ShopUp - Planned design

Afonso Pinto
up202008014@fe.up.pt

Diogo Neves
up202006343@fe.up.pt

Fábio Teixeira
up202006345@fe.up.pt

Tomás Cabral
up201800700@fe.up.pt

October 2023

1 Introduction

In this project, we will develop a shopping list application with a local-first approach. The application operates on the user's device, enabling data storage and retrieval locally. Additionally, it has a cloud component to facilitate data sharing among users and offer backup storage. This system will be inspired by the Dynamo paper by Amazon. To help deal with concurrency, our application also involves an implementation of Conflict-free Replicated Data Types (CRDTs).

2 Project plan

The first week of the project was dedicated to project planning, which involved problem analysis, framework and programming language selection, and the distribution of tasks for the upcoming weeks. The remaining time will be used for implementation work. Initially, we will use libraries such as Automerge to ease the development process, as well as get a feel for how the system will function as a whole. We will also store all data on a single server. Eventually, the plan is to implement our own version of a CRDT, as well as scale the back end horizontally using techniques such as Consistent Hashing, as well as a load-balancer so that the system could potentially handle millions of concurrent users. After much deliberation, we decided to use the Java programming language to develop the project, as it has a rich set of libraries and frameworks that can help with implementing distributed systems.

3 Implementation

3.1 Consistent Hashing and Ring Topology

Inspired by Dynamo, our system will implement consistent hashing to distribute data across a cluster of nodes, forming a ring topology. A hashing function will be applied to both data items and nodes, placing them on the ring based on their hash value. This approach minimizes reorganization when nodes are

added or removed and ensures that the data is evenly distributed. Each node will be responsible for a segment of the hash space, and data will be partitioned accordingly.

3.2 Replication Strategy

To ensure high availability, our system will replicate data across multiple nodes. A replication factor, N, determines the number of nodes across which each item will be replicated, such that when an update occurs, it first goes to the primary node, which then forwards the update to the next N-1 nodes in the sequence.

3.3 Failure Handling

Our system will be designed to handle partial failures gracefully, ensuring that the system remains operational even when some nodes fail. The system will employ a gossip-based protocol for failure detection and recovery, similar to Dynamo. In the event of a node failure, the system will automatically re-replicate the data items to maintain the desired level of redundancy.

3.4 CRDTs for Local-First Design

CRDTs will be integral to our local-first design, enabling seamless synchronization and conflict resolution. Our system will use CRDTs to manage the state of the shopping list, ensuring that changes made offline can be merged without conflicts. The CRDTs will be designed to integrate with the consistent hashing and ring topology, ensuring that local changes can be propagated and merged across the distributed system. We will have a copy of our data locally in JSON that the user will be able to edit whether online or offline. If online, the data will be constantly syncing with the copy in the server and any merge conflicts will be handled accordingly.

4 Conclusion

The planned design of our shopping list application takes inspiration from the Dynamo paper by Amazon to create a robust, scalable, and user-friendly local-first system. By leveraging a vast array of techniques, we aim to provide a seamless experience for users, with or without connectivity, while ensuring data integrity and availability.

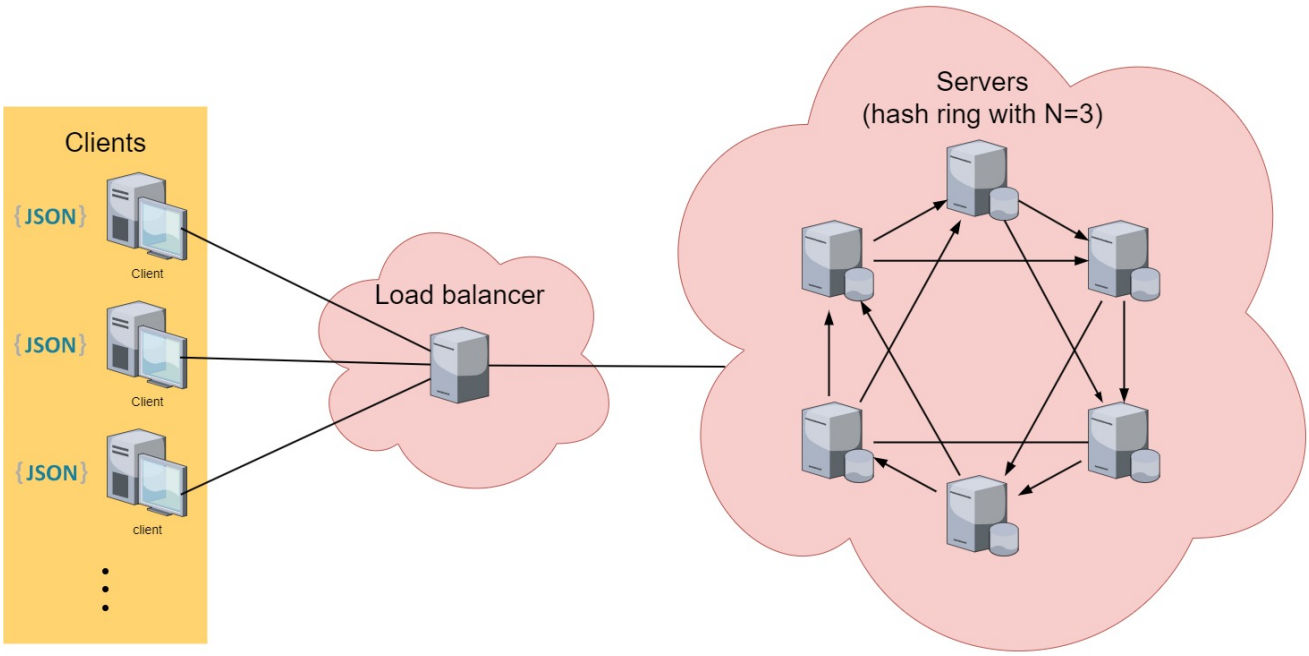

Figure 1: Local-first shopping-list application architecture