19CSE301- COMPUTER NETWORK

Electronic Payment System Inside a campus

Registration No	Name
CB.EN.U4CSE19321	G Madhuri
CB.EN.U4CSE19323	G V S Phani Teja
CB.EN.U4CSE19336	M Pavani Bai
CB.EN.U4CSE19343	R Pavan Teja

Problem statement

To understand the working of networks in Electronic payment considering the growth of the E-Commerce industry. We try to understand how networking benefits users to go cashless and provides smooth and quick transactions at any time with easy, quick and secure payments in a campus area network.

Benefits of computer networks in E payment

- Convenience and accessibility
- Faster transaction
- Easy to keep track of transaction
- Provides Payment security

Why Networking is required for the application

1) Increase in usage of mobiles.
2) Major advantage of network is central management of applications and data

The network facilitates bulk credit and debit transactions

Software/Operating System used

Programming Languages Used:-

- Javascript - (Used for both Server and Client side)
- Java

Web Technologies used:-

- HTML5

Hardware/Devices used

Mobile Payment :-

- A full wireless credit card machine/POS system with mobile capabilities
- A card reader which relies on third party hardware (smart device) to process transactions

Networking devices used

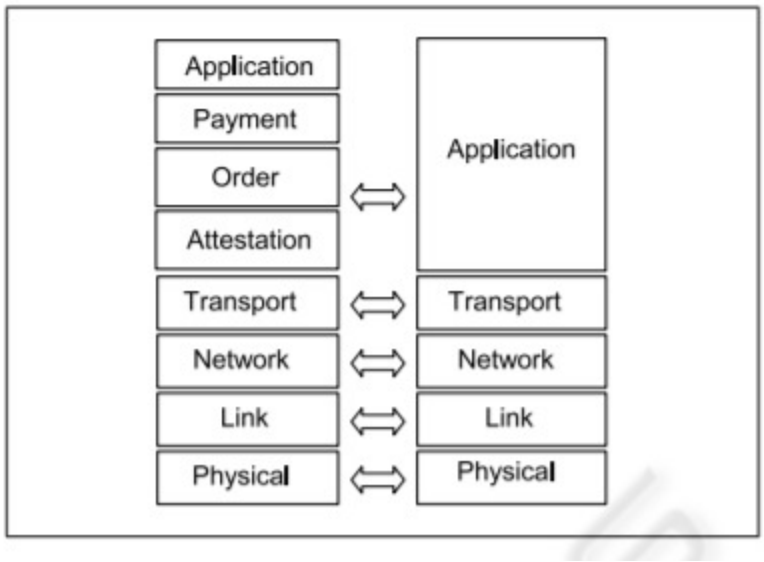
- payment Server
- FTP , DNS , WEB Server
- User device (Laptop,PC,smartphone)
- Router
- Switch

Performance parameters

Parameter	Meaning	Measured
Bandwidth	**Bandwidth** describes the maximum data transfer rate of a network or Internet connection. It measures how much data can be sent over a specific connection in a given amount of time.	It is measured in bits.It refers to the number of bits per second that a channel, a link, or rather a network can transmit.
Throughput	**Throughput** capacity indicates how many transactions a payment gateway can process per unit of time (usually 1 second).	Average rate is measured depending on bandwidth. It is measured in terms of bits transferred per second ***(bps)***.
Packet Loss	**Packet loss** occurs when one or more packets of data travelling across a computer network fail to reach their destination	

Transmission Time	**Transmission time** is the amount of time from the beginning until the end of a message transmission	The transmission time of a message relies upon the size of the message and bandwidth of the channel. ***Transmission time = Message size / Bandwidth***
Latency	**Latency is defined as the time required to successfully send a packet across a network.**	It measured in many ways like: round trip, one way, etc. ***Latency = Propagation Time + Transmission Time + Queuing Time + Processing Delay***
Propagation Time	**Propagation time** is the amount of time it takes one bit to go from the start of the link to its destination	It is measured in milliseconds***(ms)***. It is calculated as the ratio between the link length (distance) and the propagation speed over the communicating medium. ***Propagation time = Distance / Propagation speed***
Processing Delay	**Processing delay** is the time it takes routers to process the packet header. Processing delay is a key component in network delay.	
Queuing Delay	**Queuing delay** is the sum of the delays encountered by a packet between the time of insertion into the network and the time of delivery to the address.	
Jitter	**Jitter** is the variation in the time between data packets arriving, caused by network congestion, or route changes	It is measured in milliseconds***(ms)***.

Architecture diagram



E-payment layer vs Internet layer

Multiple Clients - Single Server TCP program for E-Payment:

The following multiple client - single server TCP code provides the working method of the E-Payment system.

