```
"Network Programming"
```

In [ ]:
```
"Objective:"
```

In this module,we will gain a comprehensive understanding of network programming,including client and server concepts using,By the end of this guide,we will equippped with the knowledge to handle network connetions ,transmit data,and build simple client-server applications.

In [ ]:
```
"1.Terms and Basic of Network Programming"
```

Network programming involves writting code that allows your softaware to communicate with other software over a network .It includes sending and receiving data across networks like the internet or a local area network (LAN)

```
"1.IP address:"->A unique string of numbers seperated by peroids that identifies each computer
using the internet Protocol to communcate over a network.
```

Example: 192.168.1.1 (IPv4) or 2001:0db8:85a3:0000:0000:8a2e:0370:7334 (IPv6). Function: It defines where a device is located on the network.

```
"1.Port:" Definition: A port is a number that identifies a specific process or service on a
device. It ensures data is sent to the right application.
```

Example: 80 (HTTP), 443 (HTTPS), 3306 (MySQL). Function: Ports help differentiate services running on the same IP address, like how different doors (ports) give access to different rooms (services) inside a building (device).

```
 3. Socket:: A socket is a combination of an IP address and a port number. It uniquely
identifies a specific connection between two devices.
```

Example: 192.168.1.1:8080. Function: It establishes an endpoint for communication between two systems. The socket is essential for sending/receiving data over a network connection.

```
  4. Protocol:: A protocol is a set of rules governing how data is transmitted over a network.
It defines how devices communicate and interpret the data.
```

Example: TCP (Transmission Control Protocol), UDP (User Datagram Protocol), HTTP. Function: Protocols ensure that communication happens correctly and data is exchanged in a reliable, ordered manner.

```
 IP Address: Identifies a device on a network.

 Port: Specifies a service on a device.

  Socket: Combines IP and port to establish a unique connection.

Protocol: Governs how data is transferred between devices.
```

In [ ]:
```
"The Architecture of Data Transmission between sender and receiver Using "
```

In network programming ,data transmission between a sender and receiver is typically done using sockets.

```
*Socket:->The socket library in provides low-level access to network interfaces ,allowing you
to send and receive data over a network.
```

The architecture generally involves:

```
 * Client:->The clients initiates a connection to the server and requests services or
resources.

 * Server:-> The server listens for incoming data on a specific port ,processes requests from
the client and send the appropriate responses.
```

In [ ]:
```
"Example Architecture"
```

```
 *1.Client:-> Connects to the server's IP address and Port

  *2.Server:-> Accepts the connection ,process the                request ,and send back a
response.


 "Client:"
```

A client is device or application that initiates a request for data or services from the server.It relies on the server to provide resources, such as files, web pages, or computational tasks.

```
 Examples:
```

1.Web browsers (e.g., Chrome, Firefox) act as clients when requesting web pages.

2.A desktop email client (e.g., Outlook) requests email from a mail server.

3.A MySQL client sends queries to a MySQL database server.

```
    2. Server:
```

Definition: A server is a device or application that provides resources or services to clients upon their request. It listens for client requests and responds by providing the required data or service. Examples: A web server (e.g., Apache, Nginx) delivers web pages to browsers. A mail server stores and delivers emails. A database server (e.g., MySQL server) processes and responds to database queries. Function: The server waits for client requests, processes them, and sends back the requested data or performs the necessary task. Role: Provider of services/resources

In [ ]:
```
"Getting Data from the Remote Server"
```

```
 we can retreive the data from a remote server using network sockets or high-level libraries
like "requests" for 'HTTP' comminication
```

In [ ]:
```
Example :-> Retrieving Data Using Sockets
```

In [114…
```python
import socket

# Create a socket object
client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

# Define the server address and port
server_address = ('www.github.com', 80)

# Connect to the server
client_socket.connect(server_address)

# Send a GET request to the server
request = "GET / HTTP/1.1\r\nHost: www.github.com\r\n\r\n"
client_socket.sendall(request.encode())

# Receive the response from the server
response = client_socket.recv(4096)

# Print the server response
print(response.decode())

# Close the socket
client_socket.close()
```

```
HTTP/1.1 301 Moved Permanently
Content-Length: 0
Location: https://www.github.com/
```

In [116…
```python
import socket

# Create a socket object
client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

# Define the server address and port
server_address = ('www.example.com', 80)

# Connect to the server
```

```python
client_socket.connect(server_address)

# Send a GET request to the server
request = "GET / HTTP/1.1\r\nHost: www.example.com\r\n\r\n"
client_socket.sendall(request.encode())

# Receive the response from the server
response = client_socket.recv(4096)

# Print the server response
print(response.decode())

# Close the socket
client_socket.close()
```

```
HTTP/1.1 200 OK
Accept-Ranges: bytes
Age: 390371
Cache-Control: max-age=604800
Content-Type: text/html; charset=UTF-8
Date: Tue, 17 Sep 2024 07:09:50 GMT
Etag: "3147526947+gzip"
Expires: Tue, 24 Sep 2024 07:09:50 GMT
Last-Modified: Thu, 17 Oct 2019 07:18:26 GMT
Server: ECAcc (nyd/D157)
Vary: Accept-Encoding
X-Cache: HIT
Content-Length: 1256

<!doctype html>
<html>
<head>
    <title>Example Domain</title>

    <meta charset="utf-8" />
    <meta http-equiv="Content-type" content="text/html; charset=utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <style type="text/css">
    body {
        background-color: #f0f0f2;
        margin: 0;
        padding: 0;
        font-family: -apple-system, system-ui, BlinkMacSystemFont, "Segoe UI", "Open Sans", "Helvetica Neue", He
lvetica, Arial, sans-serif;

    }
    div {
        width: 600px;
        margin: 5em auto;
        padding: 2em;
        background-color: #fdfdff;
        border-radius: 0.5em;
        box-shadow: 2px 3px 7px 2px rgba(0,0,0,0.02);
    }
    a:link, a:visited {
        color: #38488f;
        text-decoration: none;
    }
    @media (max-width: 700px) {
        div {
            margin: 0 auto;
            width: auto;
        }
    }
    </style>
</head>

<body>
<div>
    <h1>Example Domain</h1>
    <p>This domain is for use in illustrative examples in documents. You may use this
    domain in literature without prior coordination or asking for permission.</p>
    <p><a href="https://www.iana.org/domains/example">More information...</a></p>
</div>
</body>
</html>
```

In [ ]:

```python
"Example: Retrieving Data Using equests Library"
```

In [3]:

```python
import requests

#send a GET requests to the server
response=requests.get('https://linkedin.com/')
```

```
#print the response content
print(response.text)
```

Here we can get indetals inspects of linkedin application

In [ ]:
```
"4.Client &Server-Side Programming
```

Network communication typically involves a client-server model, where the client sends a request to the server, and the server processes the request and returns a response.

In [ ]:
```
"4.1. Client-Side Programming"
```

The client-side program connects to the server, sends requests, and handles responses.

