Name: Gaurav Kumar Singh

Registration Number: 19BCE2119

Course: CSE3501-ELA-L21+L22 (ISAA)

Lab FAT

Question set: 2a

Develop a Machine Learning based Malware detection system using any of the following Logistic Regression [30]

b. Using Cisco Packet Tracer configure LAN that acts as a connector between two computers to enable message transfer. [20]

1)

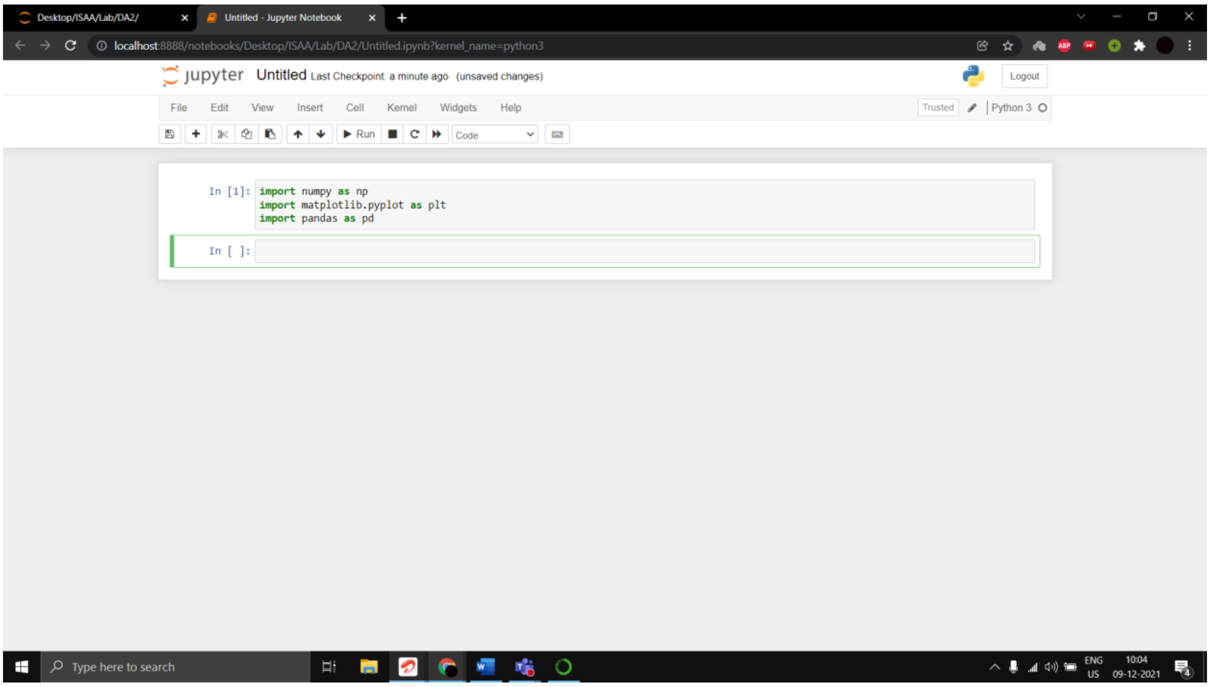
Logistic Regression (Full Code)

Importing python libraries such as NumPy, matplotlib and pandas for data reading and manipulation.

import numpy as np

import matplotlib.pyplot as plt

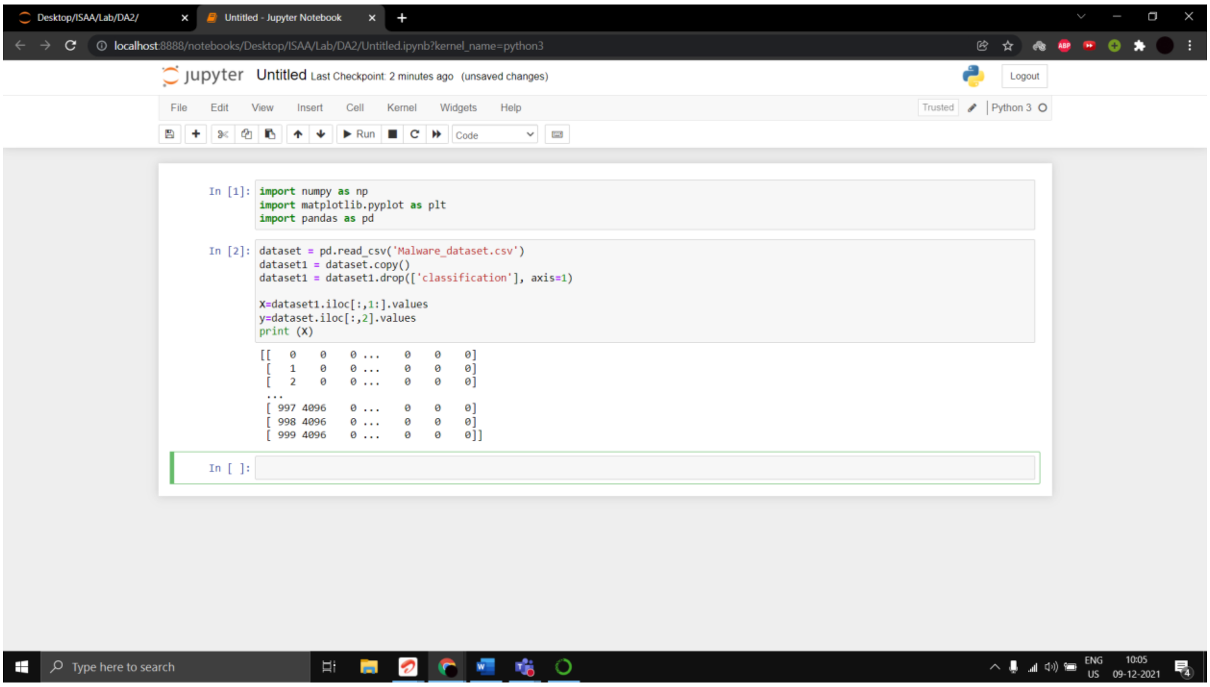
import pandas as pd


Desktop/ISAA/Lab/DA2/
Untitled - Jupyter Notebook
localhost:8888/notebooks/Desktop/ISAA/Lab/DA2/Untitled.ipynb?kernel_name=python3
jupyter Untitled Last Checkpoint: a minute ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Type here to search
ENG US
10:04
09-12-2021

```
In [1]: import numpy as np
        import matplotlib.pyplot as plt
        import pandas as pd

In [ ]:
```

Reading dataset and truncating irrelevant columns

```
dataset = pd.read_csv('Malware_dataset.csv')

dataset1 = dataset.copy()

dataset1 = dataset1.drop(['classification'], axis=1)

X=dataset1.iloc[:,1:].values

y=dataset.iloc[:,2].values

print (X)
```

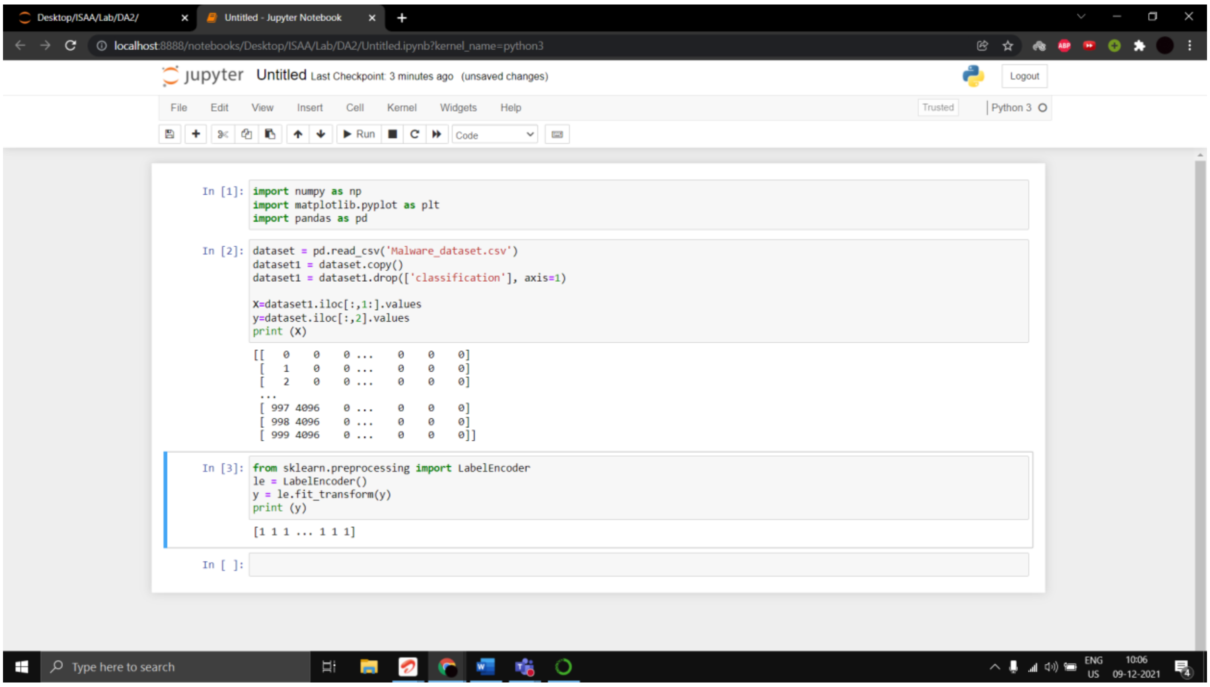


Desktop/ISAA/Lab/DA2/
Untitled - Jupyter Notebook
localhost:8888/notebooks/Desktop/ISAA/Lab/DA2/Untitled.ipynb?kernel_name=python3
jupyter
Untitled Last Checkpoint: 2 minutes ago (unsaved changes)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3
Run
Code
Type here to search
ENG
US
10:05
09-12-2021

```
In [1]: import numpy as np
        import matplotlib.pyplot as plt
        import pandas as pd

In [2]: dataset = pd.read_csv('Malware_dataset.csv')
        dataset1 = dataset.copy()
        dataset1 = dataset1.drop(['classification'], axis=1)

        X=dataset1.iloc[:,1:].values
        y=dataset.iloc[:,2].values
        print (X)

        [[   0    0    0 ...    0    0    0]
         [   1    0    0 ...    0    0    0]
         [   2    0    0 ...    0    0    0]
         ...
         [ 997 4096    0 ...    0    0    0]
         [ 998 4096    0 ...    0    0    0]
         [ 999 4096    0 ...    0    0    0]]

In [ ]:
```

Importing sklearn.preprocessing module to transform classification dataset using label encoder for classification.

```
from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()

y = le.fit_transform(y)

print (y)
```



