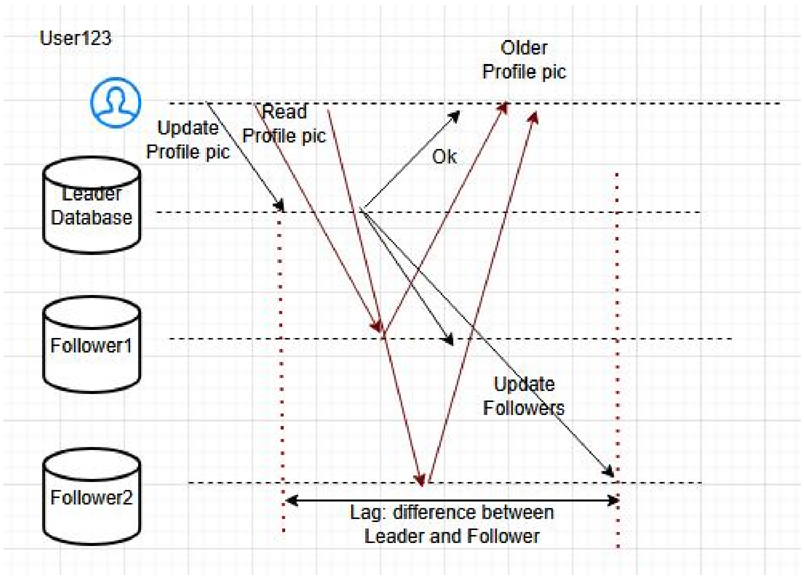
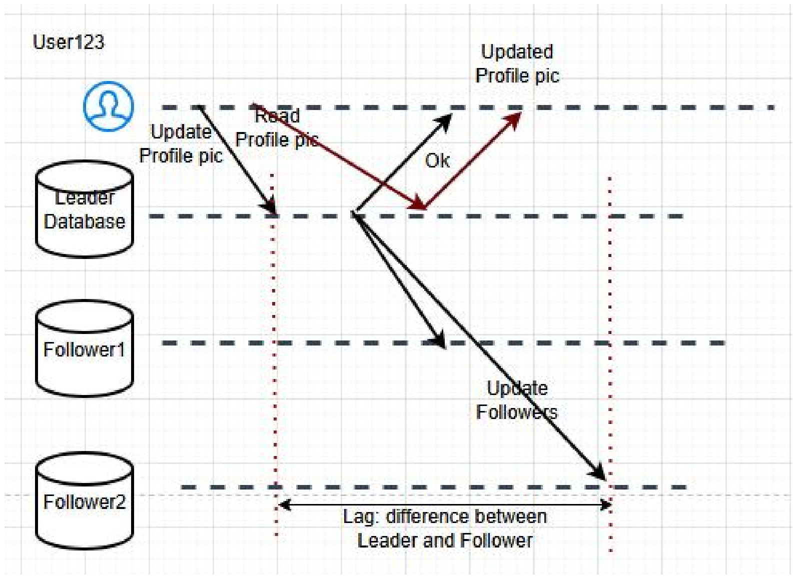
Replication3:

Replication Lag
We update Leader on time then to update Follower nodes, there will be some time-lag. This is Async, we respond Ok to User after updating Leader only