The options provided to client in the code include:

- Check account balance
- Make a payment
- View transaction history

The server maintains two files Payment.txt which store the details of the user (phone number and account balance) and transactions.txt which has the list of transactions and their details.

Client CODE:

```
package case_study;
import java.io.*;
import java.net.*;
import java.util.Scanner;

//Client class
public class Client {
    public static void main(String[] args) throws IOException
    {
        try
        {

            Scanner scn = new Scanner(System.in);

            // getting localhost ip
            InetAddress ip = InetAddress.getByName("localhost");

            // establish the connection with server port 8080
            Socket s = new Socket(ip,8080 );

            // obtaining input and out streams
            DataInputStream dis = new DataInputStream(s.getInputStream());
            DataOutputStream dos = new DataOutputStream(s.getOutputStream());

            // the following loop performs the exchange of
            // information between client and client handler
            while (true)
            {
                System.out.println(dis.readUTF());
                String tosend = "";
                String details;
                details = scn.nextLine();
                if(!details.equals("Exit")) {
                if(details.equals("1")) {
                    tosend+=details+" ";
                    System.out.println("Enter your phone number");
```

```
                    details = scn.nextLine();
                    while(details.length()<10) {
                        System.out.println("Enter valid phone number:");
                        details = scn.nextLine();
                    }
                    tosend+=details;
                }
                if(details.equals("2")) {
                tosend+=details+" ";
                System.out.println("Enter your phone number:");
                details = scn.nextLine();
                while(details.length()<10) {
                    System.out.println("Enter valid phone number:");
                    details = scn.nextLine();
                }
                tosend+=details+" ";
                System.out.println("Enter your recipent phone number:");
                details = scn.nextLine();
                while((tosend.equals("2 "+details+" ")) || details.length()<10  )
{
                    System.out.println("Enter valid recipient phone number:");
                    details = scn.nextLine();
                }
                tosend+=details+" ";
                System.out.println("Enter money:");
                details = scn.nextLine();
                while(Integer.parseInt(details)<0) {
                    System.out.println("Enter amount:");
                    details = scn.nextLine();
                }
                tosend+=details+" ";
                System.out.println("Reason:");
                details = scn.nextLine();
                tosend+=details;
                }
                if(details.equals("3")) {
                    tosend+=details+" ";
                    System.out.println("Enter your phone number:");
                    details = scn.nextLine();
                    while(details.length()<10) {
                        System.out.println("Enter valid phone number:");
                        details = scn.nextLine();
                    }
                    tosend+=details;
```

```
                    System.out.println("----------------------------------------
\nTransaction History\nPhnum\tRec_Phnum\tMoney\tReason\n");
                }
                }
                else {
                    tosend=details;
                }
                dos.writeUTF(tosend);

                // If client sends exit,close this connection
                // and then break from the while loop
                if(tosend.equals("Exit"))
                {
                    System.out.println("Closing this connection : " + s);
                    s.close();
                    System.out.println("Connection closed");
                    break;
                }

                String received = dis.readUTF();
                System.out.println(received);
                System.out.println("----------------------------------------");
            }

            // closing resources
            scn.close();
            dis.close();
            dos.close();
        }catch(Exception e){
            e.printStackTrace();
        }
    }
}
```