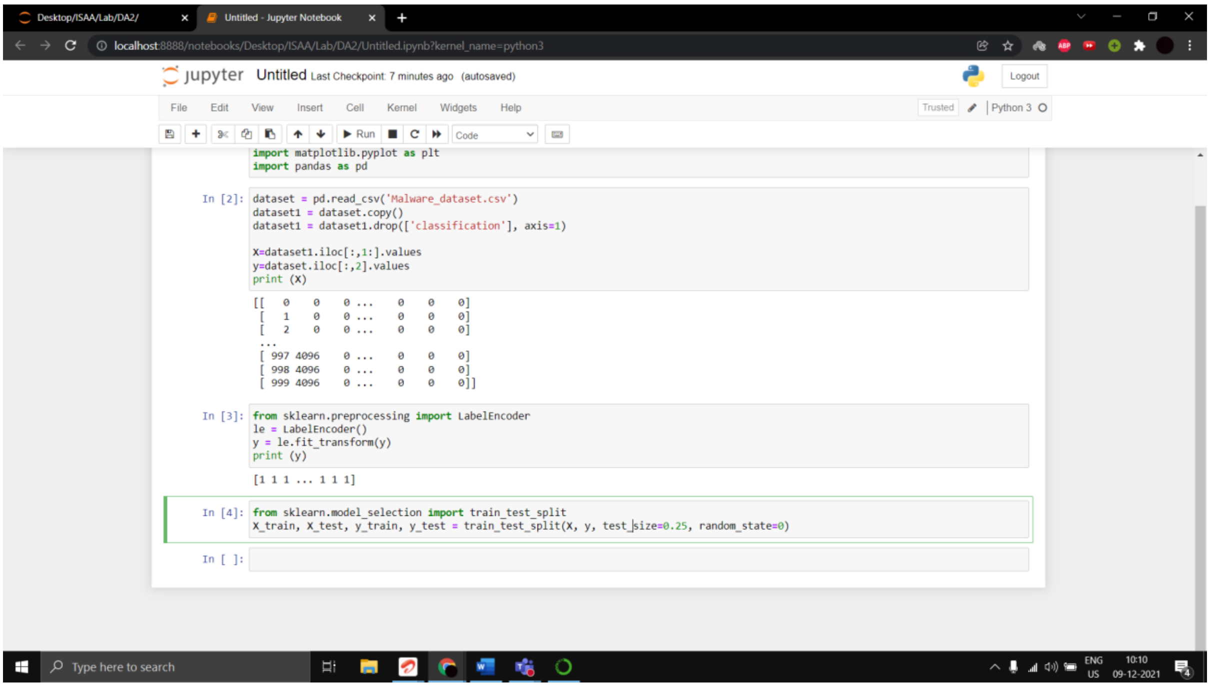
Desktop/ISAA/Lab/DA2/
Untitled - Jupyter Notebook
localhost:8888/notebooks/Desktop/ISAA/Lab/DA2/Untitled.ipynb?kernel_name=python3
jupyter
Untitled Last Checkpoint: 3 minutes ago (unsaved changes)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3
Run
Code
Type here to search
ENG
US
10:06
09-12-2021

```
In [1]: import numpy as np
        import matplotlib.pyplot as plt
        import pandas as pd

In [2]: dataset = pd.read_csv('Malware_dataset.csv')
        dataset1 = dataset.copy()
        dataset1 = dataset1.drop(['classification'], axis=1)

        X=dataset1.iloc[:,1:].values
        y=dataset.iloc[:,2].values
        print (X)

        [[   0    0    0 ...    0    0    0]
         [   1    0    0 ...    0    0    0]
         [   2    0    0 ...    0    0    0]
         ...
         [ 997 4096    0 ...    0    0    0]
         [ 998 4096    0 ...    0    0    0]
         [ 999 4096    0 ...    0    0    0]]

In [3]: from sklearn.preprocessing import LabelEncoder
        le = LabelEncoder()
        y = le.fit_transform(y)
        print (y)

        [1 1 1 ... 1 1 1]

In [ ]:
```

Splitting dataset into training and testing dataset to calculate accuracy precision and recall on. Test size: Train size:: 1: 3

from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=0)


Desktop/ISAA/Lab/DA2/
Untitled - Jupyter Notebook
localhost:8888/notebooks/Desktop/ISAA/Lab/DA2/Untitled.ipynb?kernel_name=python3
jupyter
Untitled Last Checkpoint: 7 minutes ago (autosaved)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3
Run
Code
Type here to search
ENG
US
10:10
09-12-2021

```
import matplotlib.pyplot as plt
import pandas as pd
```

```
In [2]: dataset = pd.read_csv('Malware_dataset.csv')
        dataset1 = dataset.copy()
        dataset1 = dataset1.drop(['classification'], axis=1)

        X=dataset1.iloc[:,1:].values
        y=dataset.iloc[:,2].values
        print (X)

        [[   0    0    0 ...    0    0    0]
         [   1    0    0 ...    0    0    0]
         [   2    0    0 ...    0    0    0]
         ...
         [ 997 4096    0 ...    0    0    0]
         [ 998 4096    0 ...    0    0    0]
         [ 999 4096    0 ...    0    0    0]]
```

```
In [3]: from sklearn.preprocessing import LabelEncoder
        le = LabelEncoder()
        y = le.fit_transform(y)
        print (y)

        [1 1 1 ... 1 1 1]
```

```
In [4]: from sklearn.model_selection import train_test_split
        X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=0)
```

```
In [ ]:
```

Printing the training and testing datasets

print("Xtrain")

print(X_train)

print("Xtest")

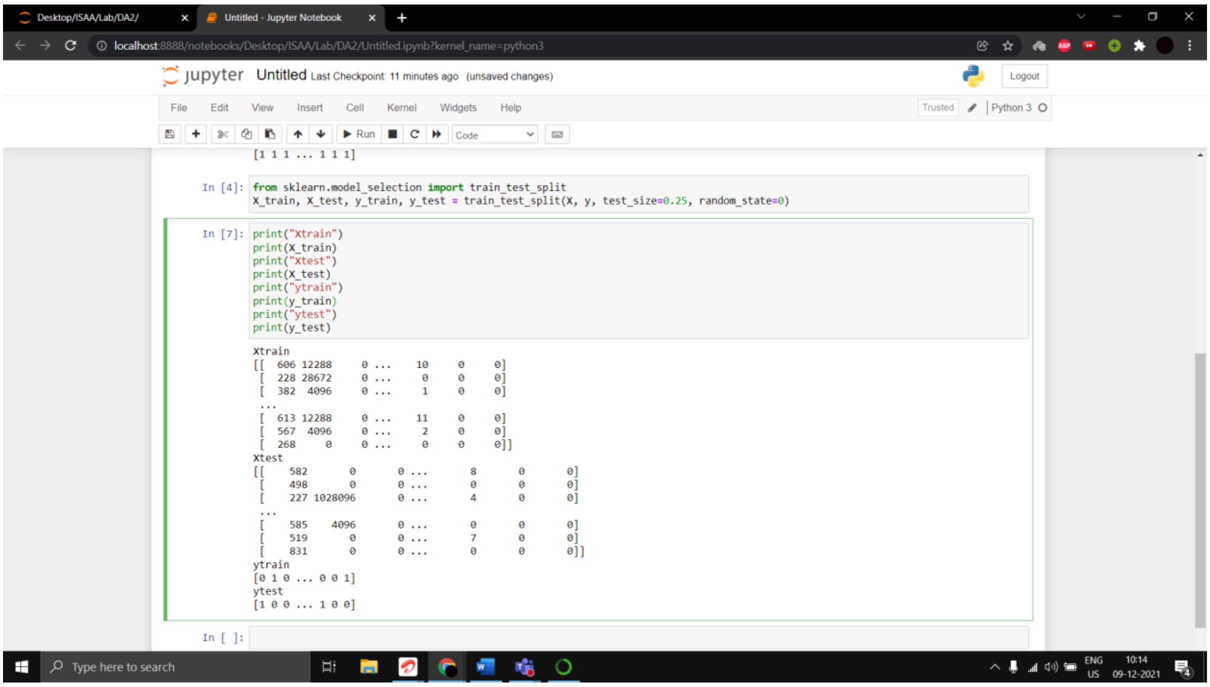
print(X_test)

print("ytrain")

print(y_train)

print("ytest")

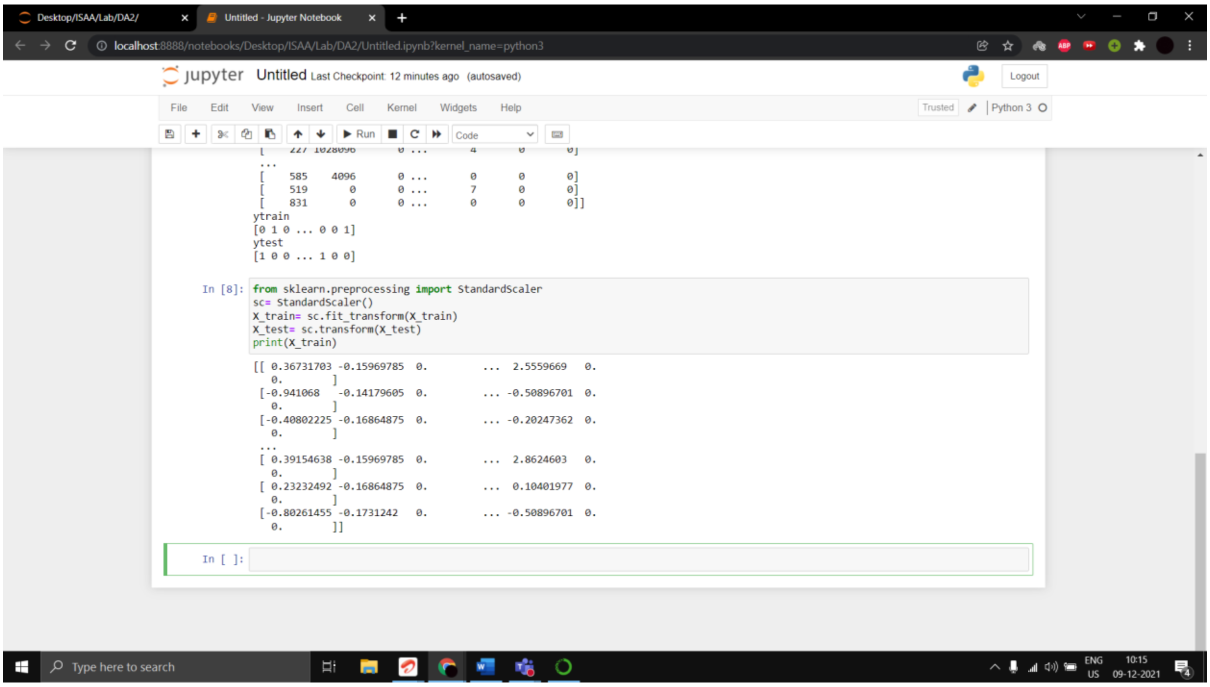
print(y_test)



```
[1 1 1 ... 1 1 1]

In [4]: from sklearn.model_selection import train_test_split
        X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=0)

In [7]: print("Xtrain")
        print(X_train)
        print("Xtest")
        print(X_test)
        print("ytrain")
        print(y_train)
        print("ytest")
        print(y_test)

        Xtrain
        [[  606 12288     0 ...    10     0     0]
         [  228 28672     0 ...     0     0     0]
         [  382  4096     0 ...     1     0     0]
         ...
         [  613 12288     0 ...    11     0     0]
         [  567  4096     0 ...     2     0     0]
         [  268     0     0 ...     0     0     0]]
        Xtest
        [[    582       0       0 ...       8       0       0]
         [    498       0       0 ...       0       0       0]
         [    227 1028096       0 ...       4       0       0]
         ...
         [    585    4096       0 ...       0       0       0]
         [    519       0       0 ...       7       0       0]
         [    831       0       0 ...       0       0       0]]
        ytrain
        [0 1 0 ... 0 0 1]
        ytest
        [1 0 0 ... 1 0 0]

In [ ]:
```

Scaler transform of the training dataset

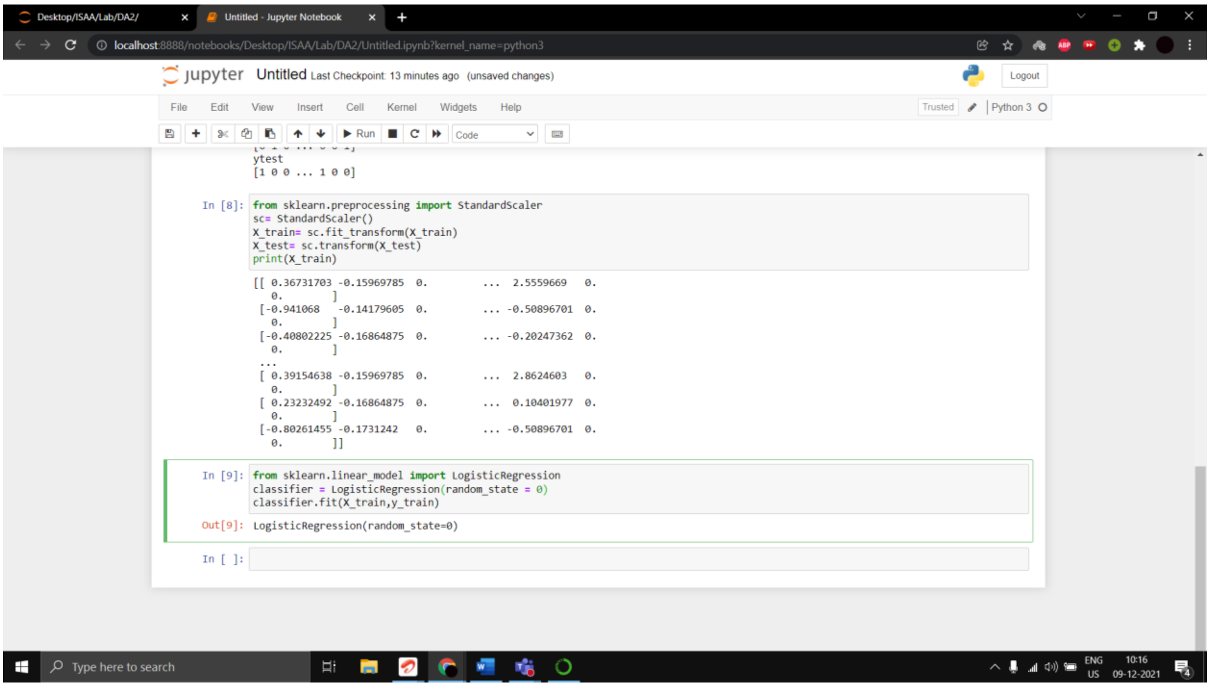
```
from sklearn.preprocessing import StandardScaler
sc= StandardScaler()
X_train= sc.fit_transform(X_train)
X_test= sc.transform(X_test)
print(X_train)
```



```
         [    227 1028096       0 ...       4       0       0]
         ...
         [    585    4096       0 ...       0       0       0]
         [    519       0       0 ...       7       0       0]
         [    831       0       0 ...       0       0       0]]
        ytrain
        [0 1 0 ... 0 0 1]
        ytest
        [1 0 0 ... 1 0 0]

In [8]: from sklearn.preprocessing import StandardScaler
        sc= StandardScaler()
        X_train= sc.fit_transform(X_train)
        X_test= sc.transform(X_test)
        print(X_train)

        [[ 0.36731703 -0.15969785  0.         ...  2.5559669   0.
           0.        ]
         [-0.941068   -0.14179605  0.         ... -0.50896701  0.
           0.        ]
         [-0.40802225 -0.16864875  0.         ... -0.20247362  0.
           0.        ]
         ...
         [ 0.39154638 -0.15969785  0.         ...  2.8624603   0.
           0.        ]
         [ 0.23232492 -0.16864875  0.         ...  0.10401977  0.
           0.        ]
         [-0.80261455 -0.1731242   0.         ... -0.50896701  0.
           0.        ]]

In [ ]:
```

