largest network LAN A requires 60 hosts. From the Host section (row) of our subnetting chart below, the closest to the required 60 hosts is 64, which corresponds to 4 subnets and a new CIDR value of /26 (the column is in bold). From this relevant information, we will build a new table containing Network ID, Subnet Mask in CIDR notation, Usable, and Name of Local Area Network affected. Keep in mind the first host identifier is reserved for the network ID and the last host ID is reserved for the broadcast ID, so the total number of usable host IDs for each subnet in this particular case is 62 (64-2).

Subnet	1	2	4	8	16	32	64	128	256
Hosts	256	128	64	32	16	8	4	2	1
SubnetMask	/24	/25	/26	/27	/28	/29	/30	/31	/32

Given the IP range: 192.168.4.0/24

Network ID	Subnet Mask	Total Host	Usable Host Range	Name of LAN
192.168.4.0	/26	64	192.168.4.1-192.168.4.62	LAN A
192.168.4.64	/26	64		Unassigned
192.168.4.128	/26	64		Unassigned
192.168.4.192	/26	64		Unassigned

Table 5 IP plan for LAN A (60 hosts)

Now let's list a network ID for each subnet. Keep in mind that only the fourth octet changes; the first three octets remain the same:

- The first network ID is always the original given ID which is 192.168.4.0
- The second network ID is 192.168.4.64
- The third network ID is 192.168.4.128
- The fourth network ID is 192.168.4.192

Here is the pattern: The first network ID is always the original one. The next network ID is obtained by adding 64 to the previous one. We can assign any of these for subnets to LAN A since they are all equal in size, but for the sake of simplicity, we assign the first subnet (192.168.4.0) to LAN A. The other three available subnets can be marked as unassigned and reserved for future use. We have completed the task of designing the IP plan for the largest LAN – LAN B.

Step 3: Implement VLSM subnetting for the second-largest network (LAN B)

The second-largest network, LAN B, requires 29 hosts. The minimum number of hosts which can satisfy LAN B with the 29 hosts on our subnetting chart is 32. This corresponds to eight subnets and a new CIDR value of /27 (the column is in bold).

Now select the first unassigned large subnet in Table 5.0 above and subdivide into two smaller subnets. This gives us 192.168.4.64 and 192.168.4.96 marked in green in Table 5 below. Again the pattern is simple: The first network ID is always the original one. The next network ID is obtained by adding 32 to the previous one. We can then assign 192.168.4.64 to LAN B, and mark the second one (192.168.4.96) as unassigned and reserved for future use. We have completed designing the IP plan for LAN A.

Subnet	**1**	**2**	**4**	**8**	**16**	**32**	**64**	**128**	**256**
Hosts	256	128	64	**32**	16	8	4	2	1
SubnetMask	/24	/25	/26	**/27**	/28	/29	/30	/31	/32

Table 5

Network ID	**Subnet Mask**	**Total Host**	**Usable Host Range**	**Name of LAN**
192.168.4.64	/27	32	192.168.4.65-192.168.4.94	LAN B
192.168.4.96	/27	32		Unassigned

Table 6 IP plan for LAN B (29 hosts)

Step 4: Implement VLSM subnetting for LAN C

This step repeats the process above. The minimum number of hosts which can satisfy LAN C with the 14 hosts on our subnetting chart is 16. This corresponds to 16 subnets and a new CIDR value of /28 (the column is in bold).

Now select the first unassigned subnet in Table 6.0 above and subdivide into two smaller subnets. This gives us 192.168.4.96 and 192.168.4.112 in Table 7 below. Again the pattern is simple: The first network ID is always the original one. The next network ID is obtained by adding 16 to the previous one. We can then assign 192.168.4.96 to LAN C, and mark the second one (192.168.4.112) as unassigned and reserved for future use. We have completed designing the IP plan for LAN C.