In [ ]:
```
"Example: Simple Client Program"
```

In [ ]:
```
import socket

#create a socket object
client_socket=socket.socket(socket.AF_INET,socket.SOCK_STREAM)

#define the server address and port

server_address=('localhost',65432)

#connect to the server

client_socket.connect(server_address)

try:
    #send data to the server
    message='Hello ,Server!'
    client_socket.sendall(message.encode())
    #receive the servers response
    data=client_socket.recv(1024)
    print(f"Received:{data.decode()}")
finally:
    #close the connection
    client_socket.close()
```

In [ ]:
```
"Server=Side Programming"
```

The server-side program listens for incoming connections ,procesess client requests and sends responses

In [ ]:
```
"Simple server Program"
```

In [1]:
```
import socket

#create  a socket object
server_socket=socket.socket(socket.AF_INET,socket.SOCK_STREAM)

#Bind the socket to an address and port

server_address=('localhost',8888)
#Listen for incoming connections
server_socket.listen()
print("Server is listening..")
while True:
    #Accept the connection
    client_socket,client_address=server_socket.accept()
    try:
        print(f"Connection from {client_address}")
        #recive the data
        data=client_socket.recv(1024)
        print(f"Received:{data.decode()}")
        #send response
        response='Hello,Client'
        client_socket.sendall(response.encode())
    finally:
        #close the connection
        client_socket.close()
```

```
---------------------------------------------------------------------------
OSError                                   Traceback (most recent call last)
Cell In[1], line 10
      8 server_address=('localhost',8888)
      9 #Listen for incoming connections
---> 10 server_socket.listen()
     11 print("Server is listening..")
     12 while True:
     13     #Accept the connection

OSError: [WinError 10022] An invalid argument was supplied
```

In [ ]:
```
import socket

# Create a TCP/IP socket
server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

# Bind the socket to the address and port
server_address = ('localhost', 8888)
server_socket.bind(server_address)

# Listen for incoming connections
server_socket.listen(5)  # Set the backlog to 5
print("Server is listening...")

while True:
    # Accept the connection
    client_socket, client_address = server_socket.accept()
    print(f"Connection established with {client_address}")
    # Handle the connection
    # ...
```

In [ ]:
```
"5. Hands-on: Using Networking Module in "
```

In , networking is commonly handled using the socket module, which provides access to the BSD socket interface. This module allows you to work with both TCP and UDP protocols to create clients and servers.

In [ ]:
```
"5.1. Data Transmission Between Client and Server"
```

Using the socket module, you can easily transmit data between a client and a server.

```
    •    TCP (Transmission Control Protocol): A connection-oriented protocol that ensures data
is reliably delivered in the correct order.

•       UDP (User Datagram Protocol): A connectionless protocol that is faster but does not
guarantee data delivery or order.
```

Example: TCP Client-Server Communication

Refer to the client and server examples provided above.

```
Key Concepts Recap:

  •     Sockets: Fundamental for network programming, used to establish a connection between a
client and server.

  •     IP Address and Ports: Essential for identifying machines and services on a network.

    •   Client-Server Model: A common network architecture where clients request services, and
servers provide them.
```

In [ ]:
```

```

```
               "Regex with Python"


 "Objective:"
 Learn how to use and write Regular Expressions(Regex) in Python


* Regex Syntax:
   1.Rgex (Regular Expressions) are patterns used to match character combinations in strings.
```

- Basic Syntax:

1. .:Matches any character except a newline

2. ^:Matches the start of the string.
3. $:Matches the end of the string.
4. :Escapes special characters.

- Quantifiers:

1. *:Matches 0 or more repetitions.
2. +:Matches 1 or more repetitions.
3. ?:Matches 0 or 1 occurence.
4. {n}:Matches exactly n occurences.
5. {n,}:Matches n or more occurences

In [ ]:
```
*Metacharacters
```

1. []: Defines a set of characters to match.
2. | : Acts like a logical OR.
3. (): Groups Expressions and captures matched text.
4. \b:Word boundary.
5. \d:Matches any digit (euqilent to [0-9])

In [ ]:
```
* Special Characters
```

1. \s: Matches any white space characters.
2. \S: Matches any non-white space characters.

In [ ]:
```
*Sets:
```

1. [a-z]: Mactches any lowercase letter.
2. [A-Z]: Matches any uppercase letter.
3. [0-9]: Matches any digits.

In [ ]:
```
*Python Module:
```

```
1. import re: To use regex in Python
```

In [ ]:
```
*Common Methods
```

1. re.match(): Determine if the regex matches at the beginning of a string.
2. re.search():Searches for the regex pattern in the string.
3. re.findalll(): Returns all matches of the pattern.
4. re.sub(): Replaces matches with a string.

In [ ]:
```
"Examples"
```

```
1.re.matach()
```

re.match() checks if the regec pattern matches at the 'beggining' of the string.

In [28]:
```
import re

pattern=r'Python'
text="Python is powerfull language."
match=re.match(pattern,text)
if match:
    print("Match found:",match.group())
else:
    print("No  match")
```

```
Match found: Python
```

In [44]:
```
import re

pattern=r'is'
text="Python is powerfull language."
match=re.match(pattern,text)
if match:
    print("Match found:",match.group())
```

```
else:
    print("No match")
```

```
No match
```

Explanation: • re.match() returns a match only if the pattern is found at the start of the string. If the pattern is elsewhere, it will return None.

```
2.re.search()
```

re.search() looks for the regex pattern 'anywhere ' in the string.

In [26]:
```
import re

pattern=r'Powerfull'

text="Python is very Powerfull language"

search=re.search(pattern,text)

if search:
    print("Search found:",search.group())
else:
    print("Not found")
```

```
Search found: Powerfull
```

In [40]:
```
import re

pattern=r'ower'

text="Python is very Powerfull language"

search=re.search(pattern,text)

if search:
    print("Search found:",search.group())
else:
    print("Not found")
```

```
Search found: ower
```

In [46]:
```
import re

pattern=r'Pfull'

text="Python is very Powerfull language"

search=re.search(pattern,text)

if search:
    print("Search found:",search.group())
else:
    print("Not found")
```

```
Not found
```

Explanation: • re.search() will return the first match it finds anywhere in the string, unlike re.match().

```
3.re.findall()
```

re.findall() returns 'all' matches of the pattern in the string as a list.

In [59]:
```
import re

pattern=r'\b[a-zA-Z]{3}\b' #matches any 3-letter word
text="The cat sat on the mat"
matches=re.findall(pattern,text)
print(matches)
```

```
['The', 'cat', 'sat', 'the', 'mat']
```

In [65]:
```
import re

pattern=r'\b[a-zA-Z]{3,}\b' #contains 3 or more letters.
text="T catis sat on the mat"
matches=re.findall(pattern,text)
print(matches)
```

```
['catis', 'sat', 'the', 'mat']
```

4.re.sub()

re.sub() is used to 'replace' all occurences of the regex pattern in the string with a replacement string.

In [73]:
```
import re

pattern ="cat|dog"
text="I have cat and a dog"

replaced_text=re.sub(pattern,"pet",text)
print(replaced_text)
```

```
I have pet and a pet
```

In [75]:
```
import re

pattern ="dog"
text="I have cat and a dog"

replaced_text=re.sub(pattern,"pet",text)
print(replaced_text)
```

```
I have cat and a pet
```

Explanation: • re.sub() replaces all matches of the pattern with the given replacement string. Here, both "cat" and "dog" are replaced by "pet".

5.Using re.match(),re.search(),re.findall(),re.sub() together

In [91]:
```
import re

text="I am vasu and My email is srinivasulu@gamil.com. Another email is gadamvasu@gmail.com"
#re.match()
pattern_match=r'I'
match=re.match(pattern_match,text)

print("Macth found:",match.group() if match else "No match")

#re.search()
pattern_search=r'\w+@\w+\.\w+'
search=re.search(pattern_search,text)
print("Search found:",search.group() if search else "No match")

#re.findall()

pattern_findall=r'\w+@\w+\.\w+'
findall=re.findall(pattern_findall,text)
print("Findall result:",findall)

#re.sub()

pattern_sub=r'\w+@\w+\.\w+'
replaced_text=re.sub(pattern_sub,"[email]",text)

print("Replaced text:",replaced_text)
```

```
Macth found: I
Search found: srinivasulu@gamil.com
Findall result: ['srinivasulu@gamil.com', 'gadamvasu@gmail.com']
Replaced text: I am vasu and My email is [email]. Another email is [email]
```

In [ ]:
```
```

"API(Application Programming Interface) access with Python"

