Ledger processor task

Background:
PUBLC's token distribution is based on clicks on content from the platform.
For every content click, the click reward is divided among the people who took part in that click, i.e - the content's creator, the content's curator, etc.
The processor's job is to take the click reward distribution and create transactions to the click beneficiaries, adding the tokens they earned to their wallets' balance.

The processor listens to a queue with the click info, grabs the action, creates the transactions, and updates the beneficiaries' balance in the ledger bucket.

Task:
Please implement the following service using Nodejs and Couchbase.

- This service should be running constantly and check for actions registered to it in a Couchbase bucket called "queue", by this reference: "action:<actionId>".
- For each action it increases the balance in the wallets by this reference: "balance:<hash>". This document resides in a couchbase bucket called "ledger". The hash is retrieved from the action in the queue.
- In your solution, consider that the processor will scale and have multiple instances running simultaneously, all grabbing actions from the same queue.
- If and only if all wallet's balance were successfully registered, it should remove the document from the queue bucket.
- Not all users' would already have an existing wallet in the ledger, wallets are created only on the first transaction they recieve. So need to check if the hash has a balance, and create a new one if it doesn't.

The database should support the following document structures:

Ledger bucket:
"balance:<hash>" => <balance>

Queue bucket:
"action:<actionId> => { (add more fields to this document if necessary)
"beneficiaries": [
<hash1>: <share>,
<hash2>: <share>,
etc...
]
}
*Consider that 0<share<=100

In this task we have excluded the part where the price is reduced from the total supply, but please consider the fact that the total supply is 100,000,000,000 (100 billion) tokens and that we support 8 decimal points accuracy (like bitcoin).

Please pay attention to:

- Be extremely accurate. That means, under NO CASE (including all edge cases), any wallet should get more or less than what it deserves.
- Pay attention to atomicity, everywhere that is necessary.
- Index wherever is needed (Bonus if you configure the indexes with high performance)
- Please supply a github/bitbucket/etc. link with a detailed README guide with instructions how to run the app.

Please let us know if you have any questions.

achi@publc.com 0524233307
lior@publc.com 0546873694

Good Luck :)