The goal of the System design for Web Privacy feature is to provide a way for a "thin client" connected to a server to generate and return query results in a reliable & trustworthy way.

Following three issues must be taken into account :

1) The system depends on a server where queries will be sent and the server will add the transactions to a memory pool, where a thread will loop through the pool and execute them.

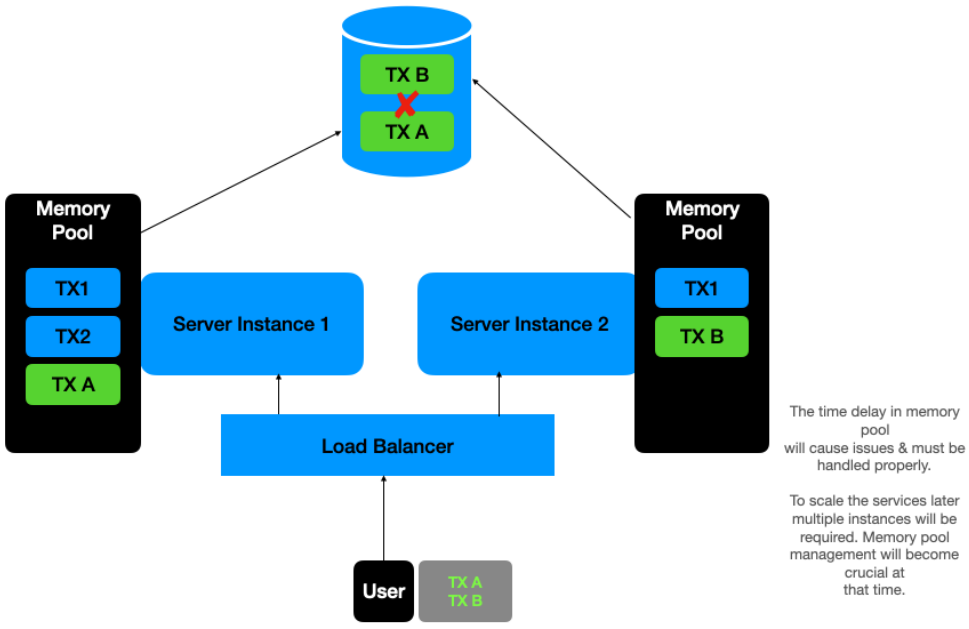
This has two main issues :

a) **Potential DOS** **HIGH**

In case a bad actor creates a lot of transactions in a short amount of time, the server can face a DOS like scenario where multiple transactions will fill the memory pool of the server.

b) **Potential Synchronization Issue** **HIGH**

To scale this up to execute a large number of transactions at some point in future multiple instances of the server must be run. So two continuous transactions (A -> B) can exist in multiple memory pools on the server instances (in case a user tries to send multiple transactions). In that scenario there will be no guarantee that transactions B will not be executed before transaction A and can lead to failed transactions. This scenario must be handled carefully.



2) **Proof of valid response** **NOTE**

The client should also receive a "proof" along with the transaction response to make sure that query is returned by an "honest" server and the server is not compromised in any case. (The suggestion would be that this proof can be based on Merkel tree and validating server response from different servers running the full node, the client should also do a conflict resolution in case the proof is not valid)

3) **Storing Private keys** **NONE**

Storing keys on the server is not ideal and QURAS should educate the end user about how the keys are generated and destroyed.
TECHFUND suggests keeping an eye on homophoric encryption systems. Using a homomorphic encryption scheme, the data owner encrypts their data and sends it to the server. The server performs the relevant computations on the data without ever decrypting it and sends the encrypted results to the data owner. The data owner is the only one able to decrypt the results, since they alone have the secret key. Using such method might help prevent sending keys in future.