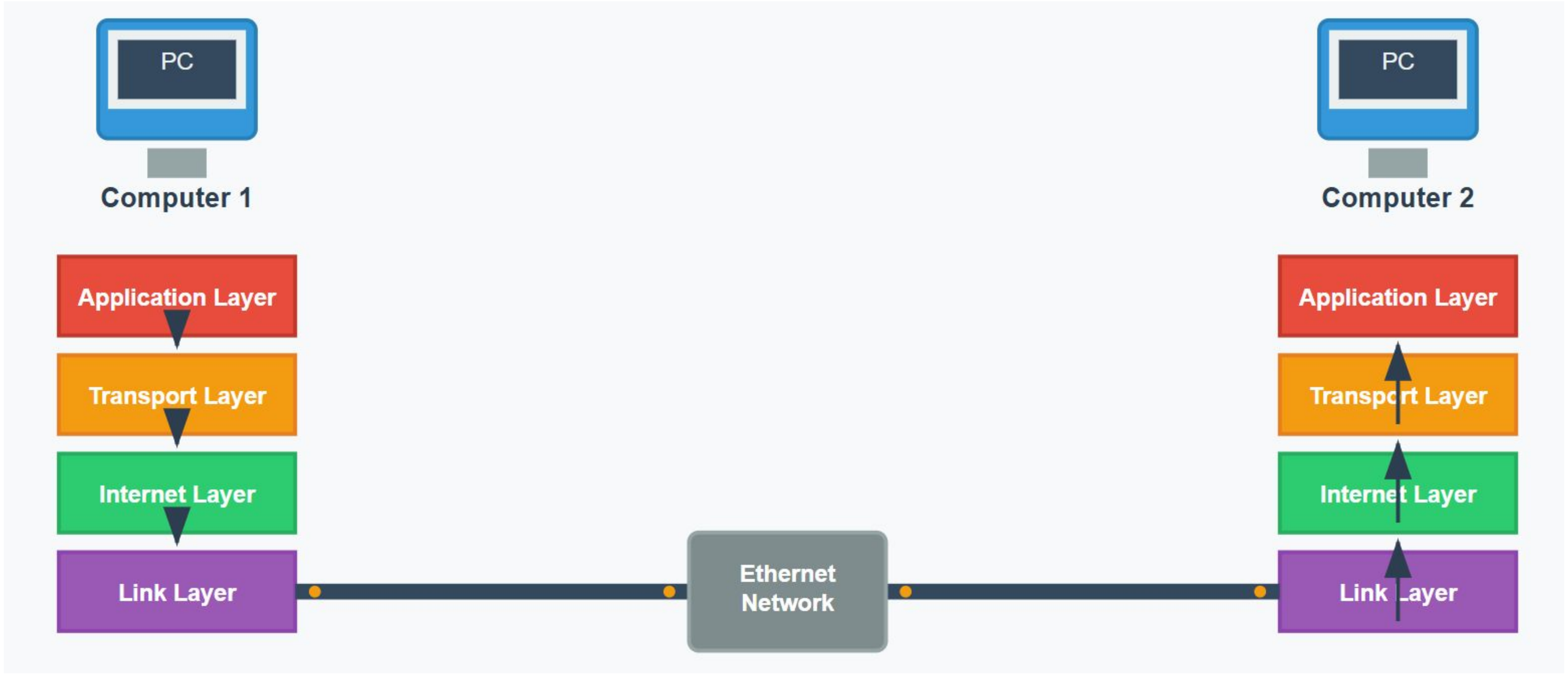
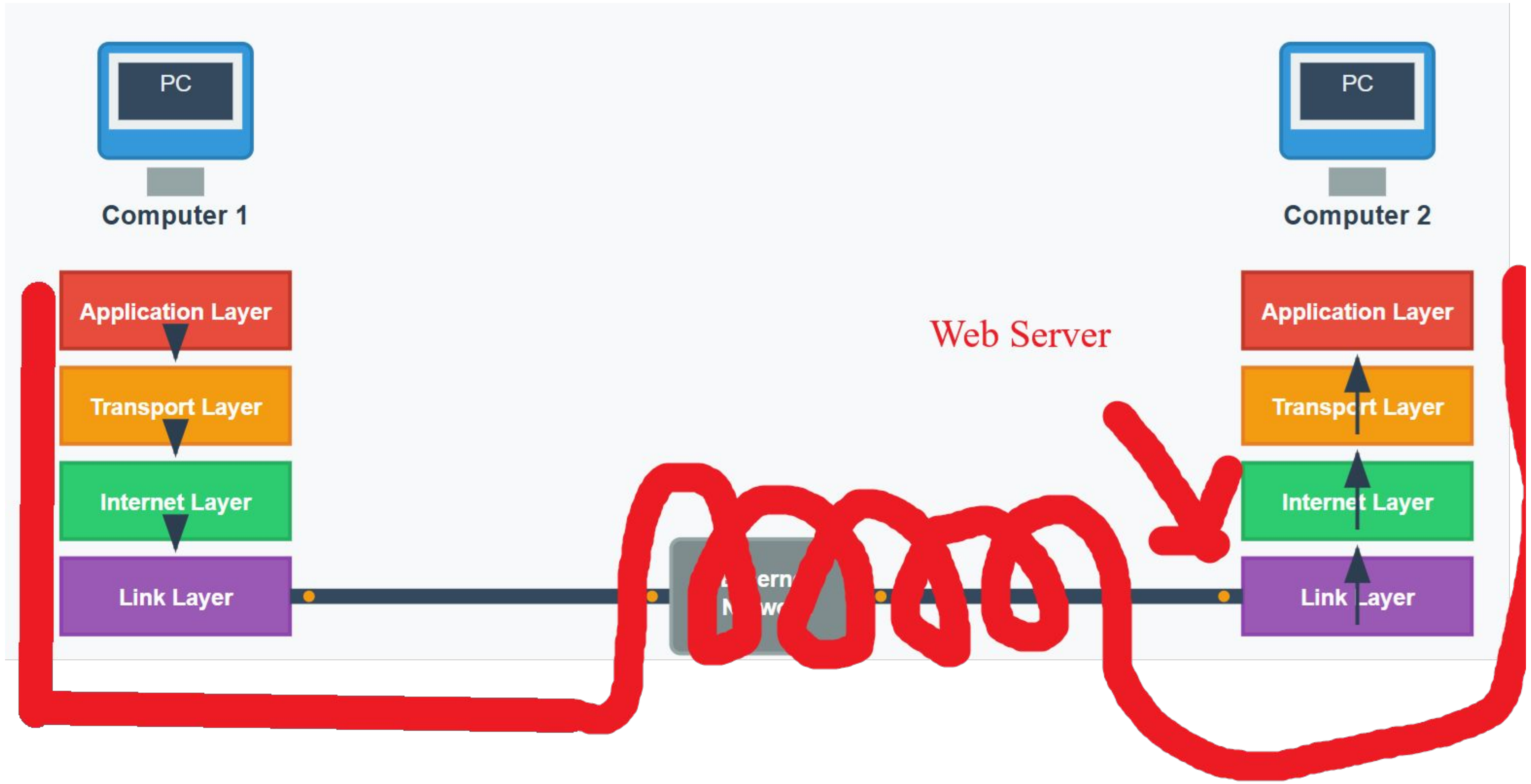
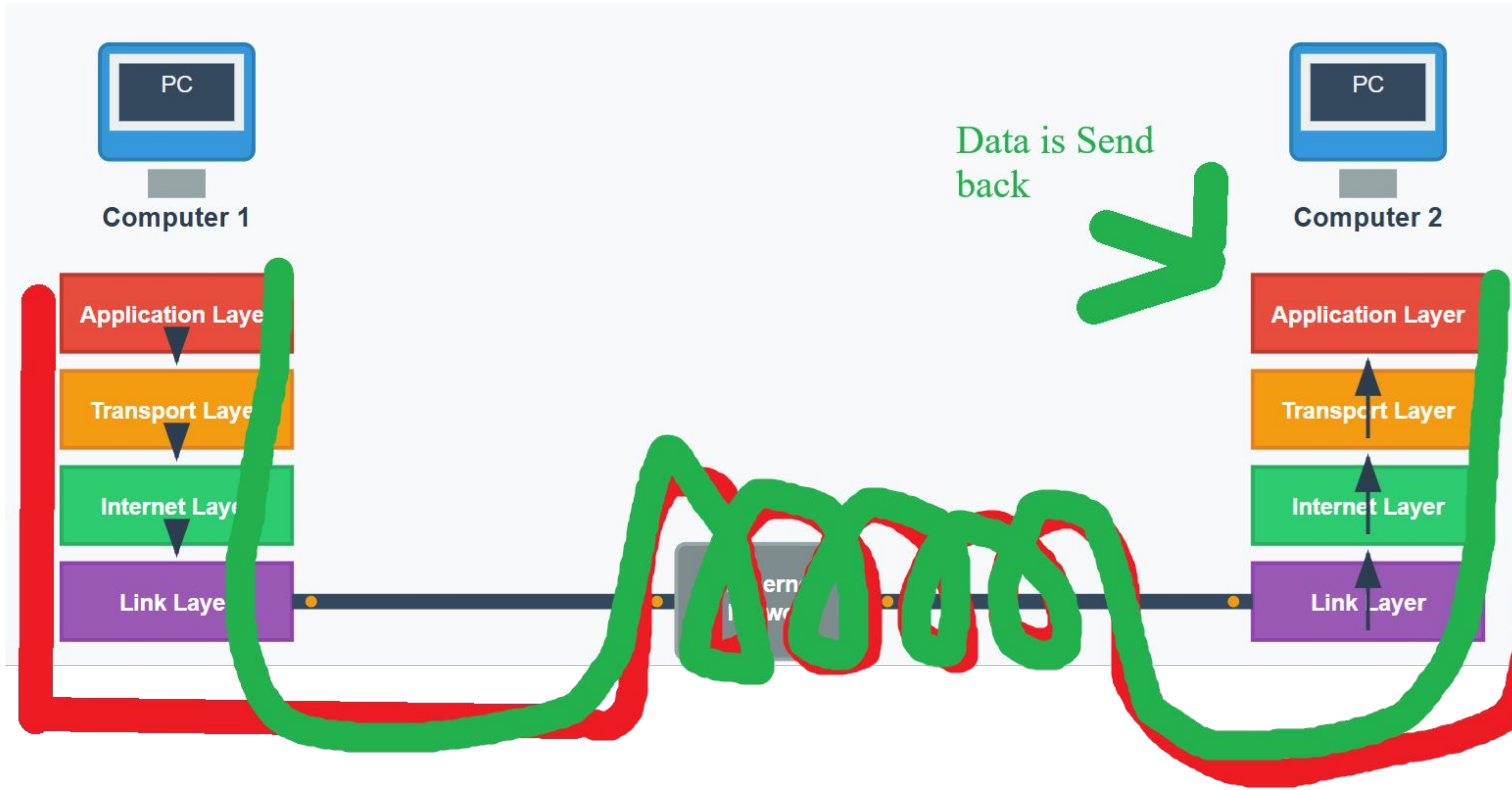
