Mini-project: HTTP GET Request Using C++ Sockets

1. Project Goal:

In this mini-project, you will build a simple HTTP GET request using C++ sockets. This will help you understand how HTTP requests and responses work at a low level by directly handling sockets.

2. Steps to Build the HTTP GET Request:

Step 1: Set up the socket

- Use the socket() function to create a new socket.
- Set up the socket to use IPv4 (AF_INET) and TCP (SOCK_STREAM).
- Connect the socket to the web server you want to send a request to using the connect() function. You will need to convert the server's address to a sockaddr_in structure.

Step 2: Create the HTTP GET request

- Construct a valid HTTP GET request string. A typical request looks like this:

```
GET / HTTP/1.1
Host: www.example.com
Connection: close
```

- Make sure the HTTP request string is correctly formatted with line breaks (\r\n). The "Connection: close" header ensures the connection is closed after the response is received.

Step 3: Send the request through the socket

- Use the send() function to send the HTTP GET request through the socket.

- Ensure that the entire request is sent by checking the return value of send(), which indicates the number of bytes successfully transmitted.

Step 4: Receive the HTTP response

- Use the recv() function to read the HTTP response from the server.
- Loop through the response, reading in chunks of data until the entire message has been received.
- The response will include the HTTP headers followed by the actual content.

Step 5: Parse and display the response

- Parse the response to separate the headers from the content (if necessary).
- Display the full response, or print specific parts such as the status code, headers, and content.

Step 6: Close the socket

- After receiving the response, close the socket using the close() function to properly release the resources.

3. Functions Required and Their Parameters:

1. socket()

- Creates a new socket.
- Syntax: int socket(int domain, int type, int protocol);
- Parameters:
  - domain: AF_INET for IPv4
  - type: SOCK_STREAM for TCP
  - protocol: Usually 0 to select the default protocol (TCP in this case).

2. connect()

- Connects the socket to a server.

- Syntax: int connect(int sockfd, const struct sockaddr *addr, socklen_t addrlen);

- Parameters:

  - sockfd: The socket file descriptor.

  - addr: A pointer to a sockaddr_in structure that contains the server's address and port.

  - addrlen: The size of the address structure.

3. send()

- Sends data through a socket.

- Syntax: ssize_t send(int sockfd, const void *buf, size_t len, int flags);

- Parameters:

  - sockfd: The socket file descriptor.

  - buf: A pointer to the buffer containing the data to send (e.g., the HTTP request).

  - len: The size of the data to send.

  - flags: Additional flags (usually set to 0).

4. recv()

- Receives data from a socket.

- Syntax: ssize_t recv(int sockfd, void *buf, size_t len, int flags);

- Parameters:

  - sockfd: The socket file descriptor.

  - buf: A pointer to the buffer where the received data will be stored.

  - len: The maximum number of bytes to receive.

  - flags: Additional flags (usually set to 0).

5. close()

- Closes the socket and releases the resources.

- Syntax: int close(int sockfd);

- Parameters:

  - sockfd: The socket file descriptor to close.