Training the algorithm using Logistic regression

```
from sklearn.linear_model import LogisticRegression

classifier = LogisticRegression(random_state = 0)

classifier.fit(X_train,y_train)
```

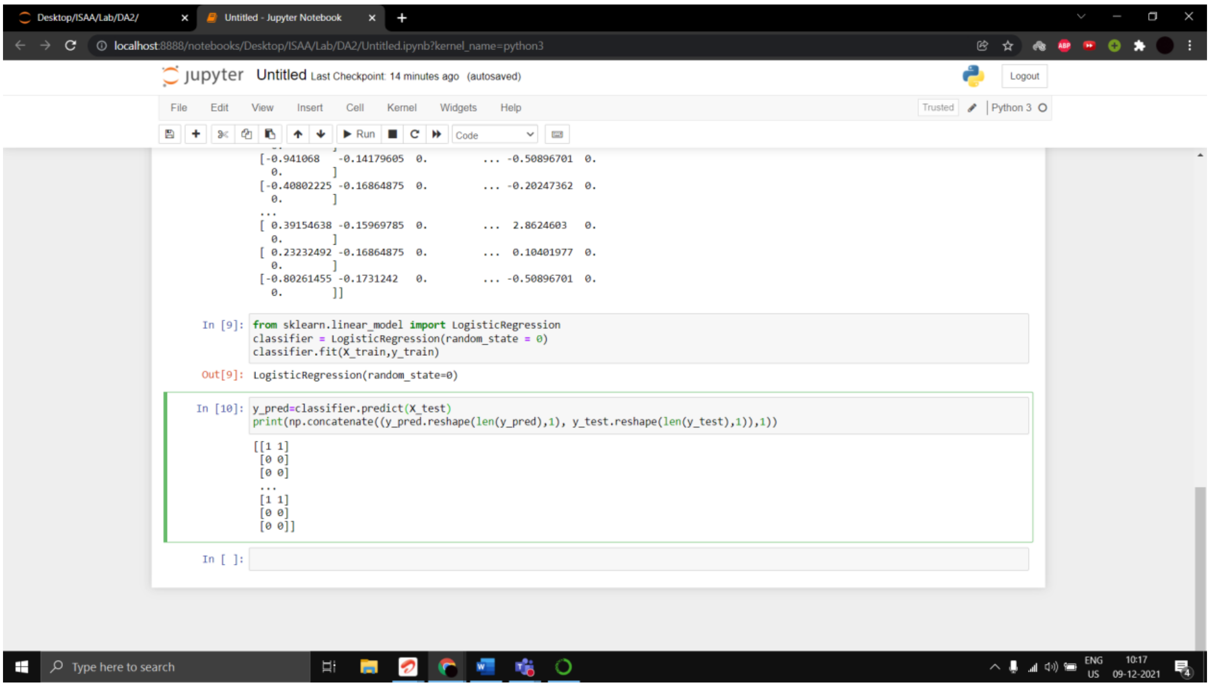


Desktop/ISAA/Lab/DA2/
Untitled - Jupyter Notebook
localhost:8888/notebooks/Desktop/ISAA/Lab/DA2/Untitled.ipynb?kernel_name=python3
jupyter Untitled Last Checkpoint: 13 minutes ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Type here to search
ENG US 10:16 09-12-2021

```
ytest
[1 0 0 ... 1 0 0]

In [8]: from sklearn.preprocessing import StandardScaler
        sc= StandardScaler()
        X_train= sc.fit_transform(X_train)
        X_test= sc.transform(X_test)
        print(X_train)

        [[ 0.36731703 -0.15969785  0.         ...  2.5559669   0.
           0.        ]
         [-0.941068   -0.14179605  0.         ... -0.50896701  0.
           0.        ]
         [-0.40802225 -0.16864875  0.         ... -0.20247362  0.
           0.        ]
         ...
         [ 0.39154638 -0.15969785  0.         ...  2.8624603   0.
           0.        ]
         [ 0.23232492 -0.16864875  0.         ...  0.10401977  0.
           0.        ]
         [-0.80261455 -0.1731242   0.         ... -0.50896701  0.
           0.        ]]

In [9]: from sklearn.linear_model import LogisticRegression
        classifier = LogisticRegression(random_state = 0)
        classifier.fit(X_train,y_train)

Out[9]: LogisticRegression(random_state=0)

In [ ]:
```

Testing the algorithm on the test split dataset

```
y_pred=classifier.predict(X_test)

print(np.concatenate((y_pred.reshape(len(y_pred),1), y_test.reshape(len(y_test),1)),1))
```



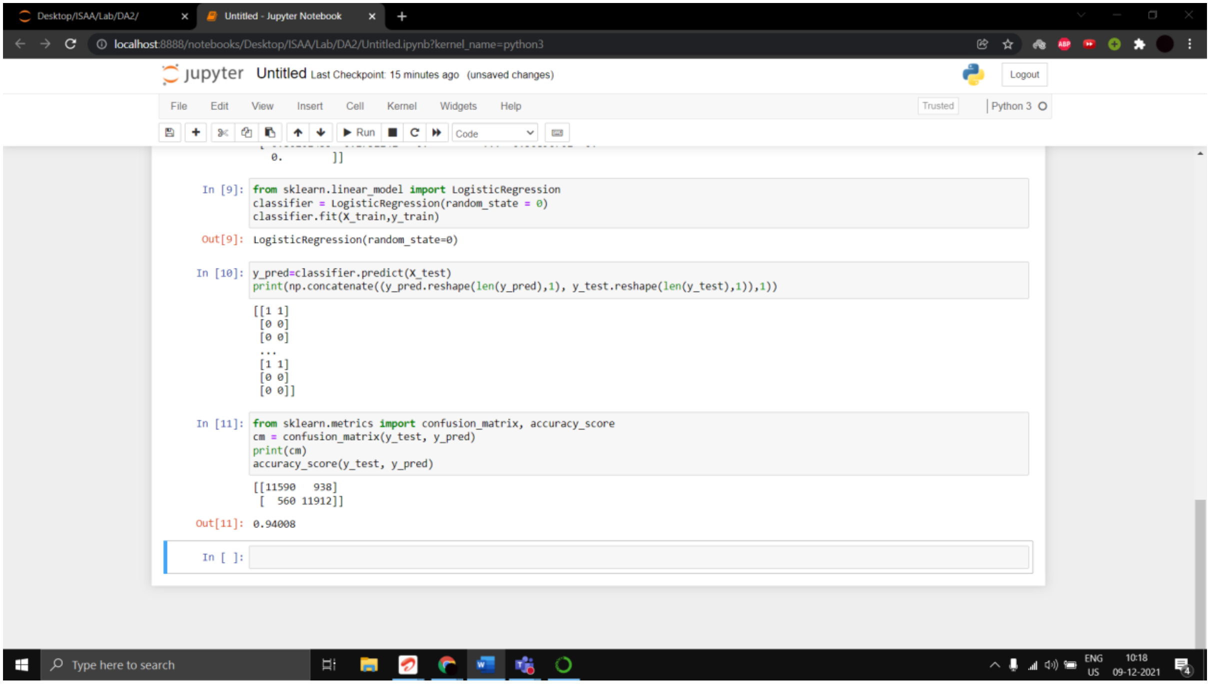
Desktop/ISAA/Lab/DA2/
Untitled - Jupyter Notebook
localhost:8888/notebooks/Desktop/ISAA/Lab/DA2/Untitled.ipynb?kernel_name=python3
jupyter Untitled Last Checkpoint: 14 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Type here to search
ENG US 10:17 09-12-2021

```
         [-0.941068   -0.14179605  0.         ... -0.50896701  0.
           0.        ]
         [-0.40802225 -0.16864875  0.         ... -0.20247362  0.
           0.        ]
         ...
         [ 0.39154638 -0.15969785  0.         ...  2.8624603   0.
           0.        ]
         [ 0.23232492 -0.16864875  0.         ...  0.10401977  0.
           0.        ]
         [-0.80261455 -0.1731242   0.         ... -0.50896701  0.
           0.        ]]

In [9]: from sklearn.linear_model import LogisticRegression
        classifier = LogisticRegression(random_state = 0)
        classifier.fit(X_train,y_train)

Out[9]: LogisticRegression(random_state=0)

In [10]: y_pred=classifier.predict(X_test)
         print(np.concatenate((y_pred.reshape(len(y_pred),1), y_test.reshape(len(y_test),1)),1))

         [[1 1]
          [0 0]
          [0 0]
          ...
          [1 1]
          [0 0]
          [0 0]]

In [ ]:
```

Accuracy Score

```
from sklearn.metrics import confusion_matrix, accuracy_score

cm = confusion_matrix(y_test, y_pred)

print(cm)

accuracy_score(y_test, y_pred)
```

```
Accuracy Score: 0.94008
```



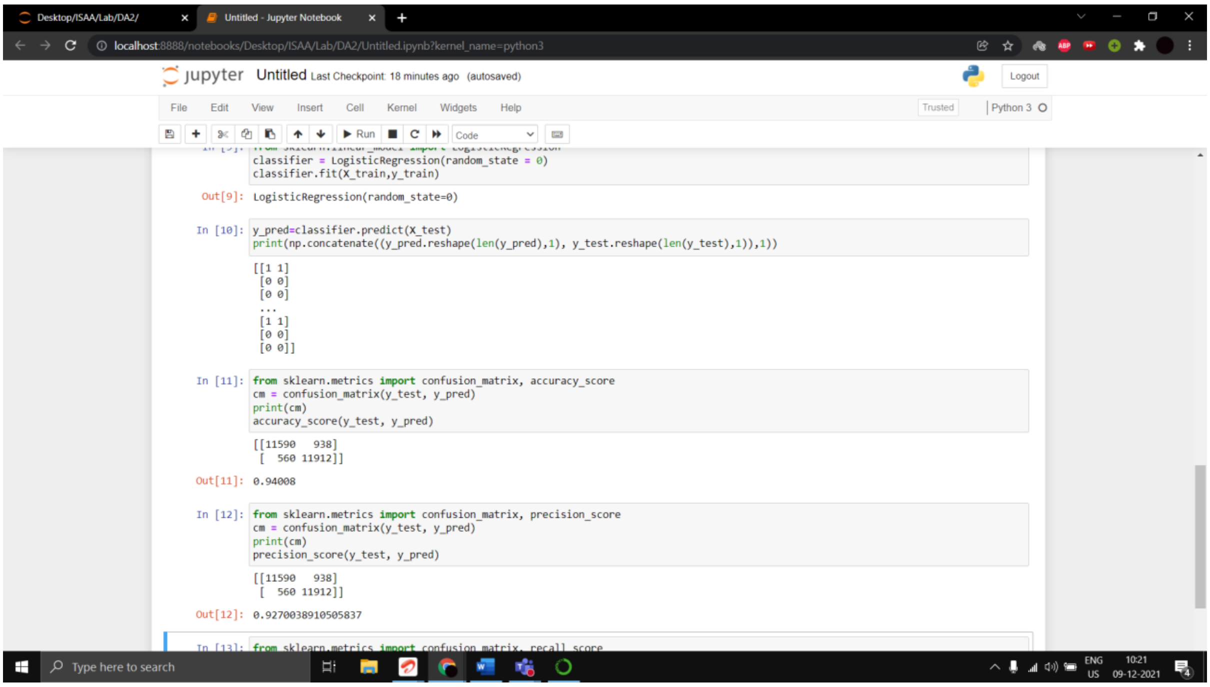
Desktop/ISAA/Lab/DA2/
Untitled - Jupyter Notebook
localhost:8888/notebooks/Desktop/ISAA/Lab/DA2/Untitled.ipynb?kernel_name=python3
jupyter
Untitled Last Checkpoint: 15 minutes ago (unsaved changes)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3
Run
Code
Type here to search
ENG
US
10:18
09-12-2021

```
   0.        ]]

In [9]:  from sklearn.linear_model import LogisticRegression
         classifier = LogisticRegression(random_state = 0)
         classifier.fit(X_train,y_train)

Out[9]:  LogisticRegression(random_state=0)

In [10]: y_pred=classifier.predict(X_test)
         print(np.concatenate((y_pred.reshape(len(y_pred),1), y_test.reshape(len(y_test),1)),1))

         [[1 1]
          [0 0]
          [0 0]
          ...
          [1 1]
          [0 0]
          [0 0]]

In [11]: from sklearn.metrics import confusion_matrix, accuracy_score
         cm = confusion_matrix(y_test, y_pred)
         print(cm)
         accuracy_score(y_test, y_pred)

         [[11590   938]
          [  560 11912]]

Out[11]: 0.94008

In [ ]:
```

