

## Endabgabe

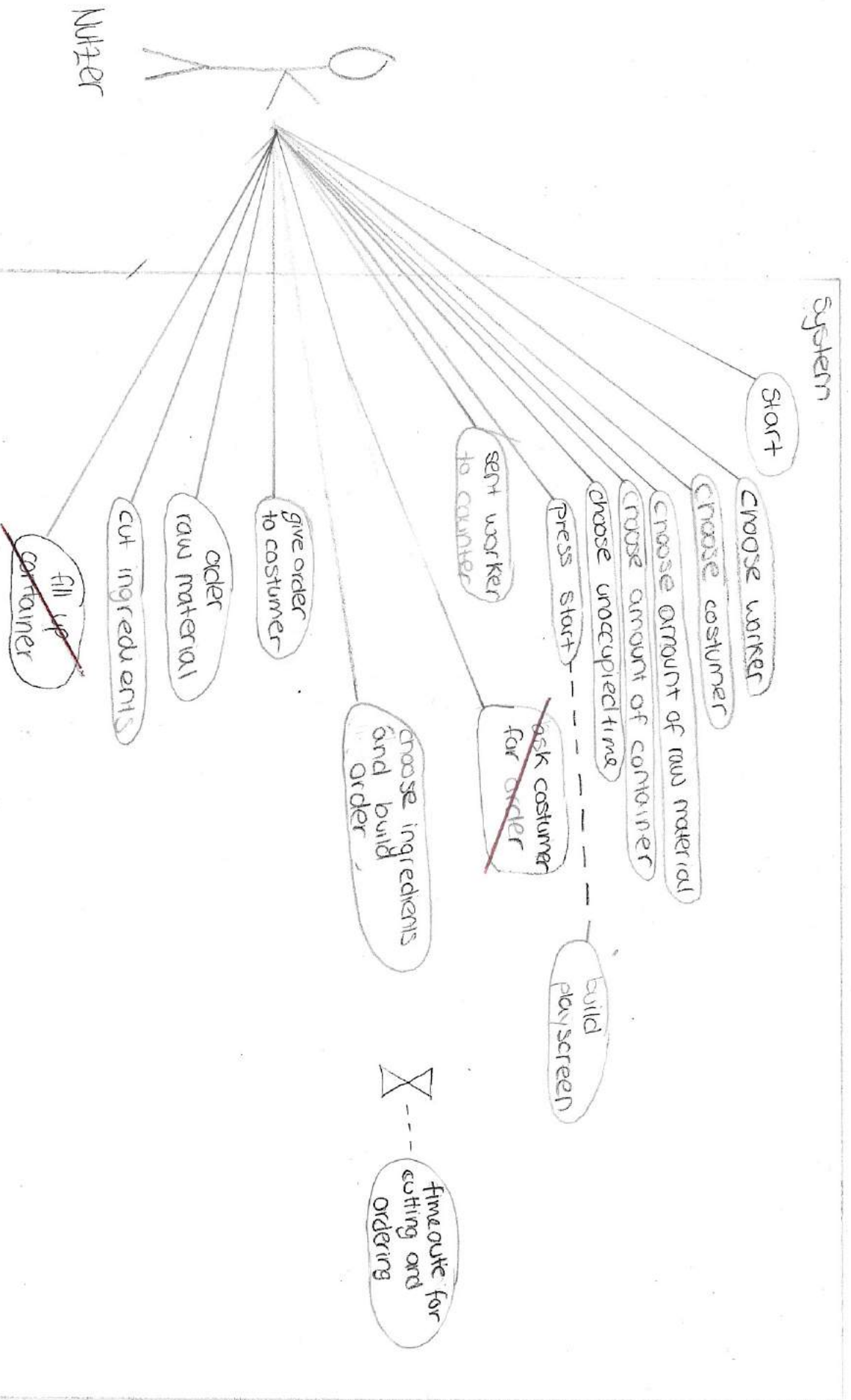
Gruppenarbeit von Asya Tetik, Christina Gabler, Lisa Fingerle, Deborah Reinbold

Aus Gründen der Übersichtlichkeit haben wir gemeinsam nur in einem Repository gearbeitet und dort rein gepusht und gepullt und die fertige Endabgabe dann in einzelne Repositories hochgeladen.

### Döner-Trainer Anleitung

1. Zu Beginn kann der Nutzer auf der Startseite die Anzahl der Kunden pro Minute, die Anzahl der Arbeiter, die Kapazität vom Rohmaterial und geschnittenem Material und der Leerlaufzeit der Mitarbeiter bestimmen.
2. Der Kunde gibt seine Bestellung auf und wird vom Mitarbeiter (durch den Nutzer gesteuert) bedient. Um die Mitarbeiter zu bewegen, muss er zuvor durch einen Klick ausgewählt werden.
3. Durch Klicken auf die einzelnen Zutaten wird die Bestellung zusammengestellt und die zur Verfügung stehenden Zutaten werden verringert. Danach kann das Gericht mit Klick auf den Kunden diesem übergeben werden.
4. Die Zufriedenheit der Kunden kann sich durch die Geschwindigkeit und die Genauigkeit des Spielers verändern und das Spiel beeinflussen. Bspw.: Wenn der Spieler zu lange braucht, um ein Gericht zu servieren, wird der Kunde wütend und verlässt die Dönerbude.
5. Auch die Mitarbeiter verändern ihre Stimmung je nach dem, wie hoch ihr Stresslevel ist.
6. Falls die Zutaten an der Theke leer sind, muss der Mitarbeiter in die Küche geschickt werden, um die Rohstoffe zu schneiden. Sind die Rohstoffe leer, können sie nachbestellt werden.
7. Die Zufriedenheit der Kunden und Mitarbeiter sowie die Anzahl der verkauften Gerichte werden fortlaufend angezeigt.

# Doner - Trainer : Use - Case - Diagram



<h1>

# Dönertrainer

Mitarbeiter

1

▼

Kunden pro Minute

1

▼

Formmaterial

1 Stock

10 Stock

Container

1 Stock

15 Stock

Leerlaufzeit

20 Sekunden

▼

START

<button>  
id="start"

<select>

id="worker"  
name="worker"

<div>

id="form"

<select>

id="costumer"  
name="costumer"

<input>

type="range"  
name="Rau"  
min="1"  
max="10"  
step="1"

