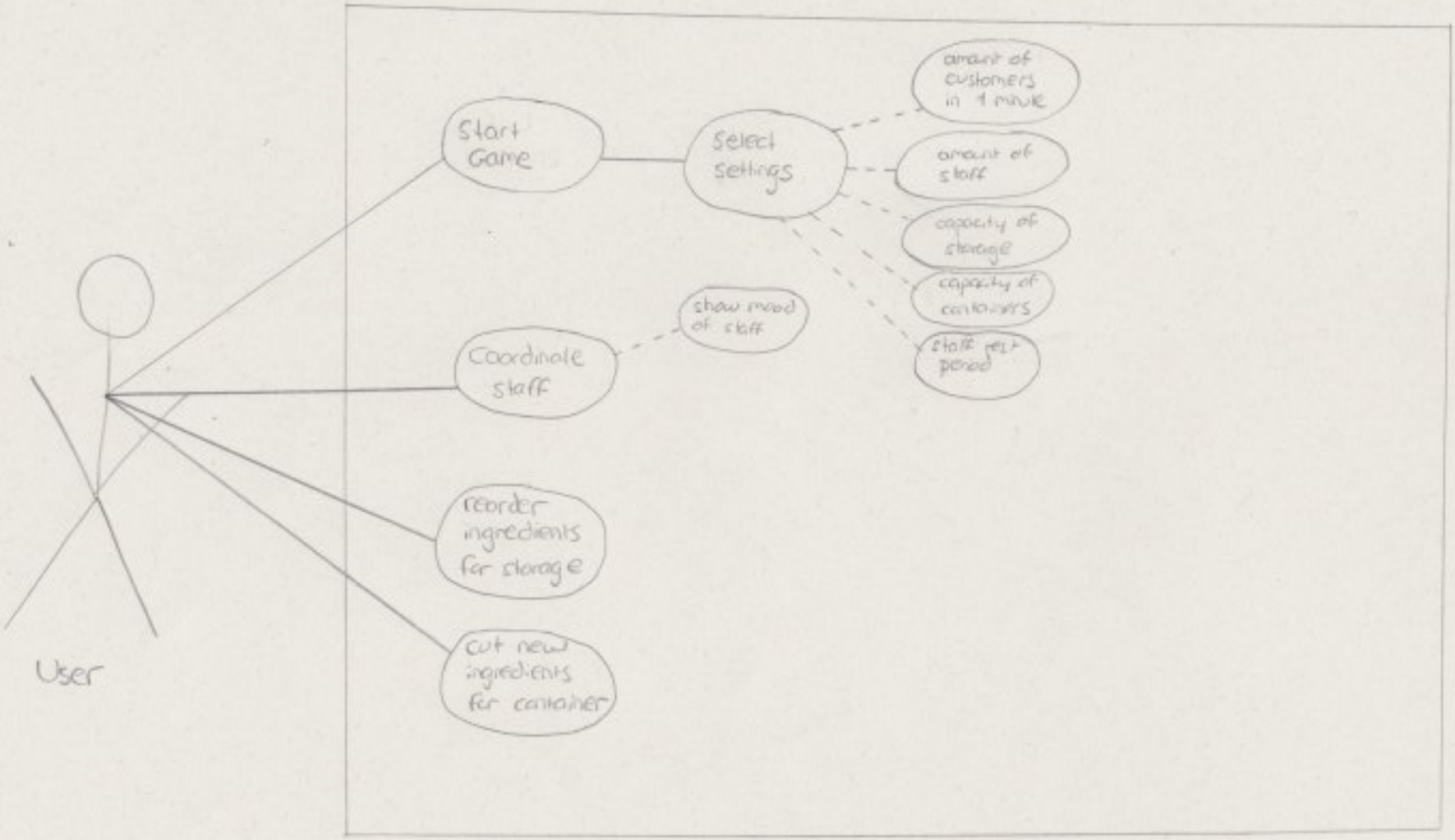
Endabgabe EIA2 – Döner-Trainer

Konzept

