~~CUSTOMER~~ WAITER AGENT DESIGN DOCUMENT

Tam Henry Le Nguyen
Tuesday Thursday 11:00 am
Milestone v2A

```
List<MyCustomer> myCustomers;

class MyCustomer
  { customer c;
    int table;
    String choice;
    enum state
    {waiting, readyToOrder, OrderRecieved, OrderSent,
DeliveringMeal, eating, leaving};
      State s=waiting;
  }
semaphore waitingForOrder(0, true);
Menu menu;
Cook chef;
Host host;
string name;
waiterGui waiterGui;
```

DATA

```
pleaseSeatCustomer(customer cust, int table)
 {myCustomers.add(new myCustomer(cust, table));
 }
readyToOrder (customer cust)
 {myCustomers.find(cust).s = readyToOrder;
 }
IWantToOrder(string food, customer c)
  {waitingForOrder.release();
   myCustomers.find(cust).choice=food;
   myCustomers.find(cust).s = OrderRecieved;
  }
OrderIsReady(string food, int table)
  {myCustomers.find(table).s=DeliveringMeal;
  }
IAmLeaving(customer c)
  {myCustomers.find(c).s=Leaving;}
```

MESSAGES

SCHEDULER

```
if there exists a myCustomer in myCustomers
   such that myCustomer.s = waiting
    SeatCustomer(myCustomer);
if there exists a myCustomer in myCustomers
   such that myCustomer.s=readyToOrder
    TakeOrder(myCustomer);
if there exists a myCustomer in myCustomers
   such that myCustomers.s=OrderRecieved
    SendOrder(myCustomer);
if there exists a myCustomer in myCustomers
   such that myCustomers.s=DeliveringMeal
    DeliverMeal(myCustomer);
if there exists a myCustomer in myCustomers
   such that myCustomers.s=leaving;
    TellHost(myCustomer);
```

ACTIONS

```
SeatCustomer(myCustomer mc)
  {
   DoSeatCustomer(mc.c);
   mc.c.followMe(new Menu;this);
  }
TakeOrder(myCustomer mc){
  DoGoToTable(mc.table);
  mc.c.whatDoYouWant();
  waitingForOrder.acquire();
  }
SendOrder(myCustomer mc){
  DoGoToCook(chef);
  chef.pleaseCook(mc.choice, mc.table, this);
  mc.s=OrderSent;
  }
DeliverMeal(myCustomer mc)
  {DoGoToTable(mc.table);
  mc.HereIsYourMeal(food choice);
  mc.s=eating;}
TellHost(myCustomer mc)
 { host.tableIsFree(mc.table);
```