Precision Score

```
from sklearn.metrics import confusion_matrix, precision_score

cm = confusion_matrix(y_test, y_pred)

print(cm)

precision_score(y_test, y_pred)
```

```
Precision Score: 0.9270038910505837
```

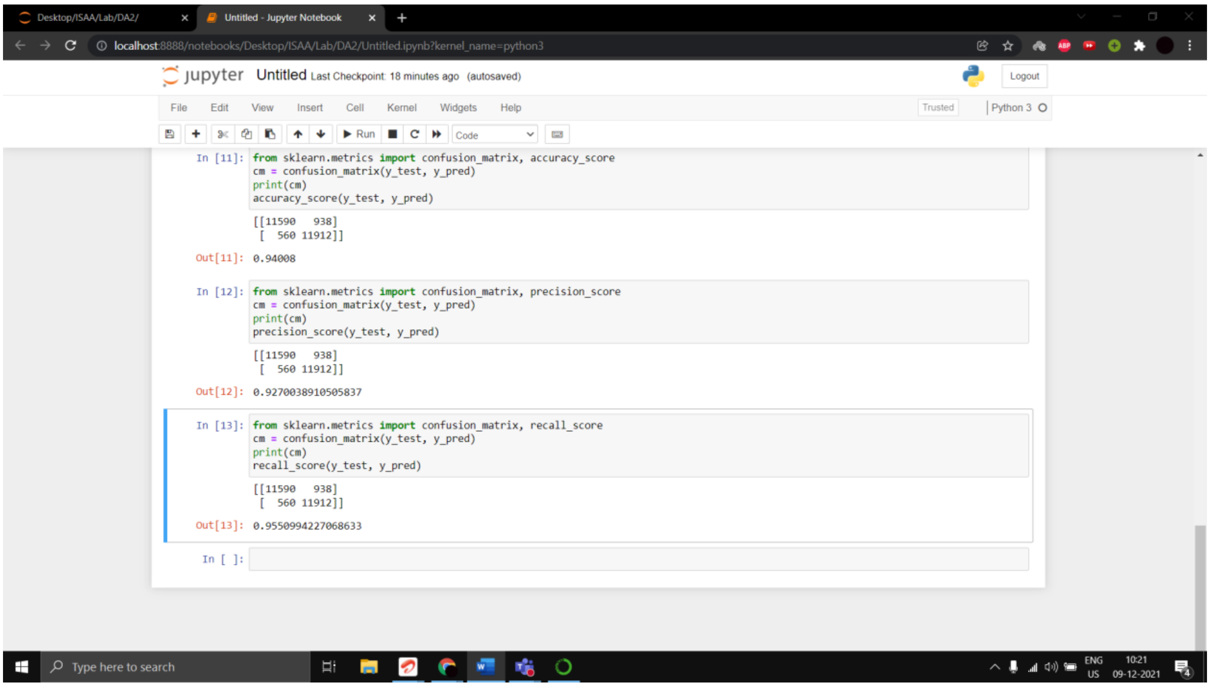

Desktop/ISAA/Lab/DA2/
Untitled - Jupyter Notebook
localhost:8888/notebooks/Desktop/ISAA/Lab/DA2/Untitled.ipynb?kernel_name=python3
jupyter Untitled Last Checkpoint: 18 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Type here to search
ENG US
10:21
09-12-2021

```
        classifier = LogisticRegression(random_state = 0)
        classifier.fit(X_train,y_train)

Out[9]: LogisticRegression(random_state=0)

In [10]: y_pred=classifier.predict(X_test)
         print(np.concatenate((y_pred.reshape(len(y_pred),1), y_test.reshape(len(y_test),1)),1))

         [[1 1]
          [0 0]
          [0 0]
          ...
          [1 1]
          [0 0]
          [0 0]]

In [11]: from sklearn.metrics import confusion_matrix, accuracy_score
         cm = confusion_matrix(y_test, y_pred)
         print(cm)
         accuracy_score(y_test, y_pred)

         [[11590   938]
          [  560 11912]]

Out[11]: 0.94008

In [12]: from sklearn.metrics import confusion_matrix, precision_score
         cm = confusion_matrix(y_test, y_pred)
         print(cm)
         precision_score(y_test, y_pred)

         [[11590   938]
          [  560 11912]]

Out[12]: 0.9270038910505837

In [13]: from sklearn.metrics import confusion_matrix, recall_score
```

Recall Score

```
from sklearn.metrics import confusion_matrix, recall_score

cm = confusion_matrix(y_test, y_pred)

print(cm)

recall_score(y_test, y_pred)
```

```
Recall Score: 0.9550994227068633
```


Desktop/ISAA/Lab/DA2/
Untitled - Jupyter Notebook
localhost:8888/notebooks/Desktop/ISAA/Lab/DA2/Untitled.ipynb?kernel_name=python3
jupyter Untitled Last Checkpoint: 18 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Type here to search
ENG US
10:21
09-12-2021

```
In [11]: from sklearn.metrics import confusion_matrix, accuracy_score
         cm = confusion_matrix(y_test, y_pred)
         print(cm)
         accuracy_score(y_test, y_pred)

         [[11590   938]
          [  560 11912]]

Out[11]: 0.94008

In [12]: from sklearn.metrics import confusion_matrix, precision_score
         cm = confusion_matrix(y_test, y_pred)
         print(cm)
         precision_score(y_test, y_pred)

         [[11590   938]
          [  560 11912]]

Out[12]: 0.9270038910505837

In [13]: from sklearn.metrics import confusion_matrix, recall_score
         cm = confusion_matrix(y_test, y_pred)
         print(cm)
         recall_score(y_test, y_pred)

         [[11590   938]
          [  560 11912]]

Out[13]: 0.9550994227068633

In [ ]:
```

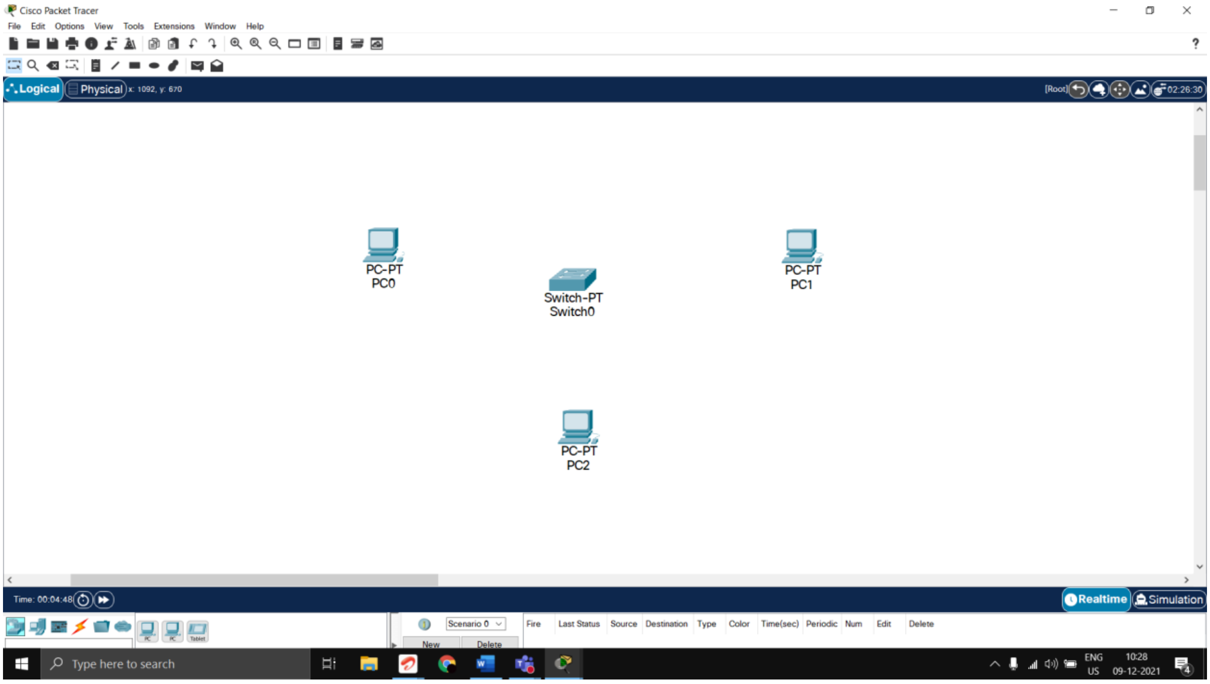
2)

Components used: PC

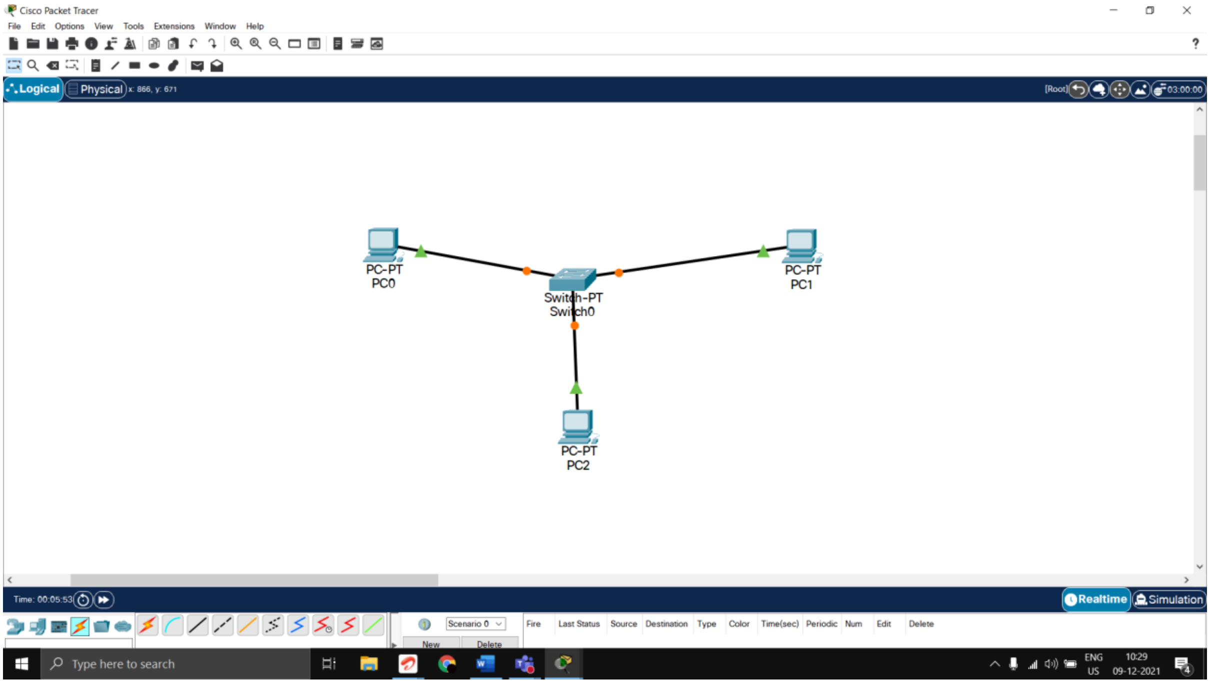
Switch

STEPS:

