## Introduction

For the auction system passive replication is used. The system is programmed so clients and replica managers can be opened, where one replica manager is chosen as leader. The clients interact with the primary manager and send their “bid” or “result” queries to the primary manager. Every other second a heartbeat/ping is sent from the replica managers to the primary manager to ensure that the primary manager has not crashed. When a primary manager crashes an election is called with the Ring method.

## Architecture

The architecture of the program can be split into three parts, the auction system and the passive replication system that includes logic for elections.

There are two proto methods for the auction system, there is the bid method and the getResult method. The bid method takes the amount, the host name and a Lamport Timestamp from the client and sends it to the primary manager. The latter takes a host name and a Lamport Timestamp and sends it to the primary manager. Both methods get corresponding and fitting response messages from the manager that are shown to the user. The messages show if the auction is over, if the bid was not big enough and so on. An auction starts when the first manager is opened and lasts 30 seconds.

The passive replication system firstly has a ping/heartbeat proto method. This method takes a host and a Lamport timestamp and returns a host and a Lamport timestamp. All the replication managers pings the primary manager every other second ensuring that the primary manager has not crashed. The first manager to not get a ping back from the primary manager can conclude that the manager has crashed and calls for an election. The manager calls the AskForLeadership method and sends it to all the other managers. This method takes a bestHost, bestHostLamportTimestamp and queue and an acknowledge message is sent back. These values are the values for the ring election datastructure. When a fitting leader has been found the SetNewLeader method is called with the same method values as the AskForLeadership method.

## Correctness 1

The implementation makes the illusion of a single copy. The program satisfies linearizability consistency as the program order is prioritized with the lamport timestamps and then afterwards the ids (ports).

## Correctness 2

The implementation is correct in the absence and in the presence of failures. When a failure or crash happens on any of the backup replication managers no faulty behavior happens. When a failure or crash happens on the primary manager the passive replication system with heartbeats quickly finds out that something is wrong and an election is called with the ring datastructure.

## Github link

https://github.com/laugepetersen/auction-system