Project Overview: Basic Load Balancer

In this project, I developed a basic Load Balancer (LB) using C, complemented by three server implementations in Python. The primary objective was to create a system that efficiently distributes incoming traffic among multiple servers, enhancing the performance and reliability of service delivery.

Key Components:

1. **Load Balancer Implementation (C)**:
    - Designed a load balancer that listens for incoming client requests.
    - Utilized a round-robin scheduling algorithm to distribute requests sequentially across the available servers.
    - Managed socket connections to handle multiple clients simultaneously, ensuring optimal resource utilization.
2. **Server Implementations (Python)**:
    - Developed three Python-based servers that interact with the Load Balancer.
    - Each server processes incoming requests, retrieves the required data, and sends back responses in the specified HTTP format.
    - Implemented error handling for invalid requests, returning appropriate HTTP status codes.
3. **Client Interaction**:
    - The Load Balancer communicates with clients using the HTTP protocol, receiving and forwarding requests to the appropriate server.
    - The servers respond with the necessary data, which the Load Balancer relays back to the clients.
4. **Testing and Validation**:
    - Conducted thorough testing using automated scripts to ensure functionality and performance under various load conditions.
    - Verified that the Load Balancer correctly routes request and handles simultaneous connections without error.