Operating System IT-308
Project Report

Project Name : **Multi Threaded Server**

Participants : **Dhaval Prajapati (201401138)**
Dhruv Koshiyar (201401136)

Objective : A server which can serve multiple user requests simultaneously using threads for sending and receiving Messages.

Introduction :

A server with the single thread can serve only one user request at time. So if multiple users' request simultaneously then server will serve only one request at a time and put others in queue. After sending the response of that request, server will pick another request from the queue. But this is impractical. For get rid of that problem we can use multi threaded server. In multi-threaded server, there are N threads which are running parallel. When request comes to the server that request is allocated to available thread from threadpool to execute. While server is busy in serving that request, if another request comes then that will be served using other thread which runs parallel to all threads in the system. By using multi-threading concept we can reduce response time and improve throughput of the server. We can achieve this by building Thread pool , which contains a fixed number of threads. The server will only serve client requests equal to the size of the threadpool simultaneously.

Functionality :

In our code we have five classes named,

1. MY_main.java
2. My_server.java
3. My_client.java
4. Helper.java
5. Request.java
6. Worker.java

Now we will discuss about the functionality of every classes.

1. **My_main.java** : It just creates a server thread on specific port number and specifies the size of thread pool.Then it calls run() method of My_server class.Then we will open the server for fix amount of time and then we close the server.During this time period

server handle the requests of clients.At last we print the total request handled by the server and number of requests handled by each threads.

2. **My_server.java** : It will start server on given port number by initialize the size of Threadpool. It overrides the run() method of Runnable class. In this method server will accept the incoming connection request from clients, send the acknowledgement of the received request through replying the message and put sockets into pending list. One more thing it does is to start new thread of Helper.

3. **My_Client.java** : It connects client to the server on given port number. It establish the inputstream for client inputs and then set outputstream to the server.Client send the message to the server and wait for the acknoledgement.As acknowledgement comes client will send another request. Here we also calculates the response time of the server.

4. **Request.java** : In this class request contains client socket,server text and arrival time.In this class we have three methods.first is setstartTime(), which return the setup time for the request.second is setendTime(),which return the finish time of any method.Third one is ServicedBy(), which return the serving thread of given request.

5. **Helper.java** : This class will check the thread pool for availability of free slot(thread) and if there is some request pending in the list then it will remove the first request in the FIFO (First In First Out) Manner and provide it to the free thread in the threadpool.

6. **Worker.java** : Basically this class work on the synchronization of the threads.It notes the start and the end time of each request.It prints the message which shows which request run by which thread.

Conclusion :

Here we have created the Multi threaded server.Which handles the requests with synchronization. We have implemented First in First out(FIFO) algorithm to handle the pending requests.