<input>

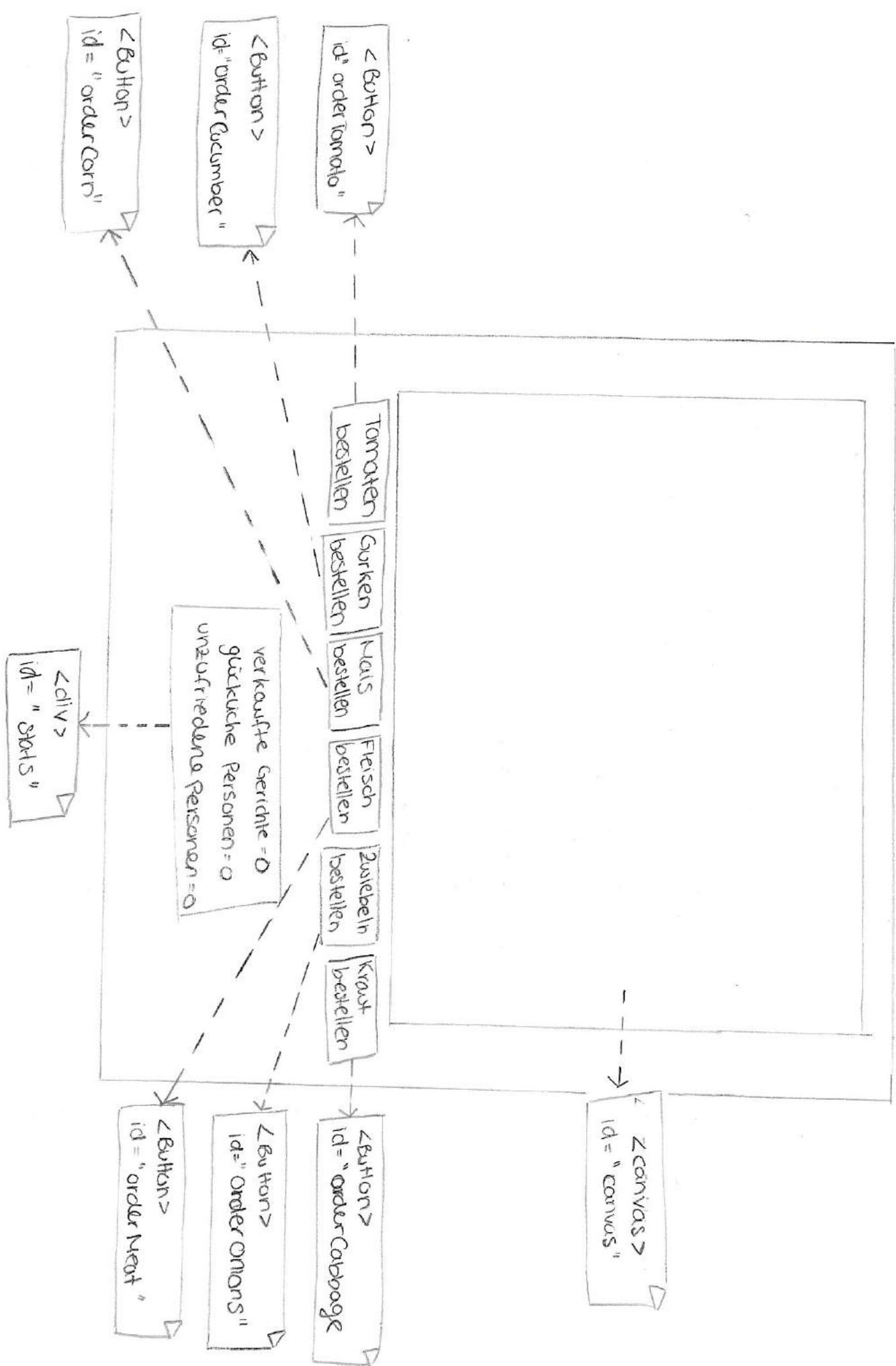
type="range"  
name="container"  
min="1"  
max="15"  
step="1"

<input>

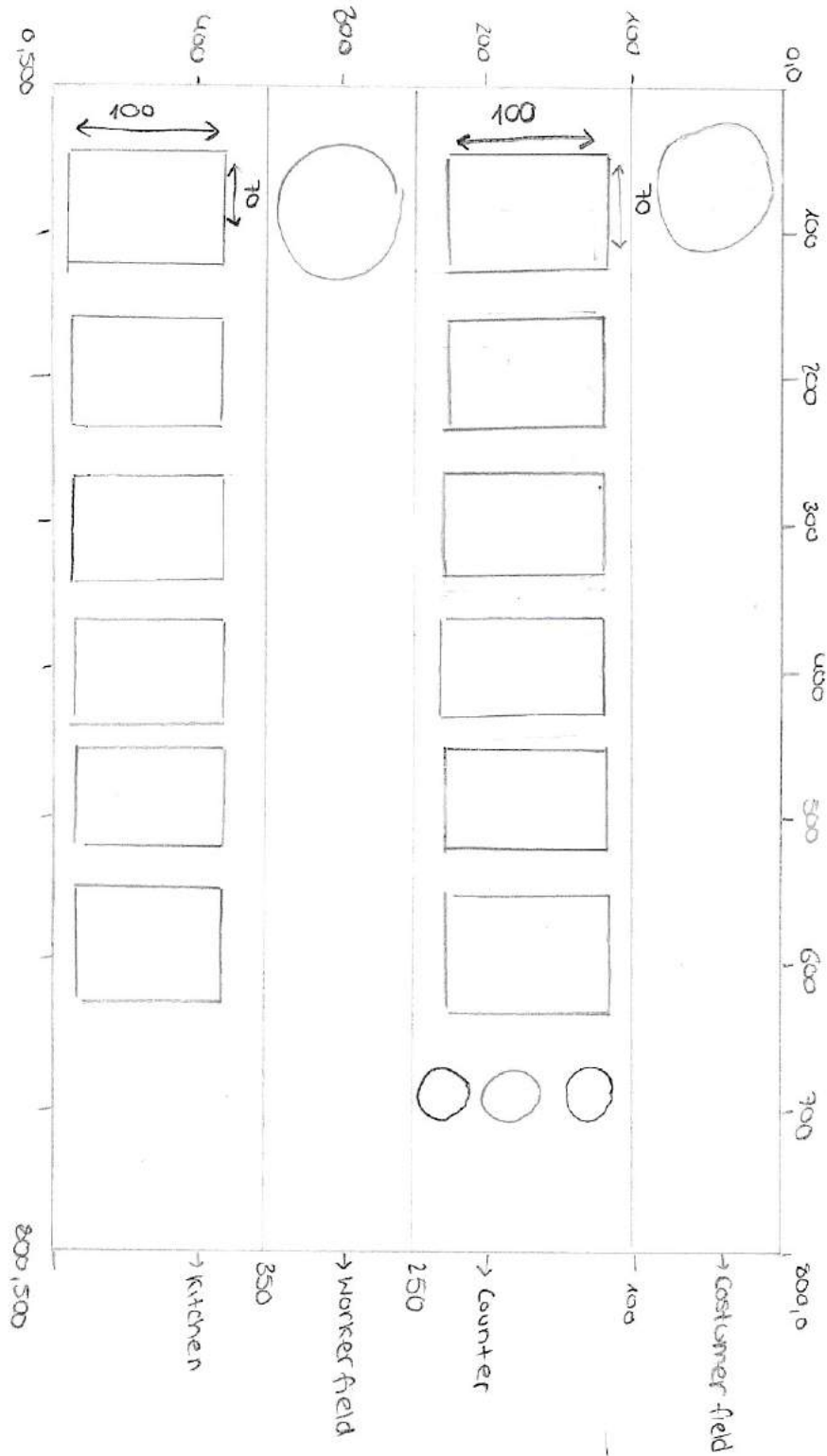
type="text"  
name="unoccupied"  
id="unoccupied"

<button>

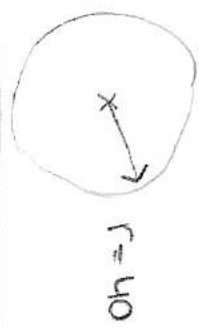
id="start"



