INTRODUCTION TO HLD (High Level Design)

WE WILL START AT 09:04 PM

Agenda:-

1. Introduction to HLD
2. case study → Pinboardin
   - ↳ 2005
   - ↳ Architectural changes
3. Importance of HLD
4. Concepts → Load Balance
   - ↳ <MAGIC TECHS>

Case Study:-

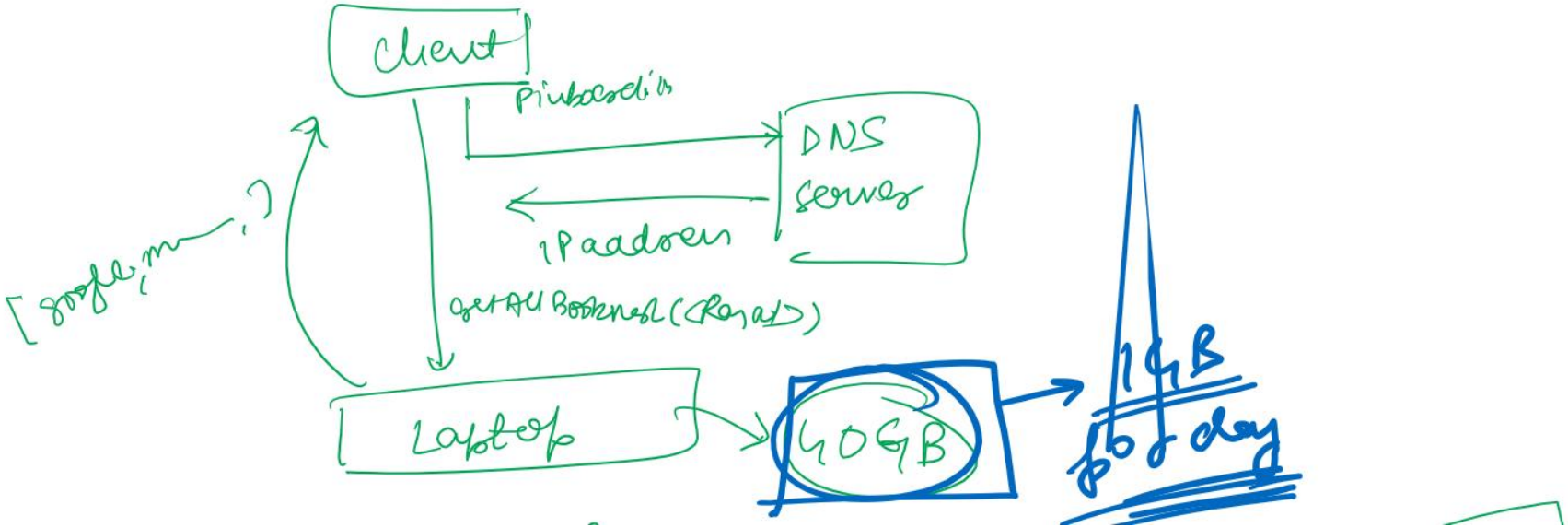
2005

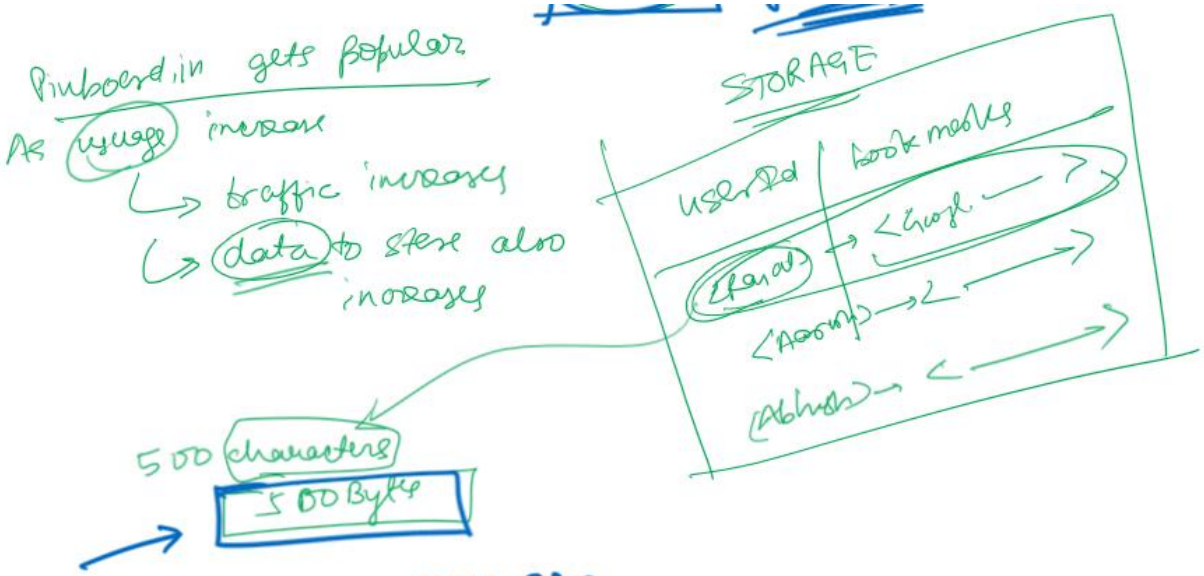
Pinboard.in

↳ bookmarking website

↳ addBookmark(user_id, bookmark)

↳ getAllBookmark(user_id)