Server CODE:

```
package case_study;

import java.io.*;

import java.net.*;
```

```java
// Server class
public class Server
{
    public static void main(String[] args) throws IOException
    {
        // server is listening on port 8080
        ServerSocket ss = new ServerSocket(8080);

        // running infinite loop for getting
        // client request
        while (true)
        {
            Socket s = null;

            try
            {
                // socket object to receive incoming client requests
                s = ss.accept();

                System.out.println("A new client is connected : " + s);

                // obtaining input and out streams
                DataInputStream dis = new DataInputStream(s.getInputStream());
                DataOutputStream dos = new DataOutputStream(s.getOutputStream());


                System.out.println("Assigning new thread for this client");

                // create a new thread object
                Thread t = new ClientHandler(s, dis, dos);

                // Invoking the start() method
                t.start();

            }
            catch (Exception e){
                ss.close();
                e.printStackTrace();
            }
        }
    }
}

// ClientHandler class
```

```
class ClientHandler extends Thread
{

    final DataInputStream dis;
    final DataOutputStream dos;
    final Socket s;


    // Constructor
    public ClientHandler(Socket s, DataInputStream dis, DataOutputStream dos)
    {
        this.s = s;
        this.dis = dis;
        this.dos = dos;
    }

    @Override
    public void run()
    {
        String received;
        String toreturn;
        while (true)
        {
            try {

                // MENU
                dos.writeUTF("E-
Payment Service\nMenu\n1.Check Your balance\n2.Make payment\n3.View Transaction H
istory\nSelect and option or .. Type Exit to terminate connection.");

                // receive the answer from client
                received = dis.readUTF();

                if(received.equals("Exit"))
                {
                    System.out.println("Client " + this.s + " sends exit...");
                    System.out.println("Closing this connection.");
                    this.s.close();
                    System.out.println("Connection closed");
                    break;
                }

//                System.out.println(received);
                String[] data = received.split(" ");
```

```
                File fileToBeModified = new File("C:\\Users\\PhaniTeja\\Desktop\\
Payment.txt");
                File Transactionsfile = new File("C:\\Users\\PhaniTeja\\Desktop\\
transaction.txt");
                String contents = "";
                BufferedReader reader = null;
                FileWriter writer = null;

                if(data[0].equals("1")) {
                    reader = new BufferedReader(new FileReader(fileToBeModified))
;
                    String line = reader.readLine();

                    while (line != null)
                    {
                        String[] filecont = line.split(" ");
                        if(filecont[0].equals(data[1])) {
                            contents = "Your Balance is: "+filecont[1];
                            break;
                        }
                        line = reader.readLine();
                    }
                    if(contents.length()==0) {
                        contents= "Invalid Phone number";
                    }
                    toreturn =  contents;
                    dos.writeUTF(toreturn);

                }
                if(data[0].equals("2")) {
//                  System.out.println("2 selected");
                    reader = new BufferedReader(new FileReader(fileToBeModified))
;
                    String line = reader.readLine();
                    int flag=0;
                    while (line != null)
                    {
                        String[] filecont = line.split(" ");
                        if(filecont[0].equals(data[1]) && Integer.parseInt(fileco
nt[1])>=Integer.parseInt(data[3])) {
                            int bal = Integer.parseInt(filecont[1]);
                            bal = bal-Integer.parseInt(data[3]);
                            contents = contents + filecont[0]+" "+String.valueOf(
bal) + System.lineSeparator();
                            flag++;
```

```
                        }
                        else if(filecont[0].equals(data[2])) {
                            int bal = Integer.parseInt(filecont[1]);
                            bal = bal+Integer.parseInt(data[3]);
                            contents = contents + filecont[0]+" "+String.valueOf(
bal) + System.lineSeparator();
                            flag++;
                        }
                        else {
                            contents = contents + line + System.lineSeparator();
                        }
                        line = reader.readLine();
                    }
                    if(flag==2) {
                        writer = new FileWriter(fileToBeModified);
                        writer.write(contents);
                        writer.close();
//                        System.out.println(contents);
                        contents="";
                        reader = new BufferedReader(new FileReader(Transactionsfi
le));
                        line = reader.readLine();
                        while (line != null)
                        {
                            contents = contents + line + System.lineSeparator();

                            line = reader.readLine();
                        }
                        String temp="";

                        for(int i=1;i<data.length;++i) {
                            temp=temp+data[i]+" ";
                        }
                        contents = contents+ temp +System.lineSeparator();
                        writer = new FileWriter(Transactionsfile);
                        writer.write(contents);
                        writer.close();
                        dos.writeUTF("Payment succesful!");

                        //

                    }
                    else {
                        dos.writeUTF("Payment Not succesful...");
                    }
```

```
                }

                if(data[0].equals("3")) {
                    reader = new BufferedReader(new FileReader(Transactionsfile))
;
                    contents="";
                    String line = reader.readLine();
                    while (line != null)
                    {

                        String[] t = line.split(" ");
                        if(t[0].equals(data[1])) {
                        contents = contents + line + System.lineSeparator();
                        }
                        line = reader.readLine();
                    }

                    if(contents.length()==0) {
                        dos.writeUTF("Oops your Transaction History seems to be e
mpty!!!....");

                    }
                    else {
                        dos.writeUTF(contents);
                    }

            }
            }catch (IOException e) {
                e.printStackTrace();
            }
        }

        try
        {
            // closing resources
            this.dis.close();
            this.dos.close();

        }catch(IOException e){
            e.printStackTrace();
        }
}
}
```

OUTPUT :

