PROBLEM STATEMENT

SECURE SOCKET IMPLEMTATION IN DATABASE MANAGEMENT

(ONLINE LIBRARY MANAGEMENT)

GROUP 1

DESIGN DOCUMENT

**1.INTRODUCTION:**

Establish communication between the server-Client to connect and access data from the database to download the required books from the database which contains list of URLs of the books to download.

**1.1 PURPOSE:**

The purpose of this project is to get a solid grasp on the fundamentals of the Socket API. Writing such an application in C++ gives a basic understanding on how the client - server architecture works and overall, on how to use the Socket API to establish communication between client and server applications with the goal of information exchange between the two. This application also enables the client to download and access the required books from database which contains the information of the books (URL).

**1.2 FUNCTIONALITIES OF THE SYSTEM:**

Client:

client performs the operation for downloading books.

Server:

In the server terminal, it will login through credentials and then socket will be created and then socket will bind and then it will check the connections and accept it. Enable user to access the database.

Database:

It contains the list of the book URLs.

Signals:

It controls the exit and back operations.

Execv Function:

It is used to Execute the required function.

**1.3 OPERATING ENVIRONMENT:**

Operating environment are:

* Sockets
* Operating system: Linux
* Data Base (SQL)
* Signals
* Platform: Ubuntu/C++.

**2. SOFTWARE REQUIREMENTS:**

* **g++/gcc**
* **makefile**
* **valgrind**
* **gcov**
* **git**

**2.1 UNIT TEST:**

**Client:**

SRS1-UT1: Create a valid socket to initiate the communication

SRS2-UT2: If the socket () function fails then the client must receive an error

message socket creation failure.

SRS3-UT3: The port no. in the client side and server side should be same if not connection is not established

SRS4-UT4: The connect () call on a stream socket is used by the client application to

establish a connection to a server. The server must have a passive open pending. A server that is using sockets must successfully call bind () and listen () before a connection can be accepted by the server with accept ().

SRS5-UT5: If send () or rec () function fails then the client won’t be able to send or

receive the data

SRS6-UT6: Send and receive the request, The server uses the socket that is returned

from the accept () call. These functions return the amount of data that was sent or received. Because stream sockets send and receive information in streams of data, it can take more than one send () or rec () to transfer all of the data. It is up to the client and the server to agree on some mechanism to signal that all of the data has been transferred.

**Server:**

SRS1-UT1: Provide valid username and password.

SRS2-UT1: If Validation is successful, create a valid socket to initiate the

communication

SRS2-UT2: The port no. in the client side and server side should be same if not connection is not established.

SRS3-UT3: The bind () should bind the client – server,

if failed must return binding error while listen () function must listen on particular port number.

SRS4-UT4: listen () function indicates a readiness to accept client connection

requests, and creates a connection request queue of length backlog to queue incoming connection requests.

SRS\_UT5: The accept () call is used by a server to accept a connection request from a

client. When a connection is available, the socket created is ready for use to read data from the process that requested the connection. The call accepts the first connection on its queue of pending connections for the given socket.

SRS6-UT6: Send and receive the request, The server uses the socket that is returned