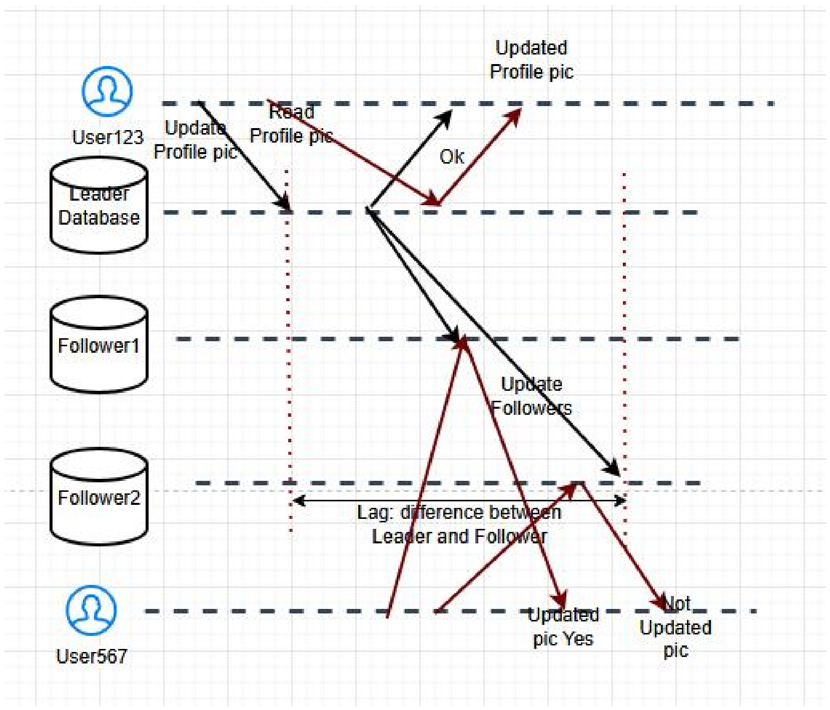


We cannot avoid LAG, it will happen and somehow we have to manage it. We don't want users to see the older data. Lets discuss some approaches to fix these issues

-- Read Your Own Write:



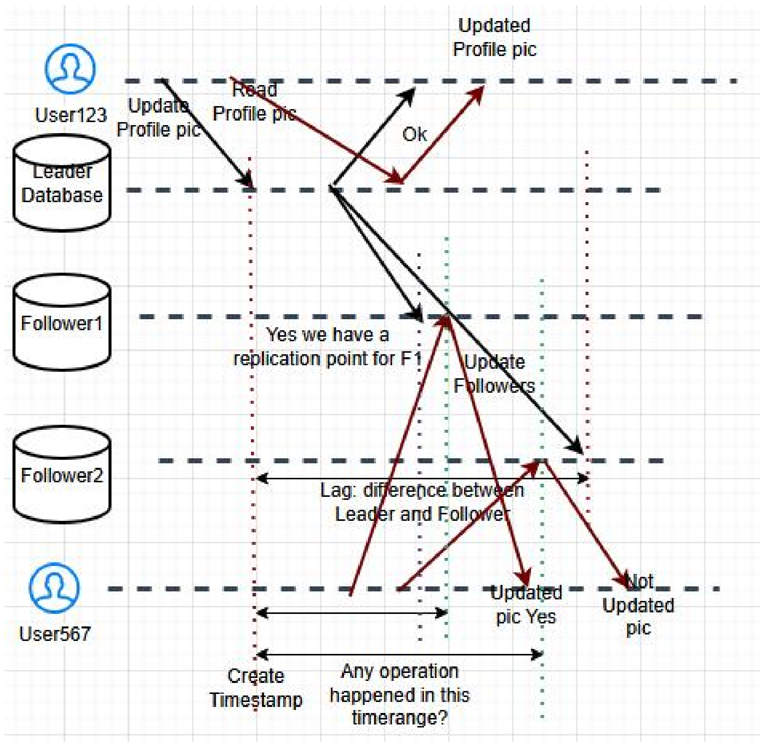
Read operation will go to the Leader only so this will fix the issue and the User always sees the Updated pic. When we have more Read operations than Write, we can implement this.



User567 can see the Updated pic of User123 at point1 not point2. Still it is considered as Ok.
We could have a hybrid approach also, say we will read from the Leader only if the User updating the value is seeing the value. For other users, we can read from the Followers.