Subnet	**1**	**2**	**4**	**8**	**16**	**32**	**64**	**128**	**256**
Hosts	256	128	64	32	**16**	8	4	2	1
SubnetMask	/24	/25	/26	/27	**/28**	/29	/30	/31	/32

Table 7

Network ID	Subnet Mask	Total Host	Usable Host Range	Name of LAN
192.168.4.96	/28	16	192.168.4.97– 192.168.4.110	LAN C
192.168.4.112	/28	16		Unassigned

Table 8

Step 5: Implement VLSM subnetting for Link A, B, and C
The last step is to assign three smaller subnets for serial links A, B, and C. Each link requires two host IDs. Therefore, the minimum number of hosts which can each link with two hosts on our subnetting chart is four. This corresponds to 64 subnets and a new CIDR value of /30 in our subnetting chart (the column is in bold).

Now select the unassigned subnet in Table 8 above and subdivide into four smaller subnets to accommodate the subnets for the three-serial links. This gives us four unique IPs as shown in Table 10 below.

Subnet	1	2	4	8	16	32	64	128	256
Hosts	256	128	64	32	16	8	4	2	1
SubnetMask	/24	/25	/26	/27	/28	/29	/30	/31	/32

Table 9

Network ID	Subnet Mask	Total Host	Usable Host Range	Name of LAN
192.168.4.112	/30	4	192.168.4.113–192.168.4.114	LINK AB
192.168.4.116	/30	4	192.168.4.117–192.168.4.118	LINK AC
192.168.4.120	/30	4	192.168.4.121–192.168.4.122	LINK BC
192.168.4.124	/30	4		Unassigned

Table 10

Again here's the pattern: The first network ID is always the original one. The next network ID is obtained by adding four to the previous one. We can then assign the first three IPs to Link A, B, and C respectively, and mark the last one (192.168.4.124) as unassigned and reserved for future use. We have completed designing the IP plan for Link A, B, and C, and indeed the entire network. The table below is the complete IP plan.

Network ID	Subnet Mask	Total Host	Usable Host Range	Name of LAN
192.168.4.0	/26	64	192.168.4.1-192.168.4.62	LAN A
192.168.4.64	/27	32	192.168.4.65-192.168.4.94	LAN B
192.168.4.96	/28	16	192.168.4.97– 192.168.4.110	LAN C
192.168.4.112	/30	4	192.168.4.113–192.168.4.114	LINK AB
192.168.4.116	/30	4	192.168.4.117–192.168.4.118	LINK AC
192.168.4.120	/30	4	192.168.4.121–192.168.4.122	LINK BC

Table 9 complete IP plan

VLSM is a crucial technique in modern network design. If you want to design and implement scalable and efficient networks, you should definitely master the art of VLSM subnetting. One of the key objectives of VLSM subnetting in IPv4 is to improve efficiency in the utilization of the space available. This has managed to keep it going in the last 30 years. But on the 25th of November 2019, RIPE Network Coordination Centre announced that it made the final /22 IPv4 address allocation, and has officially run out of IPv4 addresses. A longer-term solution to the eventual exhaustion of the 32-bit IPv4 network address space is the 64-bit IPv6 protocol.

How do you calculate VSLM?
The simplest way to calculate VLSM is by using a subnetting chart like the one shown in Table 3.0 above, and then following the steps below:

1. Arrange the requirements of IP addresses in descending order like the one shown on Table 4 above
2. Using the subnetting chart, assign the appropriate subnet masks to each subnet based on the required number of hosts.
3. Allocate one the resulting subnets to the designated LAN and reserve the rest for future use
4. Pick the next available subnet from step 3 above, and repeat the subnetting process using the chart till you get to last network on your list
5. Review and document your subnetting summary

What does it mean when it says "IP not in subnet range"?
“IP not in subnet range” simply means that you are attempting to use an IP address that doesn’t belong to the block of IP’s defined by the subnet mask in question. Based on our VLSM example above, if the network address and subnet mask for LAN B is 192.168.4.0 and 255.255.255.192 (/26) respectively, and you are trying to use an ip address of 192.168.2.2 then you will get an “ip not in subnet range” error. The only usable host IP addresses in range are 192.168.4.1–192.168.4.62 as shown on Table 9.

How would the use of VLSM impact your choice of routing protocols?
Well, the bad news is that not all routing protocols support VLSM. Classful routing protocols such as RIPv1 and IGRP, do not support VLSM. Therefore, it's important to ensure that you configure your router for VLSM with one of the supported protocols. But the good news is that all current generation of routing protocols such as RIPv2/v3, OSPF, IS-IS, EIGRP, BGP, and even Static routes, are classless and therefore support VLSM.