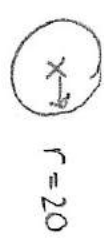
# Canvas: Kebab - House



worker & customer

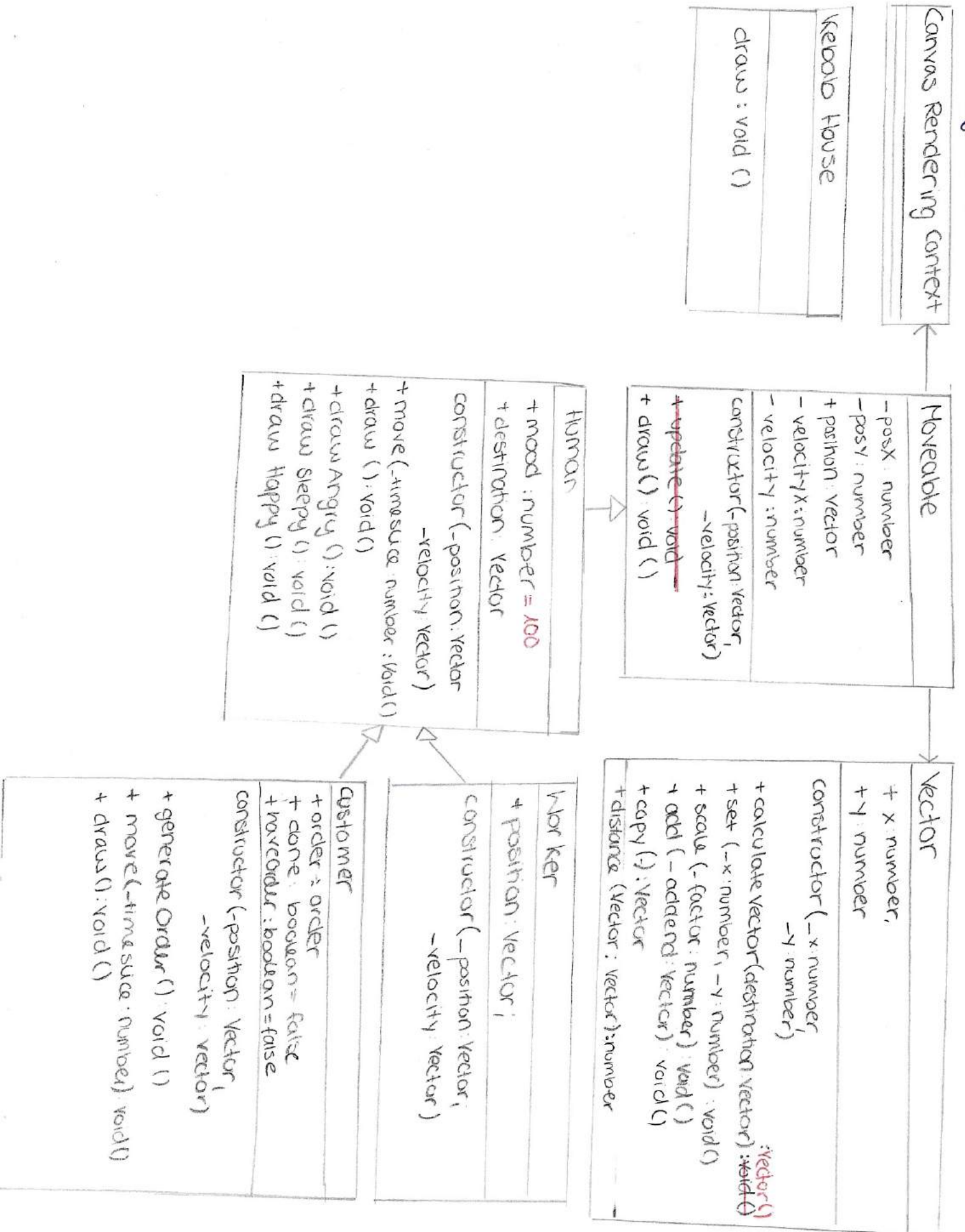


Döner, Yufka, Lahmacun





# Class diagram ②



Classdiagram ②

Doner
draw: void()

Yufka
draw: void()

Lahmacun
draw: void()

<<interface>> order name: string ingredients: IngredientsList[]
--

Ingredients
+ maxIngredients: number + maxRawIngredients: number + used Tomatoes: number = 0 + used Raw Tomatoes: number = 0 + used Cucumbers: number = 0 + used Raw Cucumbers: number = 0 + used Corn: number = 0 + used Raw Corn: number = 0 + used Meat: number = 0 + used Raw Meat: number = 0 + used Onions: number = 0 + used Raw Onions: number = 0 + used Cabbage: number = 0 + used Raw Cabbage: number = 0 + draw(): void()

<<enum>> ingredientsList DÖNER, YUFKA, LAHMACUN, TOMATDES, CUCUMBER, CORN, MEAT, ONION, CABBAGE
---