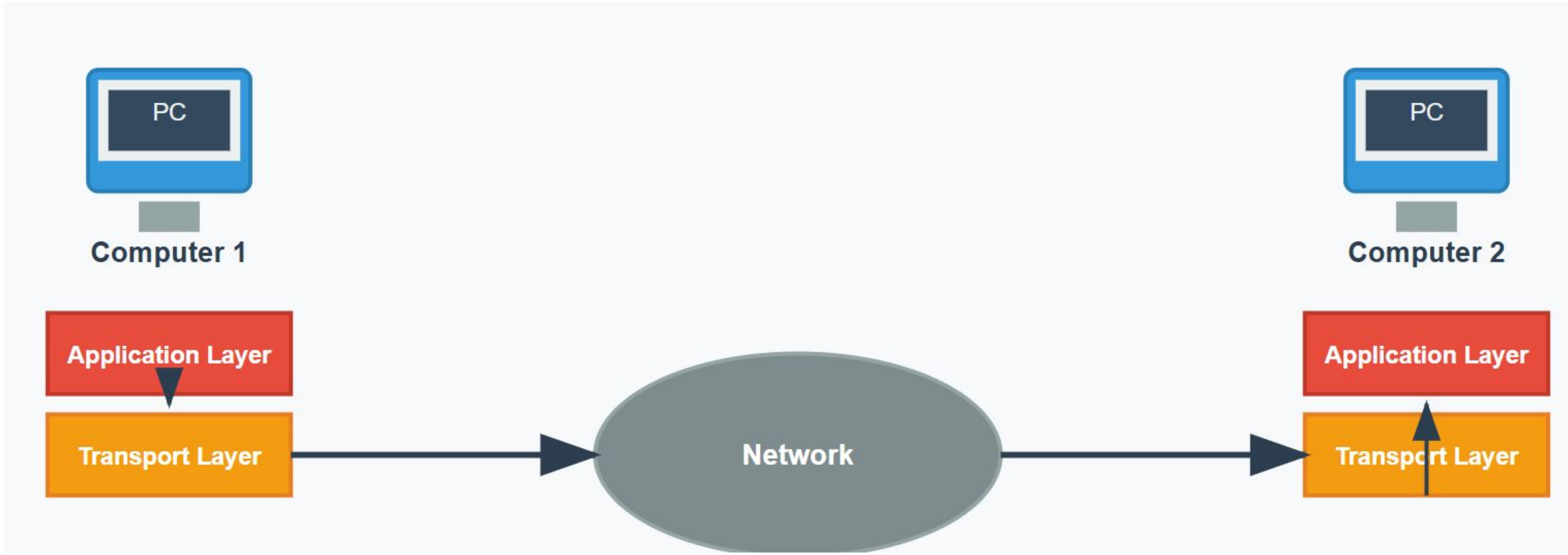
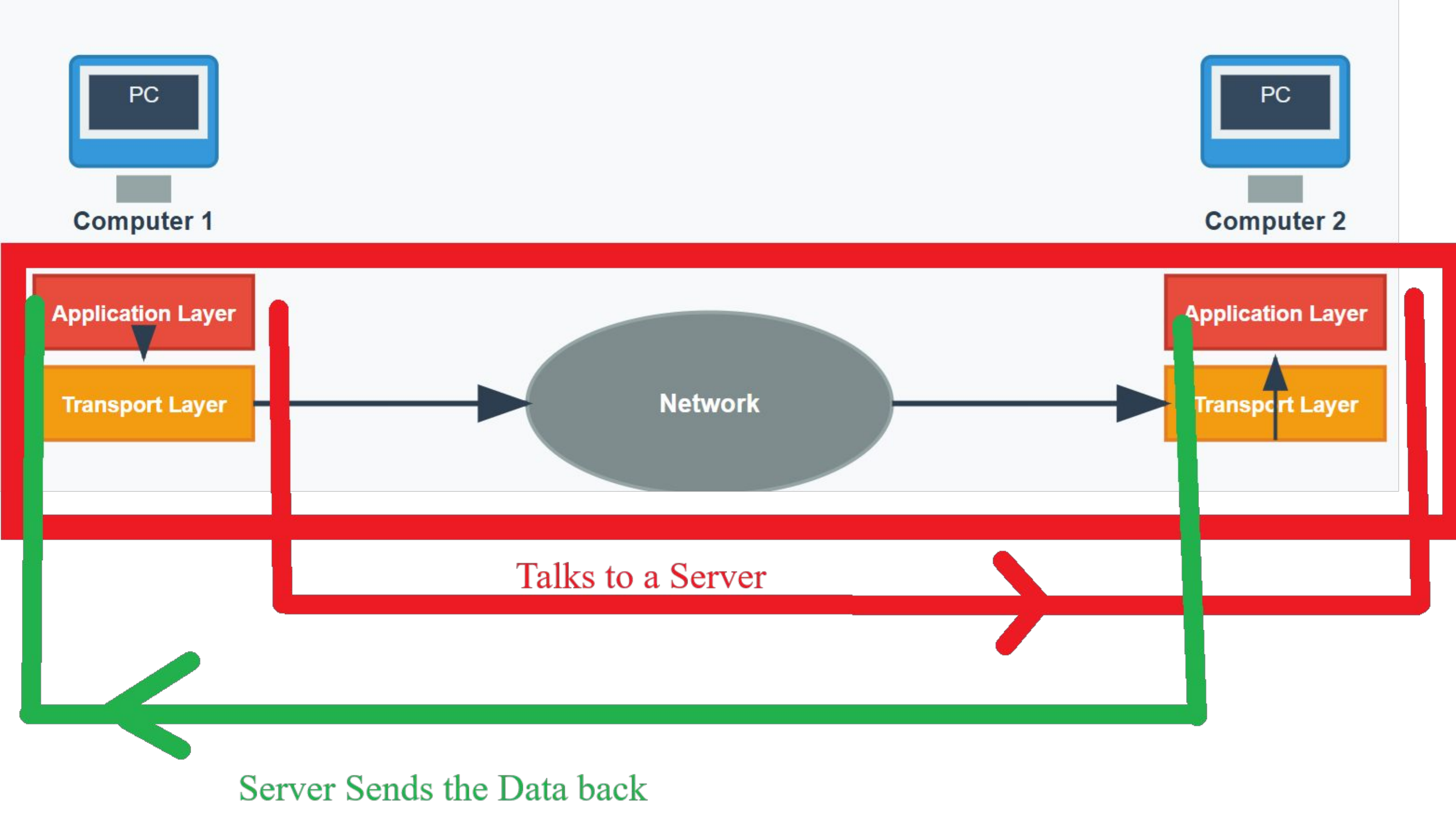
Network Architecture
(TCP Transport Control Protocol)