Everyday → 2 million users are coming to their website

1 Million = 10^6
1 Billion = 10^9

Data storing everyday = 2 million * 500 Bytes

= 1000 Bytes × 10^6

= Bytes × 10^3 × 10^3

KB MB

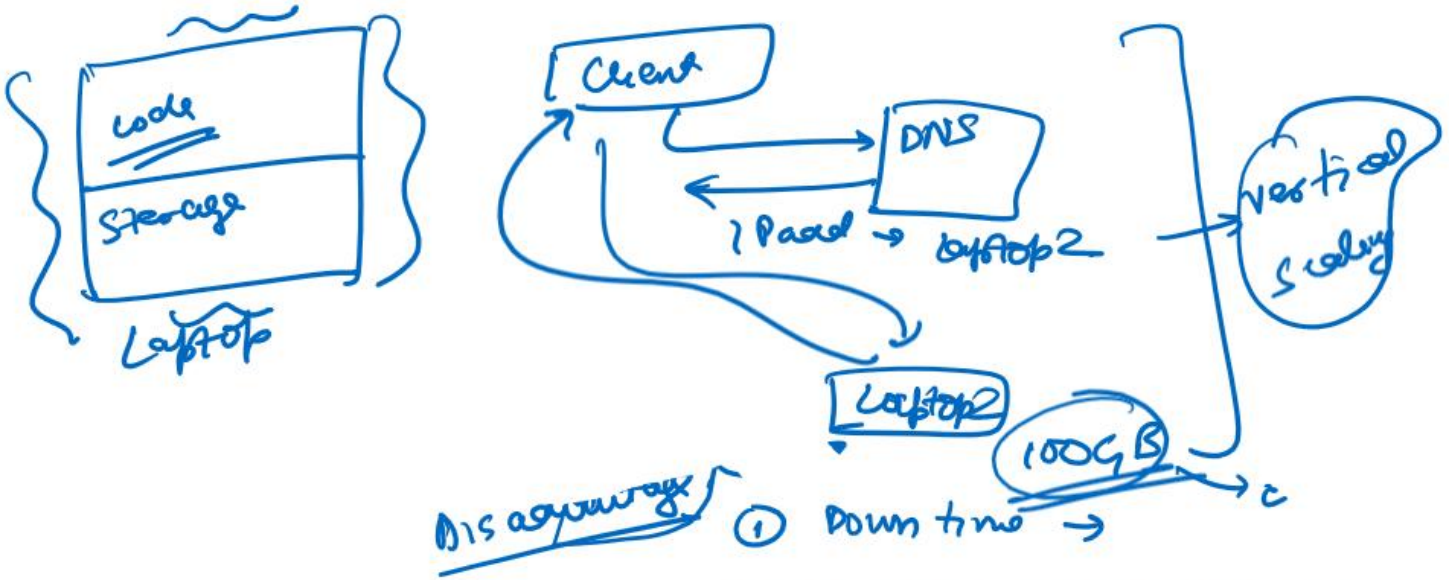
= 1000 MB = 1GB of data

everyday on the storage boxes of Pinboardin

Problem

Laptop → 40GB
1GB per day

Solution! → (1) Add memory

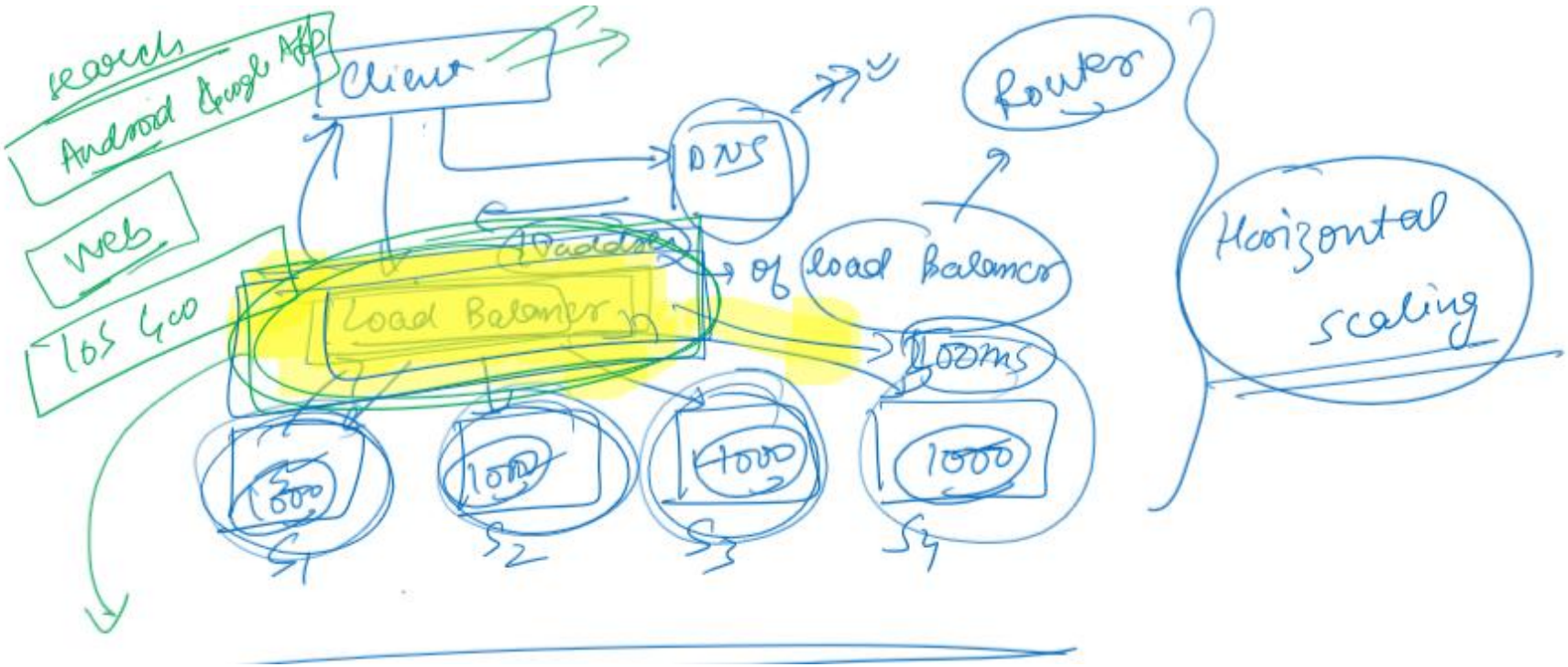


Disadvantage

(1) Down time →

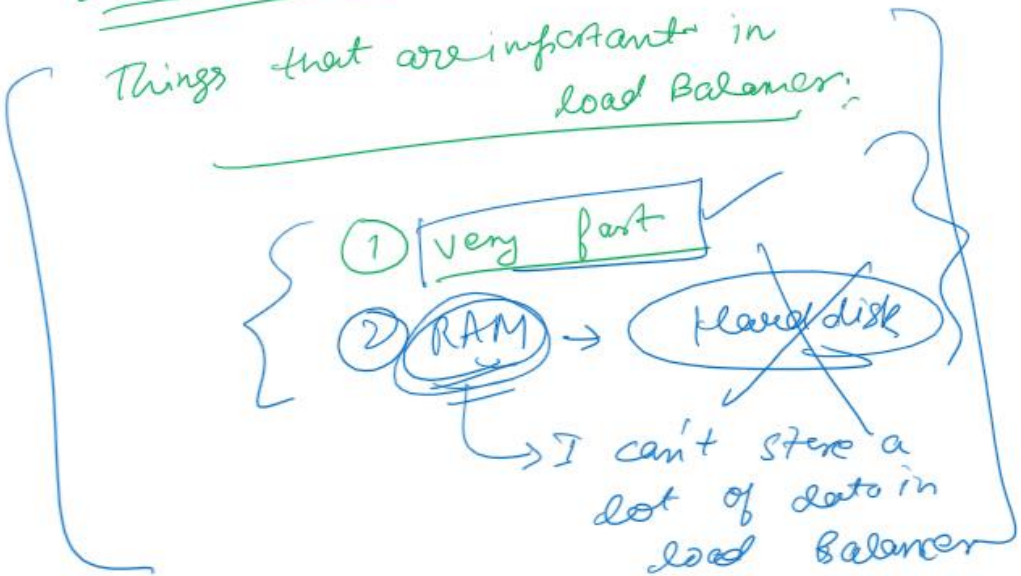
(2) Laptop2 memory will also get filled

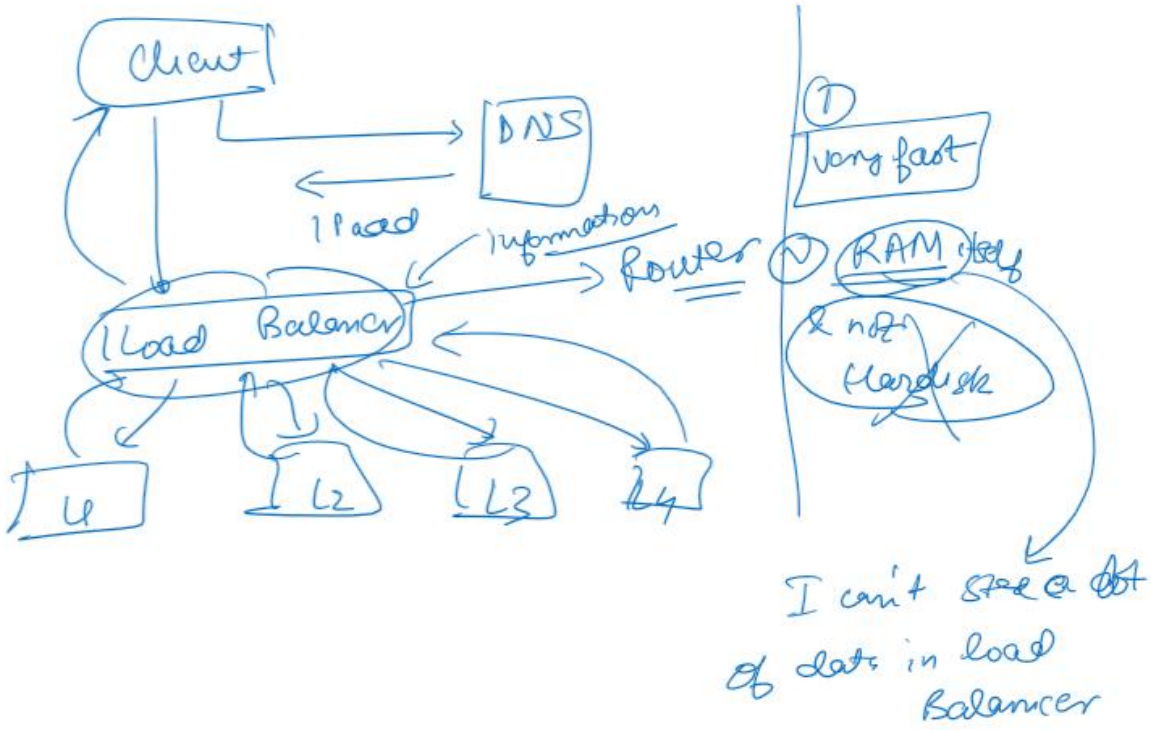
(2) Get more machines
(Add multiple machines)