In the above introduction, how VLSM is performed, the algorithm for the application is developed .

3. Overview of the application

In developing of the application, the following technologies are used:

1. HTML5 for creating the structure of the application
2. JavaScript, and jQuery for functionality
3. CSS, and Bootstrap for user experience
4. PHP for server-side development, i.e., storing results of calculation in MySQL database.

Index.html:

Is divided in three parts:

- Header
- Working part where you can view the content of intro.html, vlsm.html where user can perform the calculations, and to see the stored results from performed calculations in the MySQL database.
- MENU is structured in a way that user can choose how to use the application, it consists of three options: introduction, Calculate and Read Database.

In creating of index.html Is used HTML5, JavaScript and jQuery JavaScript library.

Functionality of the index.html mainly is archived with the following jQuery functions:

- function loadIntro(),
- function calculate()
- function readDatabase()

function loadIntro(), performs the following operations:

- Clears the content of the “ segment in the working part of the index.html
- Loads the intro.html page into the working part of the index.html

function calculate(), performs the following operations:

- Clears the content of the “ segment in the working part of the index.html
- Loads the vlsm.html page from where user can perform calculations for the entered IP address range.

function readDatabase(), performs the following operations:

- Clears the content of the “ segment in the working part of the index.html
- Loads the readDatabase.php file in order user to see content of the MySQL database where results of the VLSM calculations are stored.

vlsm.html

Technology used in creating of the vlsm.html is HTML5, JavaScript, Bootstrap, and jQuery.

vlsm.html contains HTML “form” elements where label for IP address, and input fields for the four values of the IP address space, label and input element for the CIDR value, dynamically created input field for the number of hosts in order networks to be created, button named “Remove Network” is used if user wants to remove the hosts and network.

At the bottom of the vlsm.html there are four buttons:

- Add Network: Serves to add network in order user to enter the hosts for that network.
- Submit to MySQL: Serves to store results of the calculations into MYSQL database.
- Calculate VLSM: Serves to calculate results and to display the results of the calculations in Bootstrap “modal”.
- New Calculation: Serves to reset the form and reload the vlsm.html in order to start new calculations.

Adding the input field’s dynamically for number of hosts is achieved with JavaScript jQuery library.

Functionality of the Bootstrap modal where results are presented as well as is achieved with JavaScript jQuery library.

In the form element there is an “onsubmit” option where function validateForm() is called in order to validate the values of input fields.

All input fields are of type number with predefined range and option “required”, which is another type of validation that all inputs are correct.

Intro.html

Is created with HTML5 it contains paragraph with basic information and introduction to VLSM.

getData.js

Is created with JavaScript and jQuery library. getData.js gives the functionality of the vlsm.html page.

Form line 1 to 7 are declared 6 JavaScript arrays which are used to perform necessary calculations and to store the intermediate results which latter on are used to calculate the final result of the VLSM.

- lanHostsArray : Store values entered in vlsm.html in input fields for “Number of hosts” per network,
- hostsPerLan: Stores adjusted values of number of hosts values to fit appropriate number of hosts
- subnets: Contains values for number of subnets,
- hosts: Contains values for hosts for each subnet,
- submask: Contains values for the submask.

function availableHosts()

Is reading the value of the input field “CIDR” in vlsm.html, returns the value of the total available hosts.

function networkID()

Is used to calculate the network id for entered IP address space in the vlsm.htm.

It reads the values od “CIDR” input field from vlsm.html, as well as read the value of the fourth number of the IP address, and calculates the network ID, and returns the result.

function validateForm()

Is used to validate user input in order correct calculation to be performed. Function read the entered values for all input fields from the form in vlsm.htm and performs necessary validation using the “if else” statement on entered values and returns “true” if values are correct, or “false” if values are incorrect.

This function is used in the function test().

function resetForm()

Uses JavaScript function reset() to reset the form in event of wrong input or when new calculation is required.