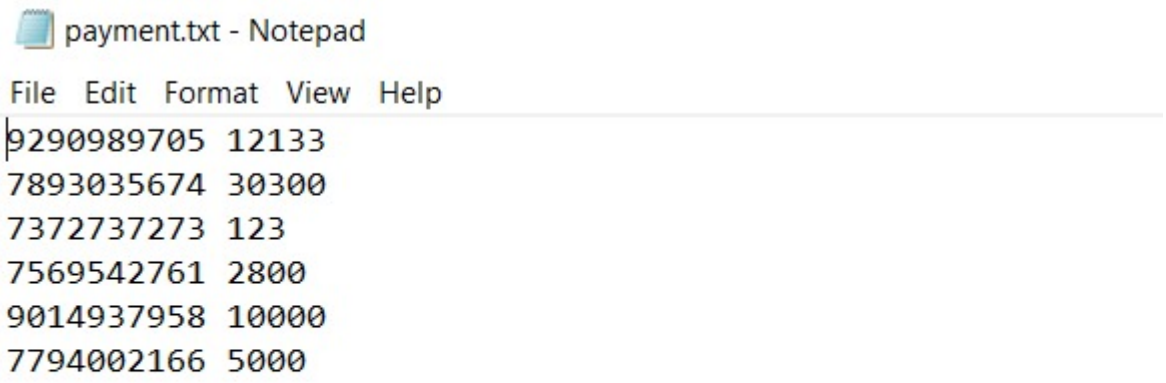
Client side:

Client

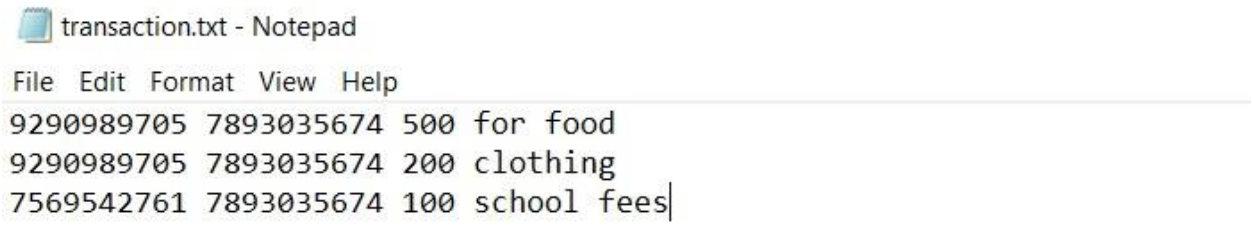
```
Client [Java Application] C:\Program Files\Java\jre1.8.0_152\bin\javaw.exe  (23 Nov, 2
E-Payment Service
Menu
1.Check Your balance
2.Make payment
3.View Transaction History
Select and option or .. Type Exit to terminate connection.
1
Enter your phone number
7569542761
Your Balance is: 2900
----------------------------------------
E-Payment Service
Menu
1.Check Your balance
2.Make payment
3.View Transaction History
Select and option or .. Type Exit to terminate connection.
2
Enter your phone number:
7569542761
Enter your recipent phone number:
7893035674
Enter money:
100
Reason:
school fees
Payment succesful!
----------------------------------------
E-Payment Service
Menu
1.Check Your balance
2.Make payment
3.View Transaction History
Select and option or .. Type Exit to terminate connection.
3
Enter your phone number:
7569542761
----------------------------------------
Transaction History
Phnum   Rec_Phnum       Money   Reason

7569542761 7893035674 100 school fees
```

Files:



Cisco Packet Tracer:-



Application Protocols used:

1. SMTP:

- Transfers the mail from users to server.
- Transfers the mail from Server to users.

2. DNS:

- To Access the website both from users side and employees side.
- Link: amrita.edu

3. FTP:

- payments.txt contains the users account status such balance and active mobile number.
- Transfer of the file from Server to users.

Routing Protocols:

Routing Information Protocol (RIP) is a dynamic routing protocol that uses hop count as a routing metric to find the best path between the source and the destination network. It is a distance-vector routing protocol that has an AD value of 120 and works on the application layer of the OSI model.

Features of RIP :