Android App
Web
Client
DNS
Router
Load Balancer
Load Balancer
Horizontal scaling
1000
1000
1000
1000

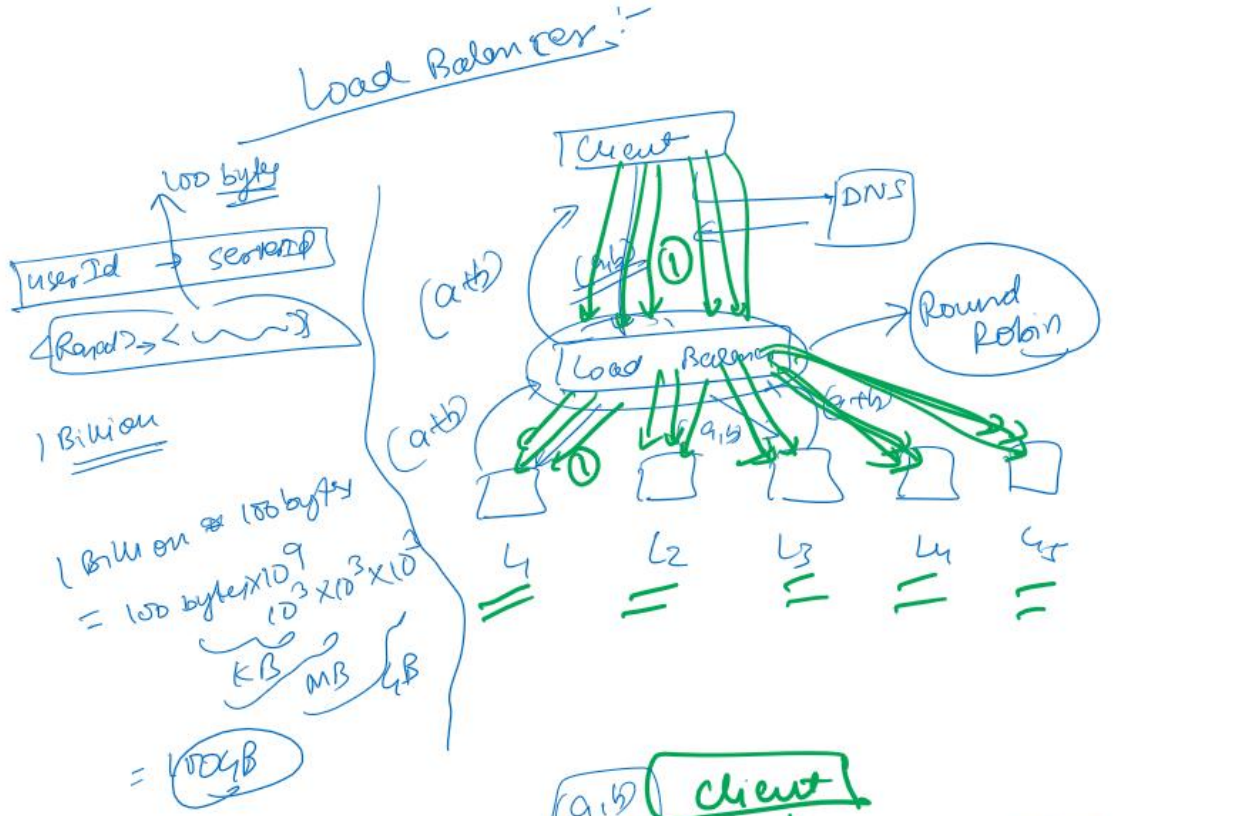
Load Balancer :-

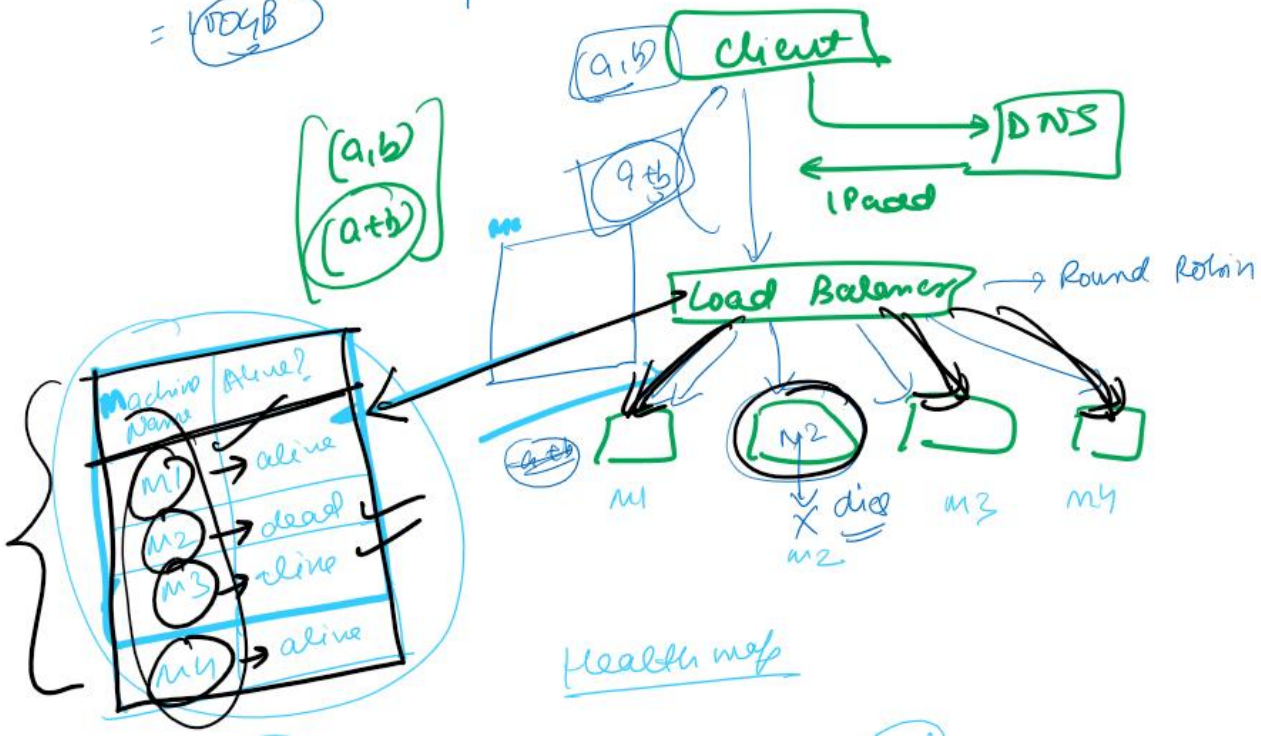
Things that are important in load Balancer :