This function is used in the function test().

function sumHosts()

Is used to calculate the number of total available hosts, and it is used in the function validateForm(), uses it as one of the conditions for validating if returned value of the sumHosts() is less than 0, function validateForm() return value is set to “false”, and resets the user input and reloads the vlsm.html page.

function numLans()

Reads the values of the entered hosts from the vlsm.html HTML5 form element, validates that entered values are numbers and there are not empty for the number of hosts, then it stores these values into lanHostsArray.

This function is also used in the function test().

function lanHosts()

Adjusts the values of the number of hosts for the values of entered hosts les or equal to 8 adding 2 to the entered value in order accurate calculations to be performed, this function as well as is used in the function test().

function setHosts()

Is filling the array hostsPerLan with values from the lanHostsArray using the actual available hosts by cross referencing the values in hosts array, by rounding the entered values of hosts in the vlsm.htm HTML5 form element on nearest larger value in the hosts array.

Function setHosts() is used in the function test().

function getSubAndMask()

First this function is sorting in accenting order the hostsPerLan array where actual values of the hosts are stored.

Declared variable innerTable is used to dynamically create the HTML5 table where results of VLSM calculations are displayed.

Function reads the values of the IP address input fields, and uses function networkID() to calculate the network ID for the entered IP address space.

Further on function is using the algorithm described in Outline section of this document to perform necessary calculations by reaching the final result which will be displayed in the modal element.

function test ()

Is called on button click of the “Calculate VLSM” button in the vlsm.htm HTML form element.

First checks the returned value of the function validateForm() if it is true it calls in sequence the following functions:

- sumHosts (),
- numLans(),
- lanHosts(),
- setHosts(),
- getSubAndMask().

And user receives the result of the calculation, if return value is false than the form is reset and vlsm.htm is reloaded.

function reFresh()

Is used to refresh and reload the vlsm.htm when button “New Calculation” is clicked as well as when return value of the validateForm() is false.

inDatabase.php

The purpose of the inDatabase.php is to store the results of the calculation in the MySQL database.

It uses the same logic and algorithm like in getData.js to perform the VLSM calculations, with values entered in vlsm.html, after calculating, the result is stored in MySQL database table.

If result is successfully stored user is notified.

readDatabase.php

Purpose of readDatabase.php is to read the content of the database where result of the calculations are stored and display the stored database on user screen.

The CSS code in readDatabase is used to format the output table which contains the information stored I the database.

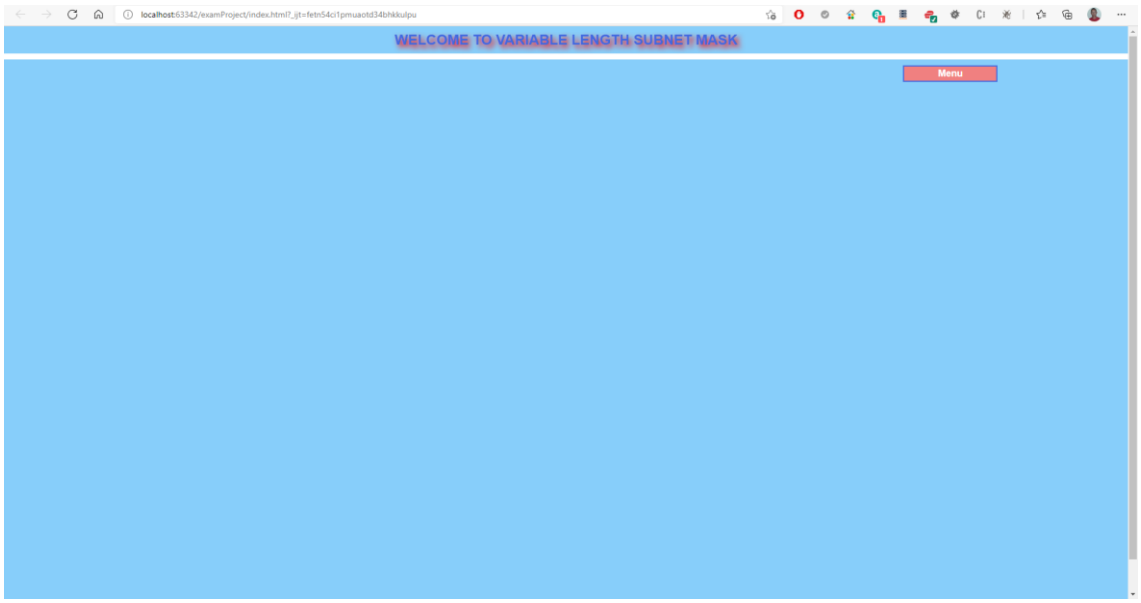
styles.css

Contains CSS code for the table style which contain results of the VLSM calculation.