Read from Leader:
- Always updated
- In Write-intensive Application, our Leader could break

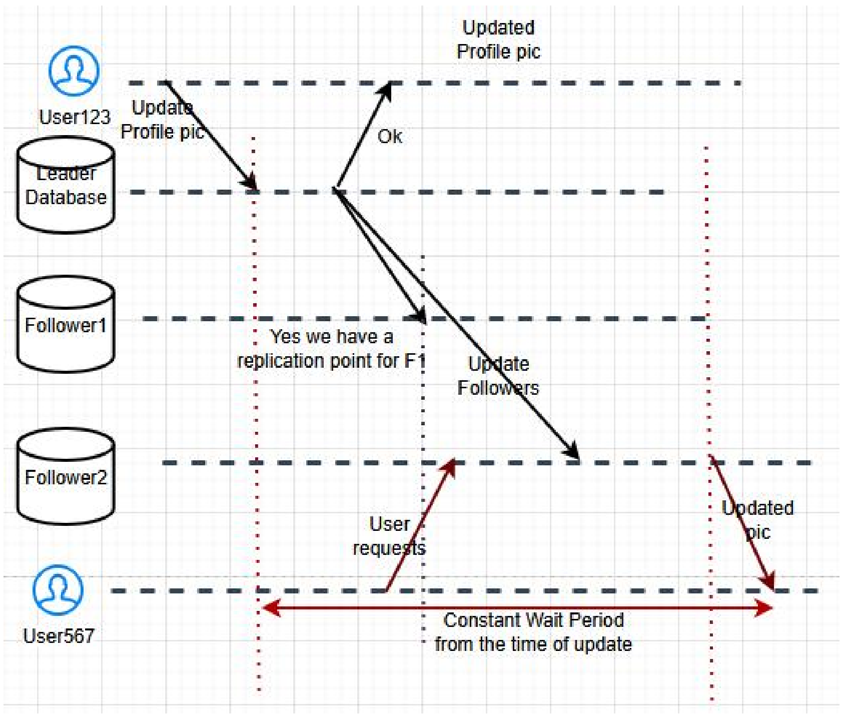
-- Queries via Timeline:
Whenever the Leader is updated, we will create a Timestamp at that time and we will see whether there is any updation in our Followers between the Timestamp and time of User567 checking.



Yes there is a Replication point within the timeline for F1 but not for F2. that means, User567 can read from F1 and not F2. this is Query on Timeline approach. We will allow User to read only if replication point is present within the timeline.

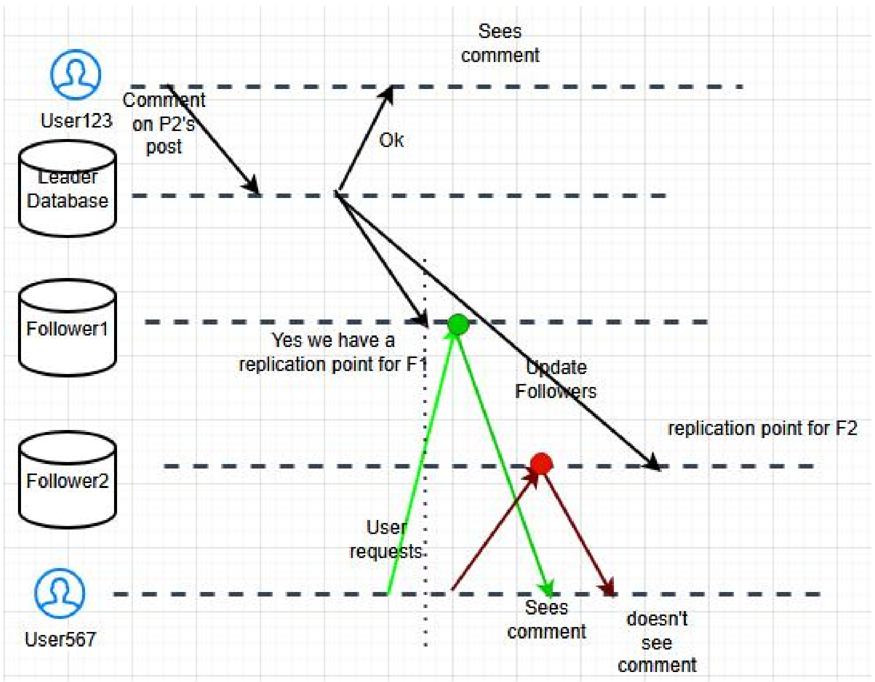
-- Redirect our query to next node:
User567 queries ---> F1 ---> redirects to F2 if it fails ---> it could redirect to Leader also to get some response. Basically, User queries go into one node if it is not updated, it gets redirected into another node and so on. Drawback is it will increase Latency

