myTicket

myTicket is a platform where users can buy and sell ticket for events. Each user can sell his own ticket creating an auction. Every user can partecipate to an auction and the winner will get the ticket.

Functional requirements:

- A user can register to the platform.
- A registered user can join multiple auction.
- A registered user can create or delete an auction wrt his own tickets.
- A registered user cannot create a bid on his auction.
- A registered user can create a bid for an auction.
- A registered user can add a ticket to his own list of tickets.
- Every auction has a setted duration time.
- Every auction is accessible to all registered user.
- Every auction has a winner.
- A registered user can see every auction.
- A registered user needs to registrate to partecipate to an auction.
- During an auction, every user registered to it, can see the remaining time.
- During an auction, every user registered to it, can see the older bids.
- A user can login with username and password.

Non functional requirements:

- Communication between client and server use HTTP protocol. (HTTPS preferred)
- When a user join an auction, communication between client ad server use WebSocket. (WSS preferred)
- System need to be available even if one node fails.
- System need to be consistent even if one node fails.
- There will be a load balancer to manage the request load.
- Each node is going to have the same data.

Erlang Usage

About Erlang, each node will have a erlang process, called master node. The master node will create a new process, called worker node for each auction. This is going to happen on all physical node. In this way each worker node can communicate with his respective worker node to exchange info about last bids and auction evolution. Only masters node will communicate with Java Spring Web Server.