PC
Computer 1
Application Layer
Transport Layer
Internet Layer
Link Layer
PC
Computer 2
Application Layer
Transport Layer
Internet Layer
Link Layer
Web Server

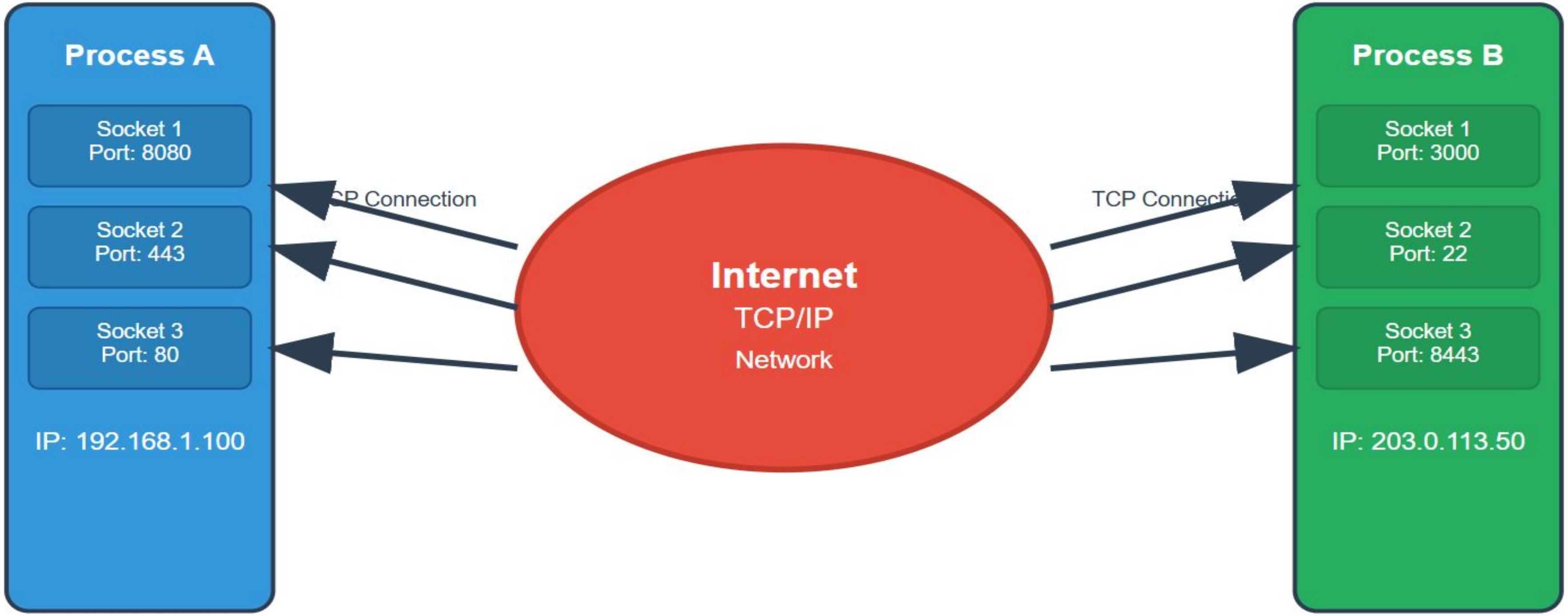

PC
Computer 1
Application Laye
Transport Laye
Internet Laye
Link Laye
PC
Computer 2
Data is Send back
Application Layer
Transport Layer
Internet Layer
Link Layer


PC
Computer 1
PC
Computer 2
Application Layer
Transport Layer
Network
Application Layer
Transport Layer

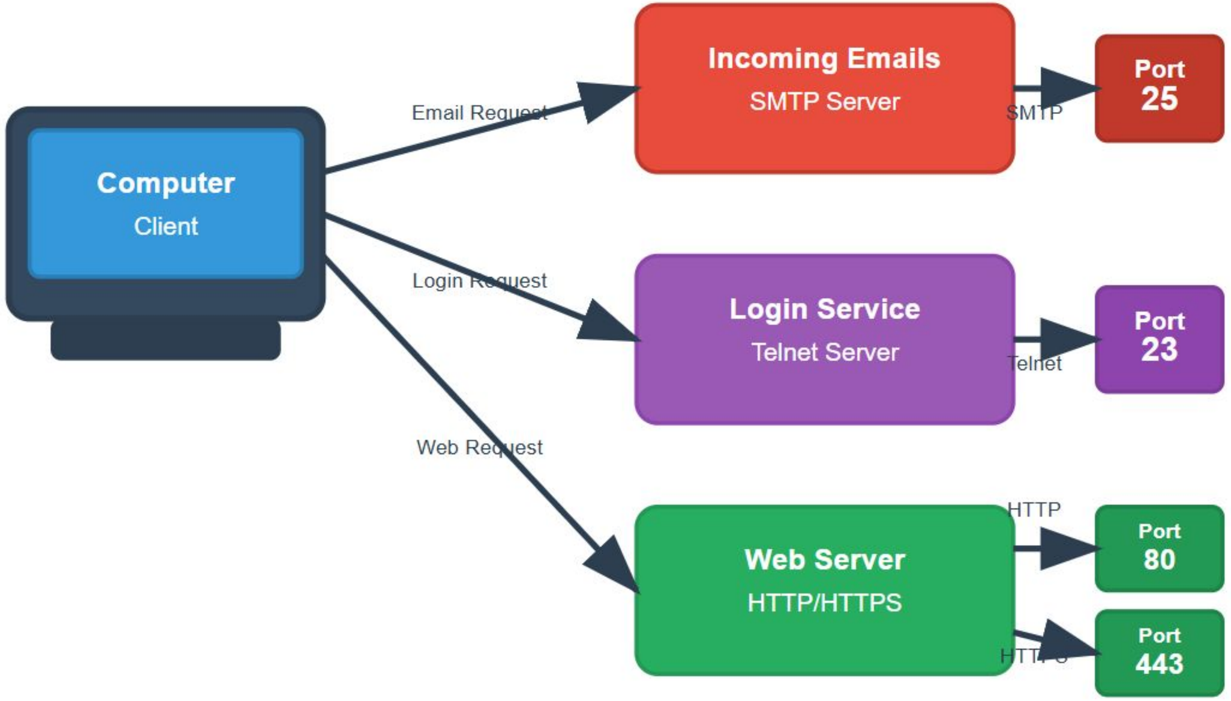

PC
PC
Computer 1
Computer 2
Application Layer
Transport Layer
Network
Application Layer
Transport Layer
Talks to a Server
Server Sends the Data back

1 Computer running python is chatting/Communicating with another computer running php etc(100 or 1000 times a second) and we call this connection a <u>SOCKET</u> .

How Sockets Work ?



We need a Webserver (Names and Points) and also which application and there can be different applications on that server are listening on what are called as PORTS .