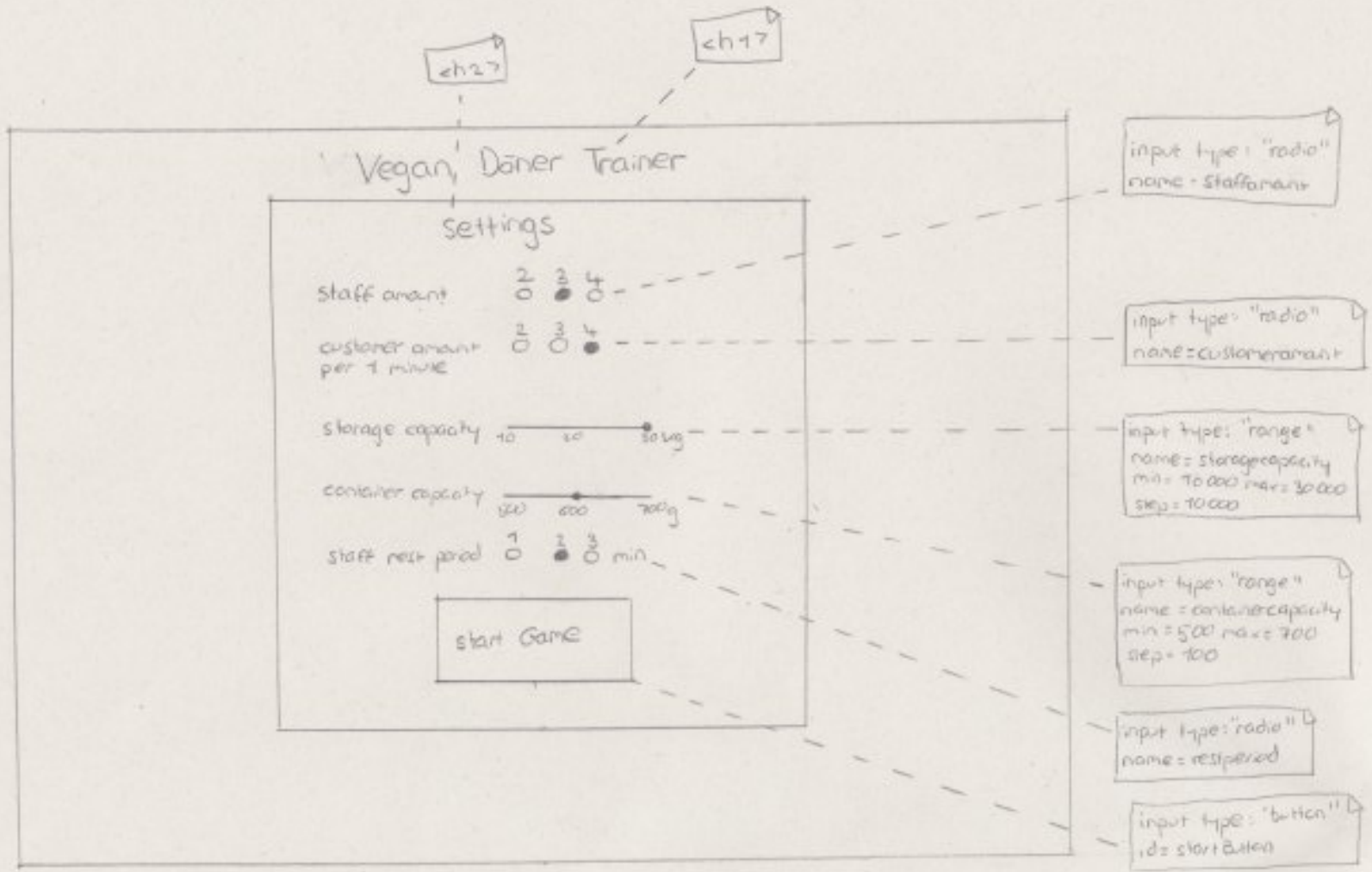
Verena Rothweiler
Matrikelnummer: 270156