# Activity Diagram main



```

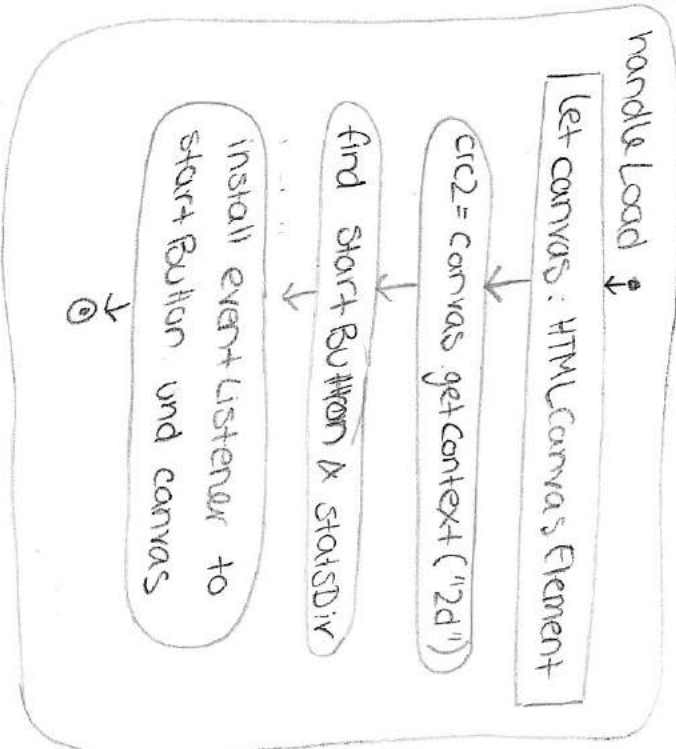
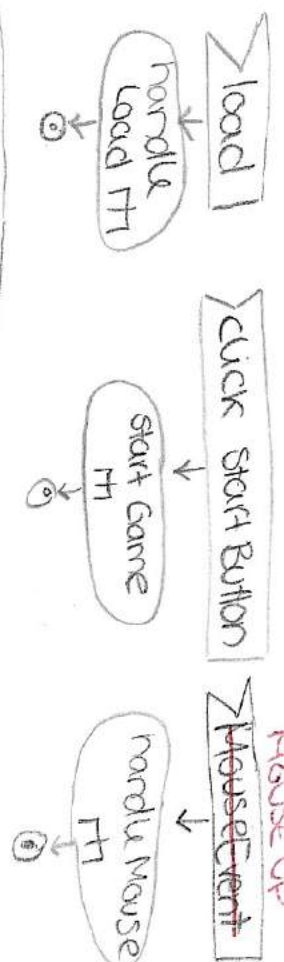
let crc2: CanvasRenderingContext2D
let startButton: HTMLButtonElement
let keyboardouse: keyboardouse

let ingredients: ingredients
let doener: Doener
let yufka: Yufka

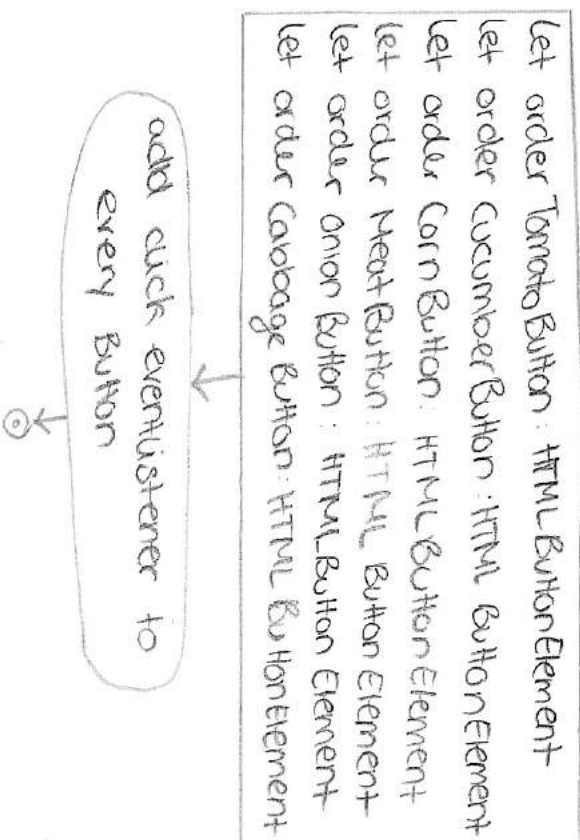
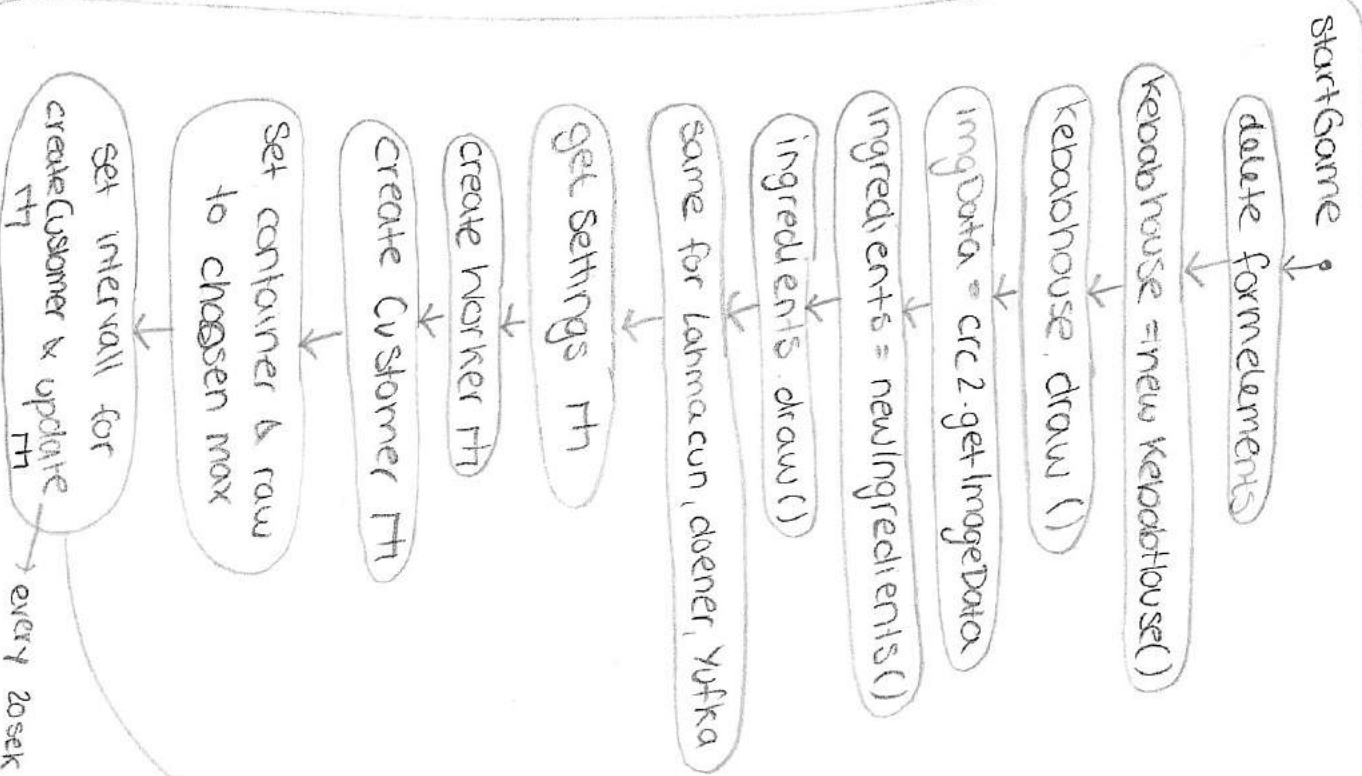
let Lahmacun: Lahmacun

let worker: number = 2
let customer: number = 3
let rau: number = 10
let container: number = 15
let unoccupied: number = 20
let moveable: Moveable[] = []

let animation: boolean = false
let allCustomer: number = 0
let imgData = imageData
let usingredients: ingredientsList[] = []
let activeWorker: Worker
let doneOrder: number = 0
let happyCustomer: number = 0
let angryCustomer: number = 0
let happyWorker: number = 0
let unhappyWorker: number = 0
let statsDiv: HTMLDivElement
  
```



# Activity Diagram: Main ②



# Activity Diagram: Main ③

get settings ↓

let formData: FormData = new FormData

worker = Number(formData.get("worker"))

customer = Number(formData.get("customer"))

raws = Number(formData.get("raws"))

container = Number(formData.get("container"))

unoccupied = Number(formData.get("unoccupied"))



Create Worker ↓

nWorker = 0

[nWorker < worker]

draw newWorker

push to Moreable[]

increase  
nWorker  
++



Create Customer ↓

[allCustomer < 5]

draw Customer

push to Moreable[]

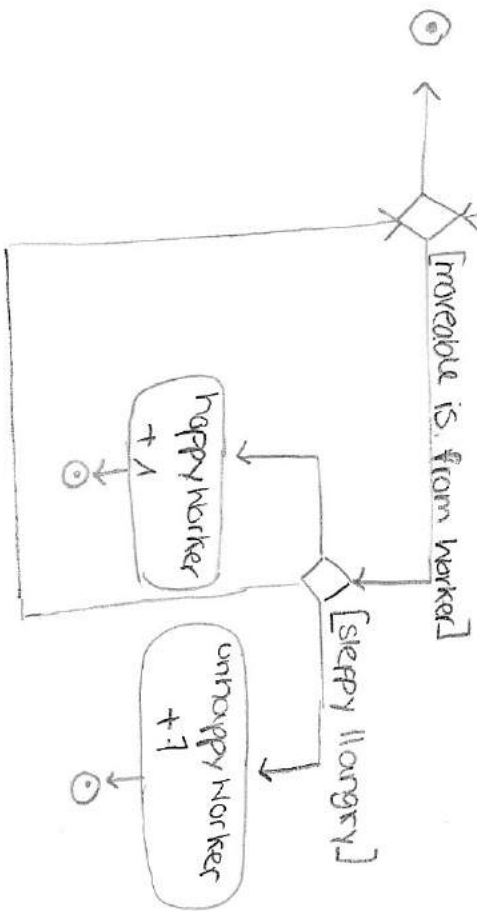


⑤



set happyMarker &  
unhappyMarker to 0

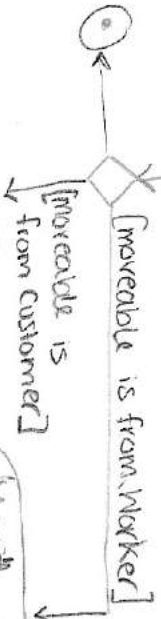
let movable of moveables



update

Let moveable of moveables

draw moveable



moor  
decreases

generate order

more Customer

[customer angry]  $\times$  [customer angry & no order]

move to door

angry customer

decrease  
Customer

clutter ← from measurable [ ]