Time-based read:



We have a standard wait period since the time of updating the Leader, before we read from Followers, this will guarantee we always read the Updated value. Based on SLA, SLO and SLI, we can determine the wait period.

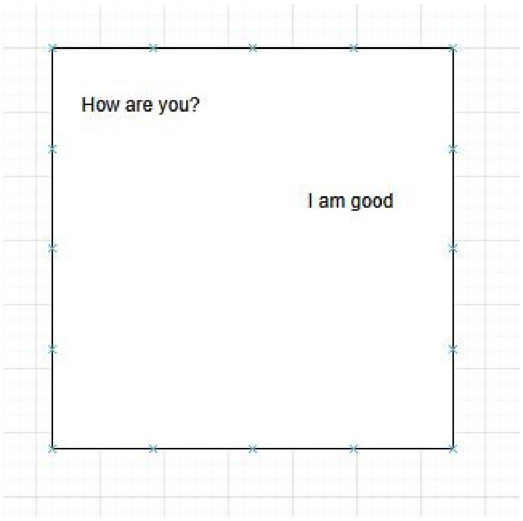
First issue: we wanted to read only the updated value
Second issue: Monotonic read



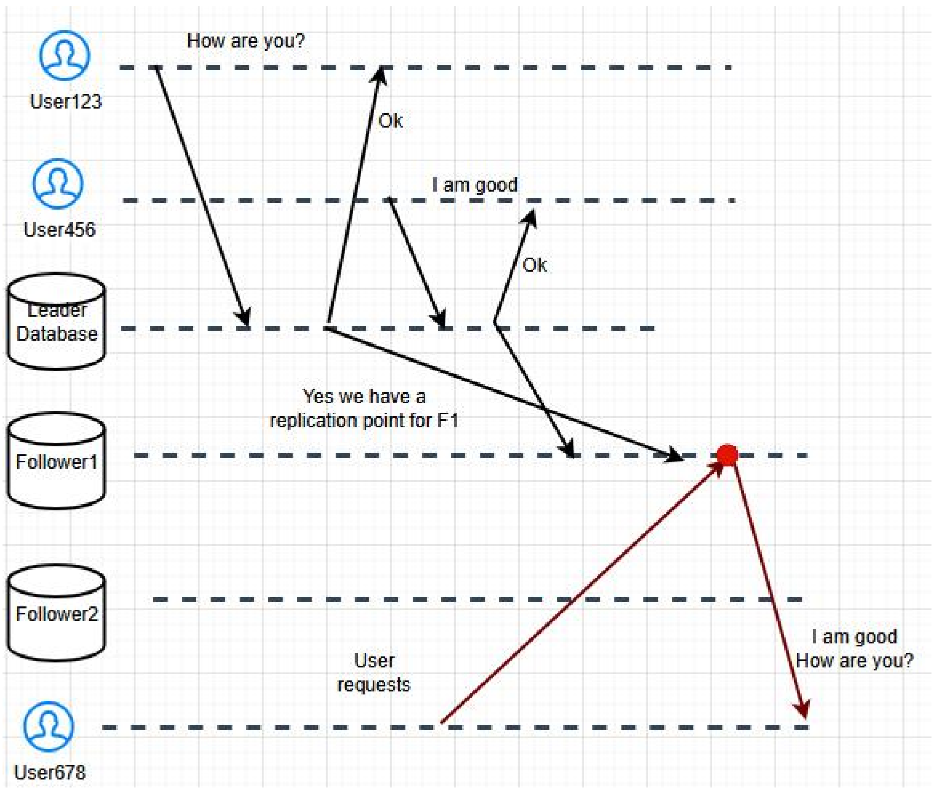
First time, User567 is directed to Follower1 then he sees the comment, next time, the read request is directed to Follower2 where he doesn't see the comment since replication was not done at that point.

