SDLE

U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

SHOPUP

A Local-First Application

CLASS 6

Afonso Pinto Diogo Neves Fábio Teixeira Tomás Torres

.”It is important to feel ownership of that data, because the creative expression is something so personal.”

Martin Kleppmann

Problem Definition

- **Local and Cloud Functionality -** The application will run on user devices, allowing data to be stored locally. It will also have a cloud component for data sharing and backup.
- **CRUD Operations -** Users can create, edit or delete shopping lists and products alike through a user interface. Each list will have a unique ID for easy sharing and access. Lists remain active until they are deleted.
- **Shared Access -** Users with access to a list's unique ID can add or delete items. This feature facilitates collaborative list management.

Problem Definition

- **Concurrency and Data Integrity -** To handle concurrent modifications and ensure high availability, we will use Conflict-free Replicated Data Types (CRDTs) for better data integrity.
- **Scalable Cloud-Like Architecture -** With the goal of serving millions of users, the cloud architecture must be designed to avoid bottlenecks. This includes considering independent list management and data sharding, similar to Amazon Dynamo.

Solution

Server Side

1

ReverseProxy

Reverse proxy that receives client requests, forwards them to the server nodes, and vice-versa. When a new node is added, the node requests the state of the ring to the proxy.

2

ServerNode

Nodes that contain the shopping lists.
Communicate with each other in order to know state of the ring.
“Heartbeat” between nodes: if a node stops responding, considered dead and information transmitted to other nodes.
Implemented replication, rebalancing, consistent hashing and sharding, important pillars of distributed systems.

Conflict-free Replicated Data Type

GCounters

- Each GCounter has a **HashMap** which is composed of:
  - **UUID(key)**: The user ID of every user who edited the product
  - **Counter(value)**: An integer value that increments for each operation done(addition or remotion)
- Merge method between counters

PNCounters

- Each product has a PNCounter associated
- Are composed of two GCounters
  - **Negative**: counts remotions
  - **Positive**: counts additions

How it works

- When a user adds or changes the quantity of a product, the counter with the key equal to the userID inside the Hashmap of the positive or negative counter will be updated
- For merge operation, the shopping list is iterated checking new products and their PNCounters
  - If there is no conflict, just add the alteration
  - If there is conflict, compare GCounters of both versions and accept the key with a higher counter

Solution

Limitations

- With more time, we could improve adding/removing synchronization with AWSet.
- Broker is single poinf of failure.
- Assumption that there is always a Broker online.

Conclusion

Developed a local-first, distributed shopping list application that adeptly balances local data persistence with robust sharing and backup capabilities.

Achieved **consistency**, **scalability**, and **concurrent** user collaboration while using a CRDT.

Although we faced many difficulties, we are proud of what we accomplish and learnt throughout this project.