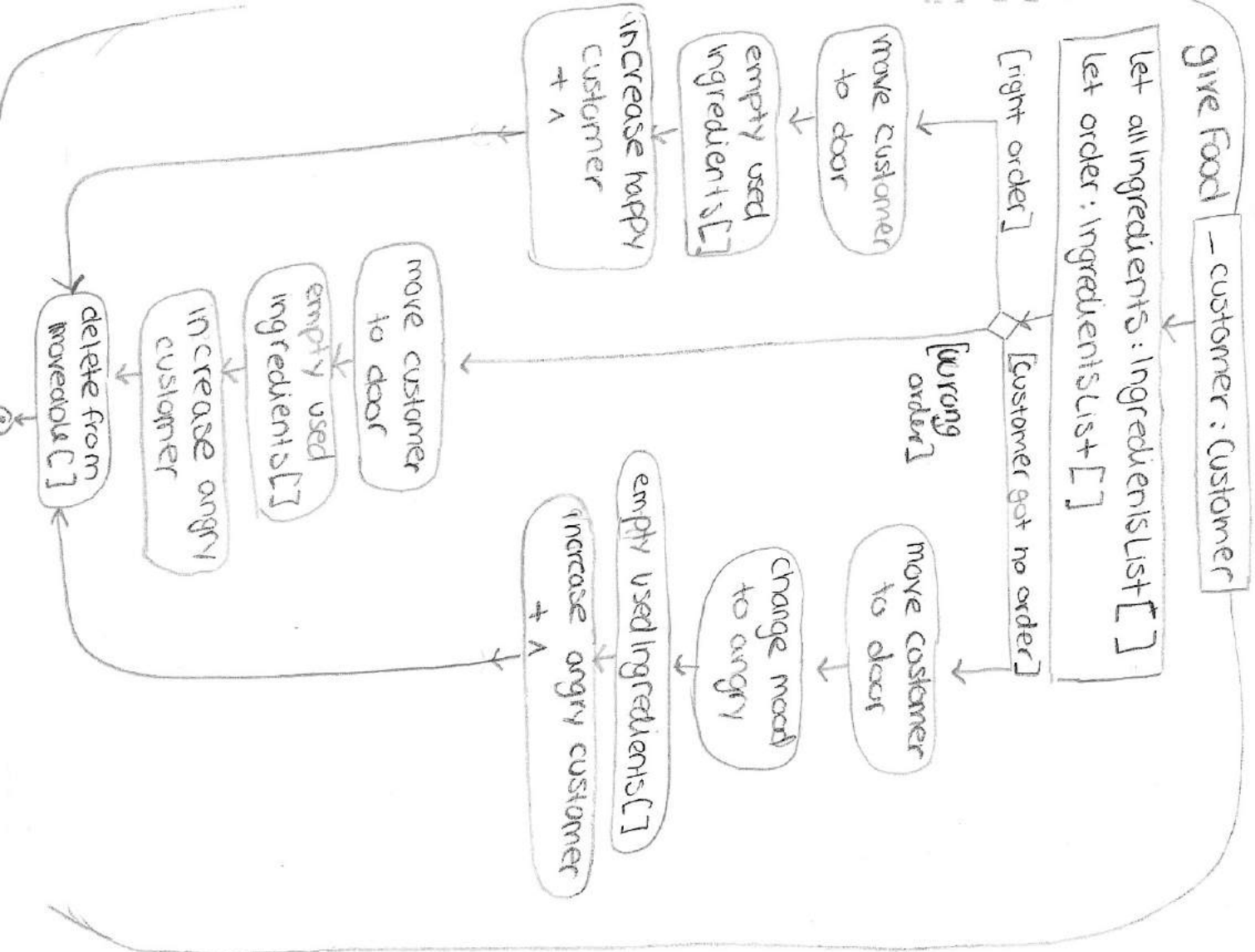
Count worker mood

fill statsdir

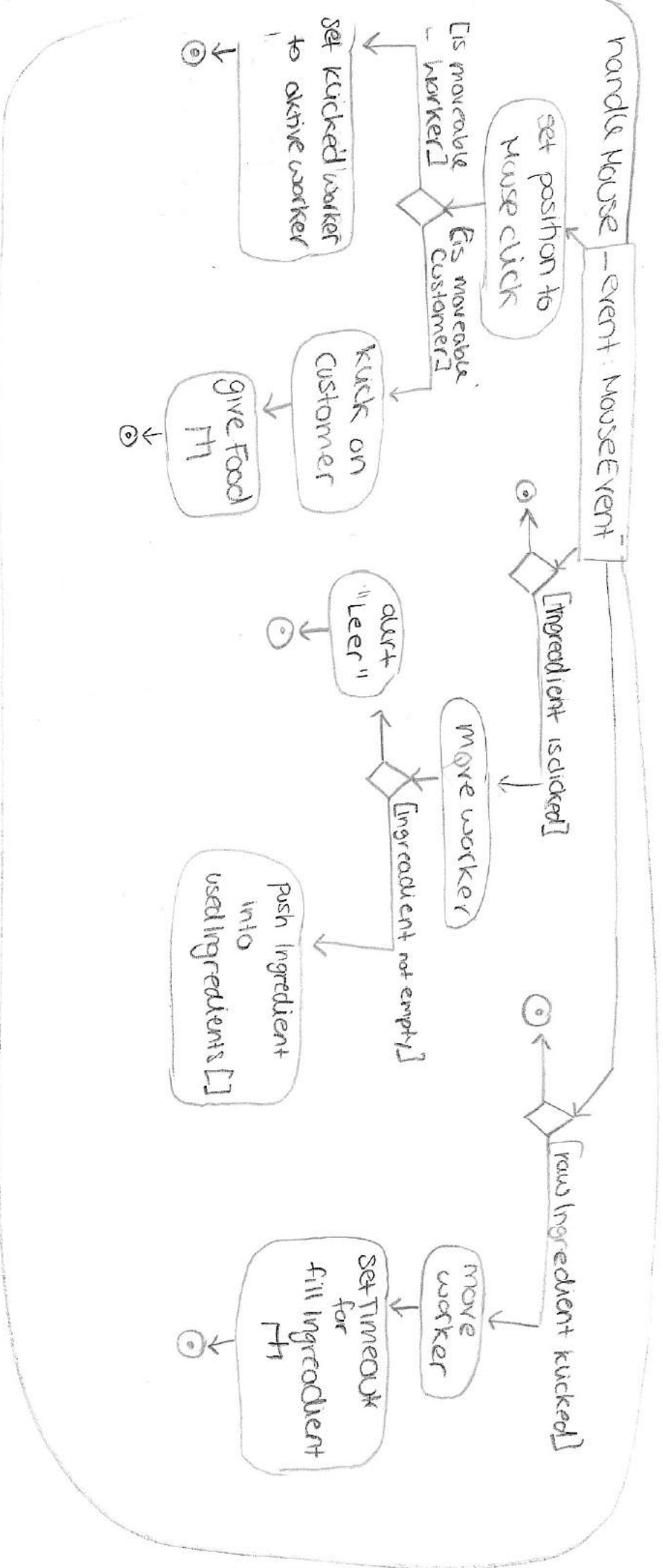
draw ingredients, ~~Donner~~,  
~~Yufka, tahmacun~~