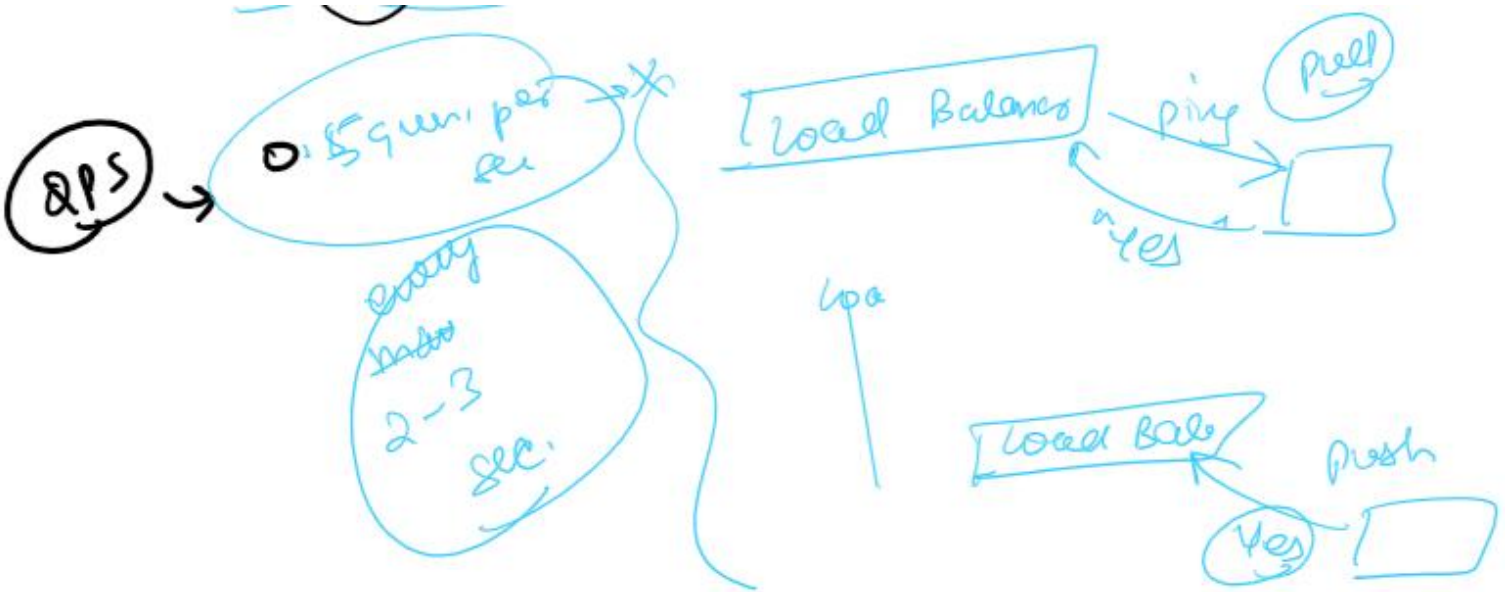


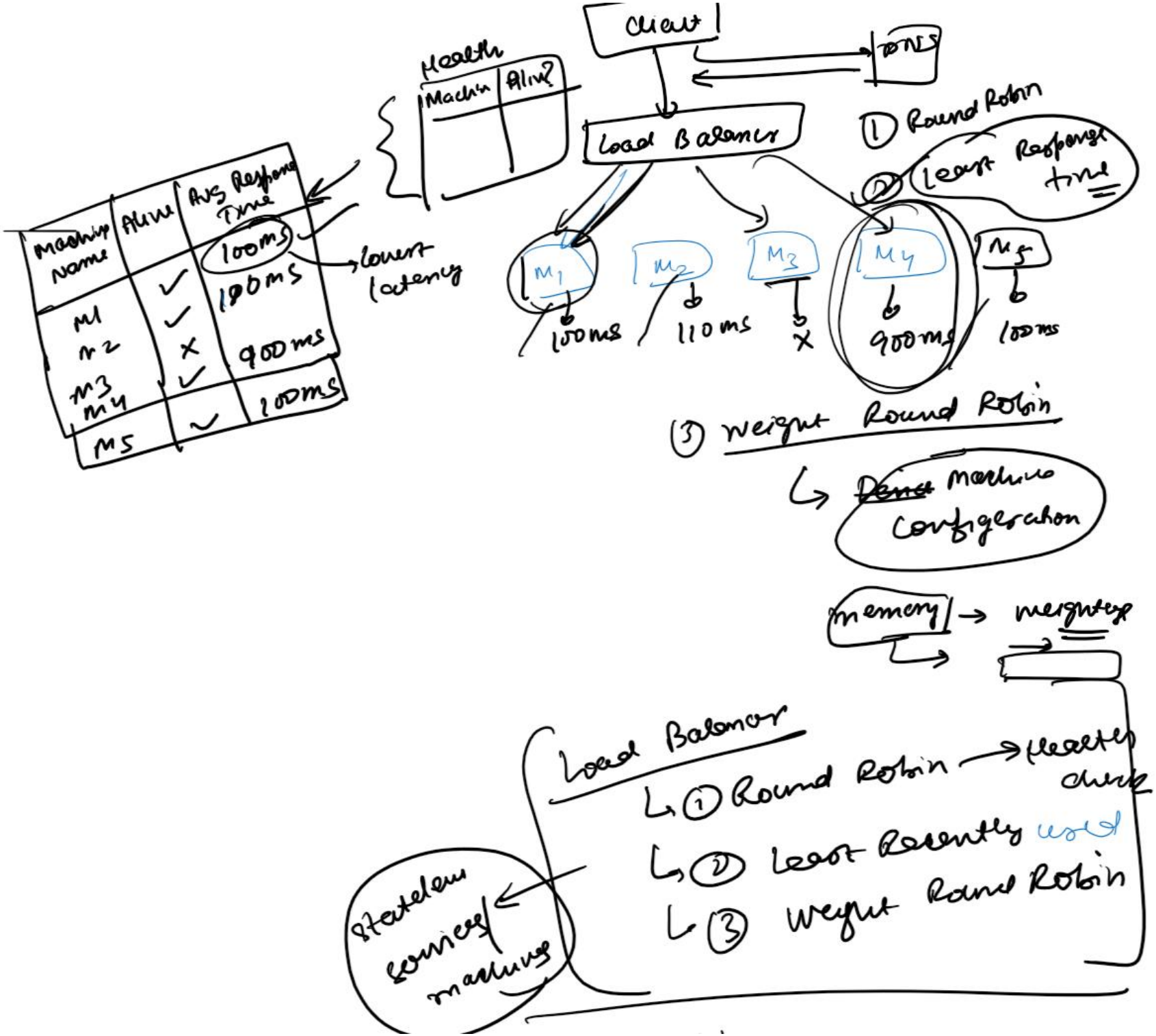
Load Balancer :-





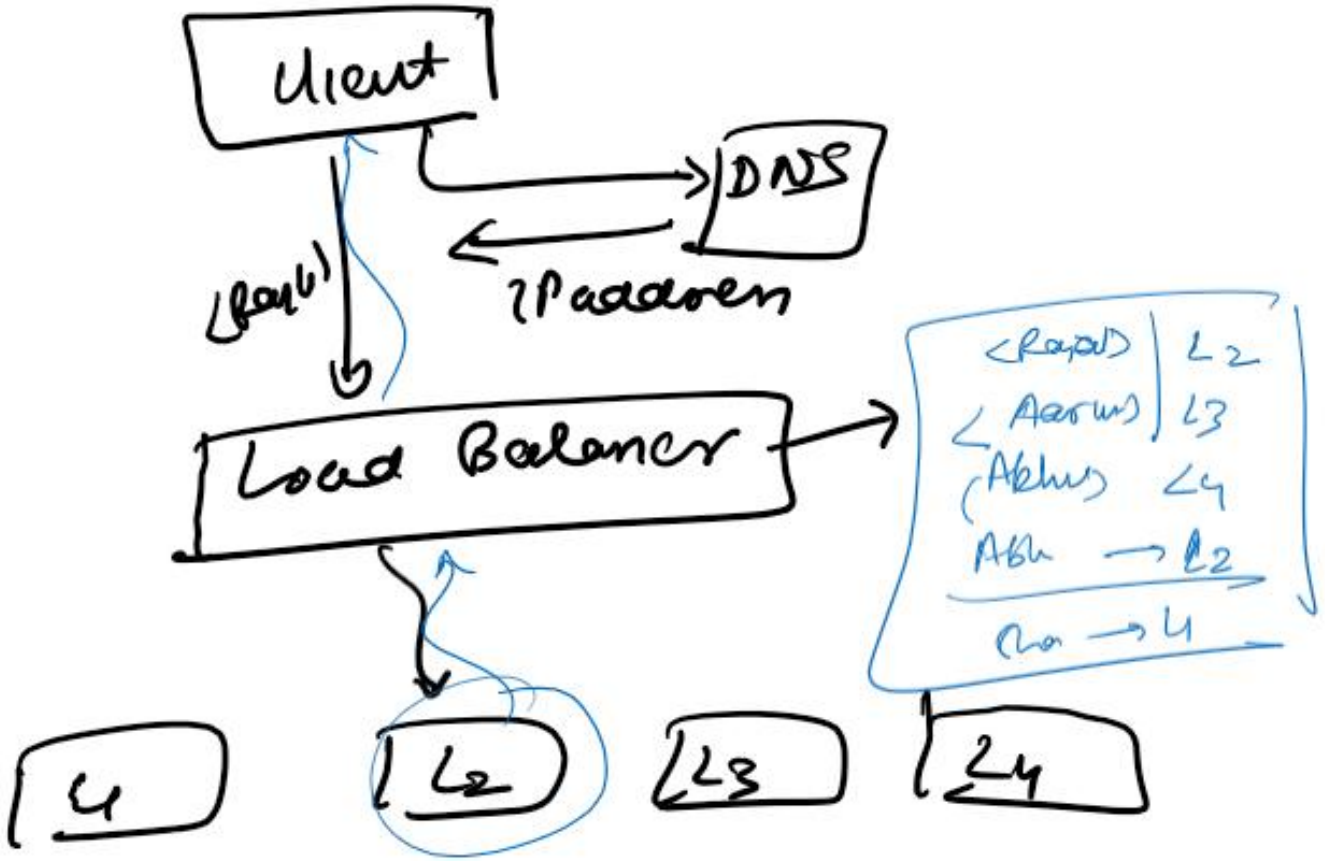
Health map


QPS
Load Balancer
ping
Poll
yes
query
2-3
sec
Load Bala
push
yes


Client
DNS
Health
Machine
Alive?
Load Balancer
1) Round Robin
2) Least Response time
Machine Name
Alive
Avg Response Time
M1
M2
M3
M4
M5
100ms
lowest latency
100ms
110ms
X
900ms
100ms
3) Weight Round Robin
machine configuration
memory
Load Balancer
1) Round Robin
2) Least Recently used
3) Weight Round Robin