1. Updates of the network are exchanged periodically.

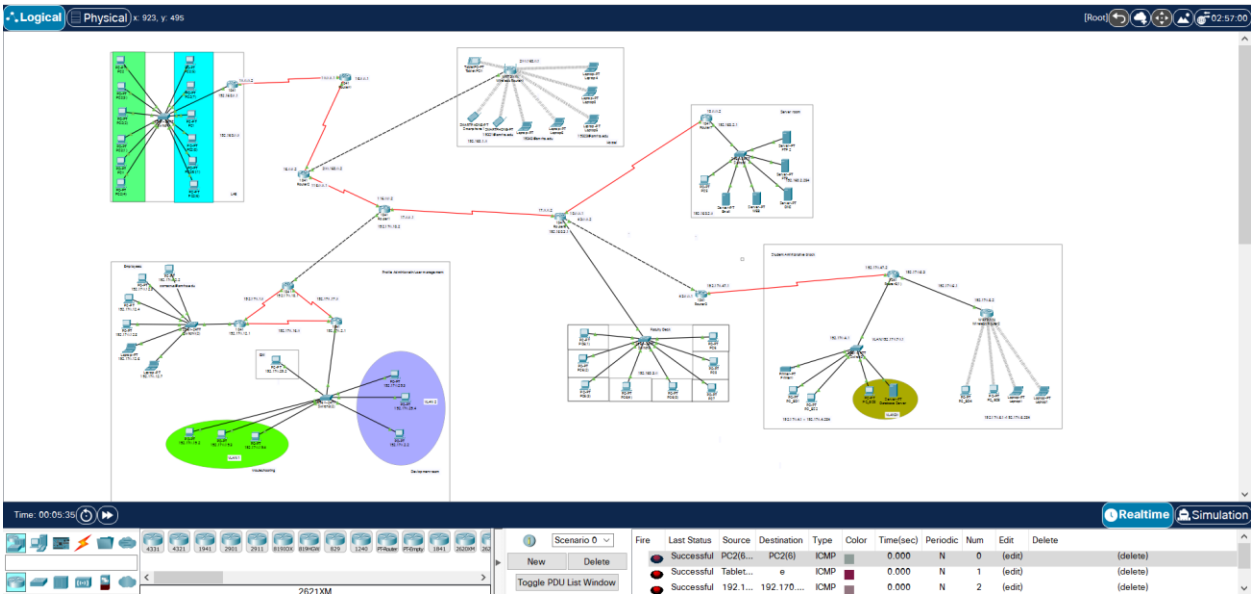
2. Updates (routing information) are always broadcast.

3. Full routing tables are sent in updates.

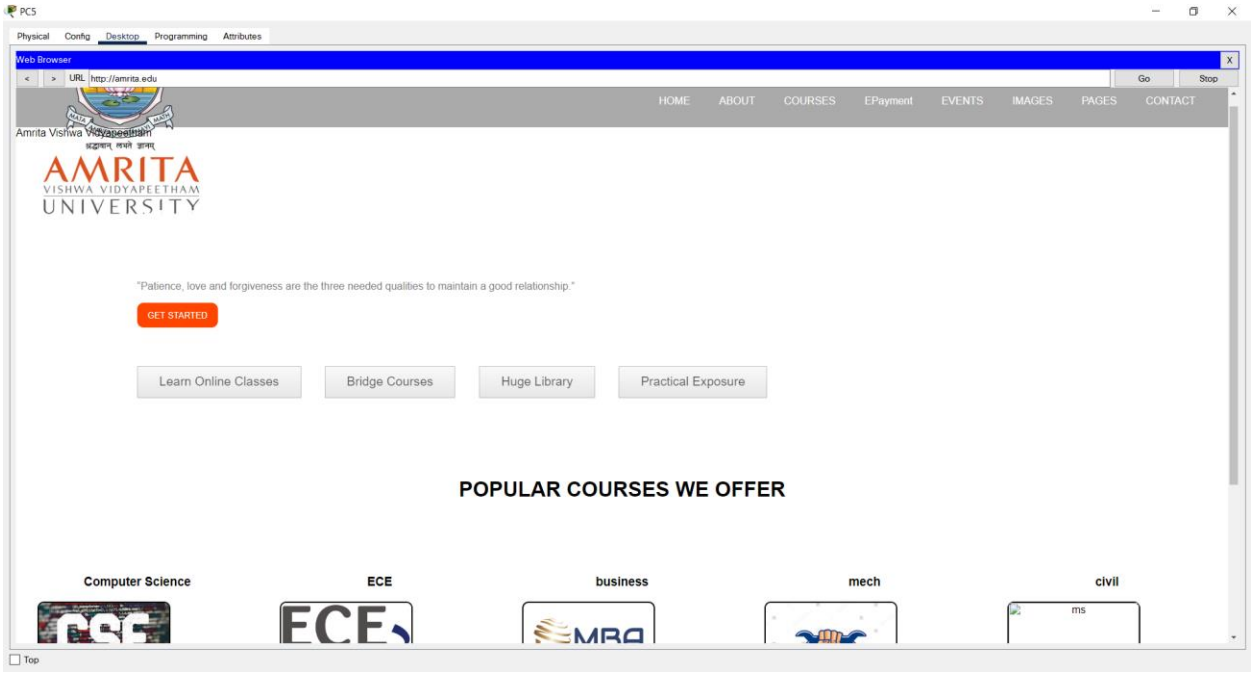
4. Routers always trust routing information received from neighbor routers. This is also known as Routing on rumours.