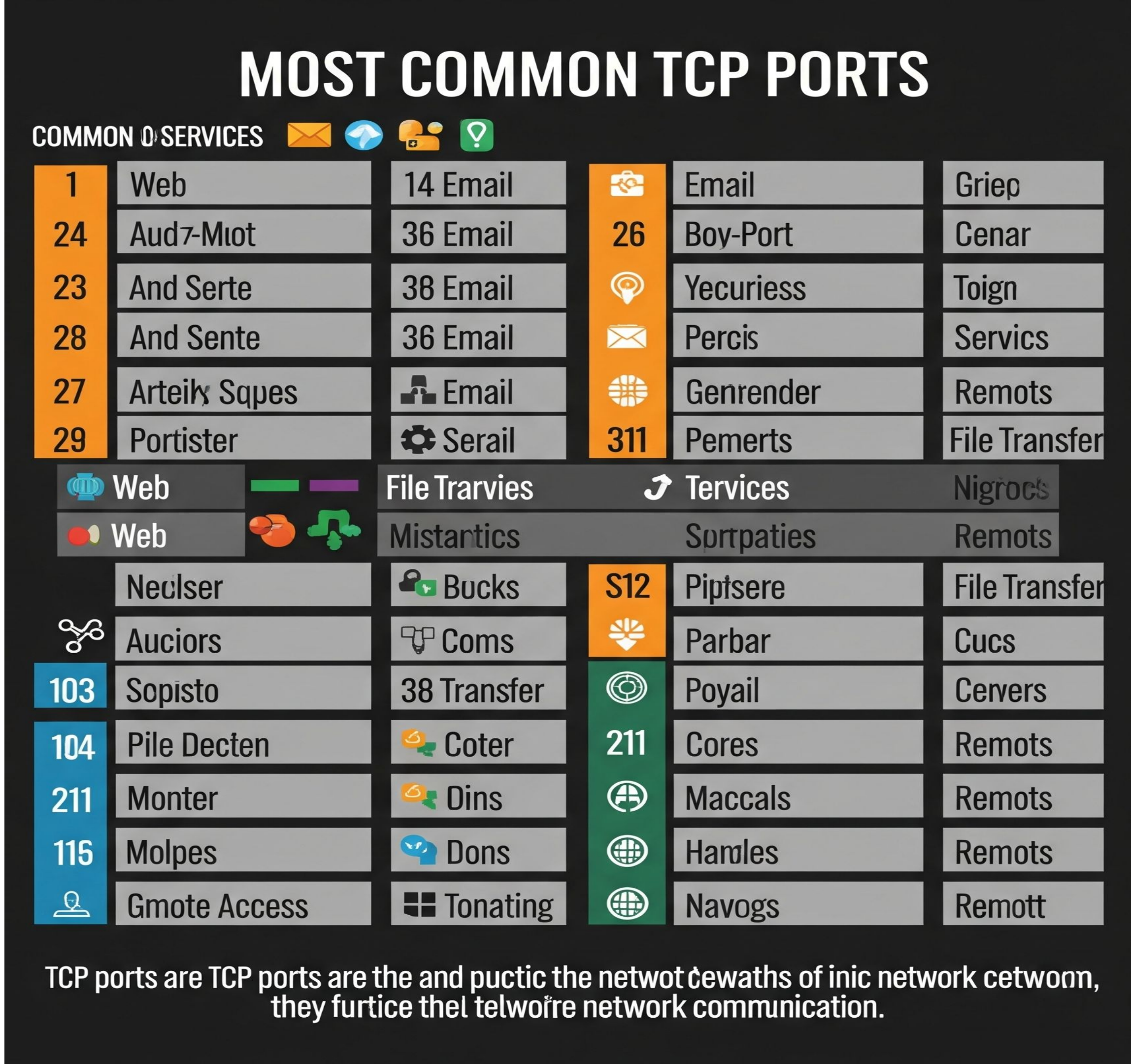
Server Details

Email Service:
- Port 25: SMTP
- Handles incoming mail

Login Service:
- Port 23: Telnet
- Remote login access

Web Server:
- Port 80: HTTP
- Port 443: HTTPS
- Web content delivery

Common TCP Ports :-

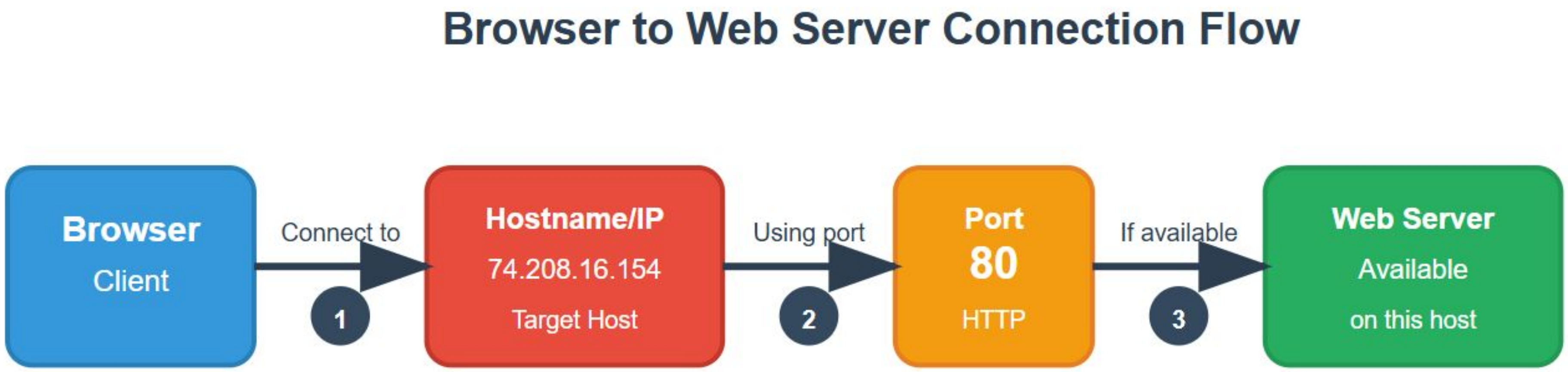


MOST COMMON TCP PORTS

COMMON 0 SERVICES

1	Web	14 Email		Email	Griep
24	Aud7-Mıot	36 Email	26	Boy-Port	Cenar
23	And Serte	38 Email		Yecuriess	Toign
28	And Sente	36 Email		Percis	Servics
27	Arteik Sqpes	Email		Genrender	Remots
29	Portister	Serail	311	Pemerts	File Transfer
Web		File Trarvies		Tervices	Nigrocs
Web		Mistartics		Sorpaties	Remots
	Neulser	Bucks	S12	Piptsere	File Transfer
	Auciors	Coms		Parbar	Cucs
103	Sopisto	38 Transfer		Poyail	Cervers
104	Pile Decten	Coter	211	Cores	Remots
211	Monter	Dins		Maccals	Remots
115	Molpes	Dons		Hamles	Remots
	Gmote Access	Tonating		Navogs	Remott

TCP ports are TCP ports are the and puctic the netwot cewaths of inic network cetwoon, they furtice thel telwoṙe network communication.



Eg :- `http://localhost:8080/`

How to Link , Network Layer , Transport Layer , Entire Internet , Some Server on Other Side with Data and we want to talk to it .

How to Link , Network Layer , Transport Layer , Entire Internet , Some Server on Other Side with Data and we want to talk to it .

It Takes only 3 Lines of Code .

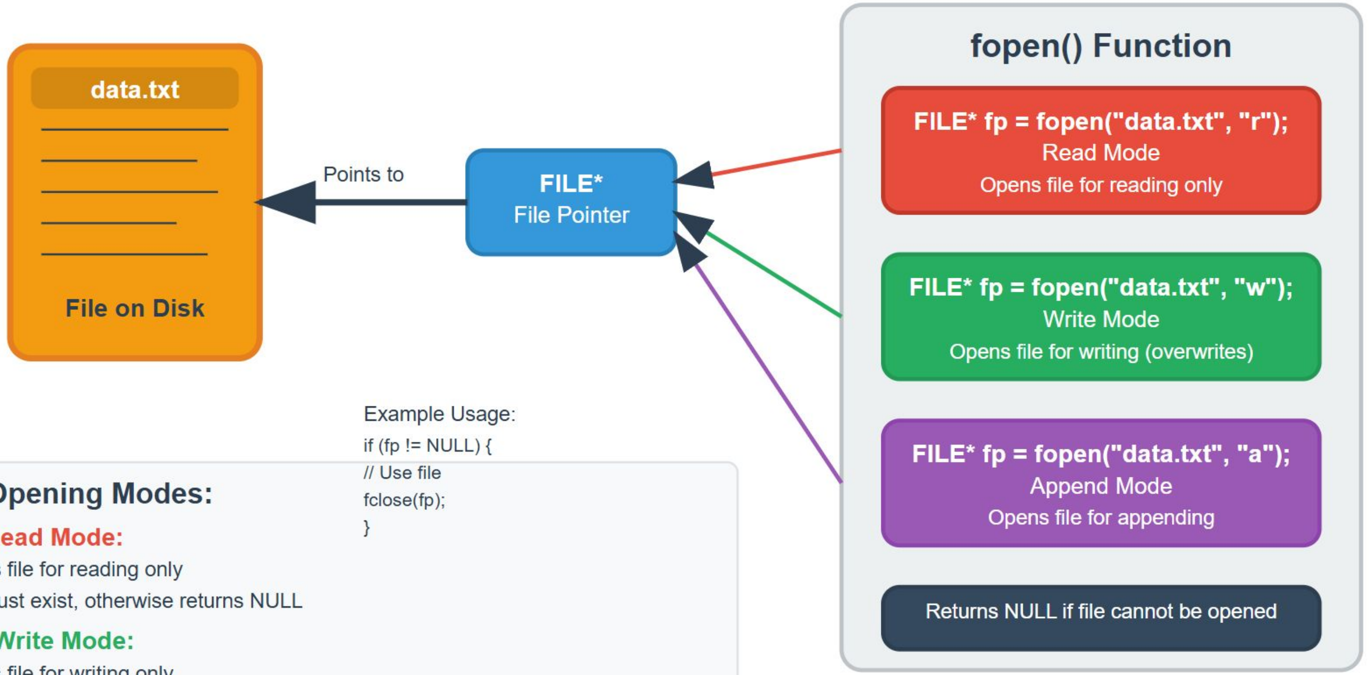
Using a Library Known as “Socket”

```python
import socket

mysock = socket.socket(socket.AF_INET,
socket.SOCK_STREAM)
mysock.connect(('icio.us', 80))
```