Stateless
servers
machines

Pinboard.in

NIO



Approach ① :-

userId → serverId

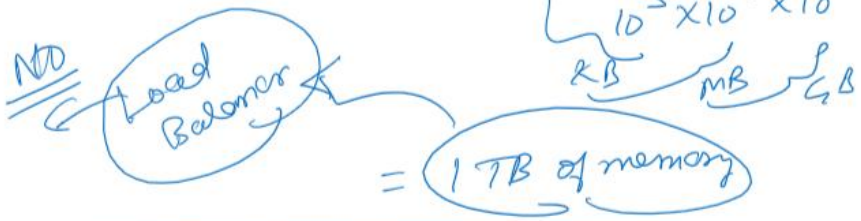
100 Bytes

10 billion * 100 Bytes =

1000 × Bytes × 10^9

$10^3 \times 10^3 \times 10^3$

KB MB GB



Approach ② :-

Nikhil → take mod

serverId = userId % (N) → no. of servers

→ 5 servers → N = 6

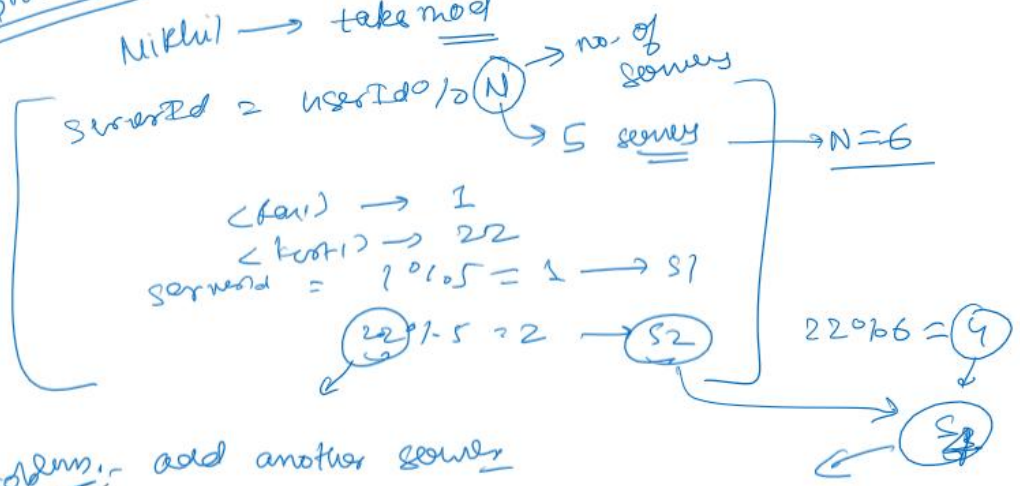
<fail> → 1

<test> → 22

serverId = 1 % 5 = 1 → S1

22 % 5 = 2 → S2

22 % 6 = 4



problem :- add another server

N → N+1

(userId % N) ≠ (userId % N+1)