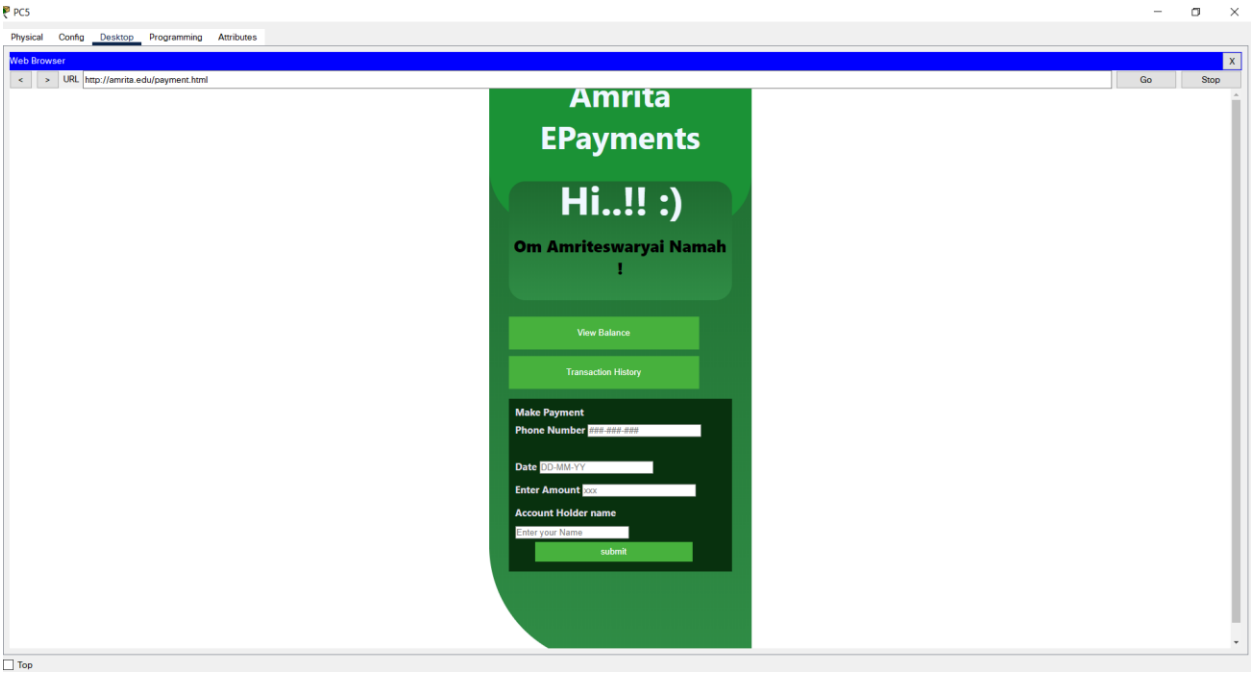
Used VLAN for restricting access to profile administration.

Overall configurations:

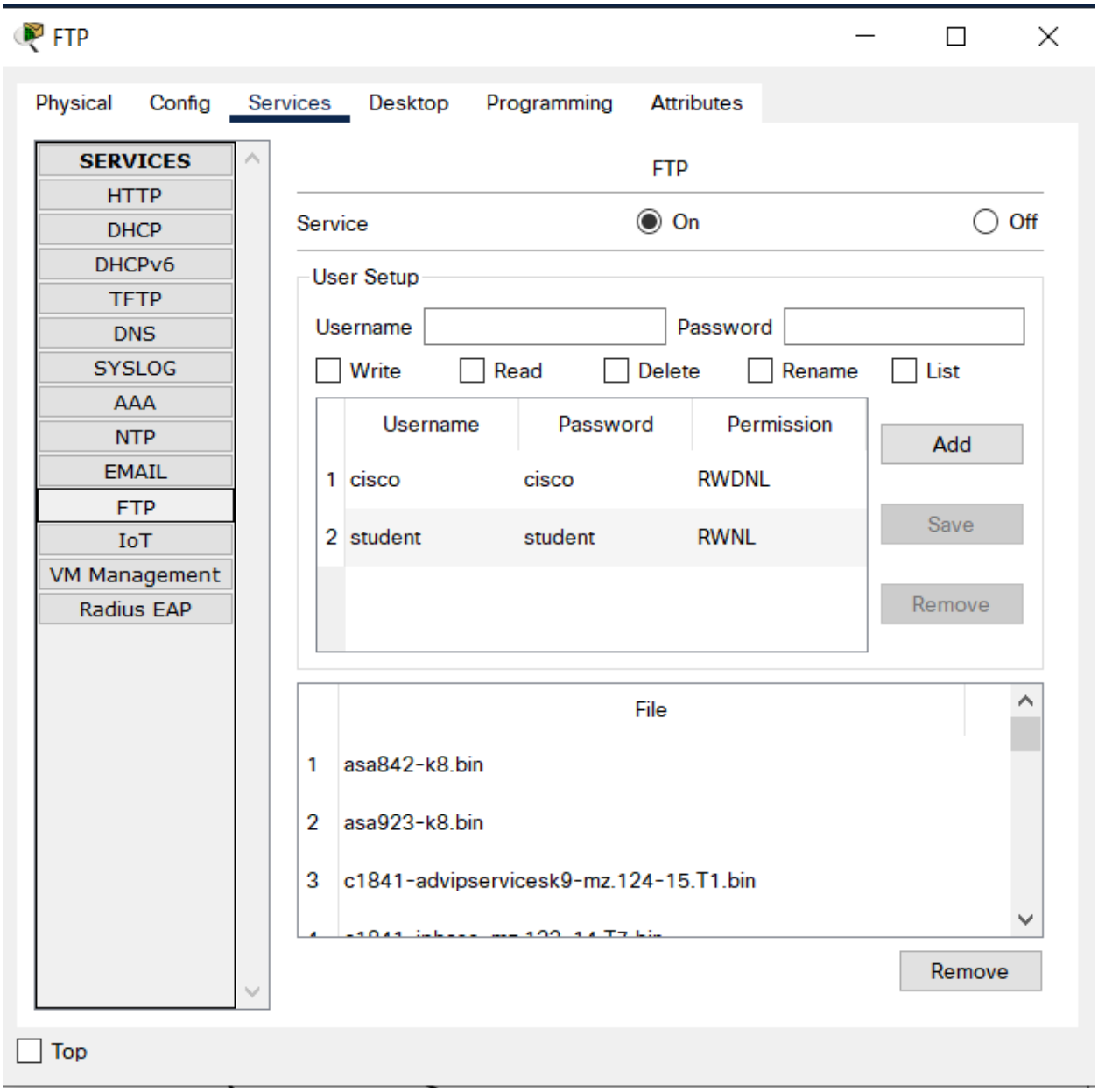


Website output (DNS):