Refer to :- https://docs.python.org/3/library/socket.html

Similar to File handling in C++


data.txt
File on Disk
Points to
FILE*
File Pointer
fopen() Function
FILE* fp = fopen("data.txt", "r");
Read Mode
Opens file for reading only
FILE* fp = fopen("data.txt", "w");
Write Mode
Opens file for writing (overwrites)
FILE* fp = fopen("data.txt", "a");
Append Mode
Opens file for appending
Returns NULL if file cannot be opened

Example Usage:

```
if (fp != NULL) {
// Use file
fclose(fp);
}
```

File Opening Modes:

"r" - Read Mode:

- Opens file for reading only
- File must exist, otherwise returns NULL

"w" - Write Mode:

- Opens file for writing only
- Creates new file if it doesn't exist
- Overwrites existing file content

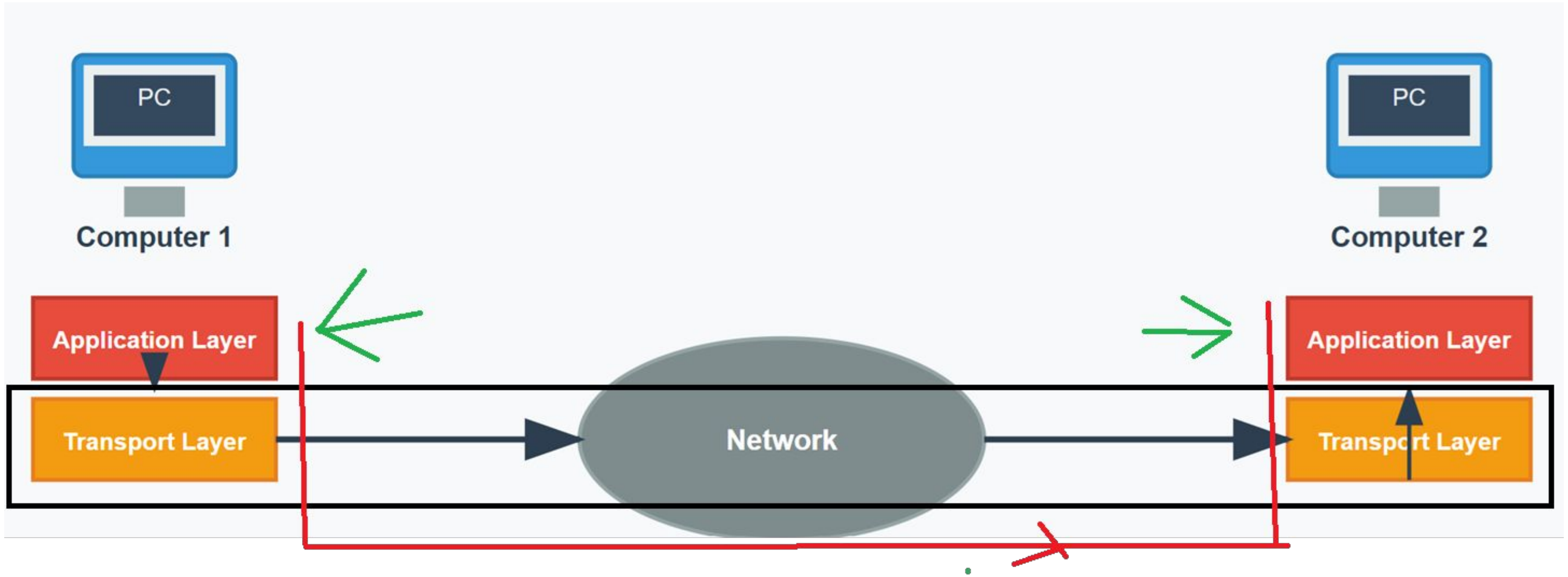
"a" - Append Mode:

- Opens file for writing at the end

Run a python Program and made a connection with the socket then connect it to a particular port on a far away computer then we can start sending data back and forth (i.e Connection)

This Moves up Back to Application Layer

And in Application Layer there are some rules .



In Application Layer we Follow Certain Protocols.

Protocols :- Set to Rules that all parties follow so we can predict each other's behavior.

Eg :-

1. Security Checkup in Airport / IIT Bhilai Gate.
2. Saying “Hello” 1st Before Start the Conversation in phone .
3. We Show indicator to either go left or right in any vehicle.

In This Segment the Protocol that we are going to be Following is HTTP(Hypertext Transfer Protocol)

Invented to Retrieve HTML , Documents , Images etc

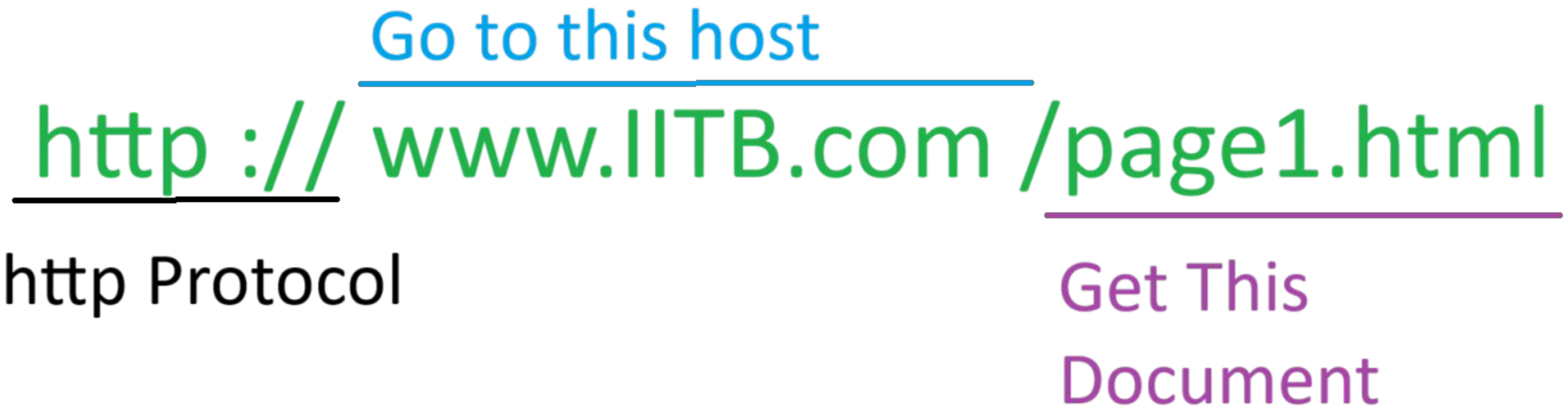
HTTP`s Hyperlink is its most powerful property .

HTTP is set of Rules that to allow browsers to retrieve web documents from servers over the internet .

HTTP have Standardized one of the things was protocol of uniform Resource Locators or URLs
They Contain Some Information Inside them.

http :// www.IITB.com /page1.html

HTTP have Standardized one of the things was protocol of uniform Resource Locators or URLs They Contain Some Information Inside them.