# Activity Diagram: Main ⑤

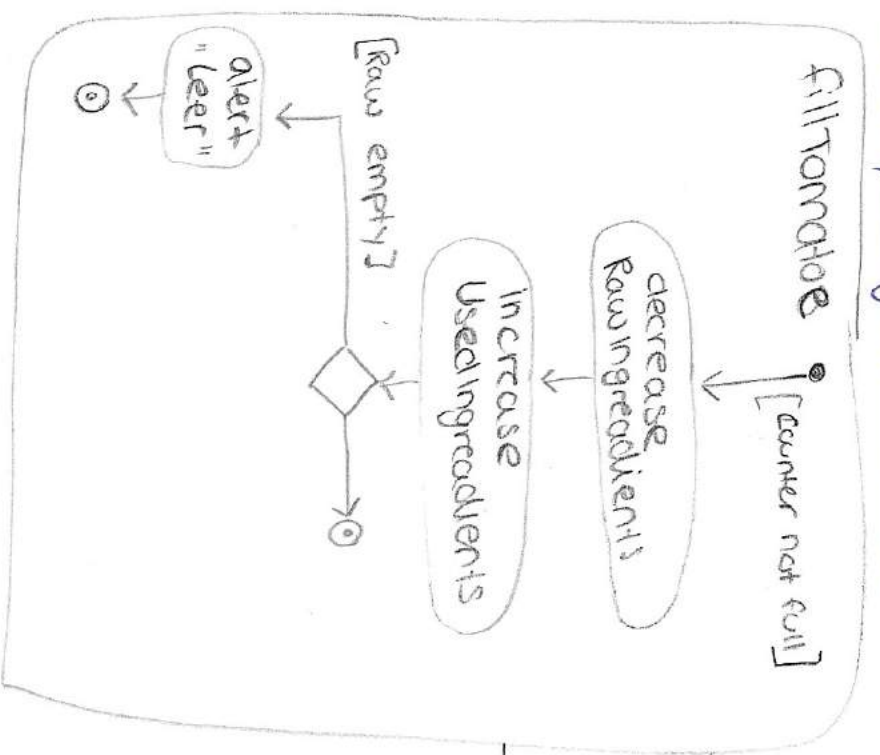


# Activity Diagram: Main ⑥

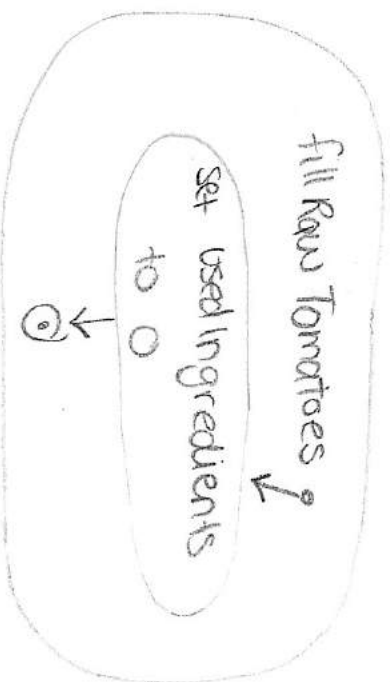
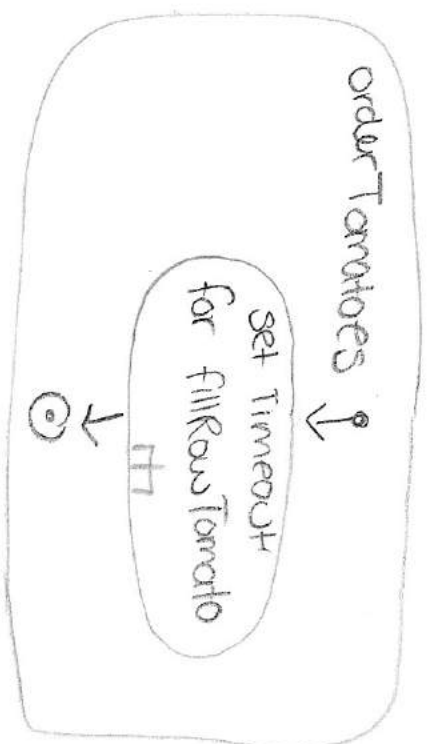




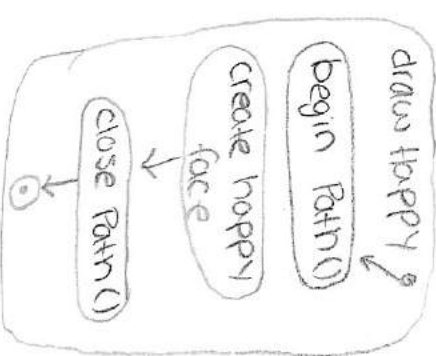
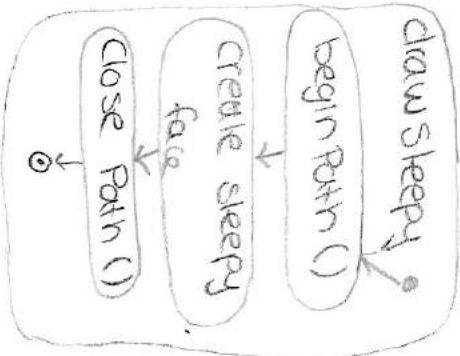
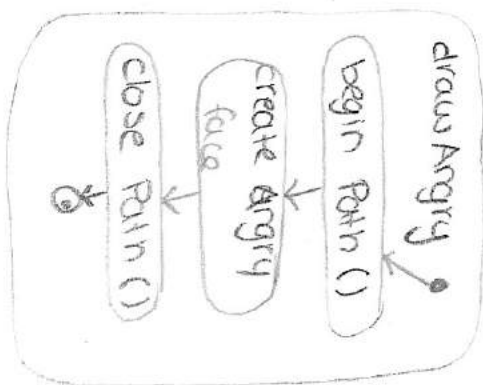
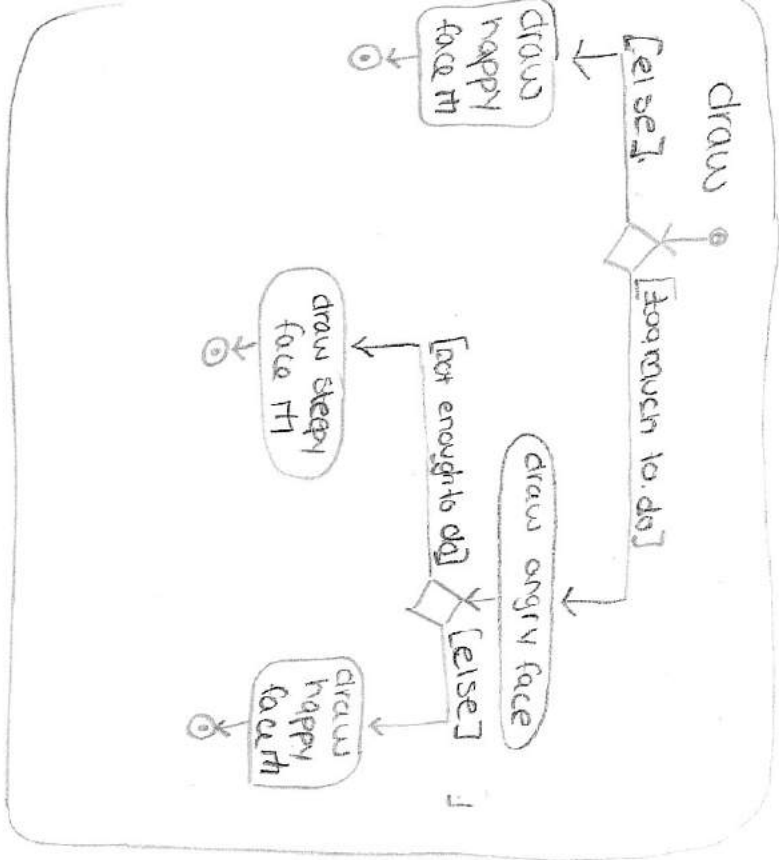
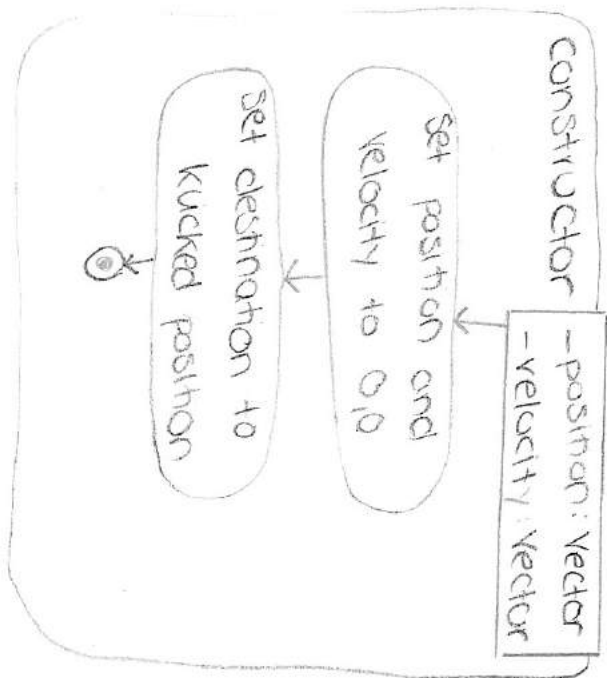
# Activity Diagram: main ①