In [ ]:
```
Objective : Learn how we access open APIs using Python
```

"* Gooogle Text to Speech

In [1]:
```
!pip install gtts
```

```
Collecting gtts
  Downloading gTTS-2.5.3-py3-none-any.whl.metadata (4.1 kB)
Requirement already satisfied: requests<3,>=2.27 in c:\users\gadam\anaconda3\lib\site-packages (from gtts) (2.32
.2)
Requirement already satisfied: click<8.2,>=7.1 in c:\users\gadam\anaconda3\lib\site-packages (from gtts) (8.1.7)
Requirement already satisfied: colorama in c:\users\gadam\anaconda3\lib\site-packages (from click<8.2,>=7.1->gtt
s) (0.4.6)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\gadam\anaconda3\lib\site-packages (from requ
ests<3,>=2.27->gtts) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\gadam\anaconda3\lib\site-packages (from requests<3,>=2.2
7->gtts) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\gadam\anaconda3\lib\site-packages (from requests<3
,>=2.27->gtts) (1.26.19)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\gadam\anaconda3\lib\site-packages (from requests<3
,>=2.27->gtts) (2024.6.2)
Downloading gTTS-2.5.3-py3-none-any.whl (29 kB)
Installing collected packages: gtts
Successfully installed gtts-2.5.3
```

In [3]:
```python
from gtts import gTTS
import os
```

In [ ]:
```python
*convert text to speech using gTTs(Google text to Speech)
```

In [103…
```python
from gtts import gTTS
import os

text = """Hi Hello vasu How are you
"""
tts = gTTS(text)
tts.save("output.mp3")
os.system("start output.mp3")
```

Out[103… 0

*Google Speech to TExt

In [27]:
```
pip install SpeechRecognition
```

```
Collecting SpeechRecognition
  Downloading SpeechRecognition-3.10.4-py2.py3-none-any.whl.metadata (28 kB)
Requirement already satisfied: requests>=2.26.0 in c:\users\gadam\anaconda3\lib\site-packages (from SpeechRecogn
ition) (2.32.2)
Requirement already satisfied: typing-extensions in c:\users\gadam\anaconda3\lib\site-packages (from SpeechRecog
nition) (4.11.0)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\gadam\anaconda3\lib\site-packages (from requ
ests>=2.26.0->SpeechRecognition) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\gadam\anaconda3\lib\site-packages (from requests>=2.26.0
->SpeechRecognition) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\gadam\anaconda3\lib\site-packages (from requests>=
2.26.0->SpeechRecognition) (1.26.19)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\gadam\anaconda3\lib\site-packages (from requests>=
2.26.0->SpeechRecognition) (2024.6.2)
Downloading SpeechRecognition-3.10.4-py2.py3-none-any.whl (32.8 MB)
   ---------------------------------------- 0.0/32.8 MB ? eta -:--:--
   ---------------------------------------- 0.1/32.8 MB 3.0 MB/s eta 0:00:12
    --------------------------------------- 0.5/32.8 MB 5.2 MB/s eta 0:00:07
   - -------------------------------------- 0.9/32.8 MB 7.3 MB/s eta 0:00:05
   - -------------------------------------- 1.4/32.8 MB 7.8 MB/s eta 0:00:05
   -- ------------------------------------- 1.8/32.8 MB 8.3 MB/s eta 0:00:04
   -- ------------------------------------- 2.4/32.8 MB 8.4 MB/s eta 0:00:04
   --- ------------------------------------ 2.9/32.8 MB 8.7 MB/s eta 0:00:04
   --- ------------------------------------ 3.2/32.8 MB 8.5 MB/s eta 0:00:04
   ---- ----------------------------------- 3.7/32.8 MB 9.2 MB/s eta 0:00:04
   ---- ----------------------------------- 4.0/32.8 MB 8.6 MB/s eta 0:00:04
   ----- ---------------------------------- 4.5/32.8 MB 8.7 MB/s eta 0:00:04
   ----- ---------------------------------- 4.8/32.8 MB 8.8 MB/s eta 0:00:04
   ------ --------------------------------- 5.3/32.8 MB 8.7 MB/s eta 0:00:04
   ------ --------------------------------- 5.7/32.8 MB 9.0 MB/s eta 0:00:04
   ------- -------------------------------- 6.2/32.8 MB 8.8 MB/s eta 0:00:04
   -------- ------------------------------- 6.6/32.8 MB 8.8 MB/s eta 0:00:03
   -------- ------------------------------- 7.0/32.8 MB 8.8 MB/s eta 0:00:03
   --------- ------------------------------ 7.5/32.8 MB 8.9 MB/s eta 0:00:03
   --------- ------------------------------ 8.0/32.8 MB 9.0 MB/s eta 0:00:03
   ---------- ----------------------------- 8.3/32.8 MB 8.9 MB/s eta 0:00:03
   ---------- ----------------------------- 8.7/32.8 MB 8.9 MB/s eta 0:00:03
   ----------- ---------------------------- 9.1/32.8 MB 8.8 MB/s eta 0:00:03
   ----------- ---------------------------- 9.5/32.8 MB 8.8 MB/s eta 0:00:03
   ------------ --------------------------- 9.9/32.8 MB 8.8 MB/s eta 0:00:03
   ------------ --------------------------- 10.3/32.8 MB 9.0 MB/s eta 0:00:03
   ------------- -------------------------- 10.7/32.8 MB 9.1 MB/s eta 0:00:03
   ------------- -------------------------- 11.2/32.8 MB 9.1 MB/s eta 0:00:03
   -------------- ------------------------- 11.5/32.8 MB 8.8 MB/s eta 0:00:03
   -------------- ------------------------- 12.0/32.8 MB 8.8 MB/s eta 0:00:03
   -------------- ------------------------- 12.3/32.8 MB 8.8 MB/s eta 0:00:03
   --------------- ------------------------ 12.7/32.8 MB 8.6 MB/s eta 0:00:03
   --------------- ------------------------ 13.1/32.8 MB 8.6 MB/s eta 0:00:03
   ---------------- ----------------------- 13.5/32.8 MB 8.7 MB/s eta 0:00:03
   ---------------- ----------------------- 13.9/32.8 MB 8.5 MB/s eta 0:00:03
   ----------------- ---------------------- 14.1/32.8 MB 8.5 MB/s eta 0:00:03
   ----------------- ---------------------- 14.6/32.8 MB 8.6 MB/s eta 0:00:03
   ------------------ --------------------- 15.0/32.8 MB 8.4 MB/s eta 0:00:03
   ------------------ --------------------- 15.3/32.8 MB 8.5 MB/s eta 0:00:03
   ------------------- -------------------- 15.8/32.8 MB 8.4 MB/s eta 0:00:03
   ------------------- -------------------- 16.2/32.8 MB 8.4 MB/s eta 0:00:02
   -------------------- ------------------- 16.6/32.8 MB 8.4 MB/s eta 0:00:02
   -------------------- ------------------- 17.0/32.8 MB 8.5 MB/s eta 0:00:02
   --------------------- ------------------ 17.3/32.8 MB 8.3 MB/s eta 0:00:02
   --------------------- ------------------ 17.7/32.8 MB 8.3 MB/s eta 0:00:02
   --------------------- ------------------ 18.0/32.8 MB 8.2 MB/s eta 0:00:02
   ---------------------- ----------------- 18.4/32.8 MB 8.3 MB/s eta 0:00:02
   ---------------------- ----------------- 18.7/32.8 MB 8.1 MB/s eta 0:00:02
   ----------------------- ---------------- 19.1/32.8 MB 8.1 MB/s eta 0:00:02
   ----------------------- ---------------- 19.4/32.8 MB 8.0 MB/s eta 0:00:02
   ----------------------- ---------------- 19.7/32.8 MB 8.0 MB/s eta 0:00:02
   ------------------------ --------------- 20.0/32.8 MB 7.9 MB/s eta 0:00:02
   ------------------------ --------------- 20.3/32.8 MB 7.8 MB/s eta 0:00:02
   ------------------------- -------------- 20.6/32.8 MB 7.8 MB/s eta 0:00:02
   ------------------------- -------------- 21.0/32.8 MB 7.7 MB/s eta 0:00:02
   -------------------------- ------------- 21.4/32.8 MB 7.7 MB/s eta 0:00:02
   -------------------------- ------------- 21.6/32.8 MB 7.5 MB/s eta 0:00:02
   -------------------------- ------------- 22.0/32.8 MB 7.6 MB/s eta 0:00:02
   --------------------------- ------------ 22.2/32.8 MB 7.4 MB/s eta 0:00:02
   --------------------------- ------------ 22.6/32.8 MB 7.6 MB/s eta 0:00:02
   --------------------------- ------------ 23.0/32.8 MB 7.5 MB/s eta 0:00:02
   ---------------------------- ----------- 23.4/32.8 MB 7.4 MB/s eta 0:00:02
   ---------------------------- ----------- 23.8/32.8 MB 7.5 MB/s eta 0:00:02
   ----------------------------- ---------- 24.2/32.8 MB 7.5 MB/s eta 0:00:02
   ----------------------------- ---------- 24.4/32.8 MB 7.4 MB/s eta 0:00:02
   ------------------------------ --------- 24.7/32.8 MB 7.4 MB/s eta 0:00:02
   ------------------------------ --------- 25.1/32.8 MB 7.4 MB/s eta 0:00:02
   ------------------------------- -------- 25.5/32.8 MB 7.4 MB/s eta 0:00:02
   ------------------------------- -------- 25.7/32.8 MB 7.3 MB/s eta 0:00:01
   ------------------------------- -------- 26.1/32.8 MB 7.2 MB/s eta 0:00:01
   -------------------------------- ------- 26.3/32.8 MB 7.1 MB/s eta 0:00:01
   -------------------------------- ------- 26.6/32.8 MB 7.0 MB/s eta 0:00:01
   -------------------------------- ------- 26.8/32.8 MB 7.0 MB/s eta 0:00:01
   -------------------------------- ------- 27.1/32.8 MB 6.9 MB/s eta 0:00:01
   --------------------------------- ------ 27.4/32.8 MB 6.8 MB/s eta 0:00:01
   --------------------------------- ------ 27.6/32.8 MB 6.9 MB/s eta 0:00:01
   --------------------------------- ------ 27.9/32.8 MB 6.8 MB/s eta 0:00:01
   ---------------------------------- ----- 28.3/32.8 MB 6.8 MB/s eta 0:00:01
   ---------------------------------- ----- 28.5/32.8 MB 6.7 MB/s eta 0:00:01
   ----------------------------------- ---- 28.8/32.8 MB 6.7 MB/s eta 0:00:01
   ----------------------------------- ---- 29.2/32.8 MB 6.7 MB/s eta 0:00:01
   ----------------------------------- ---- 29.5/32.8 MB 6.7 MB/s eta 0:00:01
   ------------------------------------ --- 29.9/32.8 MB 6.7 MB/s eta 0:00:01
   ------------------------------------ --- 30.2/32.8 MB 6.8 MB/s eta 0:00:01
   ------------------------------------- -- 30.6/32.8 MB 6.8 MB/s eta 0:00:01
   ------------------------------------- -- 31.0/32.8 MB 6.8 MB/s eta 0:00:01
   -------------------------------------- - 31.3/32.8 MB 6.8 MB/s eta 0:00:01
   -------------------------------------- - 31.6/32.8 MB 6.8 MB/s eta 0:00:01
   -------------------------------------- - 31.6/32.8 MB 6.8 MB/s eta 0:00:01
   -------------------------------------- - 32.0/32.8 MB 6.7 MB/s eta 0:00:01
   ---------------------------------------  32.2/32.8 MB 6.5 MB/s eta 0:00:01
   ---------------------------------------  32.5/32.8 MB 6.6 MB/s eta 0:00:01
   ---------------------------------------  32.7/32.8 MB 6.5 MB/s eta 0:00:01
   ---------------------------------------  32.8/32.8 MB 6.4 MB/s eta 0:00:01
   ---------------------------------------  32.8/32.8 MB 6.4 MB/s eta 0:00:01
   ---------------------------------------- 32.8/32.8 MB 6.1 MB/s eta 0:00:00
Installing collected packages: SpeechRecognition
Successfully installed SpeechRecognition-3.10.4
Note: you may need to restart the kernel to use updated packages.
```

```
In [ ]: Convert speech to text to using the 'SpeechRecognisation' library
```

```
In [99]: import speech_recognition as sr

         recognizer = sr.Recognizer()
         with sr.Microphone() as source:
             print("Speak now:")
             audio = recognizer.listen(source)
         try:
             print("You said: " + recognizer.recognize_google(audio))
         except sr.UnknownValueError:
```

```
    print("Google Speech Recognition could not understand the audio")
```

```
Speak now:
You said: Vijay Devarakonda
```

