

Assignment No: 03

Name: Dnyaneshwari Thombal

Class: SY-1

Batch: C

PRN: B25CE2005

Title:

Load Balancing:

For example, imagine you have a set of servers that handle requests for a web application. The key to load balancing is using the hash value of a client's IP address or a request ID to determine which server should handle the request. The hash function is typically designed so that the data is evenly distributed across the servers, ensuring that no single server is overloaded. Write a program of a load balancing system using hashing, where a basic hash table for mapping incoming requests to a set of servers.

Program: #include

<iostream> using

namespace std;

class LoadBalancer {

int n_servers; int

hash_table[100];

public:

LoadBalancer(int n) {

n_servers = n; for (int i = 0; i <

n_servers; i++) { hash_table[i]

= 0;

}

}

int hashFunction(int val) {

return val % n_servers;

} int processIP(char ip[]) {

int sum = 0, num = 0; for (int i

= 0; ip[i] != '\0'; i++) { if

(ip[i] == '.') { sum +=

num;

num = 0; } else {

num = num * 10 + (ip[i] - '0');

```

    }
}
sum += num;
return sum;
} void insert(char ip[]) {
int val = processIP(ip);    int
idx = hashFunction(val);    int
start = idx;

    while (hash_table[idx] != 0) {        idx = (idx + 1) % n_servers;
if (idx == start) {                    cout << "All servers are busy! Cannot
assign " << ip << endl;                return;
    }
}
    hash_table[idx] = val;    cout << "Request "
<< ip << " (sum = " << val        << ") is handled
by Server " << idx << endl;
}

void display() {    cout << "\nServer
Load Summary:\n";    for (int i = 0; i <
n_servers; i++) {    cout << "Server "
<< i << " <- ";    if (hash_table[i] !=
0) {        cout << hash_table[i];
    } else {        cout
<< "<empty>";
    }
cout << endl;
}
}
};

int main() {
    int n, r;
    cout << "Enter number of servers: ";
    cin >> n;    cout << "Enter number of IP
requests: ";    cin >> r;

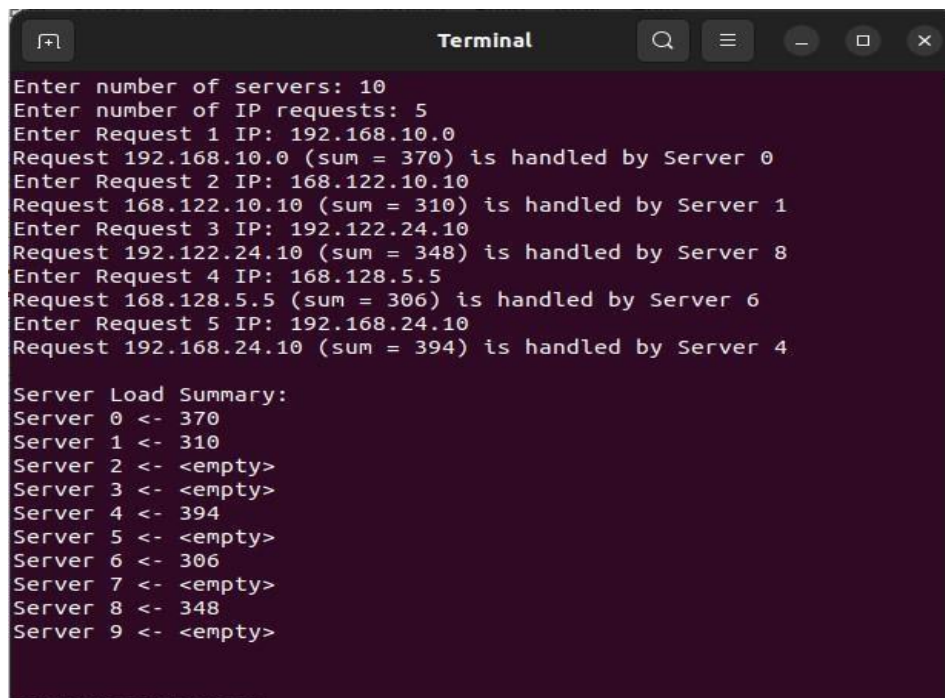
    LoadBalancer lb(n);

```

```
    for (int i = 0; i < r; i++) {        char ip[50];
cout << "Enter Request " << i + 1 << " IP: ";
cin >> ip;    lb.insert(ip);
    }

    lb.display();
return 0;
}
```

Output:

A terminal window titled "Terminal" with a dark background and light text. It shows the execution of a program. The user enters 10 for the number of servers and 5 for the number of IP requests. Then, five requests are entered with their IP addresses. The program outputs which server handled each request and the sum of the IP octets. Finally, it displays a server load summary for all 10 servers.

```
Enter number of servers: 10
Enter number of IP requests: 5
Enter Request 1 IP: 192.168.10.0
Request 192.168.10.0 (sum = 370) is handled by Server 0
Enter Request 2 IP: 168.122.10.10
Request 168.122.10.10 (sum = 310) is handled by Server 1
Enter Request 3 IP: 192.122.24.10
Request 192.122.24.10 (sum = 348) is handled by Server 8
Enter Request 4 IP: 168.128.5.5
Request 168.128.5.5 (sum = 306) is handled by Server 6
Enter Request 5 IP: 192.168.24.10
Request 192.168.24.10 (sum = 394) is handled by Server 4

Server Load Summary:
Server 0 <- 370
Server 1 <- 310
Server 2 <- <empty>
Server 3 <- <empty>
Server 4 <- 394
Server 5 <- <empty>
Server 6 <- 306
Server 7 <- <empty>
Server 8 <- 348
Server 9 <- <empty>
```