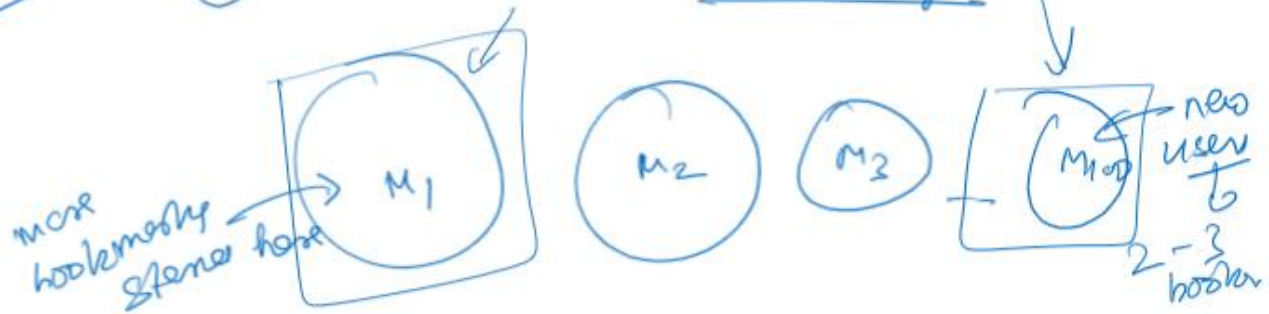
if userId > N

Approach ③ Abhishek

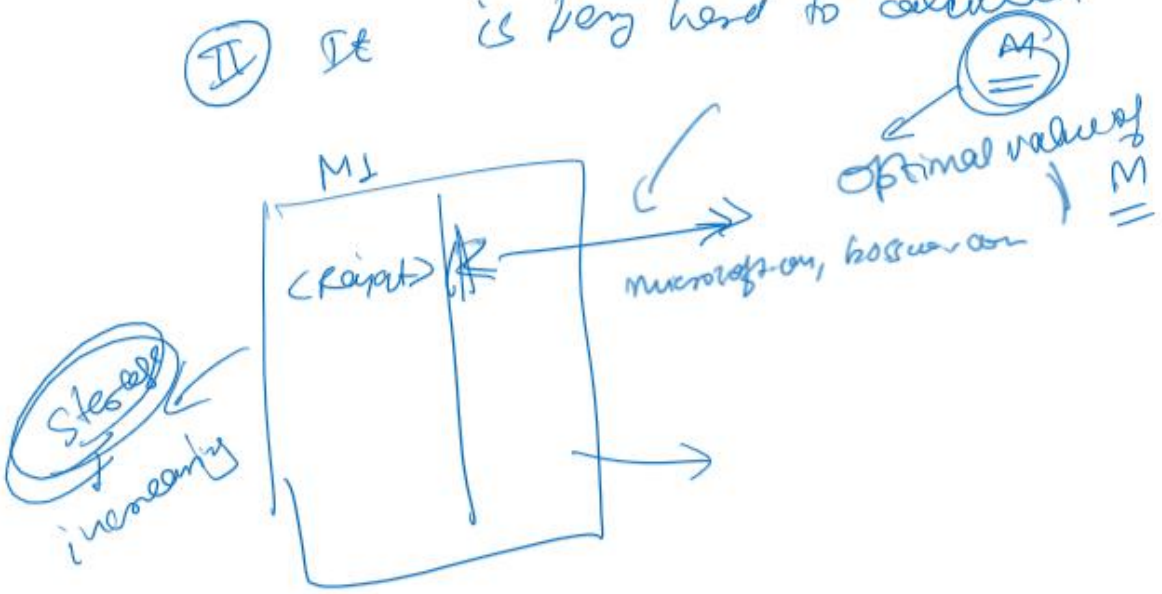
1 → M M+1 → 2M 2M+1 → 3M

Problem :-

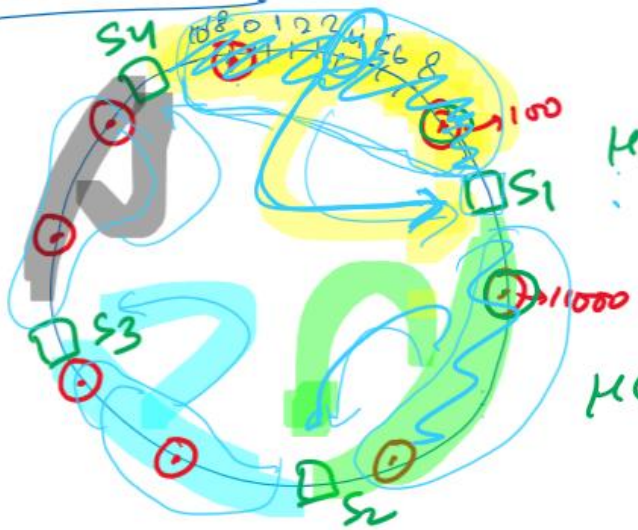
(I) Traffic is distributed unevenly



(II) It is very hard to calculate

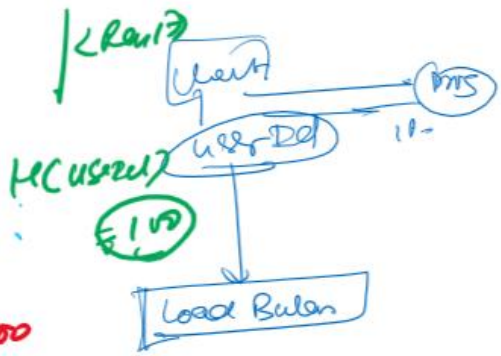


Approach 4 :- Consistent Hashing

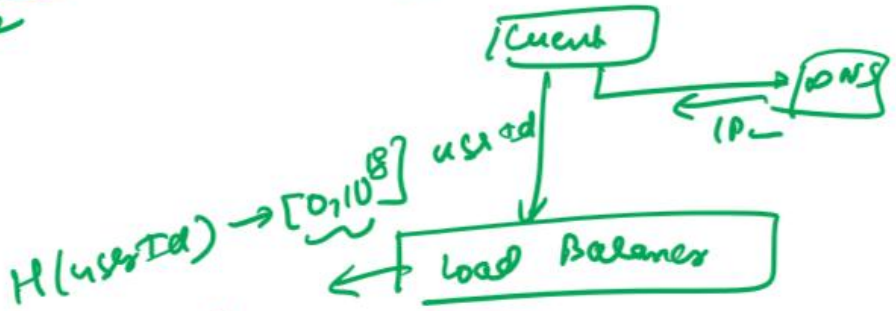


H(userId) → [0, 10^18]

H(serverId) → [0, 10^18]