In [119…

```
import speech_recognition as sr
recognizer =sr.Recognizer()
with sr.Microphone() as source:
    print("Sing a song:")
    audio=recognizer.listen(source)
try:
    print("song: " +recognizer.recognize_google(audio))
except sr.UnknownValueError:
    print("Google Speech Recognition could not understand the audio")
```

```
Sing a song:
song: I just wanna you don't know someone I am sadness
```

"*OpenWeatherMap"

In [ ]:

```
*Access weather data from OpenWeatherMap API.
```

In [121…

```
pip install OpenWeatherMap API
```

```
Collecting OpenWeatherMap
  Downloading openweathermap-0.1.4-py3-none-any.whl.metadata (621 bytes)
Collecting API
  Downloading api-0.0.7.tar.gz (2.2 kB)
  Preparing metadata (setup.py): started
  Preparing metadata (setup.py): finished with status 'done'
Requirement already satisfied: requests in c:\users\gadam\anaconda3\lib\site-packages (from API) (2.32.2)
Collecting nose (from API)
  Downloading nose-1.3.7-py3-none-any.whl.metadata (1.7 kB)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\gadam\anaconda3\lib\site-packages (from requ
ests->API) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\gadam\anaconda3\lib\site-packages (from requests->API) (
3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\gadam\anaconda3\lib\site-packages (from requests->
API) (1.26.19)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\gadam\anaconda3\lib\site-packages (from requests->
API) (2024.6.2)
Downloading openweathermap-0.1.4-py3-none-any.whl (3.2 kB)
Downloading nose-1.3.7-py3-none-any.whl (154 kB)
   ---------------------------------------- 0.0/154.7 kB ? eta -:--:--
   ------- ------------------------------- 30.7/154.7 kB 640.0 kB/s eta 0:00:01
   ---------------------------------------  153.6/154.7 kB 1.8 MB/s eta 0:00:01
   ---------------------------------------- 154.7/154.7 kB 1.5 MB/s eta 0:00:00
Building wheels for collected packages: API
  Building wheel for API (setup.py): started
  Building wheel for API (setup.py): finished with status 'done'
  Created wheel for API: filename=api-0.0.7-py3-none-any.whl size=2314 sha256=294ff6108884830ff731391daaaf9736a7
13e0fc14a36a321d3b5fff127dacd5
  Stored in directory: c:\users\gadam\appdata\local\pip\cache\wheels\75\7e\91\7b77e5f4c0f22865668b7ae91c73bb2eb5
bc1cd23555f26172
Successfully built API
Installing collected packages: nose, OpenWeatherMap, API
Successfully installed API-0.0.7 OpenWeatherMap-0.1.4 nose-1.3.7
Note: you may need to restart the kernel to use updated packages.
```

In [165…

```
pip install requests
```

```
Requirement already satisfied: requests in c:\users\gadam\anaconda3\lib\site-packages (2.32.2)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\gadam\anaconda3\lib\site-packages (from requ
ests) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\gadam\anaconda3\lib\site-packages (from requests) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\gadam\anaconda3\lib\site-packages (from requests)
(1.26.19)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\gadam\anaconda3\lib\site-packages (from requests)
(2024.6.2)
Note: you may need to restart the kernel to use updated packages.
```

In [10]:

```
import requests
#from  OpenWeatherMap API  import weather
api_key="856419db724420025d7b4fdf0d3bc6cd"
city="London"
url=f"http://api.openweathermap.org/data/2.5/weather?q={city}&appid={api_key}"
response=requests.get(url)
data=response.json()
print(f"Weather in {city}: {data['weather'][0]['description']}")
```

```
Weather in London: broken clouds
```