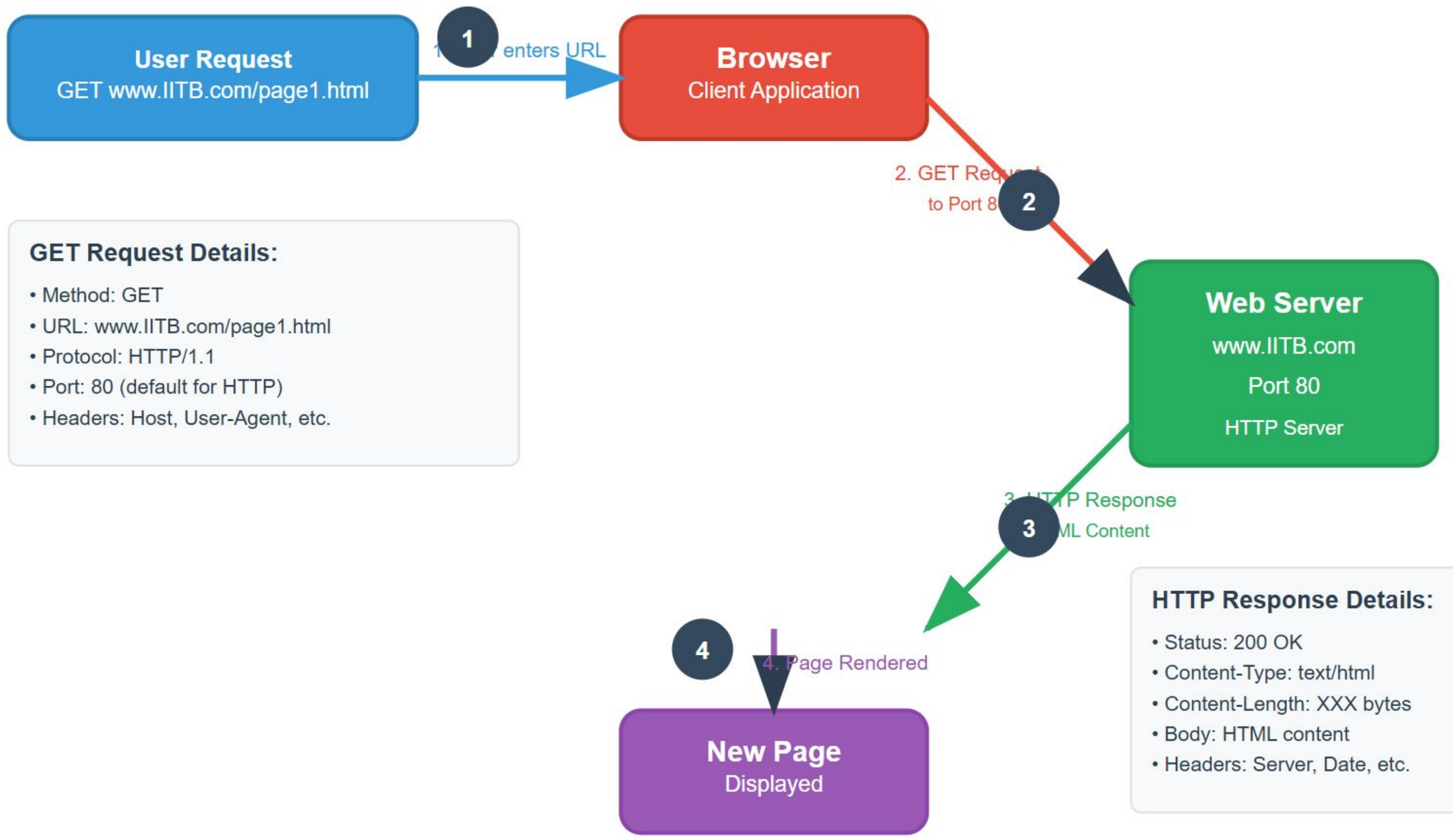
Each Time User Clicks on an anchor tag with an “href =” value to switch to a new page , The Browser makes a Connection to a web server and issues a “GET” request- to GET the content of the page at the specified URL .

Remember :- HTTP`s Hyperlink is its most powerful property .

Everytimes user click a link it gets it to a different page (href) i.e Hypertext.

The server returns the HTML Document to the browser which formats and displays the document to the user .

Request-Response Cycle



GET Request Details:

- Method: GET
- URL: www.IITB.com/page1.html
- Protocol: HTTP/1.1
- Port: 80 (default for HTTP)
- Headers: Host, User-Agent, etc.

HTTP Response Details:

- Status: 200 OK
- Content-Type: text/html
- Content-Length: XXX bytes
- Body: HTML content
- Headers: Server, Date, etc.

Eg :- (Only works in Ubuntu)

```
Telnet icio.us 80
GET http://icio.us/ HTTP/1.0
```

How to make an HTTP Request

GET http://IITB.com HTTP/1.0

How to Retrieve a Page like Browser

```
import socket

mysock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
mysock.connect(('icio.us', 80))

cmd = 'GET http://icio.us/ HTTP/1.0\r\n\r\n'.encode()
```

Request

```
#Send it to server .
mysock.send(cmd)

while True:
    data = mysock.recv(512)
    if len(data) < 1:
        break
    print(data.decode(),end='')
```

Receive 1st 512 Characters .

```
mysock.close()
```

Close the Connection.

Open a Socket , Send Command , Retreive a data , close the socket

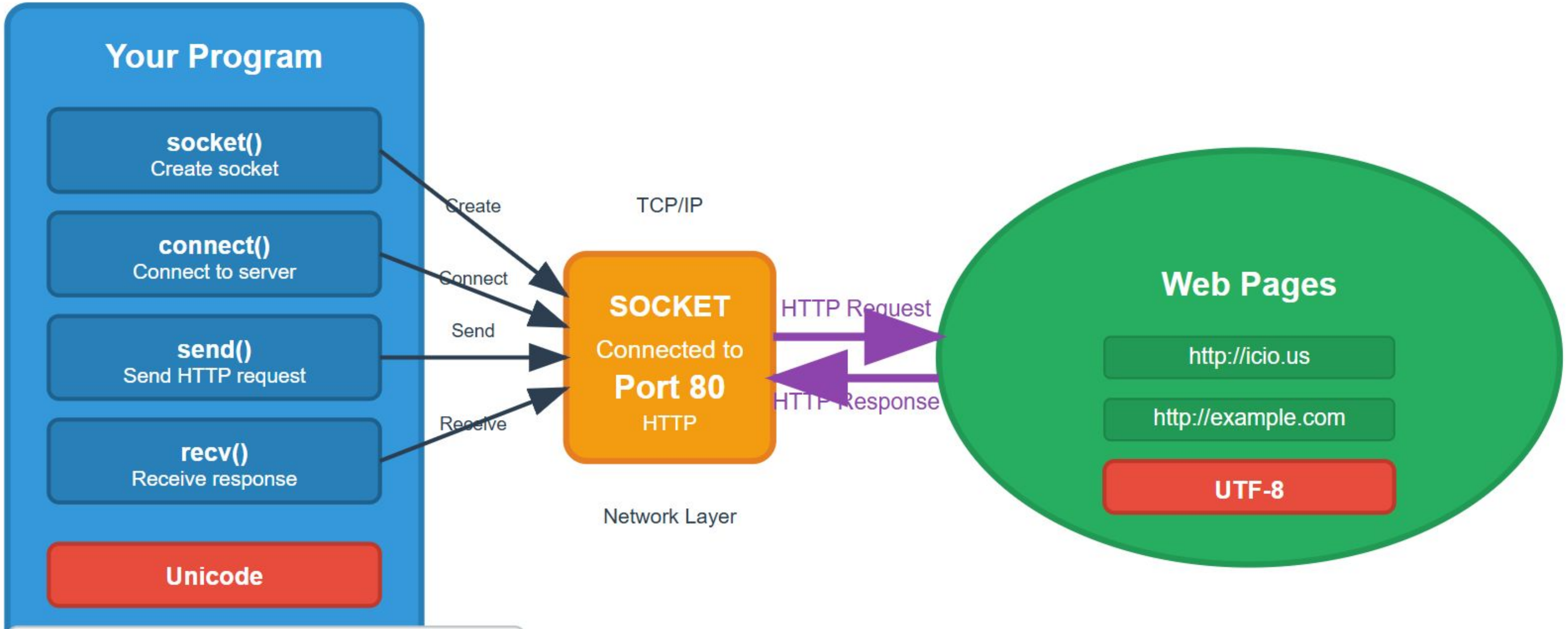
How to Use Developer Console ?