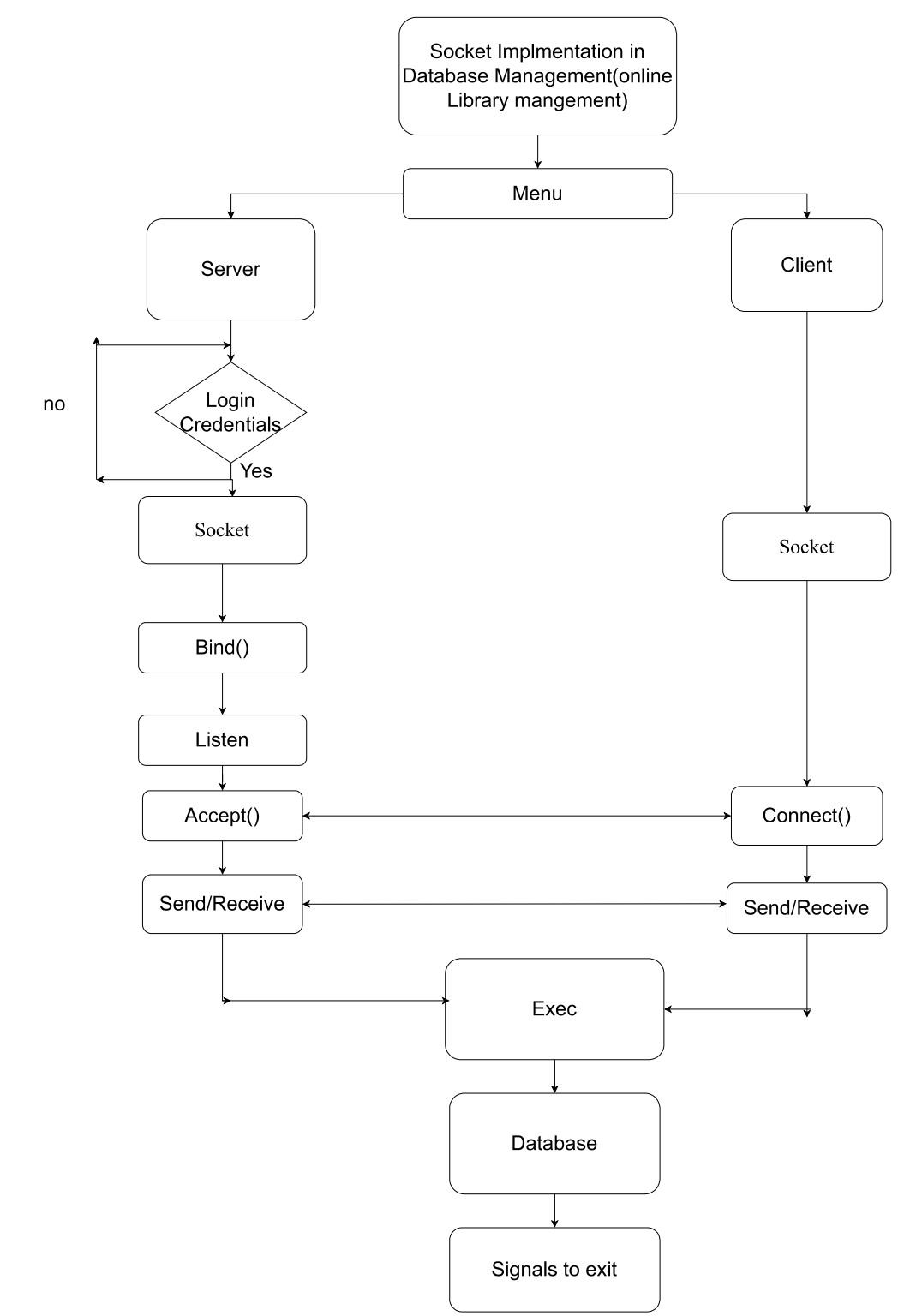
from the accept () call. These functions return the amount of data that was sent or received. Because stream sockets send and receive information in streams of data, it can take more than one send () or rec () to transfer all of the data. It is up to the client and the server to agree on some mechanism to signal that all of the data has been transferred.

**Database:**

SRS1-UT1: Creation of Database.

SRS2-UT2: Insertion of list of URLs of Books into the database.

SRS3-UT3: Display the data from database based on client requirement.

**3. FLOW DIAGRAM:**

**SOCKET:**

Socket programming is a way of connecting two nodes on a network to communicate with each other. One socket(node) listens on a particular port at an IP, while the other socket reaches out to the other to form a connection. The server forms the listener socket while the client reaches out to the server

**BIND:**

When a socket has both an IP address and a port number it is said to be 'bound to a port', or 'bound to an address'. A bound socket can receive data because it has a complete address. Binding is the process of allocating a port number to a socket.

**LISTEN:**

The listen () function applies only to stream sockets. It indicates a readiness to accept client connection requests, and creates a connection request queue of length backlog to queue incoming connection requests. Once full, additional connection requests are rejected**.**

**ACCEPT:**

The accept () call is used by a server to accept a connection request from a client. When a connection is available, the socket created is ready for use to read data from the process that requested the connection. The call accepts the first connection on its queue of pending connections for the given socket.

**CONNECT:**

The connect () call on a stream socket is used by the client application to establish a connection to a server. The server must have a passive open pending. A server that is using sockets must successfully call bind () and listen () before a connection can be accepted by the server with accept ().

**SEND/REC:**

The server uses the socket that is returned from the accept () call.

These functions return the amount of data that was sent or received. Because stream sockets send and receive information in streams of data, it can take more than one send () or rec () to transfer all of the data. It is up to the client and the server to agree on some mechanism to signal that all of the data has been transferred.

**EXEC:**

The execv function is most commonly used to overlay a process image that has been created by a call to the fork function. file. is the filename of the file that contains the executable image of the new process. argv is a pointer to an array of pointers to null-terminated character strings. exec is a functionality of an operating system that runs an executable file in the context of an already existing process, replacing the previous executable.

**DATABASE:**

A database is an organized collection of structured information, or data, typically stored electronically in a computer system. A database is usually controlled by a database management system (DBMS).

**SIGNALS:**

A signal is a software generated interrupt that is sent to a process by the OS because of when user press ctrl-c or another process tell something to this process. There are fix set of signals that can be sent to a process. signal is identified by integers.

**4.CONCLUSION:**

Successfully communication is created between the client-server using sockets and database is created to store the book URLs. Client is given access to download books from database. Termination is done by using signals.