In [12]:

```
import requests
#from  OpenWeatherMap API  import weather
```

```
api_key="856419db724420025d7b4fdf0d3bc6cd"
city="India"
url=f"http://api.openweathermap.org/data/2.5/weather?q={city}&appid={api_key}"
response=requests.get(url)
data=response.json()
print(f"Weather in {city}: {data['weather'][0]['description']}")
```

```
Weather in India: overcast clouds
```

In [8]:
```
import requests
#from  OpenWeatherMap API  import weather
api_key="856419db724420025d7b4fdf0d3bc6cd"
city="New York"
url=f"http://api.openweathermap.org/data/2.5/weather?q={city}&appid={api_key}"
response=requests.get(url)
data=response.json()
print(f"Weather in {city}: {data['weather'][0]['description']}")
```

```
Weather in New York: clear sky
```

In [14]:
```
import requests

API_KEY = '856419db724420025d7b4fdf0d3bc6cd'
city = 'London'
base_url = 'https://api.openweathermap.org/data/2.5/weather'

params = {
    'q': city,
    'appid': API_KEY,
    'units': 'metric'  # 'metric' for Celsius, 'imperial' for Fahrenheit
}

response = requests.get(base_url, params=params)

# Check if the request was successful
if response.status_code == 200:
    data = response.json()
    print(f"City: {data['name']}")
    print(f"Temperature: {data['main']['temp']}°C")
    print(f"Weather: {data['weather'][0]['description']}")
else:
    print("Error: ", response.status_code)
```

```
City: London
Temperature: 21.44°C
Weather: broken clouds
```

In [28]:
```
import datetime as dt

import requests

base_url="http://api.openweathermap.org/data/2.5/weather?"

api_key='856419db724420025d7b4fdf0d3bc6cd'
city="New York"

def kelvin_to_celcius_fahrenheit(kelvin):
    celcius=kelvin-273.15
    fahrenheit=celcius*(9/5)+32
    return celcius,fahrenheit

url=base_url + "appid=" + api_key + "&q=" +city

response=requests.get(url).json()
temp_kelvin=response['main']['temp']
temp_celcius,temp_fahrenheit=kelvin_to_celcius_fahrenheit(temp_kelvin)
feels_like_kelvin=response['main']['feels_like']

print(response)
print("temp:",temp_kelvin)
```

```
{'coord': {'lon': -74.006, 'lat': 40.7143}, 'weather': [{'id': 800, 'main': 'Clear', 'description': 'clear sky',
'icon': '01d'}], 'base': 'stations', 'main': {'temp': 292.01, 'feels_like': 291.94, 'temp_min': 289.64, 'temp_ma
x': 293.34, 'pressure': 1014, 'humidity': 76, 'sea_level': 1014, 'grnd_level': 1013}, 'visibility': 10000, 'wind
': {'speed': 4.63, 'deg': 360}, 'clouds': {'all': 0}, 'dt': 1726744963, 'sys': {'type': 2, 'id': 2083229, 'count
ry': 'US', 'sunrise': 1726742463, 'sunset': 1726786700}, 'timezone': -14400, 'id': 5128581, 'name': 'New York',
'cod': 200}
temp: 292.01
```

"GUI Programming with PyQt5 in Python"

In [1]:
```
!pip install pyaudio
```

```
Collecting pyaudio
  Downloading PyAudio-0.2.14-cp312-cp312-win_amd64.whl.metadata (2.7 kB)
Downloading PyAudio-0.2.14-cp312-cp312-win_amd64.whl (164 kB)
   ---------------------------------------- 0.0/164.1 kB ? eta -:--:--
   ---------------------------------------- 0.0/164.1 kB ? eta -:--:--
   ------- -------------------------------- 30.7/164.1 kB 1.4 MB/s eta 0:00:01
   ------------------- -------------------- 81.9/164.1 kB 1.2 MB/s eta 0:00:01
   ------------------------ -------------- 102.4/164.1 kB 991.0 kB/s eta 0:00:01
   -------------------------------------   163.8/164.1 kB 898.2 kB/s eta 0:00:01
   --------------------------------------- 164.1/164.1 kB 758.1 kB/s eta 0:00:00
Installing collected packages: pyaudio
Successfully installed pyaudio-0.2.14
```

In [ ]:
```
"GUI Programming with PyQt5 in Python"
```

PyQt5 is a powerful Python library for creating Graphical User Interface (GUI)application It Provides bindings for the Qt application framework and is widely used for craeting cross-platform desktop applications.

In [ ]:
```
"Introduction of PyQt5"
```

1.PyQt5 is a set of Python bindings for Qt libraries which can be used to create professional looking desktop applications.

2.With PyQt5 ,we can build modern and efficent graphical interfaces and manage events like button clicks ,text input ,etc.

In [ ]:
```
"Components and Events in PyQt5"
```

- Components are UI elements like buttons,text boxes,labels etc..
- Events are actions triggered by the user ,such as clicking a button ,entering text,etc.

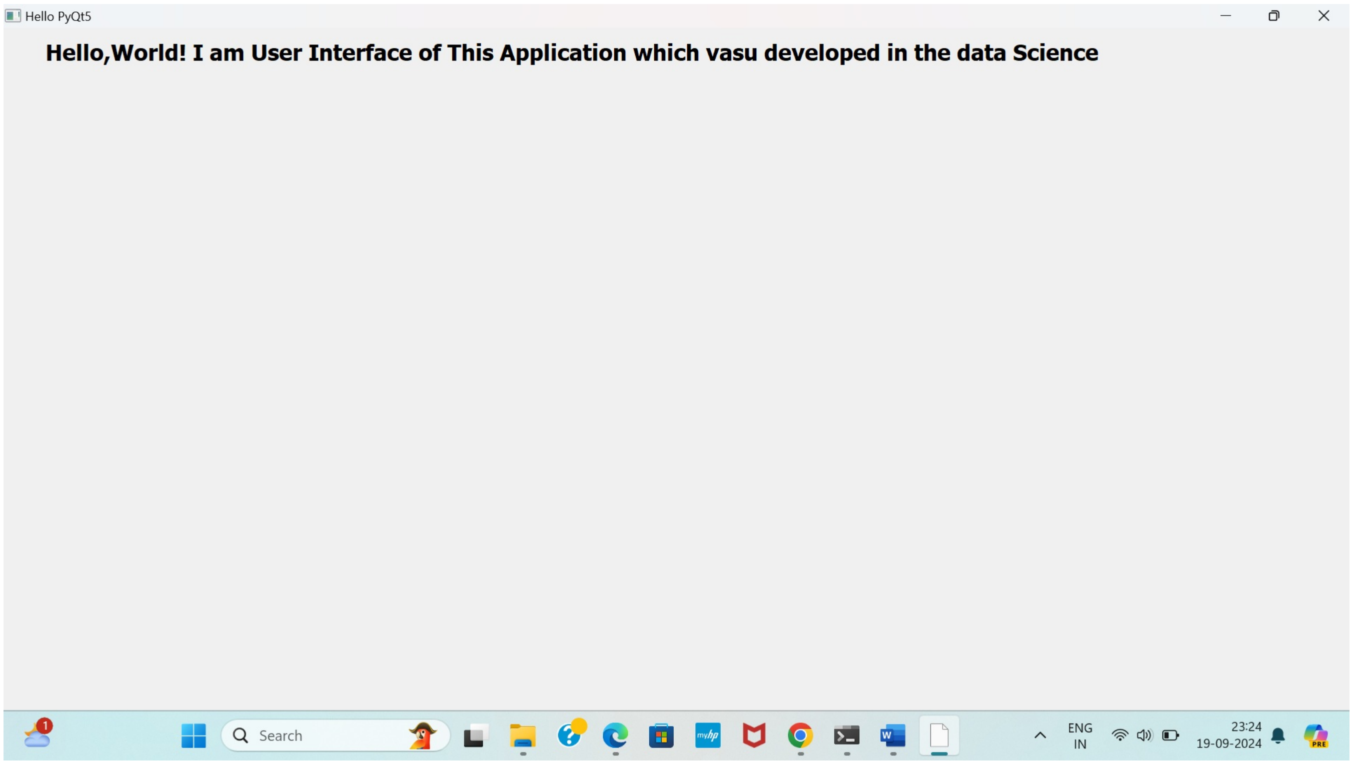
This actions can be handled using signals and slots.