1) Placing the components (PC and Switch) {PC and PT-Switch}

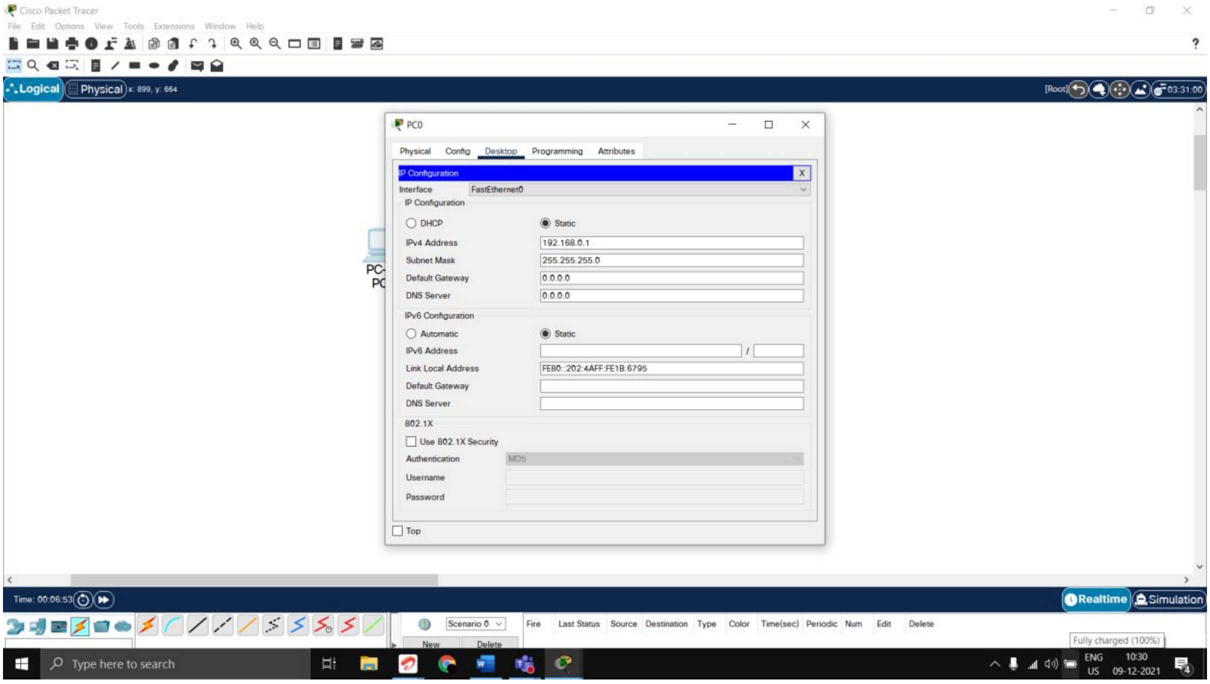


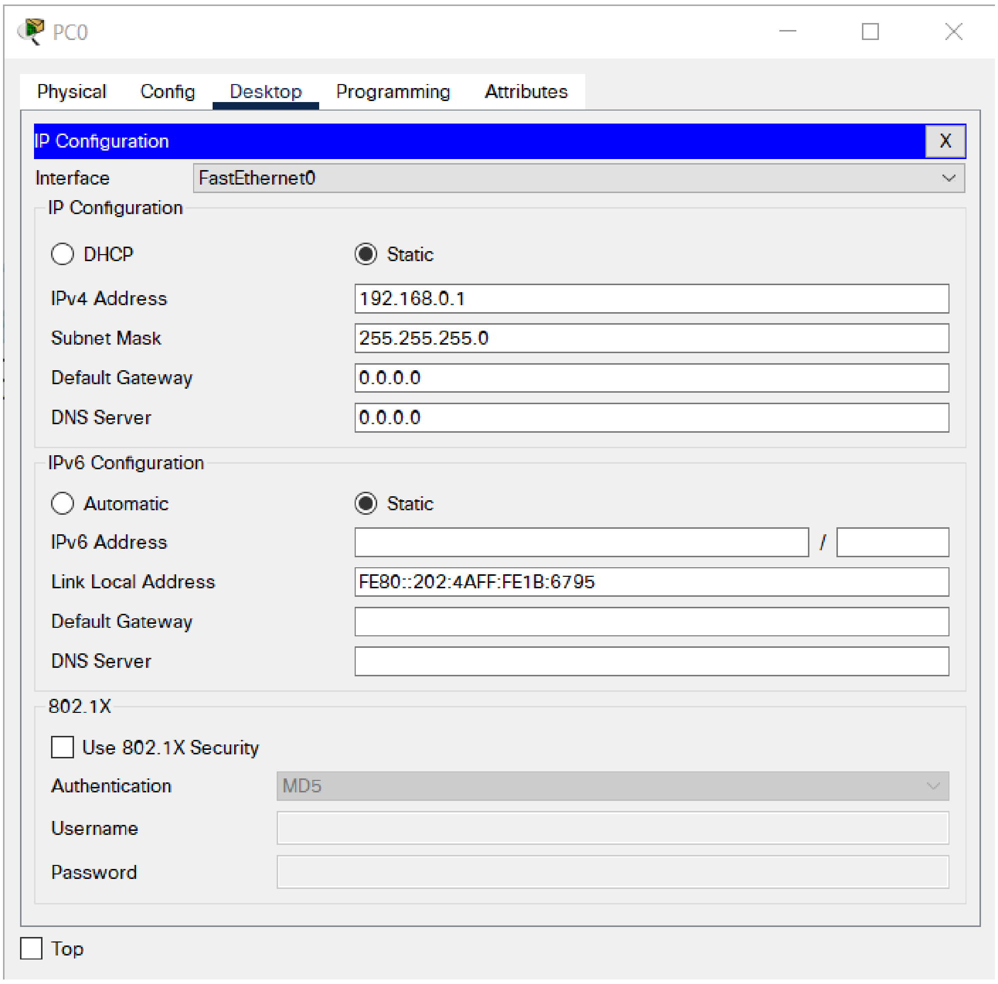
2) Connecting PCs to the Switch



3) Configuring IP addresses and Masks of the PCs

PC0





IP Configuration

Interface: FastEthernet0

IP Configuration

○ DHCP ● Static

IPv4 Address: 192.168.0.1

Subnet Mask: 255.255.255.0

Default Gateway: 0.0.0.0

DNS Server: 0.0.0.0

IPv6 Configuration

○ Automatic ● Static

IPv6 Address: /

Link Local Address: FE80::202:4AFF:FE1B:6795

Default Gateway:

DNS Server:

802.1X

□ Use 802.1X Security

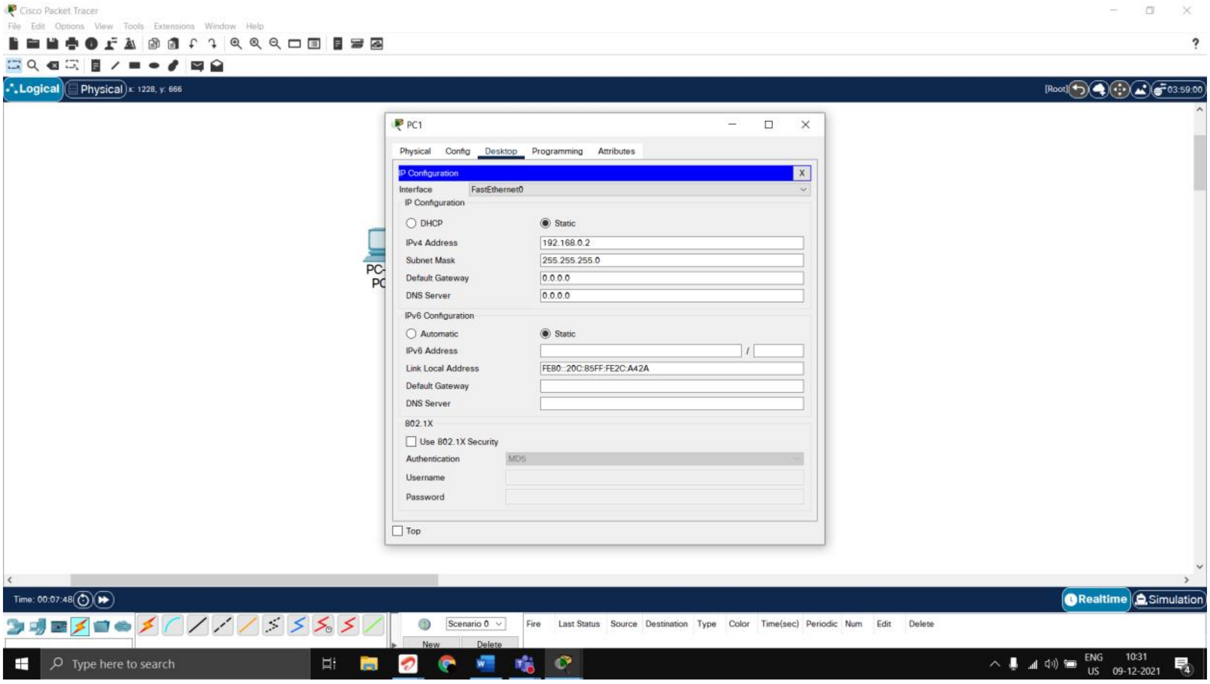
Authentication: MD5

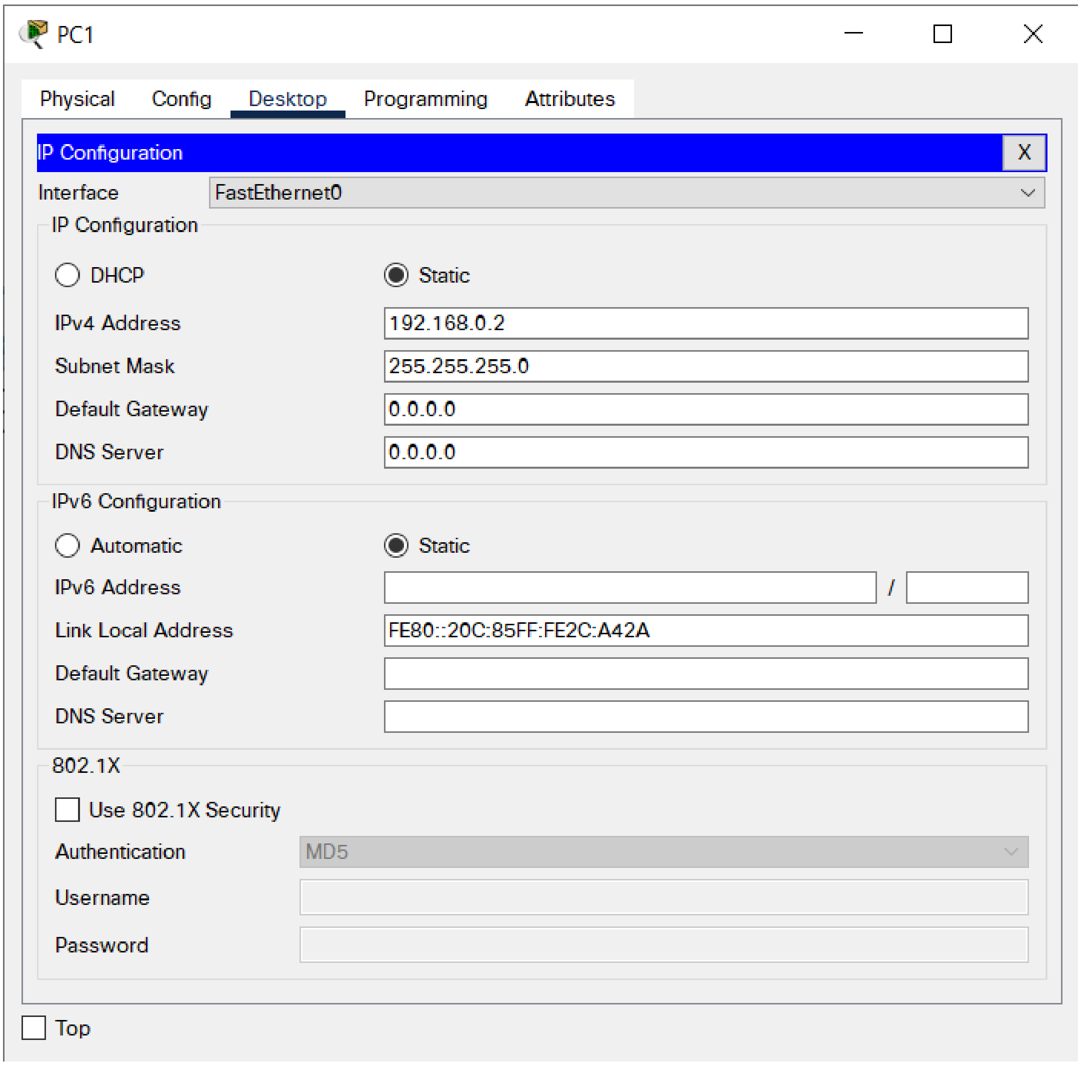
Username:

Password:

□ Top

PC1




PC1

Physical Config Desktop Programming Attributes

IP Configuration X

Interface: FastEthernet0

IP Configuration

○ DHCP ● Static

IPv4 Address: 192.168.0.2

Subnet Mask: 255.255.255.0

Default Gateway: 0.0.0.0

DNS Server: 0.0.0.0

IPv6 Configuration

○ Automatic ● Static

IPv6 Address: /

Link Local Address: FE80::20C:85FF:FE2C:A42A

Default Gateway:

DNS Server:

802.1X

□ Use 802.1X Security

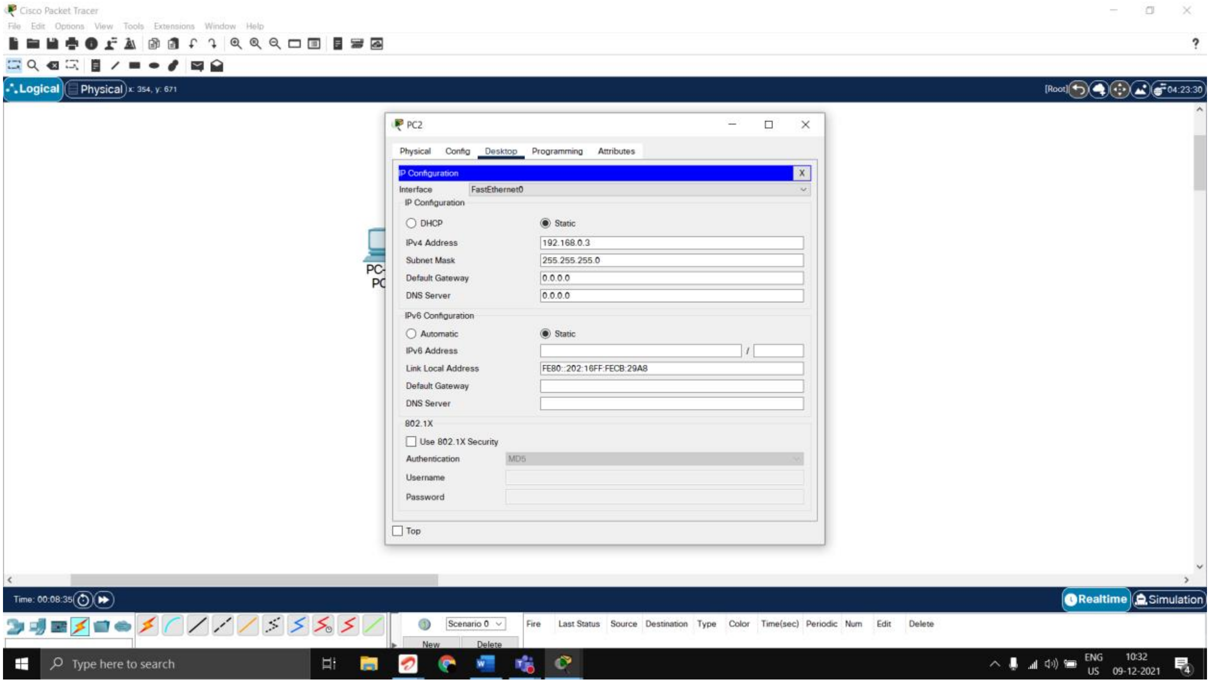
Authentication: MD5

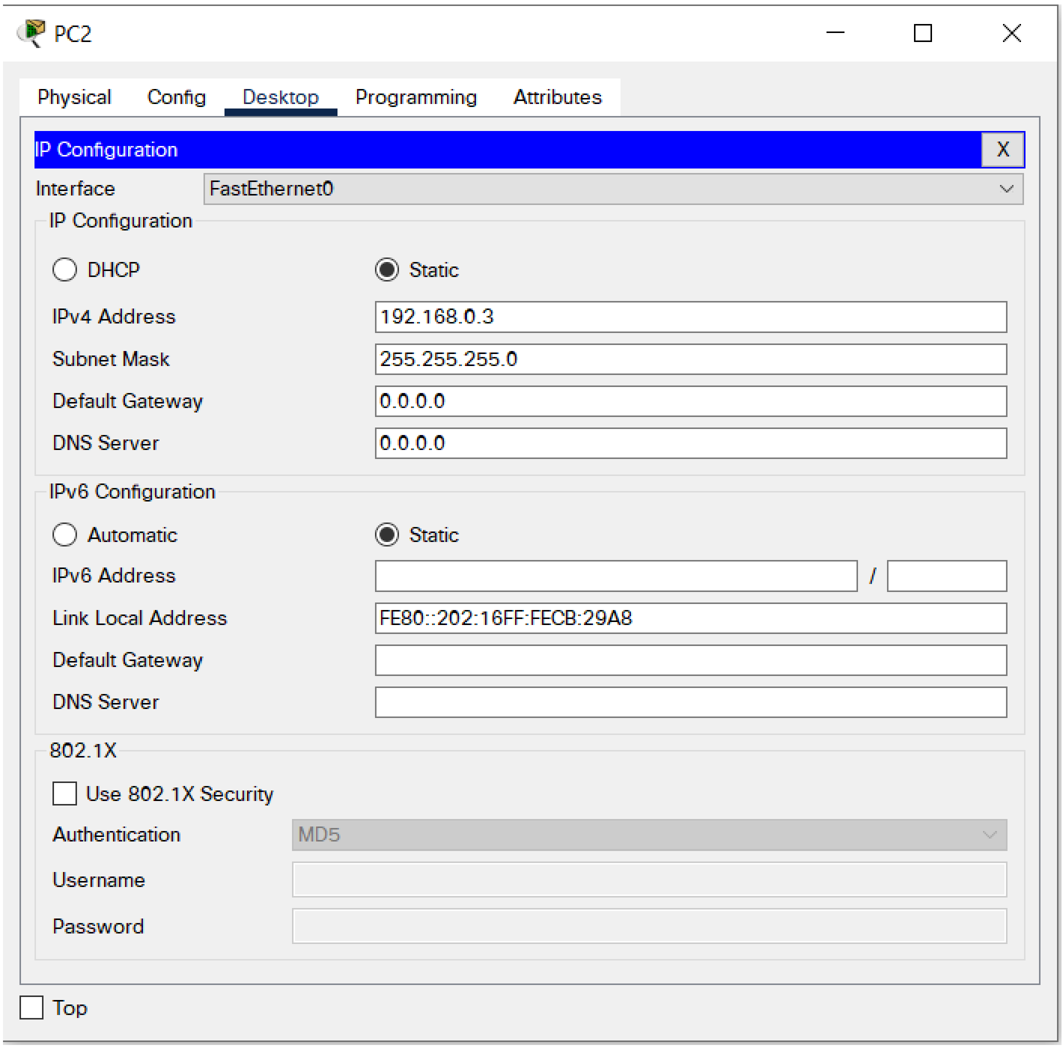
Username:

Password:

□ Top

PC2





IP Configuration X

Interface FastEthernet0

IP Configuration

○ DHCP ● Static

IPv4 Address 192.168.0.3

Subnet Mask 255.255.255.0

Default Gateway 0.0.0.0

DNS Server 0.0.0.0

IPv6 Configuration

○ Automatic ● Static

IPv6 Address /

Link Local Address FE80::202:16FF:FECB:29A8

Default Gateway

DNS Server

802.1X

□ Use 802.1X Security

Authentication MD5

Username

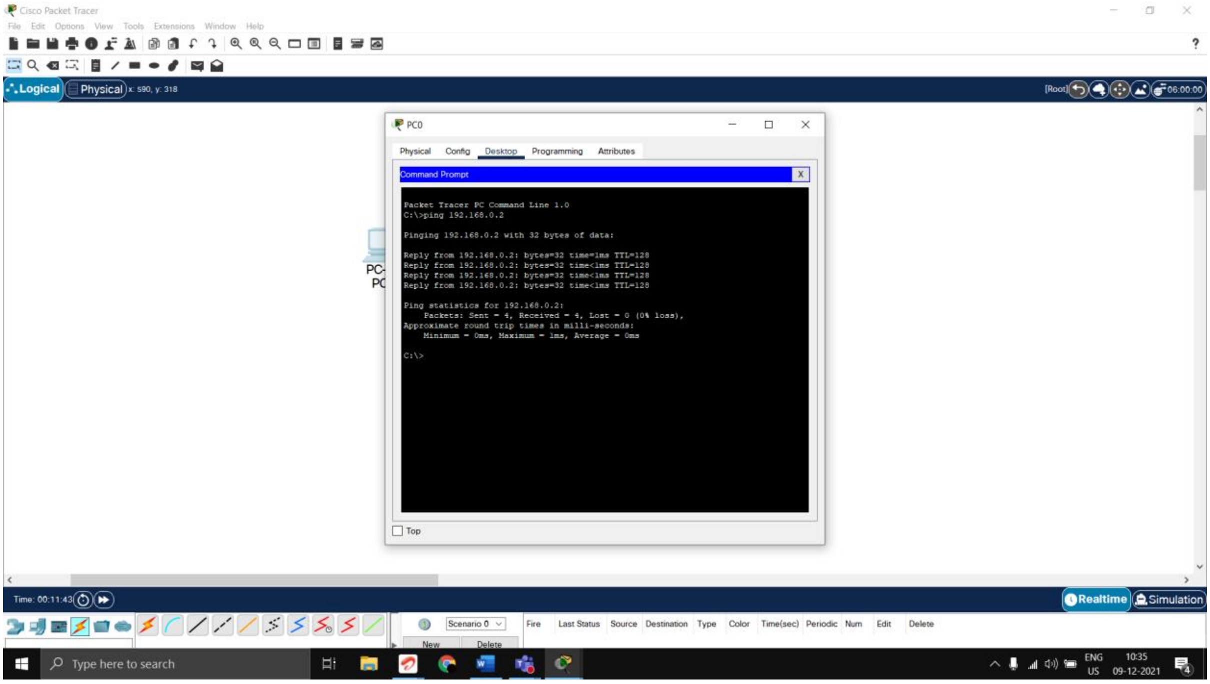
Password

□ Top

4) Checking the configuration using ping command and simulation

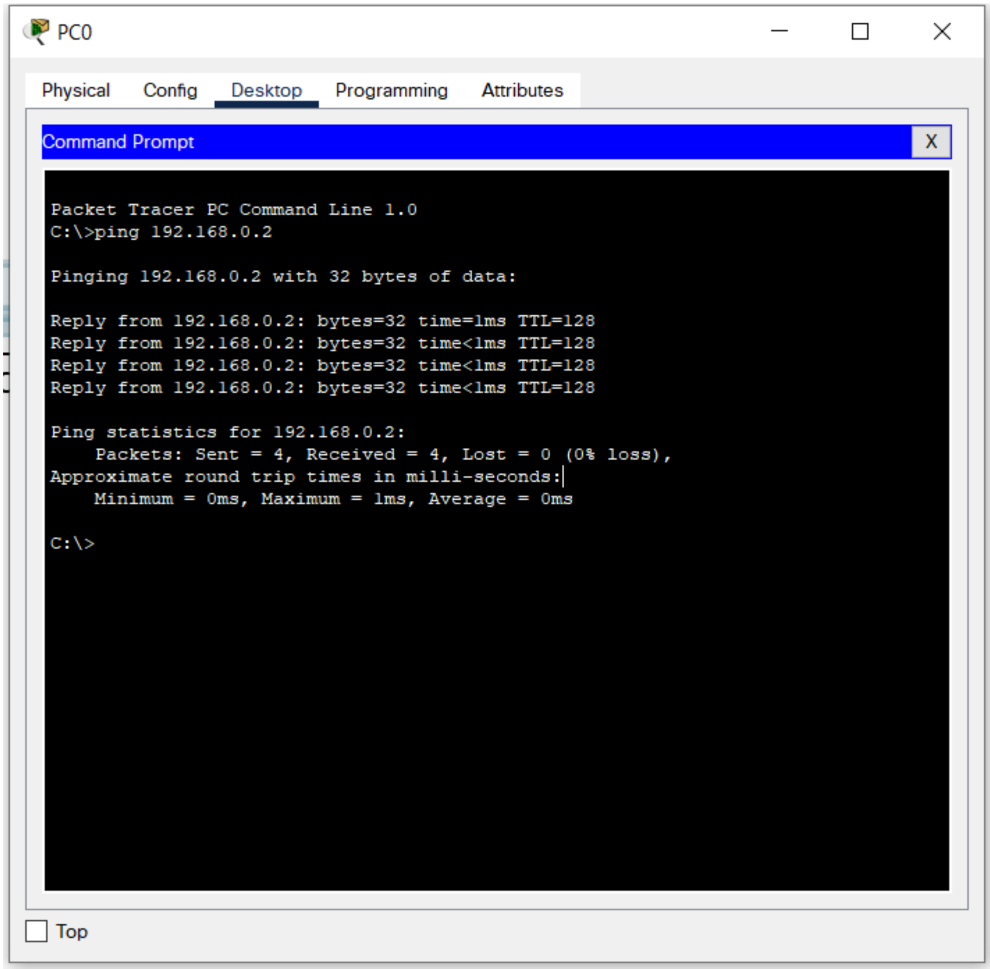
a) Ping Command (Using PC0 to ping PC1)