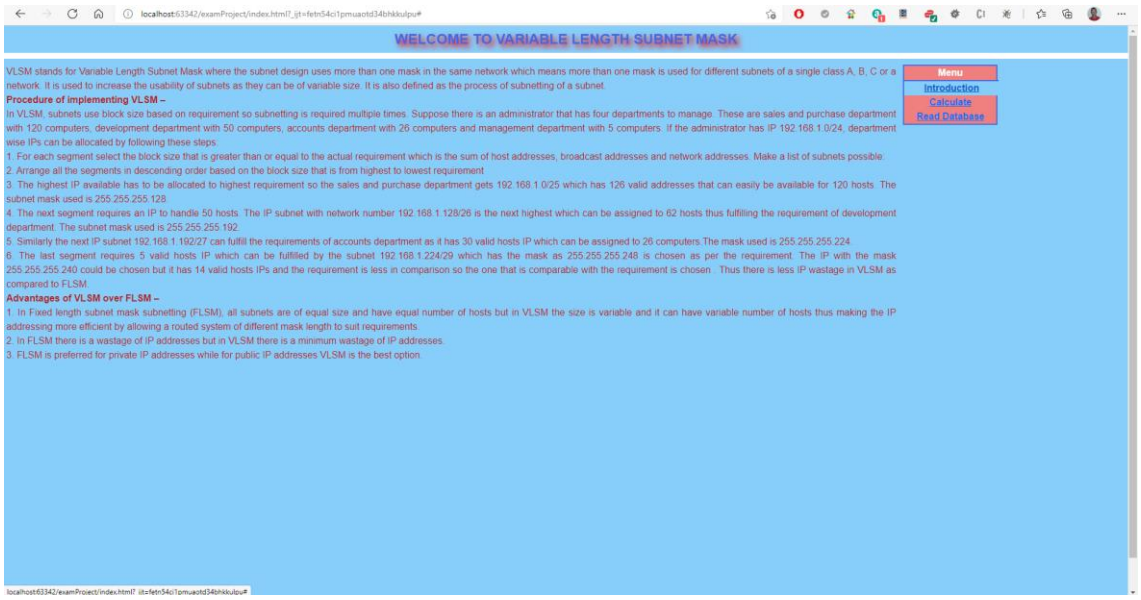
homepage.css

Contains CSS code which formats the elements displayed and used in index.htm.

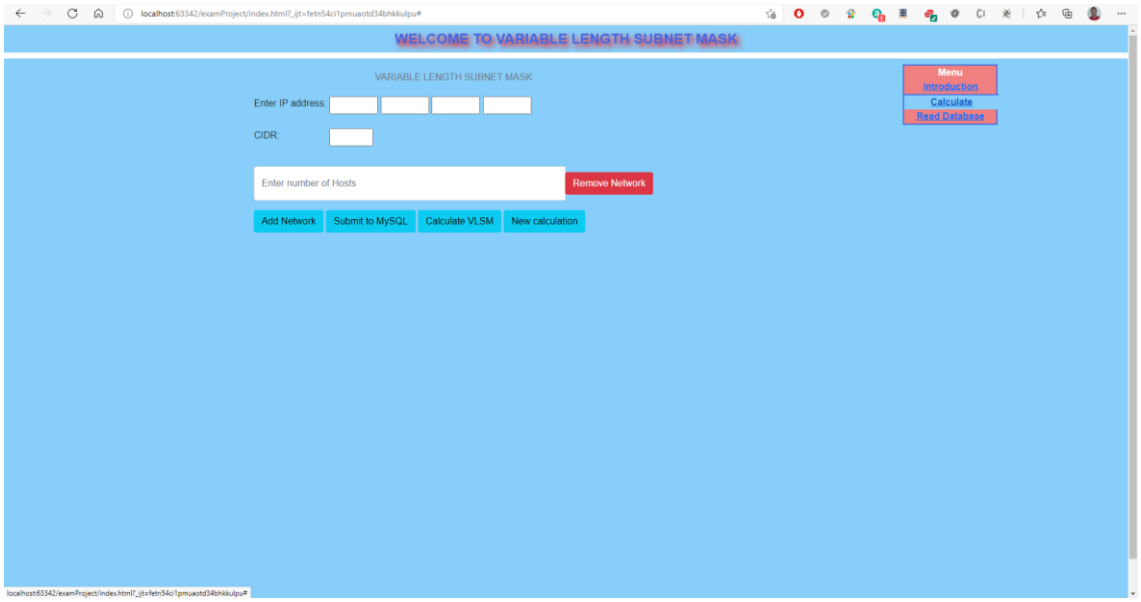
4. Documentation of the application – mini manual