In [ ]:
```
"Example: BAsic PyQt5 Applications"
```

In [ ]:
```
import sys
from PyQt5.QtWidgets import QApplication,QLabel,QWidget
#create am application object
app=QApplication(sys.argv)

#create a window (QWidget)
window=QWidget()
window.setWindowTitle('Hello PyQt5')
window.setGeometry(100,100,100,100)

#Add a label to the window
label=QLabel('<h1>Hello,World! I am User Interface of This Application which vasu developed in the data Science
label.move(60,15)

#show the window
window.show()

#start the application loop
sys.exit(app.exec_())
```

In [1]:
```
from PIL import Image
Image.open('GUI.png')
```

Out[1]:



Hello PyQt5

Hello,World! I am User Interface of This Application which vasu developed in the data Science

In [ ]:
```
"Explnation"
```

- QApplication:Initialises the application
- QWidget: Represents the window.
- QLabel: Displays static text inside the window.
- app.exec_(): Enters the application's main loop and waits for events.

In [ ]:
```
"Widgets inPyQt5"
```

- QPushButton: Button widget to trigger actions.
- QLineEdit : Text input widget to accept user input.
- QLabel : Displays static text or images.
- QCheckBox : A checkbox widget to select/deselect options

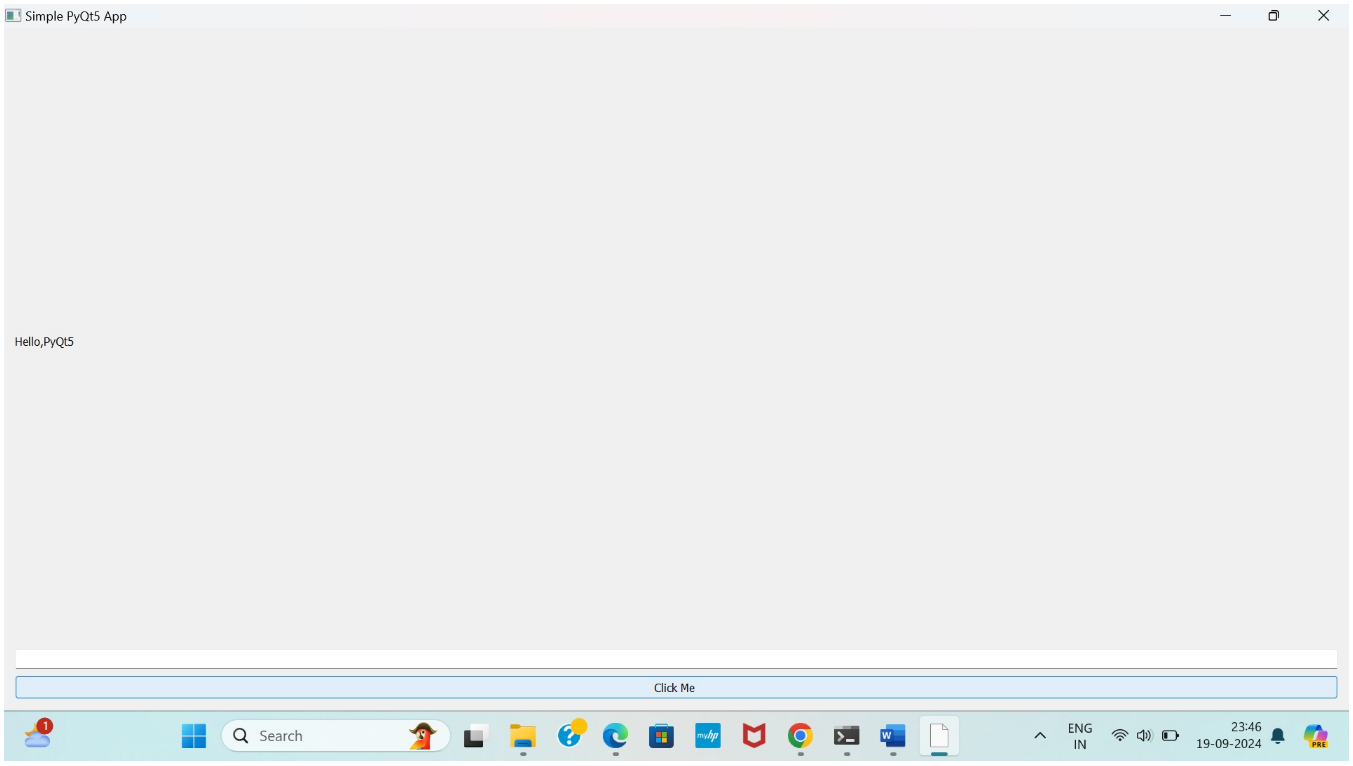
In [ ]:
```
"Example :with Multiple Widgets"
```

In [ ]:
```
import sys

from PyQt5.QtWidgets import QApplication,QWidget,QPushButton,QLabel,QLineEdit,QVBoxLayout

#Define the main applications
def on_click():
    label.setText("Button clicked!")
app=QApplication(sys.argv)

#create a window
window=QWidget()
window.setWindowTitle('Simple PyQt5 App')
window.setGeometry(100,100,1000,1000)

#Create widgets
button=QPushButton('Click Me ',window)
label =QLabel('Hello,PyQt5',window)
text_input=QLineEdit(window)

#connect button click event to the on_click function
button.clicked.connect(on_click)

#set Layout
layout = QVBoxLayout()
layout.addWidget(label)
layout.addWidget(text_input)
layout.addWidget(button)

#Apply the layout to the window

window.setLayout(layout)
window.show()
```

```
#Run the Application
sys.exit(app.exec_())
```

In [3]:
```
from PIL import Image
Image.open('GUI1.png')
```

Out[3]:



In [ ]:
```
"Layout Management"
```

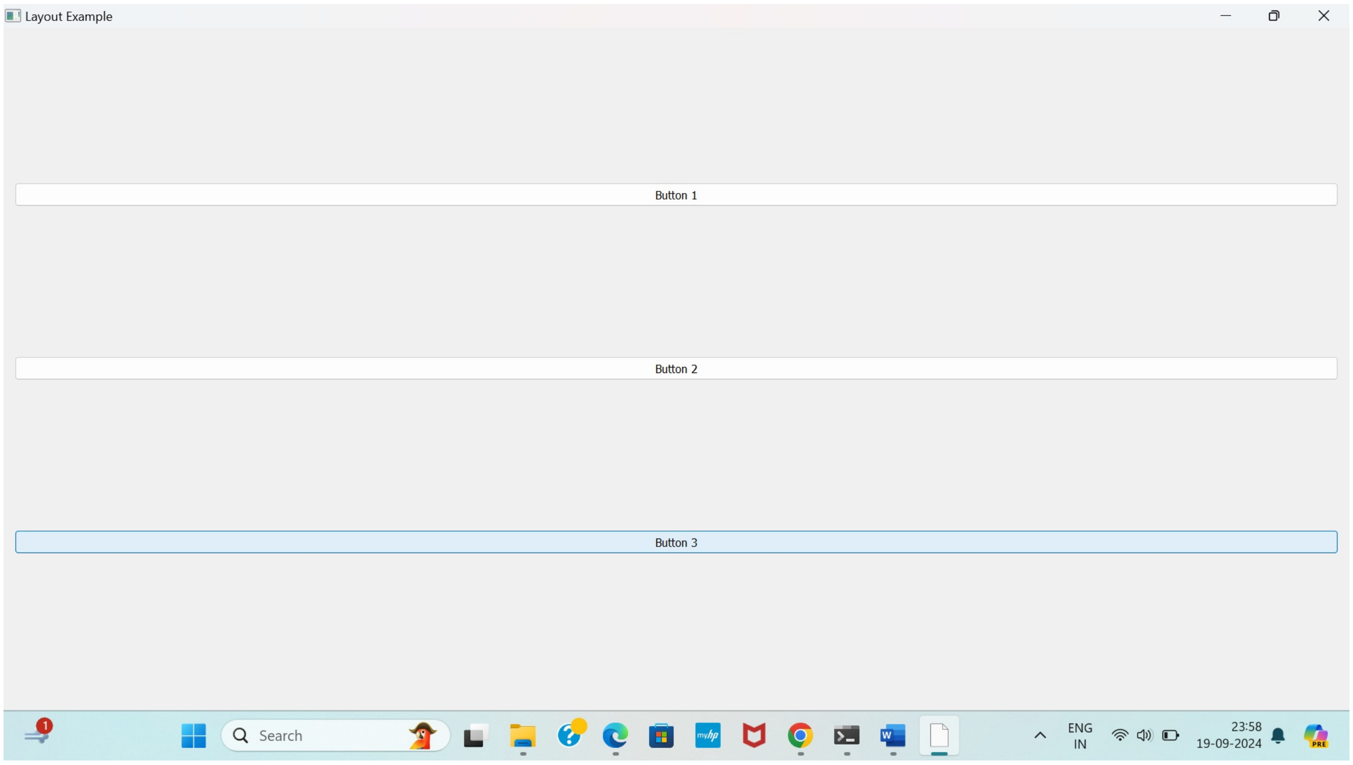
Layouts help we organise the widgets inside a window,PyQt5 provides various layouts such as:

```
    * QVBoxLayout: Arranges widgets vertically.
    * QHBoxLayout: Arranges widgets horizontally.
```