200 :- OK

404 :- Error

302 :- Found / Moved Temporarily.

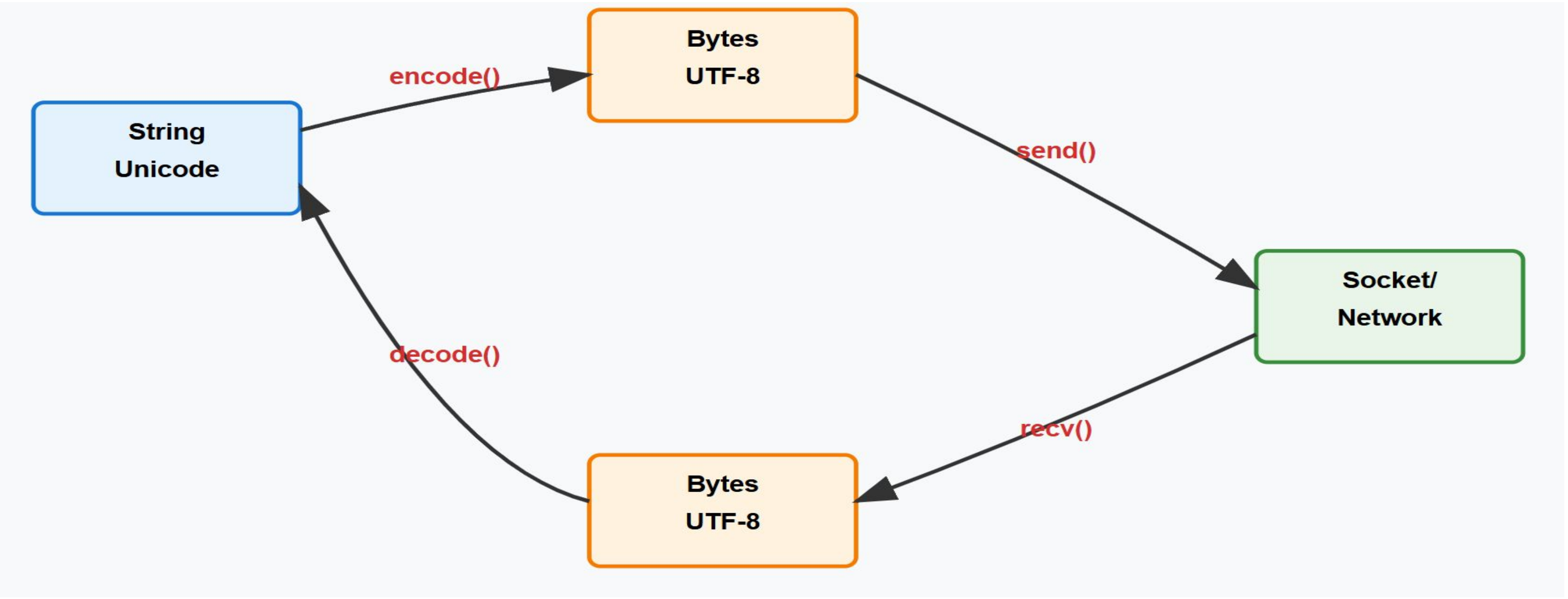
What is Encode and Decode ?



Example HTTP Request:

```
GET /page.html HTTP/1.1
Host: icio.us
User-Agent: MyProgram/1.0
```

What is Unicode and UTF-8 ?

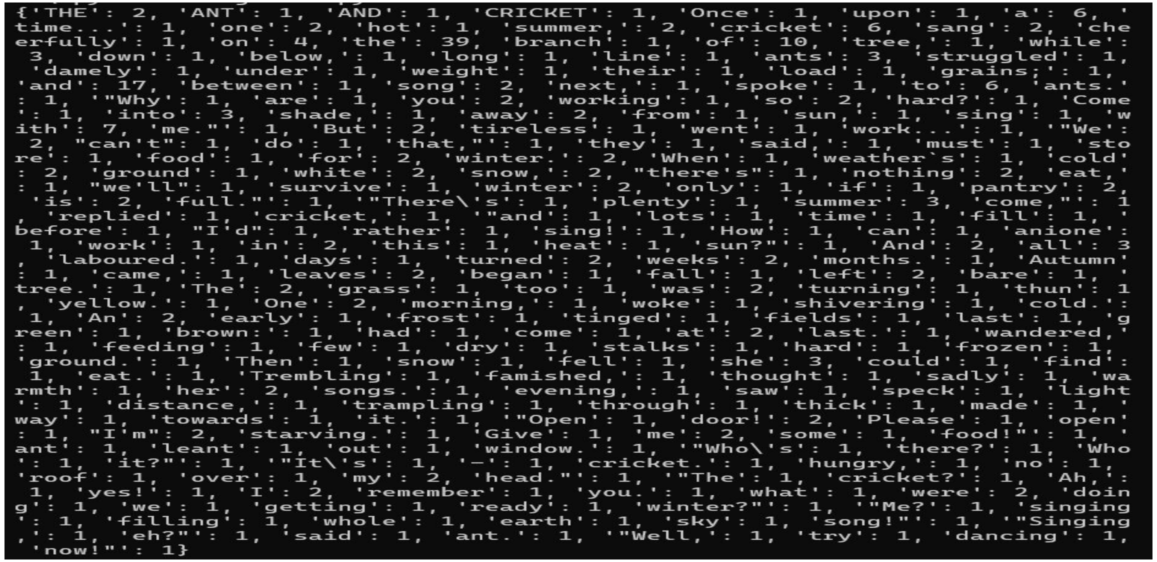


Retrieving Webpage (More Easy) Using urllib

Library that wraps our socket stuff and does all automatically .

Assignment 1

Treat it like a file and find no.of count of words in the online file .



```
{'THE': 2, 'ANT': 1, 'AND': 1, 'CRICKET': 1, 'Once': 1, 'upon': 1, 'a': 6, '
time...': 1, 'one': 2, 'hot': 1, 'summer,': 2, 'cricket': 6, 'sang': 2, 'che
erfully': 1, 'on': 4, 'the': 39, 'branch': 1, 'of': 10, 'tree,': 1, 'while':
 3, 'down': 1, 'below,': 1, 'long': 1, 'line': 1, 'ants': 3, 'struggled': 1,
 'damely': 1, 'under': 1, 'weight': 1, 'their': 1, 'load': 1, 'grains;': 1,
'and': 17, 'between': 1, 'song': 2, 'next,': 1, 'spoke': 1, 'to': 6, 'ants.'
: 1, '"Why': 1, 'are': 1, 'you': 2, 'working': 1, 'so': 2, 'hard?': 1, 'Come
': 1, 'into': 3, 'shade,': 1, 'away': 2, 'from': 1, 'sun,': 1, 'sing': 1, 'w
ith': 7, 'me."': 1, 'But': 2, 'tireless': 1, 'went': 1, 'work...': 1, '"We':
 2, "can't": 1, 'do': 1, 'that,"': 1, 'they': 1, 'said,': 1, 'must': 1, 'sto
re': 1, 'food': 1, 'for': 2, 'winter.': 2, 'When': 1, 'weather`s': 1, 'cold'
: 2, 'ground': 1, 'white': 2, 'snow,': 2, "there's": 1, 'nothing': 2, 'eat,'
: 1, "we'll": 1, 'survive': 1, 'winter': 2, 'only': 1, 'if': 1, 'pantry': 2,
 'is': 2, 'full."': 1, '"There\'s': 1, 'plenty': 1, 'summer': 3, 'come,"': 1
, 'replied': 1, 'cricket,': 1, '"and': 1, 'lots': 1, 'time': 1, 'fill': 1, '
before': 1, "I'd": 1, 'rather': 1, 'sing!': 1, 'How': 1, 'can': 1, 'anione':
 1, 'work': 1, 'in': 2, 'this': 1, 'heat': 1, 'sun?"': 1, 'And': 2, 'all': 3
, 'laboured.': 1, 'days': 1, 'turned': 2, 'weeks': 2, 'months.': 1, 'Autumn'
: 1, 'came,': 1, 'leaves': 2, 'began': 1, 'fall': 1, 'left': 2, 'bare': 1, '
tree.': 1, 'The': 2, 'grass': 1, 'too': 1, 'was': 2, 'turning': 1, 'thun': 1
, 'yellow.': 1, 'One': 2, 'morning,': 1, 'woke': 1, 'shivering': 1, 'cold.':
 1, 'An': 2, 'early': 1, 'frost': 1, 'tinged': 1, 'fields': 1, 'last': 1, 'g
reen': 1, 'brown:': 1, 'had': 1, 'come': 1, 'at': 2, 'last.': 1, 'wandered,'
: 1, 'feeding': 1, 'few': 1, 'dry': 1, 'stalks': 1, 'hard': 1, 'frozen': 1,
'ground.': 1, 'Then': 1, 'snow': 1, 'fell': 1, 'she': 3, 'could': 1, 'find':
 1, 'eat.': 1, 'Trembling': 1, 'famished,': 1, 'thought': 1, 'sadly': 1, 'wa
rmth': 1, 'her': 2, 'songs.': 1, 'evening,': 1, 'saw': 1, 'speck': 1, 'light
': 1, 'distance,': 1, 'trampling': 1, 'through': 1, 'thick': 1, 'made': 1, '
way': 1, 'towards': 1, 'it.': 1, '"Open': 1, 'door!': 2, 'Please': 1, 'open'
: 1, "I'm": 2, 'starving.': 1, 'Give': 1, 'me': 2, 'some': 1, 'food!"': 1, '
ant': 1, 'leant': 1, 'out': 1, 'window.': 1, '"Who\'s': 1, 'there?': 1, 'Who
': 1, 'it?"': 1, '"It\'s': 1, '—': 1, 'cricket.': 1, 'hungry,': 1, 'no': 1,
'roof': 1, 'over': 1, 'my': 2, 'head."': 1, '"The': 1, 'cricket?': 1, 'Ah,':
 1, 'yes!': 1, 'I': 2, 'remember': 1, 'you.': 1, 'what': 1, 'were': 2, 'doin
g': 1, 'we': 1, 'getting': 1, 'ready': 1, 'winter?"': 1, '"Me?': 1, 'singing
': 1, 'filling': 1, 'whole': 1, 'earth': 1, 'sky': 1, 'song!"': 1, '"Singing
,': 1, 'eh?"': 1, 'said': 1, 'ant.': 1, '"Well,': 1, 'try': 1, 'dancing': 1,
 'now!"': 1}
```