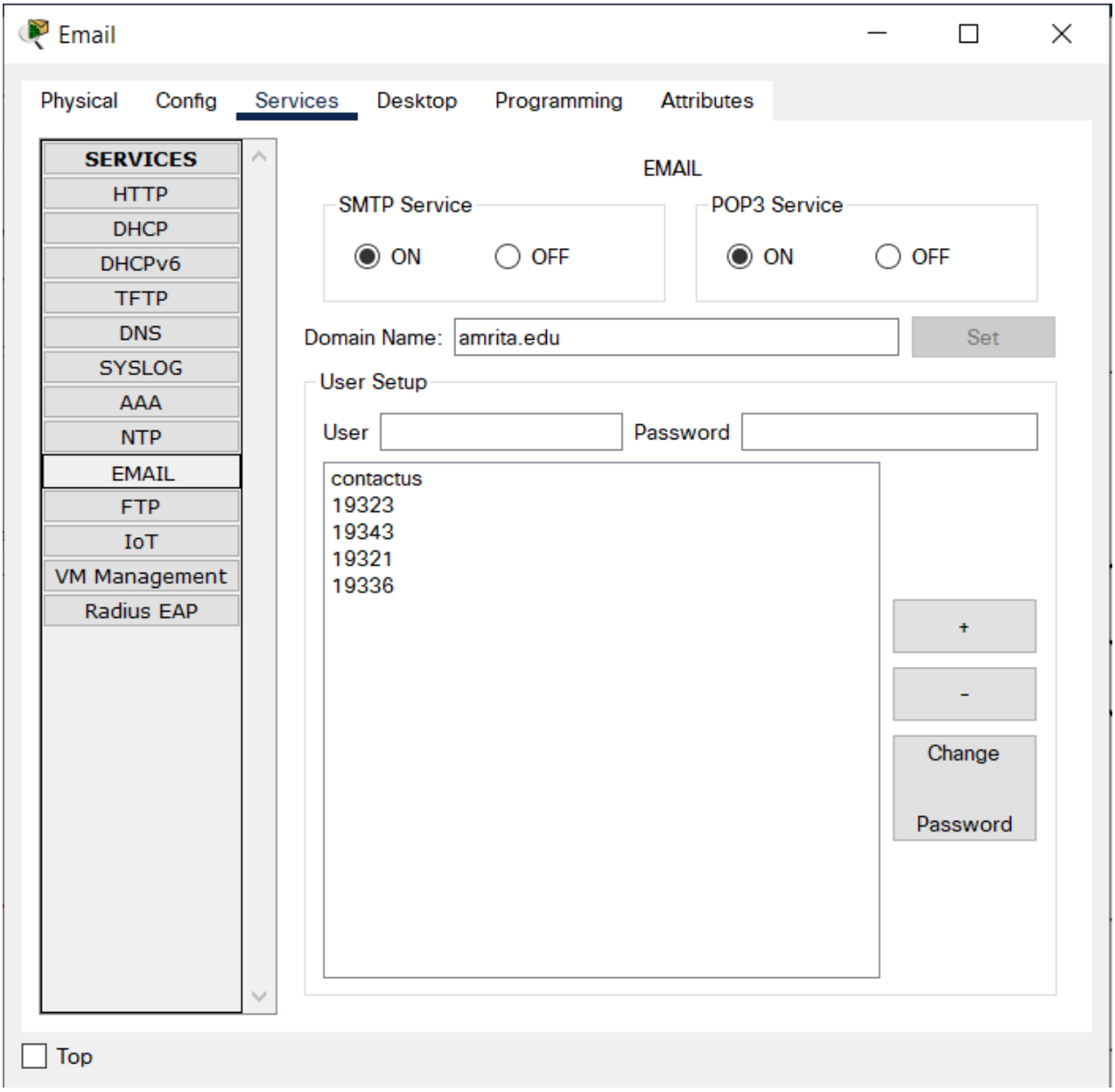




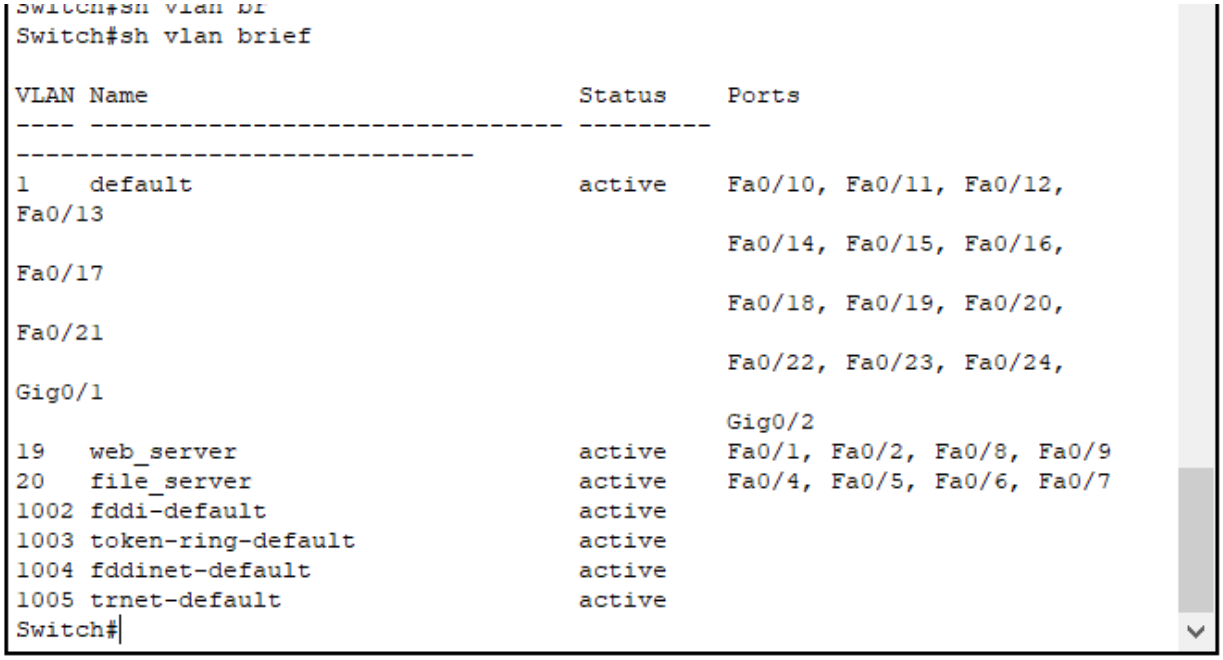
FTP:



SMTP EMail:


Email
Physical
Config
Services
Desktop
Programming
Attributes
SERVICES
HTTP
DHCP
DHCPv6
TFTP
DNS
SYSLOG
AAA
NTP
EMAIL
FTP
IoT
VM Management
Radius EAP
EMAIL
SMTP Service
ON
OFF
POP3 Service
ON
OFF
Domain Name:
amrita.edu
Set
User Setup
User
Password
contactus
19323
19343
19321
19336
+
-
Change
Password
Top

VLAN:



Ctrl+F6 to exit CLI focus

Copy Paste