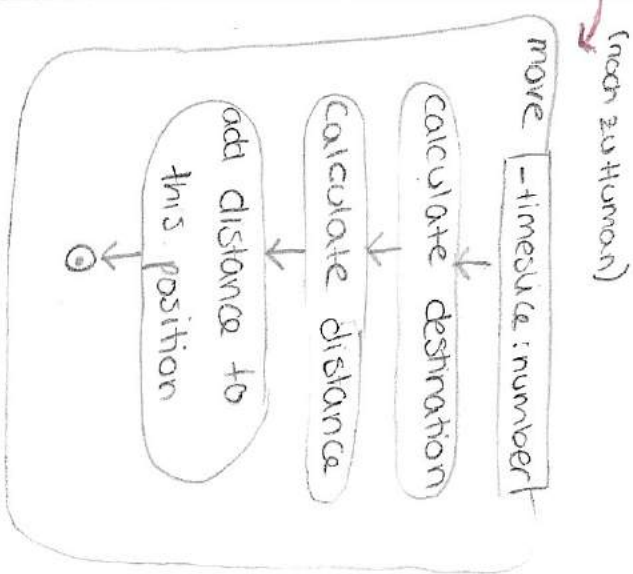
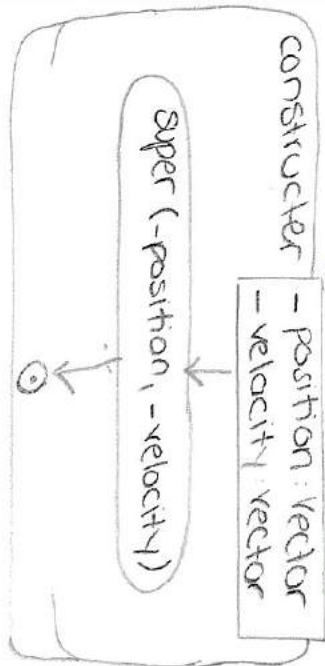
→ same for all ingredients



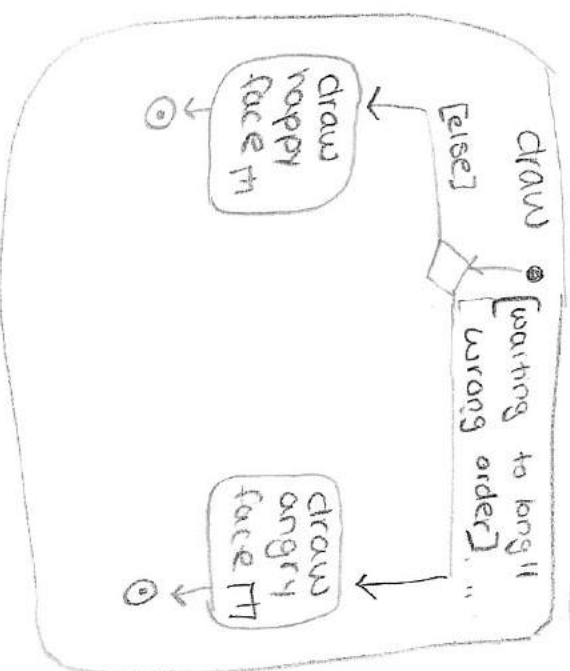
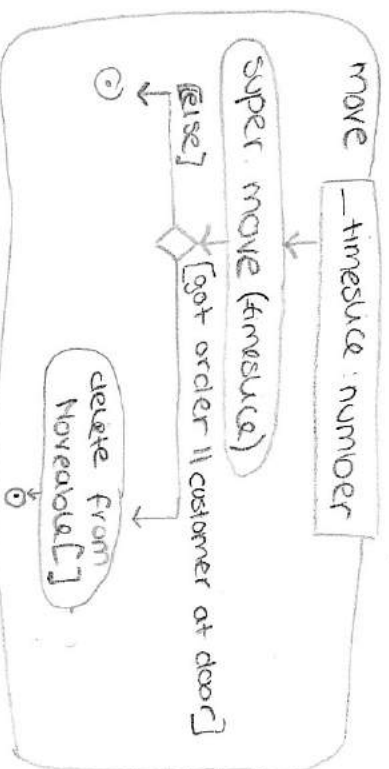
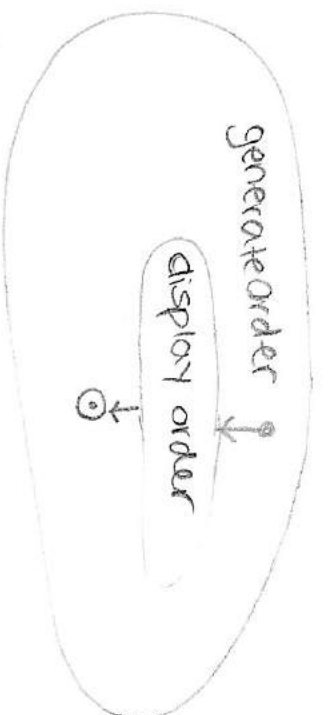
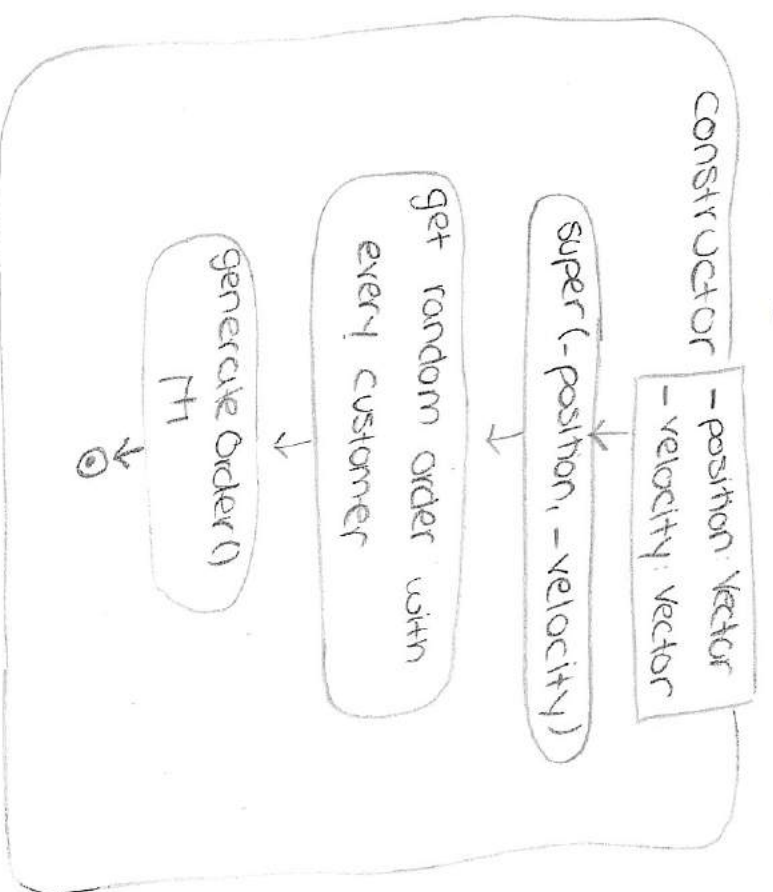
# Activity Diagram: Human