Monotonic read is the solution to this problem. The solution is bind every user to a single node only from there they can read, they cannot read from any other node. So User567 will be bind to only Follower2, so Comment appearance and disappearance will never happen. User will be bound to read from particular Follower only.

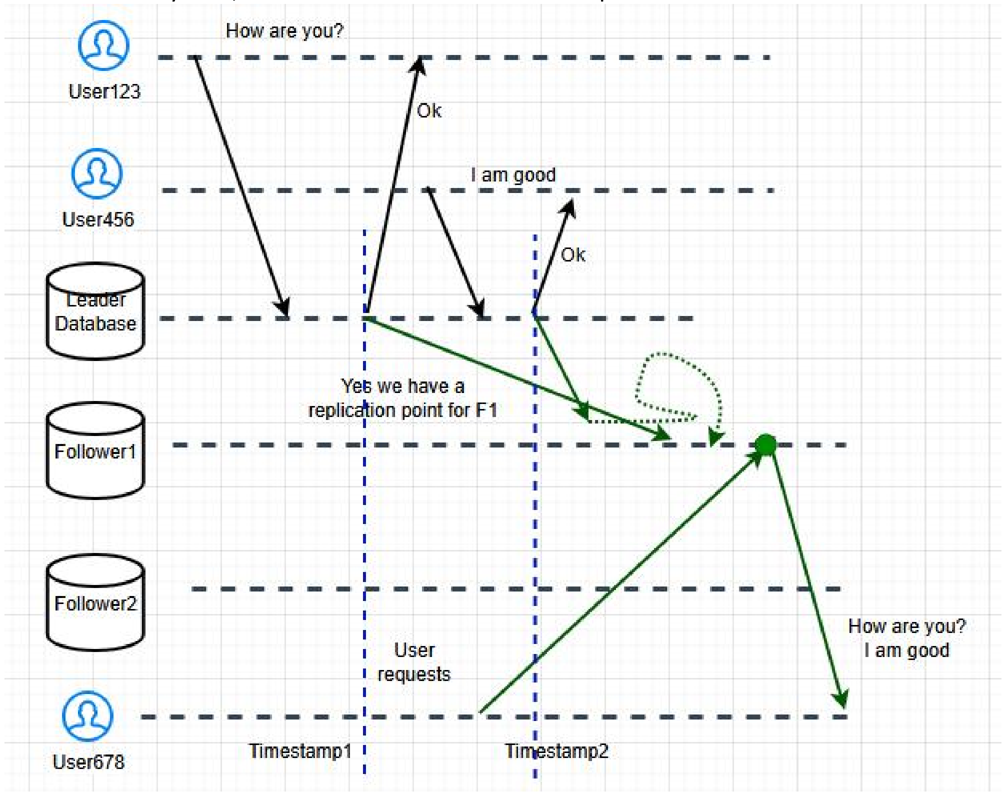
Consistent Prefix:



Consider a chat messenger, first message is "How are you?" and second message is "I am good" both must be consistent. Sequence should be maintained as well.



In a group chat or chat room, when third user User678 is trying to read, they read "I am good" first then "How are you?", which is incorrect and this is the problem.



Solution is we get the Timestamps from the Leader commits, so before replicating in the Follower, we compare the Timestamps and we swap them to maintain proper sequence.