Command: ping 192.168.0.2



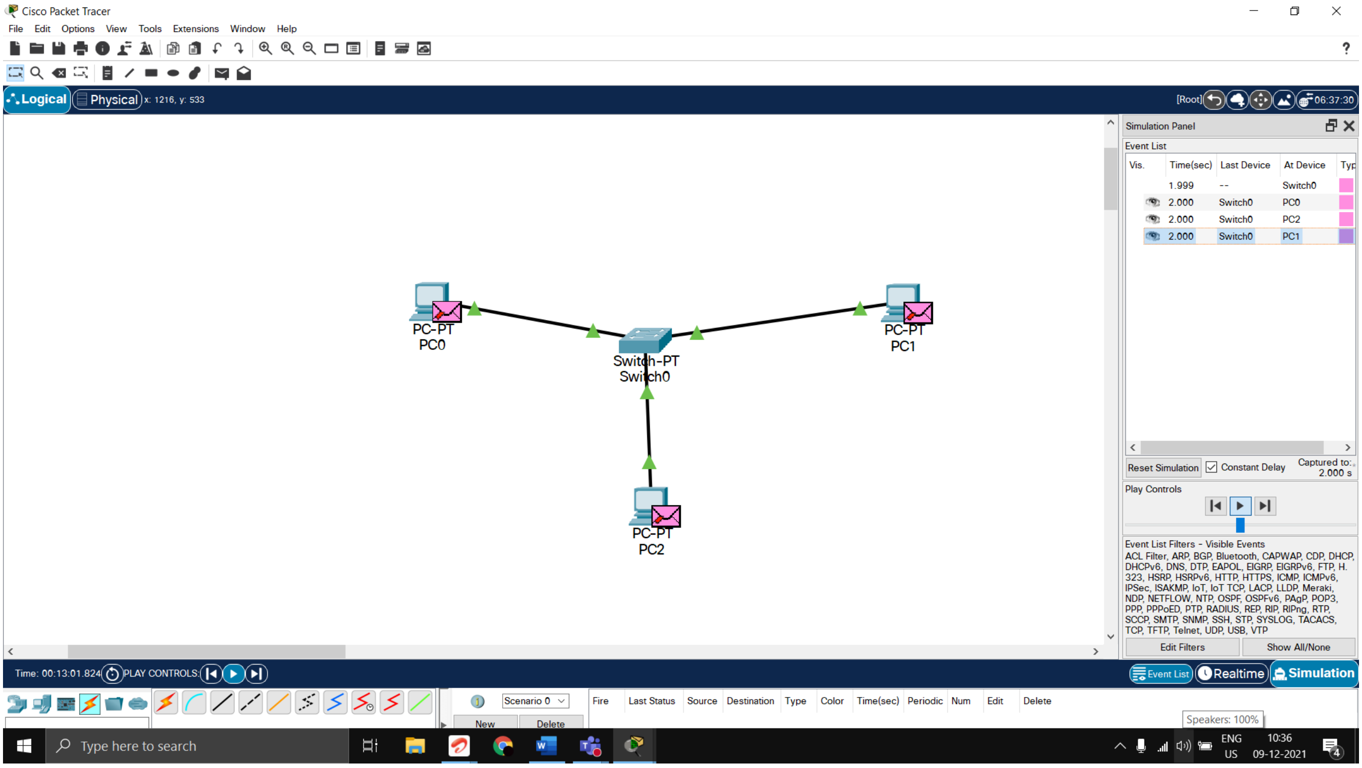
Cisco Packet Tracer
File Edit Options View Tools Extensions Window Help
Logical
Physical
PC0
Physical Config Desktop Programming Attributes
Command Prompt
Top
Time: 00:11:43
Realtime
Simulation
Scenario 0
New Delete
Fire Last Status Source Destination Type Color Time(sec) Periodic Num Edit Delete
Type here to search
ENG US
10:35
09-12-2021

```
Packet Tracer PC Command Line 1.0
C:\>ping 192.168.0.2

Pinging 192.168.0.2 with 32 bytes of data:

Reply from 192.168.0.2: bytes=32 time=1ms TTL=128
Reply from 192.168.0.2: bytes=32 time<1ms TTL=128
Reply from 192.168.0.2: bytes=32 time<1ms TTL=128
Reply from 192.168.0.2: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 1ms, Average = 0ms

C:\>
```



```
Packet Tracer PC Command Line 1.0
C:\>ping 192.168.0.2

Pinging 192.168.0.2 with 32 bytes of data:

Reply from 192.168.0.2: bytes=32 time=1ms TTL=128
Reply from 192.168.0.2: bytes=32 time<1ms TTL=128
Reply from 192.168.0.2: bytes=32 time<1ms TTL=128
Reply from 192.168.0.2: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 1ms, Average = 0ms

C:\>
```

b) Using Simulation



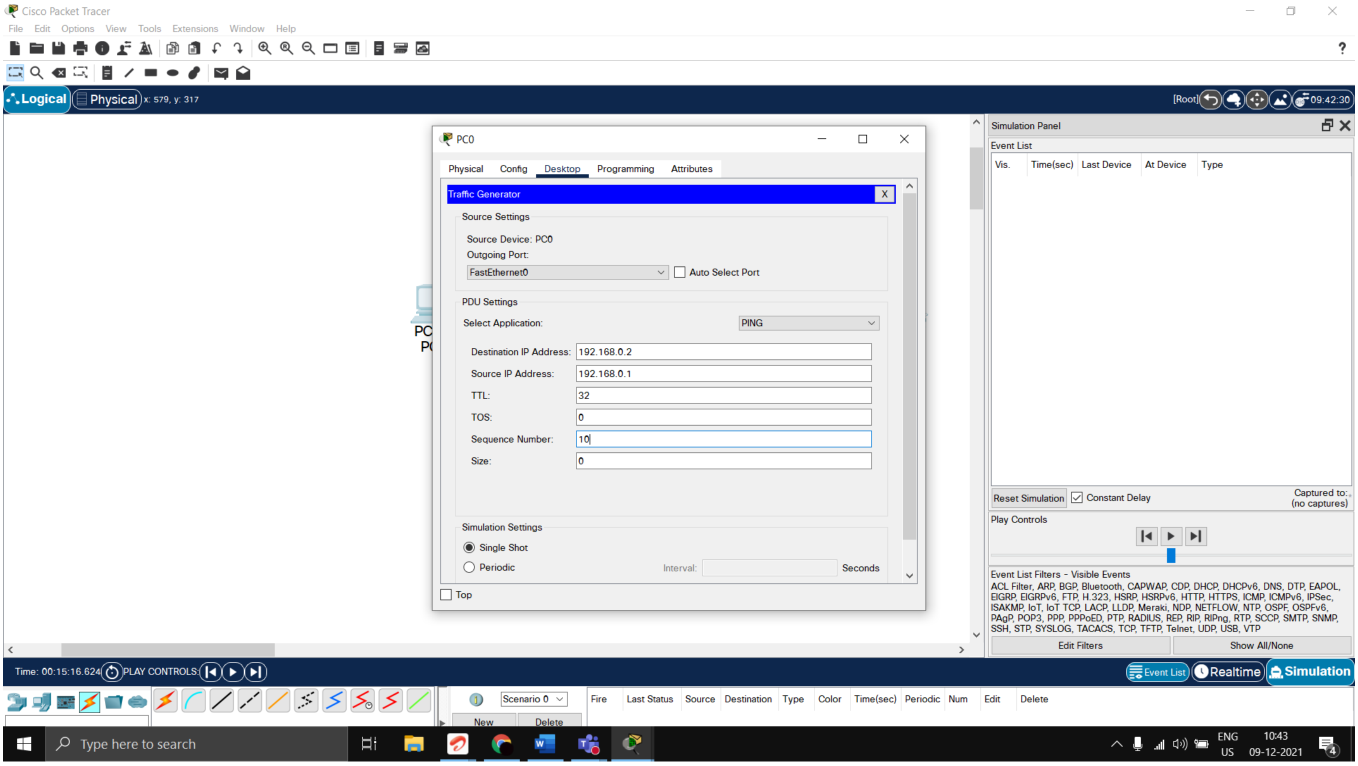
Event List

Vis.	Time(sec)	Last Device	At Device	Typ
	1.999	--	Switch0	🟪
👁	2.000	Switch0	PC0	🟪
👁	2.000	Switch0	PC2	🟪
👁	2.000	Switch0	PC1	🟪

Event List Filters - Visible Events
ACL Filter, ARP, BGP, Bluetooth, CAPWAP, CDP, DHCP, DHCPv6, DNS, DTP, EAPOL, EIGRP, EIGRPv6, FTP, H.323, HSRP, HSRPv6, HTTP, HTTPS, ICMP, ICMPv6, IPSec, ISAKMP, IoT, IoT TCP, LACP, LLDP, Meraki, NDP, NETFLOW, NTP, OSPF, OSPFv6, PAgP, POP3, PPP, PPPoED, PTP, RADIUS, REP, RIP, RIPng, RTP, SCCP, SMTP, SNMP, SSH, STP, SYSLOG, TACACS, TCP, TFTP, Telnet, UDP, USB, VTP

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit	Delete

Configuring and adding a packet to be sent during simulation



Traffic Generator

Source Settings
Source Device: PC0
Outgoing Port:
FastEthernet0 □ Auto Select Port

PDU Settings
Select Application: PING
Destination IP Address: 192.168.0.2
Source IP Address: 192.168.0.1
TTL: 32
TOS: 0
Sequence Number: 10
Size: 0