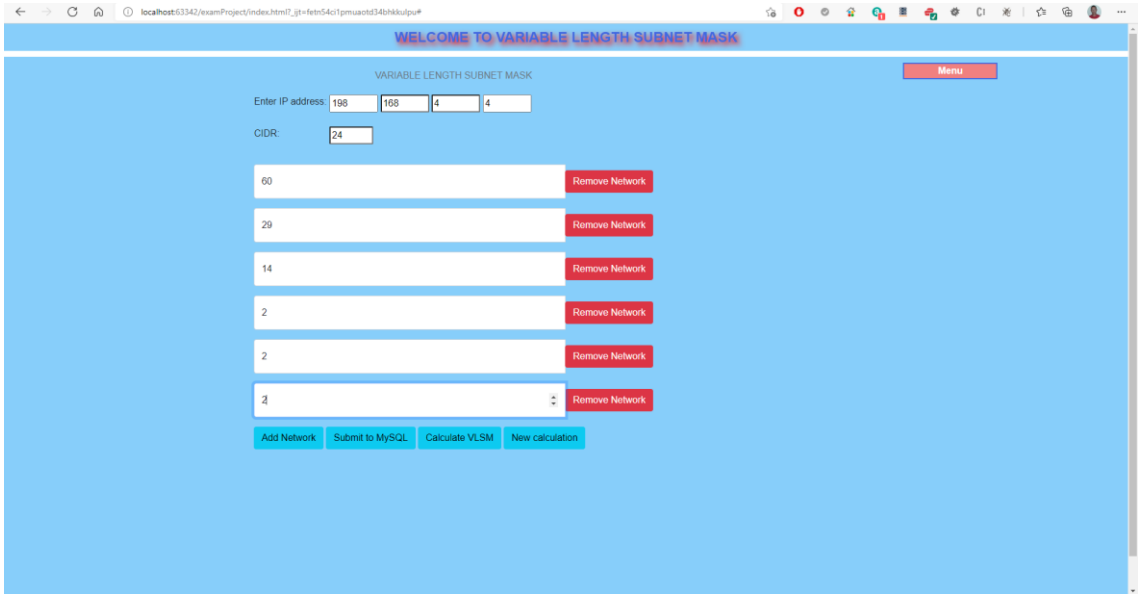
index.html 1



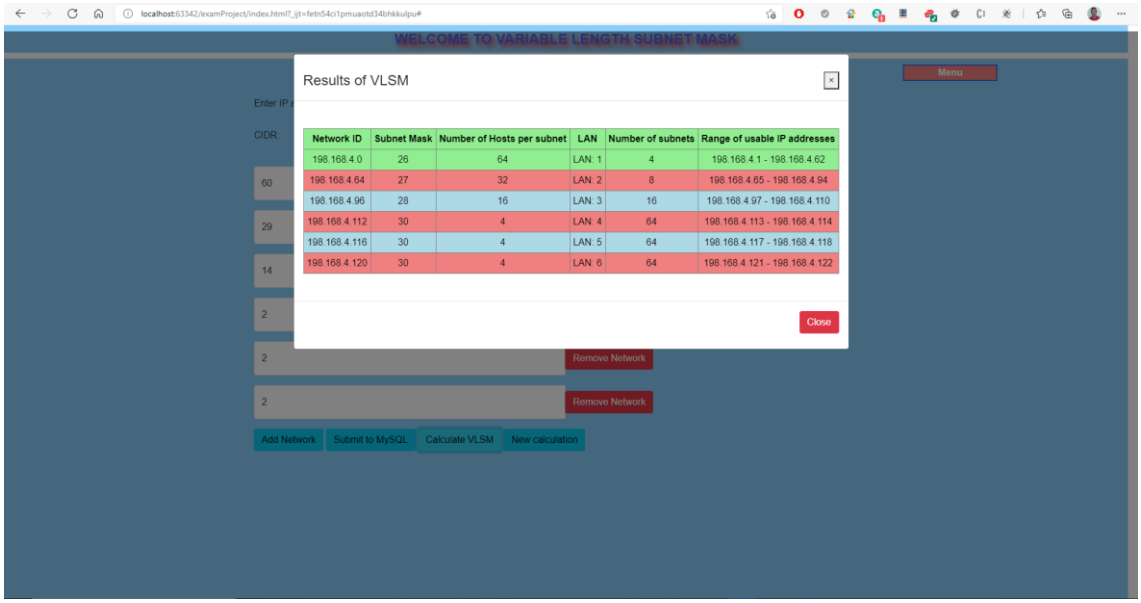
Introduction 1



vlsm.html 1

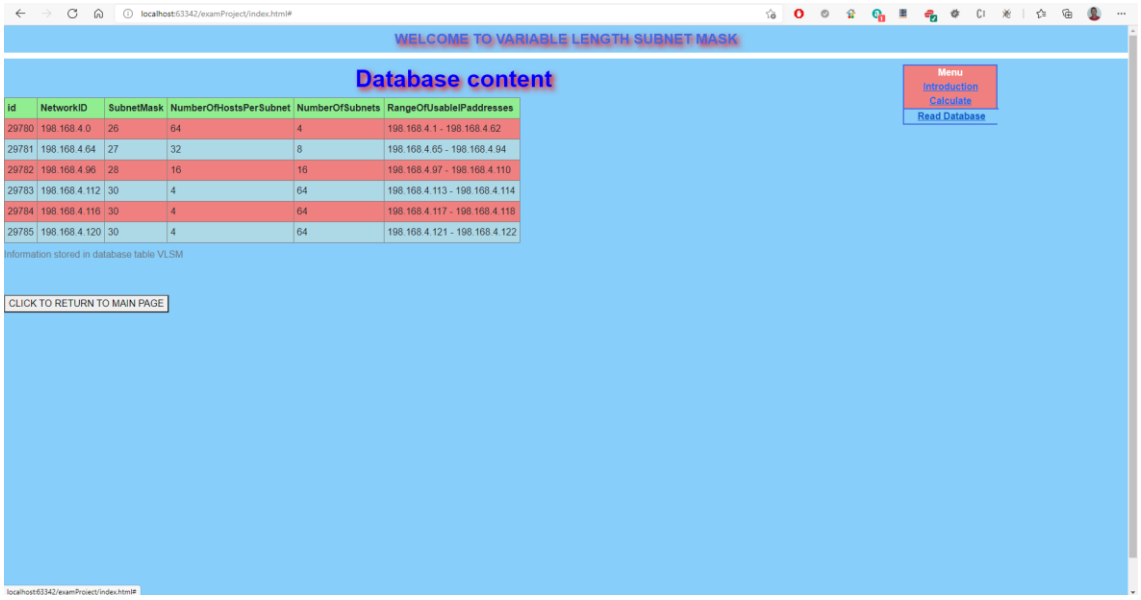


User input 1



Results of VLSM 1

Submit to MySQL 1



Database content 1

References

[1.] Variable Length Subnet Mask (VLSM) Tutorial

[2.] https://getbootstrap.com/docs/5.0/components/modal/

[3.] https://api.jquery.com/

AMAKIRI WELEKWE TECHNOLOGY ADVISOR | CYBERSECURITY EVANGELIST,
Variable Length Subnet Mask (VLSM) Tutorial - Fully Explained (comparitech.com)
UPDATED: November 12, 2020