In [ ]:
```
"Example with Layouts"
```

In [ ]:
```
import sys

from PyQt5.QtWidgets import QApplication,QWidget,QPushButton,QVBoxLayout

app=QApplication(sys.argv)

#create a window

window=QWidget()
window.setWindowTitle('Layout Example')

#create buttons
button1=QPushButton('Button 1',window)
button2=QPushButton('Button 2',window)
button3=QPushButton('Button 3',window)

#Use QVBoxLayout
layout=QVBoxLayout()
layout.addWidget(button1)
layout.addWidget(button2)
layout.addWidget(button3)

#set layout to teh window
window.setLayout(layout)
#show window
window.show()
sys.exit(app.exec_())
```

In [1]:
```
from PIL import Image
Image.open('GUI2.png')
```

Out[1]: 

In [ ]:
```
"Signals and Slots"
```

- singnals : Event or action (e.g.,button clicked)
- Slot: Function that gets called when the signal is emitted.

In [ ]:
```
"Example with Signals and Slots"
```

In [1]:
```
from PyQt5.QtWidgets import QApplication,QWidget,QPushButton

def on_click():
    print("Button clicked!")

app=QApplication([])
window=QWidget()

#create a button and connect it to the on_click slot

button=QPushButton('Click ME',window)
button.clicked.connect(on_click)

window.show()
app.exec_()
```

```
Button clicked!
Button clicked!
Button clicked!
Button clicked!
Button clicked!
Button clicked!
Button clicked!
Button clicked!
```

Out[1]: 0

We can get on Application desktop on that we have click me ,whenever we click that it prints that time of 'button clicked'

In [ ]:
```
"QMessageBOx and QDialog"
```

- QMessageBox : Displays dialog boxes like alerts,information,or confirmation.
- QDialog : Modal dialog used to gather user input.

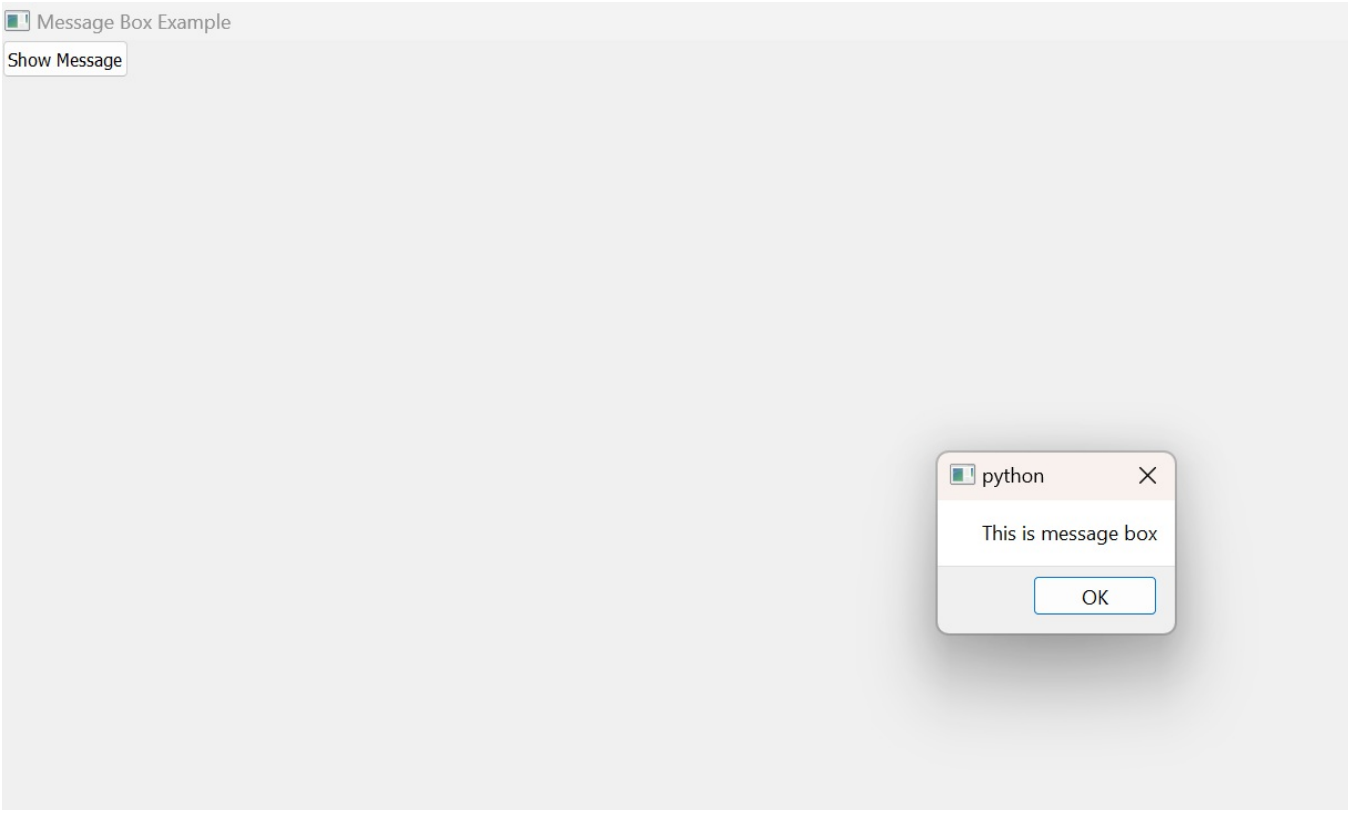
In [ ]:
```
"Example with QMessageBox"
```

In [ ]:
```
import sys
from PyQt5.QtWidgets import QApplication,QWidget,QMessageBox,QPushButton

def show_message():
    msg_box=QMessageBox()
    msg_box.setText("This is message box")
    msg_box.exec_()
```

```
app=QApplication(sys.argv)

#create a window
window=QWidget()
window.setWindowTitle('Message Box Example')

#Add Button

button=QPushButton('Show Message',window)
button.clicked.connect(show_message)

window.show()
sys.exit(app.exec_())
```

In [1]:
```
from PIL import Image
Image.open('message.png')
```

Out[1]:



In [ ]:
```
"Database Handling with PyQt5 (Mysql Intgaration)"
```

PyQt5 allows for the integration of databases,enabling you to build desktop applications that can interact with databases.The 'QSqlDatabase' class from PyQt5's 'QtSql' module provides a flexible way to interact with database like 'MySQL'.Below is an explanation of how to integrate MySQL databases with PyQt5 and perform basic operations such as displaying data in a PyQt5 application.