Simulation Settings
● Single Shot
○ Periodic Interval: Seconds

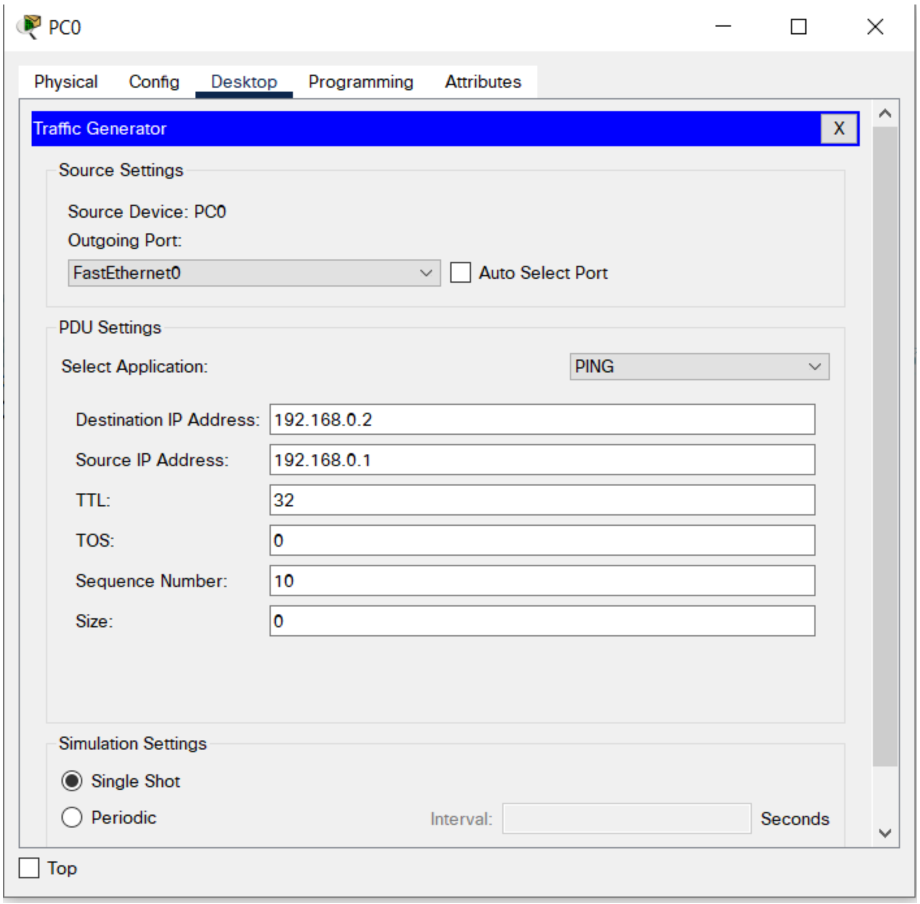
□ Top

Event List

Vis.	Time(sec)	Last Device	At Device	Type

Event List Filters - Visible Events
ACL Filter, ARP, BGP, Bluetooth, CAPWAP, CDP, DHCP, DHCPv6, DNS, DTP, EAPOL, EIGRP, EIGRPv6, FTP, H.323, HSRP, HSRPv6, HTTP, HTTPS, ICMP, ICMPv6, IPSec, ISAKMP, IoT, IoT TCP, LACP, LLDP, Meraki, NDP, NETFLOW, NTP, OSPF, OSPFv6, PAgP, POP3, PPP, PPPoED, PTP, RADIUS, REP, RIP, RIPng, RTP, SCCP, SMTP, SNMP, SSH, STP, SYSLOG, TACACS, TCP, TFTP, Telnet, UDP, USB, VTP

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit	Delete



Traffic Generator

Source Settings

Source Device: PC0

Outgoing Port: FastEthernet0 □ Auto Select Port

PDU Settings

Select Application: PING

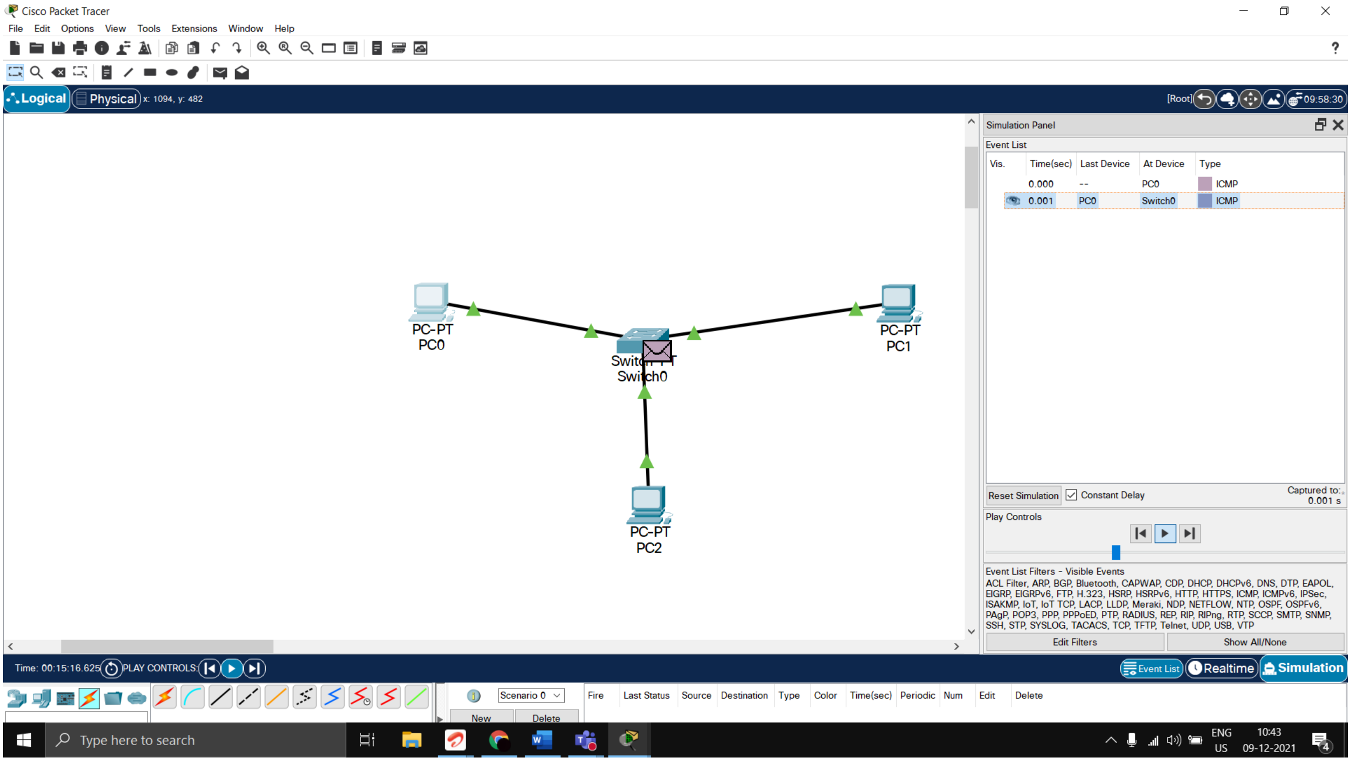
Destination IP Address:	192.168.0.2
Source IP Address:	192.168.0.1
TTL:	32
TOS:	0
Sequence Number:	10
Size:	0

Simulation Settings

● Single Shot

○ Periodic Interval: Seconds

□ Top



Simulation Panel

Event List

Vis.	Time(sec)	Last Device	At Device	Type
	0.000	--	PC0	ICMP
	0.001	PC0	Switch0	ICMP

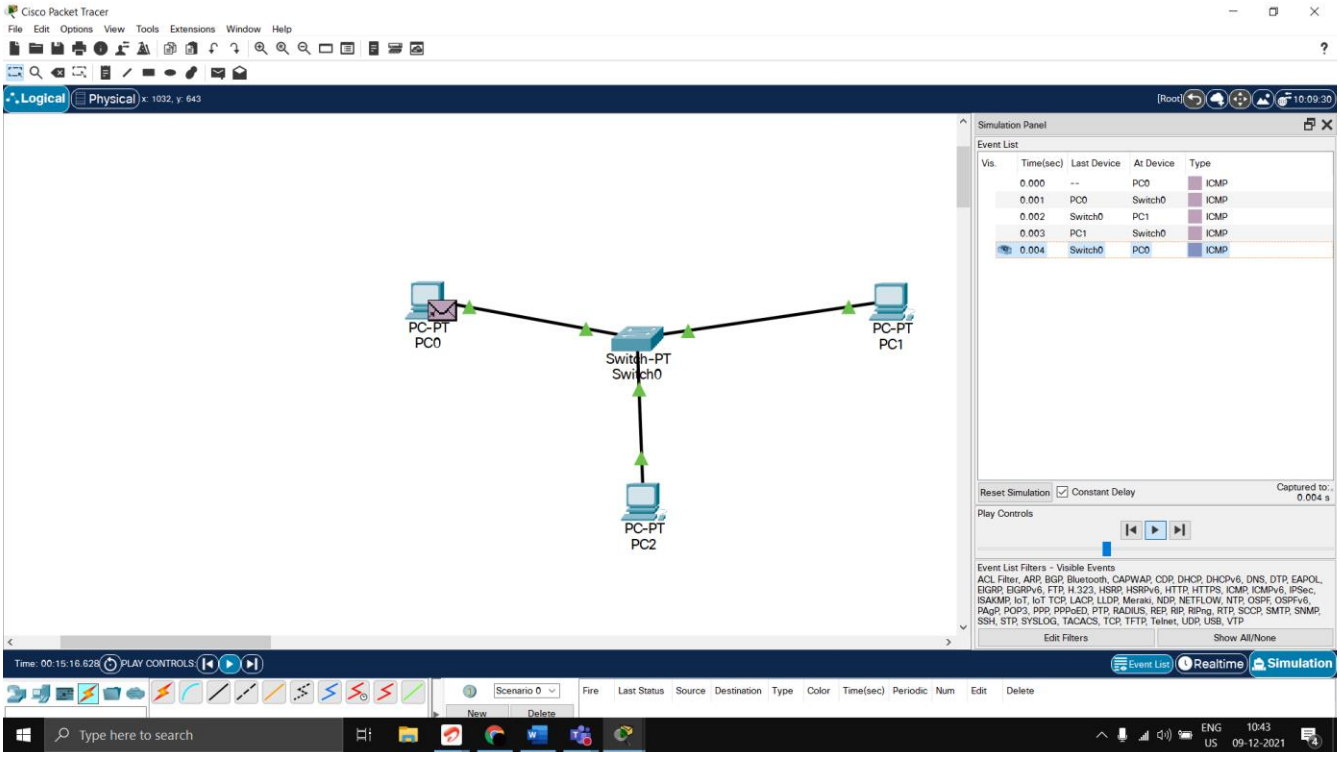
Reset Simulation ☑ Constant Delay Captured to: 0.001 s

Event List Filters - Visible Events

ACL Filter, ARP, BGP, Bluetooth, CAPWAP, CDP, DHCP, DHCPv6, DNS, DTP, EAPOL, EIGRP, EIGRPv6, FTP, H.323, HSRP, HSRPv6, HTTP, HTTPS, ICMP, ICMPv6, IPSec, ISAKMP, IoT, IoT TCP, LACP, LLDP, Meraki, NDP, NETFLOW, NTP, OSPF, OSPFv6, PAgP, POP3, PPP, PPPoED, PTP, RADIUS, REP, RIP, RIPng, RTP, SCCP, SMTP, SNMP, SSH, STP, SYSLOG, TACACS, TCP, TFTP, Telnet, UDP, USB, VTP

Time: 00:15:16.625 PLAY CONTROLS:

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit	Delete



Event List

Vis.	Time(sec)	Last Device	At Device	Type
	0.000	--	PC0	ICMP
	0.001	PC0	Switch0	ICMP
	0.002	Switch0	PC1	ICMP
	0.003	PC1	Switch0	ICMP
	0.004	Switch0	PC0	ICMP

Reset Simulation ☑ Constant Delay

Captured to:
0.004 s

Play Controls

Event List Filters - Visible Events

ACL Filter, ARP, BGP, Bluetooth, CAPWAP, CDP, DHCP, DHCPv6, DNS, DTP, EAPOL, EIGRP, EIGRPv6, FTP, H.323, HSRP, HSRPv6, HTTP, HTTPS, ICMP, ICMPv6, IPSec, ISAKMP, IoT, IoT TCP, LACP, LLDP, Meraki, NDP, NETFLOW, NTP, OSPF, OSPFv6, PAgP, POP3, PPP, PPPoED, PTP, RADIUS, REP, RIP, RIPng, RTP, SCCP, SMTP, SNMP, SSH, STP, SYSLOG, TACACS, TCP, TFTP, Telnet, UDP, USB, VTP

Edit Filters | Show All/None

QUESTION 2a FULL CODE

```
import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

dataset = pd.read_csv('Malware_dataset.csv')

dataset1 = dataset.copy()

dataset1 = dataset1.drop(['classification'], axis=1)

X=dataset1.iloc[:,1:].values

y=dataset.iloc[:,2].values

print (X)

from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()

y = le.fit_transform(y)

print (y)

from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=0)

print("Xtrain")
print(X_train)
print("Xtest")
print(X_test)
print("ytrain")
print(y_train)
print("ytest")
print(y_test)
from sklearn.preprocessing import StandardScaler
sc= StandardScaler()
X_train= sc.fit_transform(X_train)
X_test= sc.transform(X_test)
print(X_train)
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression(random_state = 0)
classifier.fit(X_train,y_train)
y_pred=classifier.predict(X_test)
print(np.concatenate((y_pred.reshape(len(y_pred),1), y_test.reshape(len(y_test),1)),1))
from sklearn.metrics import confusion_matrix, accuracy_score
cm = confusion_matrix(y_test, y_pred)
print(cm)
accuracy_score(y_test, y_pred)
from sklearn.metrics import confusion_matrix, precision_score
cm = confusion_matrix(y_test, y_pred)
print(cm)
precision_score(y_test, y_pred)
from sklearn.metrics import confusion_matrix, recall_score
cm = confusion_matrix(y_test, y_pred)
print(cm)
recall_score(y_test, y_pred)
```