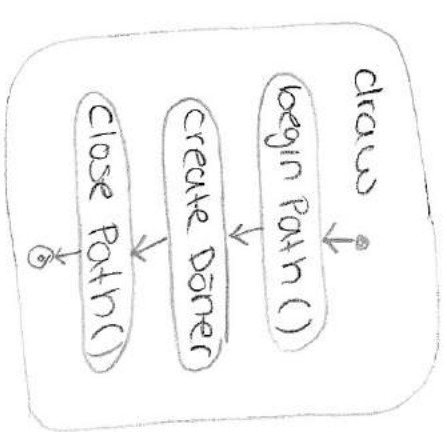
# Activity Diagram: Worker



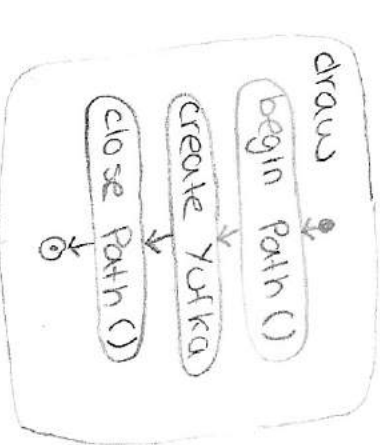
# Activity Diagram: Customer



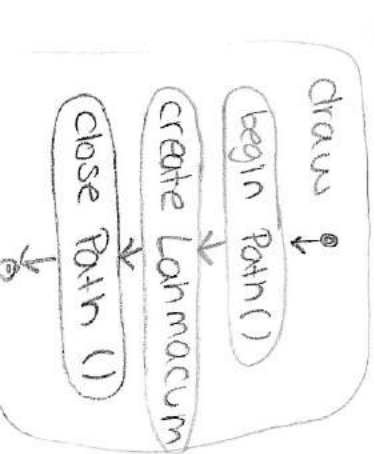
### Activity Diagram : Döner



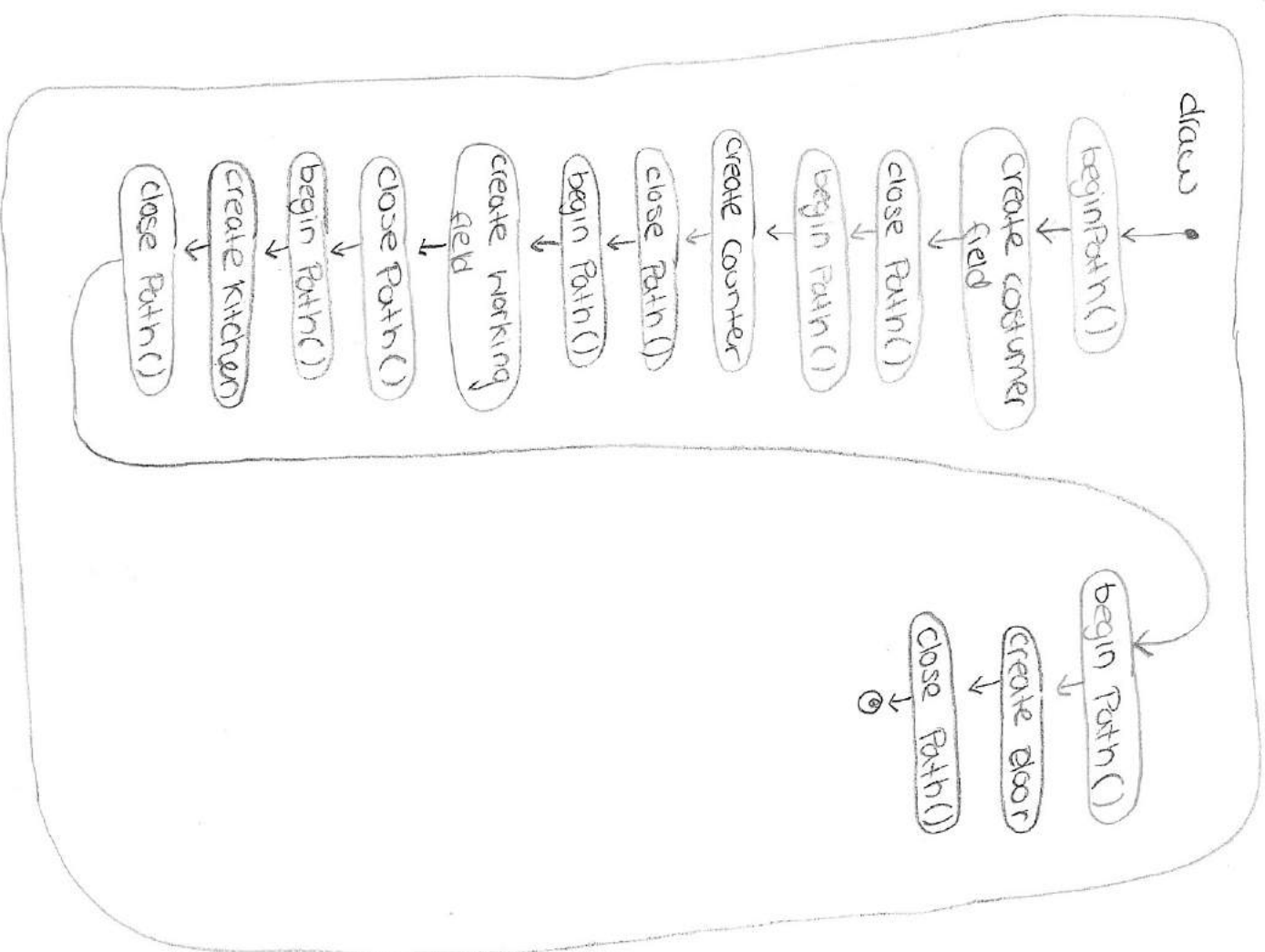
### Activity Diagram : Yufka



### Activity Diagram : Lahmacum

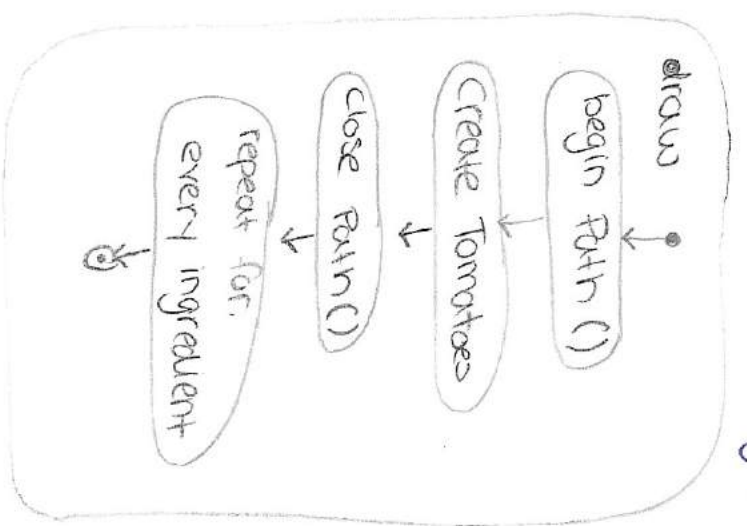


### Activity diagram : Kebab - House





## Activity Diagram: Ingredients



## Activity Diagram: Moveable