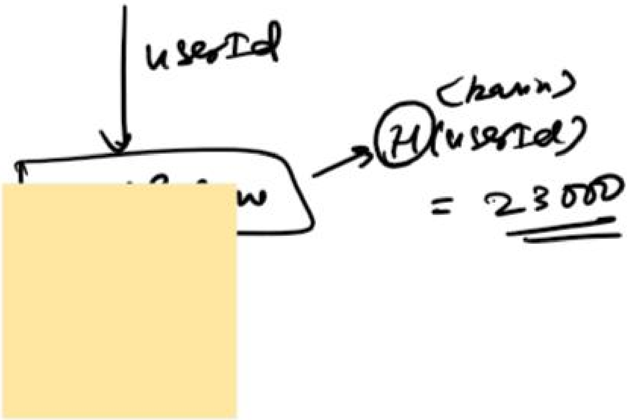


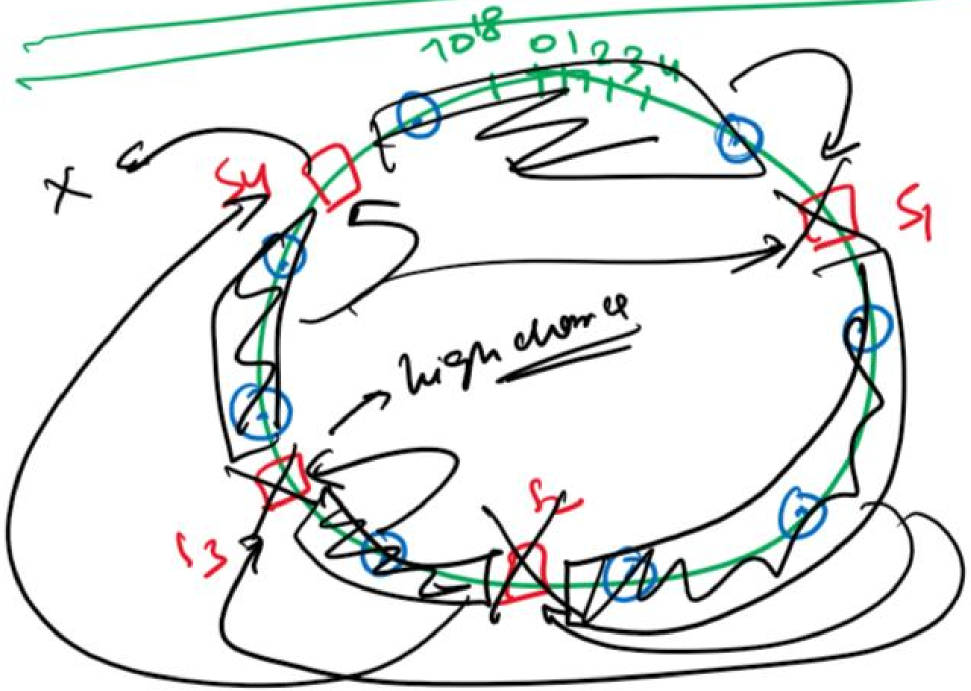
H(ReqHandler) = 11000



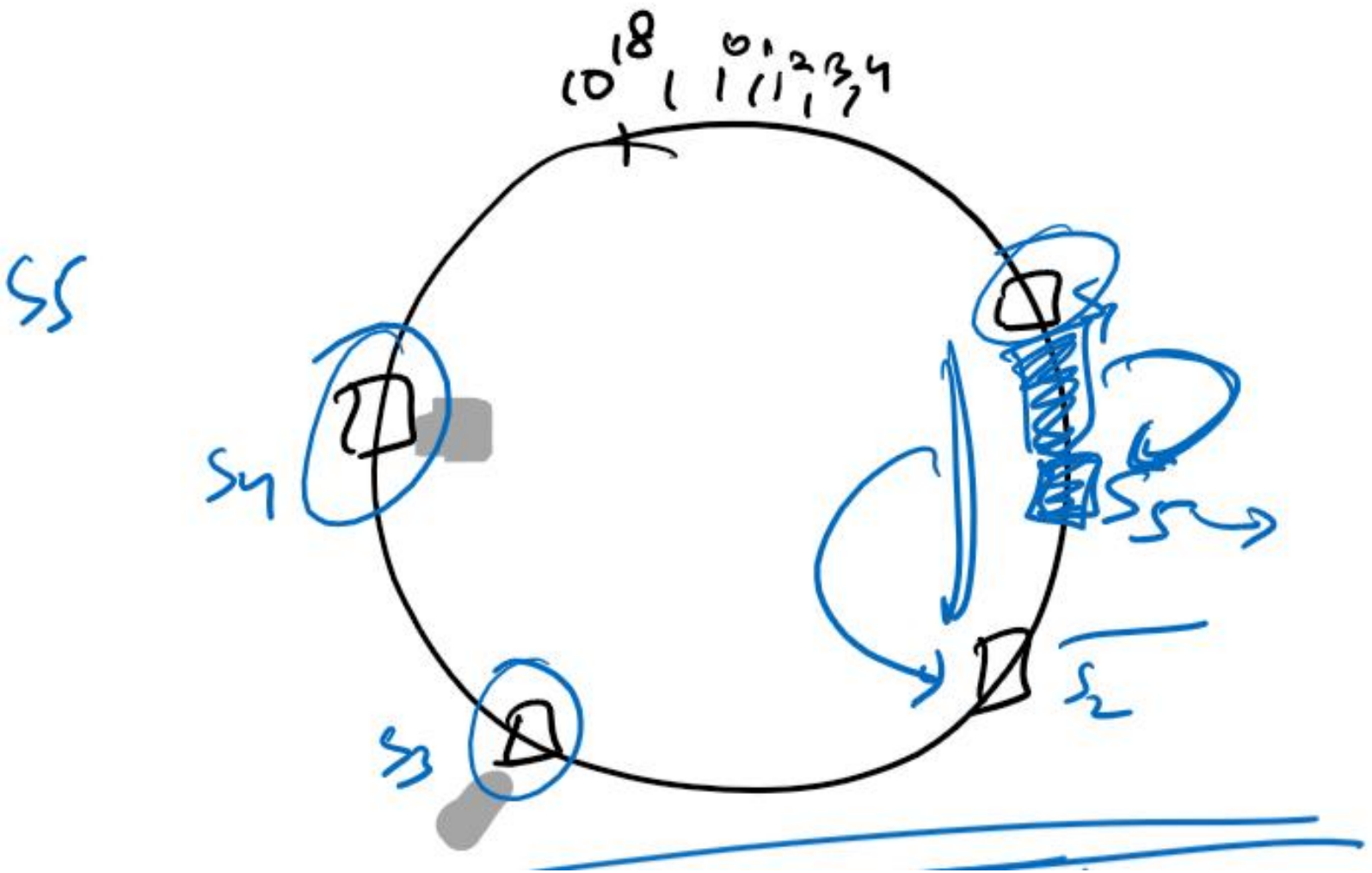
Problems :-


1018
0 1 2 3 4 5
S1
S1
double the load
will also fail
23100
S2
storage
compute


userId
(hash)
H(userId)
= 23000


1018
0 1 2 3 4
S1
S1
high chance
S2
S3

Case (ii) :- when we add a new machine :-



Modified consistent hashing

(Real world scenario)

create virtual copies of your servers.

