Experiment 4

Aim: Implementation of a Load Balancing Algorithm.

Theory:
Load balancing is a technique used in distributed systems to distribute workload among multiple nodes or servers to improve efficiency and avoid overloading any single node. There are several load balancing algorithms used in distributed systems, including:

- Round Robin: This algorithm distributes workload evenly across a set ofserversin a cyclic manner. Each new request is forwarded to the next server in the rotation.
- Least Connections: This algorithm selects the server with the fewest active connections to distribute the workload to. This ensures that heavily loaded servers are not further burdened.
- Weighted Round Robin: This algorithm assigns a weight to each server based on its capacity and distributes workload in proportion to the assigned weights. Servers with higher weights receive more requests.
- IP Hash: This algorithm uses the client's IP address to determine which server should handle the request. Requests from the same client are consistently routed to the same server.
- Least Response Time: This algorithm measures the response time of each server and directs new requests to the server with the fastest response time.
- Random: This algorithm selects a server at random to handle each new request.

This can be useful in systems where all servers have similar capabilities.

The choice of load balancing algorithm depends on the specific requirements of the distributed system. Factors such as the number of servers, server capacity, traffic patterns, and latency can all influence the choice of algorithm.

Program and Output:

```
import itertools
class LoadBalancer:
        def init (self, servers):
                self.servers = itertools.cycle(servers)
        def get_server(self):
                return next(self.servers)
if __name__ == '__main__ ':
        servers = ['server1', 'server2', 'server3']
        lb = LoadBalancer(servers)
# simulate 10 requests
for i in range(10):
        server = lb.get_server()
        print(f"Request {i} handled by {server}")
```

exp4.py > ...

```python
import itertools
class LoadBalancer:
    def __init__ (self, servers):
        self.servers = itertools.cycle(servers)
    def get_server(self):
        return next(self.servers)
if __name__ == '__main__':
        servers = ['server1', 'server2', 'server3']
        lb = LoadBalancer(servers)
# simulate 10 requests
for i in range(10):
        server = lb.get_server()
        print(f"Request {i} handled by {server}")
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Python Project> python -u "c:\Python Project\exp4.py"
Request 0 handled by server1
Request 1 handled by server2
Request 2 handled by server3
Request 3 handled by server1
Request 4 handled by server2
Request 5 handled by server3
Request 6 handled by server1
Request 7 handled by server2
Request 8 handled by server3
Request 9 handled by server1
PS C:\Python Project>
```

Conclusion:
Thus we had Successfully Implemented Load Balancing Algorithm.