Expected Output.

Web Scrapping ?

Now that we have this protocol with which we can send GET Request and get data back .

Web-Scraping is Pretending to be a Browser .
To Retreive Web Pages , Extract information , and then look at more Web pages .

What ?? Why ?? Legal ? Problem ?

Why ?

Spider the web to make a database for a search engine. (Like Google)

Pull data particularly social data - To see which links with which ?

Problem :- Parsing of HTML that comes Back .

Solution ?

Beautifulsoup from www.crummy.com

Installation :-

1. From Site
2. Have bs4 Folder on Same Directory

Code of Beautifulsoup

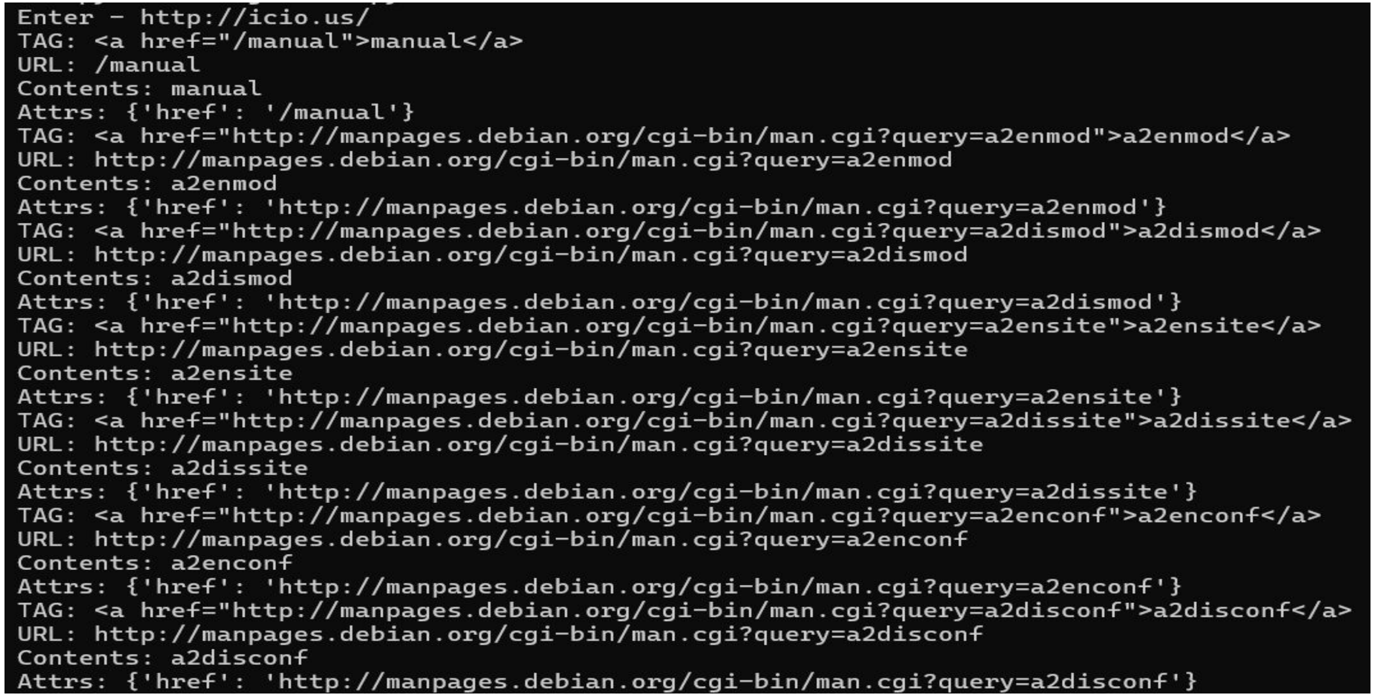
```
import urllib.request
from bs4 import BeautifulSoup

url = input('Enter - ')
html = urllib.request.urlopen(url).read()
soup = BeautifulSoup(html, "html.parser")

tags = soup('a')
for tag in tags:
    print('URL:', tag.get('href', None))
```

Assignment - 2

Retrieve Tags , Attributes and Text .



```
Enter - http://icio.us/
TAG: <a href="/manual">manual</a>
URL: /manual
Contents: manual
Attrs: {'href': '/manual'}
TAG: <a href="http://manpages.debian.org/cgi-bin/man.cgi?query=a2enmod">a2enmod</a>
URL: http://manpages.debian.org/cgi-bin/man.cgi?query=a2enmod
Contents: a2enmod
Attrs: {'href': 'http://manpages.debian.org/cgi-bin/man.cgi?query=a2enmod'}
TAG: <a href="http://manpages.debian.org/cgi-bin/man.cgi?query=a2dismod">a2dismod</a>
URL: http://manpages.debian.org/cgi-bin/man.cgi?query=a2dismod
Contents: a2dismod
Attrs: {'href': 'http://manpages.debian.org/cgi-bin/man.cgi?query=a2dismod'}
TAG: <a href="http://manpages.debian.org/cgi-bin/man.cgi?query=a2ensite">a2ensite</a>
URL: http://manpages.debian.org/cgi-bin/man.cgi?query=a2ensite
Contents: a2ensite
Attrs: {'href': 'http://manpages.debian.org/cgi-bin/man.cgi?query=a2ensite'}
TAG: <a href="http://manpages.debian.org/cgi-bin/man.cgi?query=a2dissite">a2dissite</a>
URL: http://manpages.debian.org/cgi-bin/man.cgi?query=a2dissite
Contents: a2dissite
Attrs: {'href': 'http://manpages.debian.org/cgi-bin/man.cgi?query=a2dissite'}
TAG: <a href="http://manpages.debian.org/cgi-bin/man.cgi?query=a2enconf">a2enconf</a>
URL: http://manpages.debian.org/cgi-bin/man.cgi?query=a2enconf
Contents: a2enconf
Attrs: {'href': 'http://manpages.debian.org/cgi-bin/man.cgi?query=a2enconf'}
TAG: <a href="http://manpages.debian.org/cgi-bin/man.cgi?query=a2disconf">a2disconf</a>
URL: http://manpages.debian.org/cgi-bin/man.cgi?query=a2disconf
Contents: a2disconf
Attrs: {'href': 'http://manpages.debian.org/cgi-bin/man.cgi?query=a2disconf'}
```

Expected Output :-