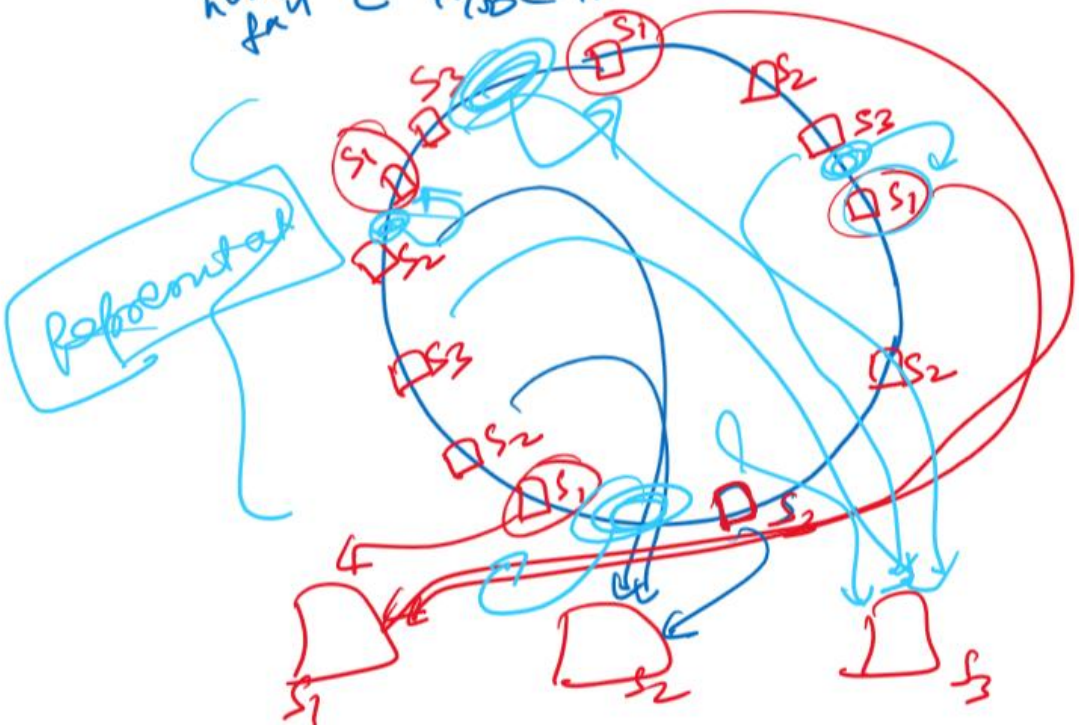
10 hash fn

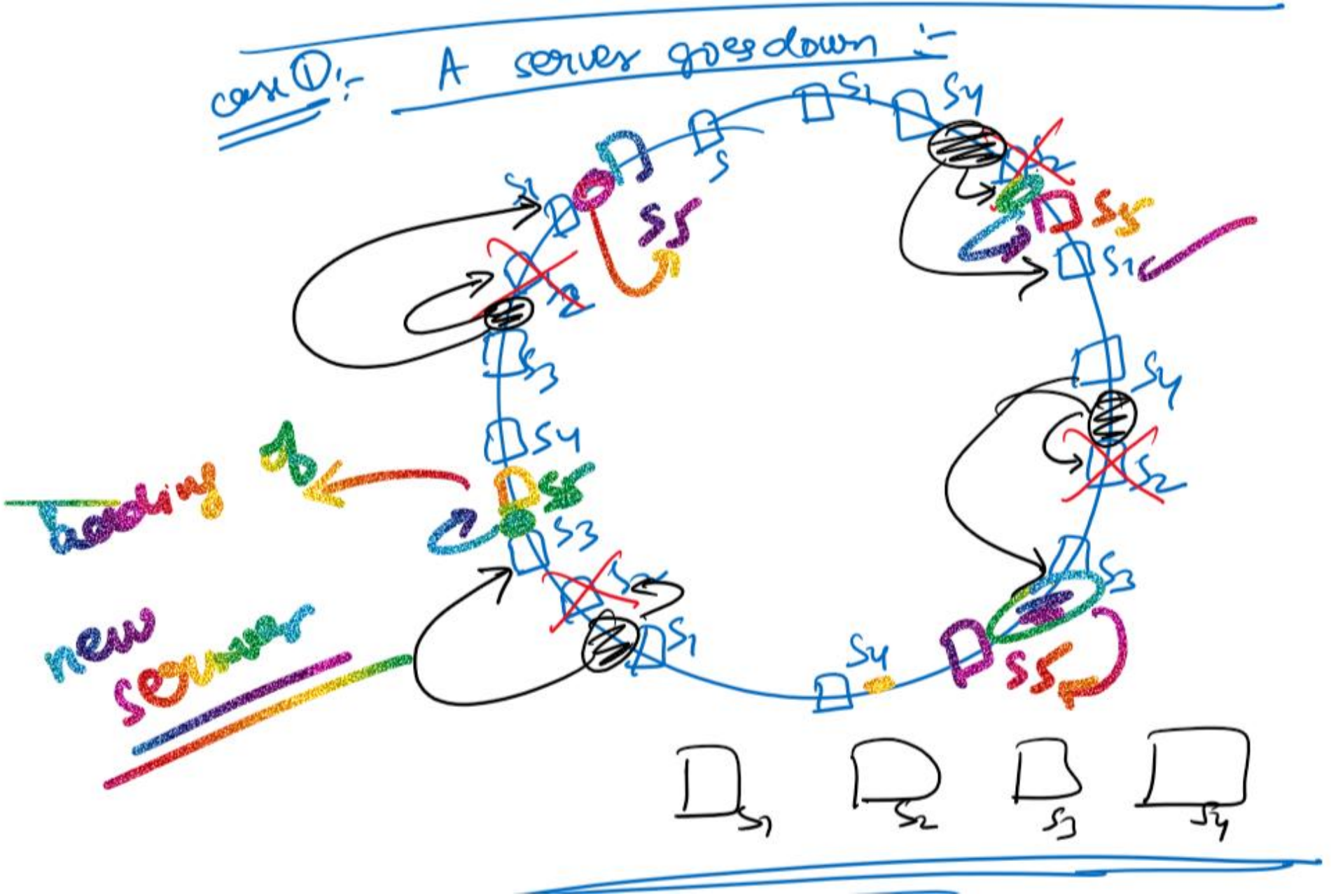
$H_1(S_1) \rightarrow [0, 10^{18}]$

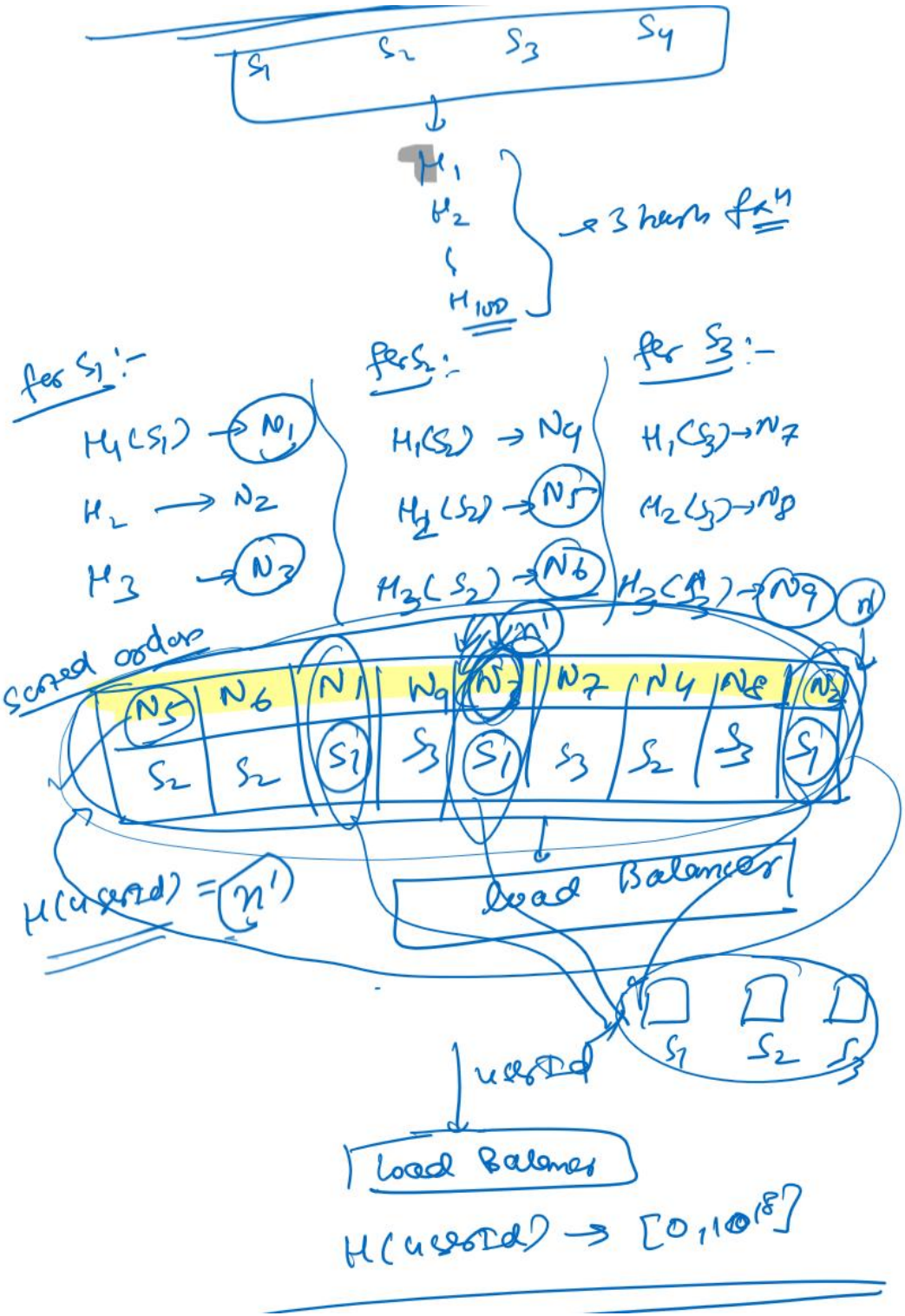
$H_2(S_1) \rightarrow [0, 10^{18}]$

⋮

$H_{10}(S_1) \rightarrow [0, 10^{18}]$




case 1:- A server goes down :-
S1
S4
S2
S3
S4
S1
S4
S2
S3
S1
S4
S2
S3
S4
S1
S2
S3
S4
looking
new server


S1 S2 S3 S4
H1
H2
H100
3 hash fxn
for S1 :-
for S2 :-
for S3 :-
H1(S1) → N1
H2 → N2
H3 → N3
H1(S2) → N4
H2(S2) → N5
H3(S2) → N6
H1(S3) → N7
H2(S3) → N8
H3(S3) → N9
n
n'
sorted order
N5 N6 N1 N9 N3 N7 N4 N8 N2
S2 S2 S1 S3 S1 S3 S2 S3 S1
H(userId) = n'
Load Balancer
S1 S2 S3
userId
Load Balancer
H(userId) → [0, 10^18]

Summarize:-

(1) Introduction to HLD

(2) Case Study: Pinboarding

- (1) Single laptop
- (2) Insufficient storage?
  - (1) Vertical scaling
  - (2) Horizontal scaling
    - Load Balancer
      - Stateless server
        - (1) Round Robin
        - (2) Least [illegible]
        - (3) Weighted Round Robin
      - Stateful server
        - (1) [illegible]
        - (2) [illegible]
        - (3) [illegible]
          1 — [illegible], [illegible] — 2
        - (4) consistent

(5) [illegible]