In [ ]:
```
"Steps to Handle MySQL Databases with PyQt5 :"
```

1. Install MySQL connector: we need to install the mysql-connector-python library for connecting your python application to a MySQl database.s

pip install mysql-connector-python

2. Create MySQl Database & Table : Ensure that your, MySQL database is set up and that you have a table from which you want to fetch data.For this example ,let's assume we have a database named sample_db and a table named employees.

In [ ]:
```
In Mysql workbench we need to create one table with records with the

CREATE DATABASE sample_db;
USE sample_db;

CREATE TABLE employees (
    id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(100),
    position VARCHAR(100),
    salary FLOAT
);

INSERT INTO employees (name, position, salary)
VALUES
```

```
('John Doe', 'Manager', 70000),
('Jane Smith', 'Engineer', 55000),
('Sam Brown', 'HR', 45000);
```

In [ ]:
```
"PyQt5 GUI to show MySQL Table Data"
```

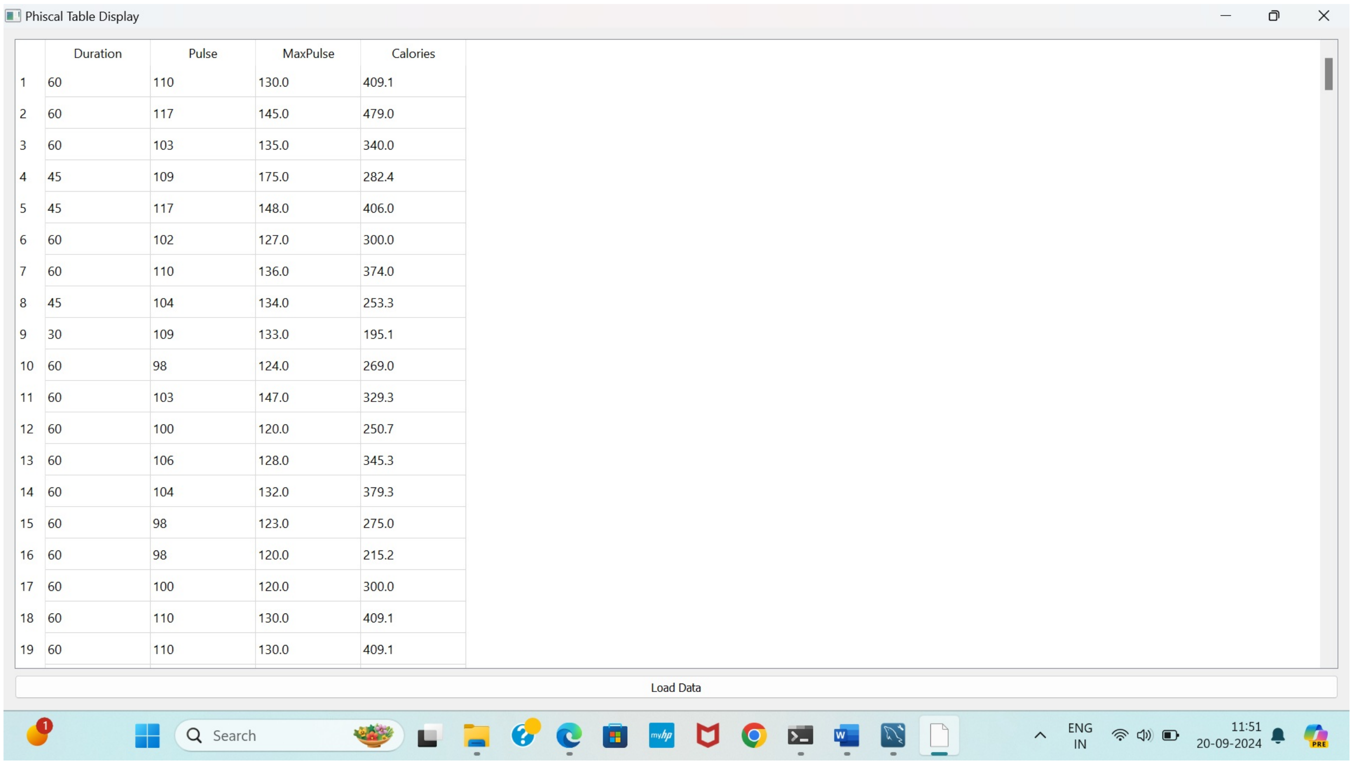
This example demonstrates how to fetch data from the MySQL database and display it in a PyQt5 application.

In [ ]:
```
"Full Example: PyQt5 GUI to Display Data from MySQL Table"
```

In [ ]:
```
import sys
from PyQt5.QtWidgets import QApplication,QWidget,QVBoxLayout,QPushButton,QLabel,QLineEdit,QTableWidget,QTableWi

import mysql.connector as conn

class MySQLTableDisplay(QWidget):
    def __init__(self):
        super().__init__()

        #Setup the PyQt5 window
        self.setWindowTitle('Phiscal Table Display')
        self.setGeometry(100,100,600,400)

        #create a vertical layout
        layout=QVBoxLayout()

        #create a table Widget
        self.table_widget=QTableWidget()
        layout.addWidget(self.table_widget)

        #button to load data
        load_button=QPushButton('Load Data')
        load_button.clicked.connect(self.load_data)
        layout.addWidget(load_button)

        #set a layout for the window
        self.setLayout(layout)
    def load_data(self):
        #Establish Mysql connection
        mydb=conn.connect(
            host="localhost",
            user="root",
            password="password",
            database="vasu1"
        )
        mycursor=mydb.cursor()
        #Query to fetch all data from the employee table
        mycursor.execute("Select* from srinu1")
        records=mycursor.fetchall()

        #Set the number of rows and columns for the table widget
        self.table_widget.setRowCount(len(records))
        self.table_widget.setColumnCount(4)
        self.table_widget.setHorizontalHeaderLabels(['Duration','Pulse','MaxPulse','Calories'])

        #populate the table with data
        for row_index,row_data in enumerate(records):
            for column_index,data in enumerate(row_data):
                self.table_widget.setItem(row_index,column_index,QTableWidgetItem(str(data)))
        #close the cursor and connection
        mycursor().close()
        mydb.close()

#Main execution of the PyQt5 application

if __name__=='__main__':
    app=QApplication(sys.argv)
    window=MySQLTableDisplay()
    window.show()
    sys.exit(app.exec_())
```

In [3]:
```
from PIL import Image
Image.open('table1.png')
```

Out[3]:



	Duration	Pulse	MaxPulse	Calories
1	60	110	130.0	409.1
2	60	117	145.0	479.0
3	60	103	135.0	340.0
4	45	109	175.0	282.4
5	45	117	148.0	406.0
6	60	102	127.0	300.0
7	60	110	136.0	374.0
8	45	104	134.0	253.3
9	30	109	133.0	195.1
10	60	98	124.0	269.0
11	60	103	147.0	329.3
12	60	100	120.0	250.7
13	60	106	128.0	345.3
14	60	104	132.0	379.3
15	60	98	123.0	275.0
16	60	98	120.0	215.2
17	60	100	120.0	300.0
18	60	110	130.0	409.1
19	60	110	130.0	409.1

Load Data

In [ ]:
```
"Exaplanation:"
```

```
1.MySQL Connection

 o      The mysql.connector.connect() function is used to establish a connection to the MySQL
database.
*o      Replace 'localhost', 'root', and 'your_password' with your MySQL host, username, and
password, respectively.y.


  2.    Fetching Data:

 *      The cursor.execute("SELECT * FROM employees") command fetches all records from the
employees table.

*       The records = cursor.fetchall() function retrieves all rows from the result set as a
list of tuples.


 3.     Displaying Data in PyQt5 Table:

 *      The QTableWidget is used to create a table in the PyQt5 GUI.

 *      setRowCount() and setColumnCount() methods define the number of rows and columns in
the table.

 *      setHorizontalHeaderLabels() defines the column names.

 *      A nested loop iterates through the fetched records and populates the QTableWidget.


4.      Triggering the Display:

o       A button labeled Load Data is connected to the load_data() function, which loads the
data from the database and displays it when clicked.
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

```

```

In [ ]:

```

```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js