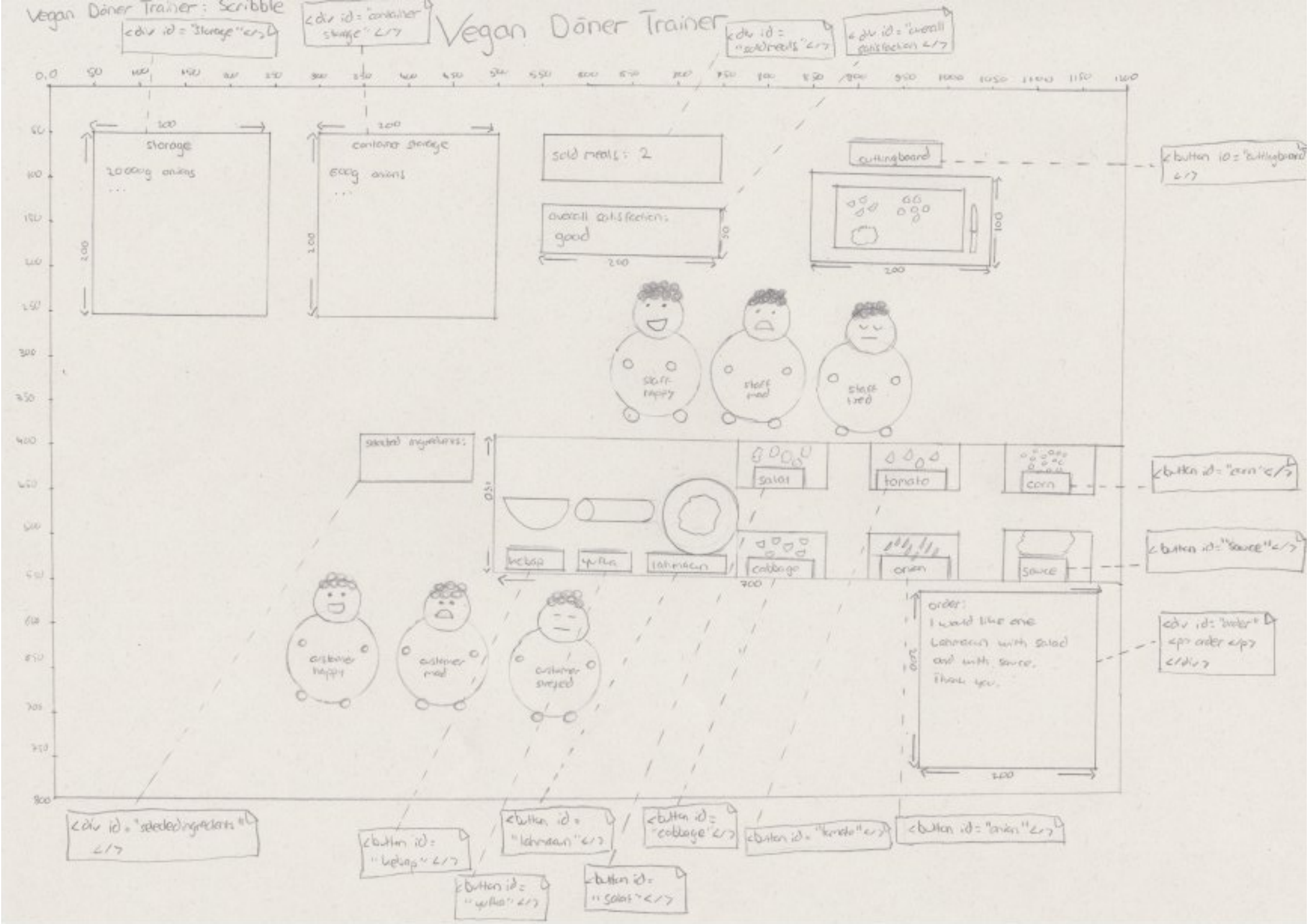
in Zusammenarbeit mit Neslisah Koc

Vegan Diner_Trainer: Use-Case-Diagram



Vegan Döner Trainer: UI - Scribble: Settings

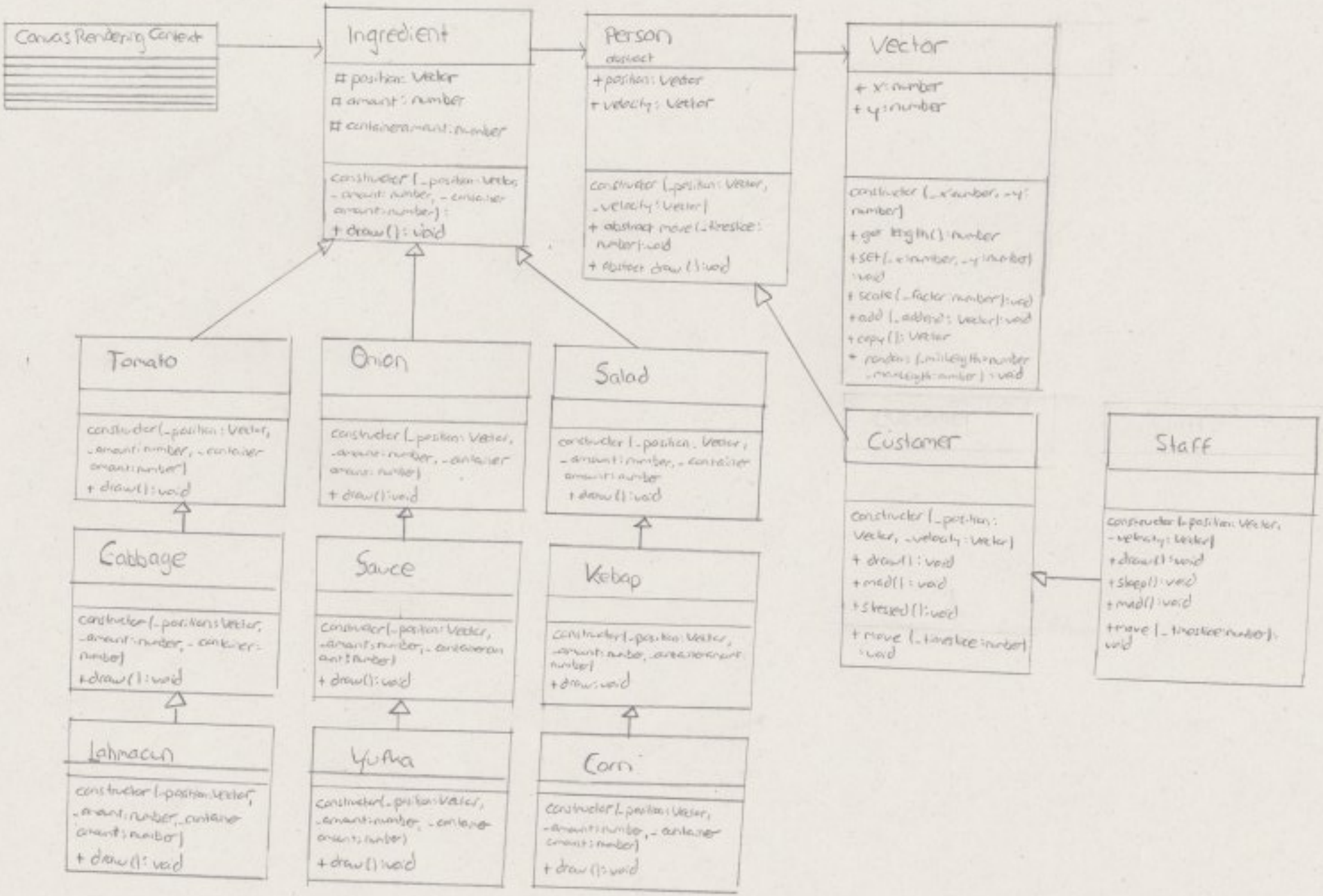



Vegan Döner Trainer: Scribble
Vegan Döner Trainer

Storage
20000g onions
...
container storage
6000g onions
...
sold meals: 2
overall satisfaction: good
cuttingboard
<button id="cuttingboard" />
staff happy
staff mad
staff tired
selected ingredients:
salat
tomato
corn
<button id="corn" />
kebap
yufka
lahmacun
cabbage
onion
sauce
<button id="sauce" />
order:
I would like one lahmacun with salad and with sauce. thank you.
  order
customer happy
customer mad
customer surprised

<button id="kebap" />
<button id="lahmacun" />
<button id="cabbage" />
<button id="tomato" />
<button id="onion" />
<button id="yufka" />
<button id="salat" />

Vegan Döner Trainer: Class Diagram



Vegan Döner Trainer: Activity Diagram: Main

load

handleLoad fn

let formData: FormData
let staffIsMad: boolean = false
let customerIsMad: boolean = false

let staffAmount: number;
let customerAmount: number;
let storageCapacity: number;
let containerCapacity: number;
let staffPatience: number;
let staffChoice: number;
let soldMeat: number = 0;
let imgData: ImageData

install
load Wiener

let ingredients: Ingredient[]
let staffs: Staff[]
let customers: Customer[]
let order: string[] = []
let customerorder: string[] = []
let storageLeft: Storage;
let ingredientsLeft: Storage;
let bools: string[] = ["Vetkip with vegan meat", "Yufka with vegan meat", "Lahmacun with vegan mixed meat"]
let topping: string[] = ["corn", "salad", "red cabbage", "onion", "tomato"]
let sauce: string[] = ["sauce"];

interface Storage{
salad: number
cabbage: number
onion: number
corn: number
tomato: number}

click on sauce Button

update Sauce fn

click on start Button

prepareGame fn

click on Lahmaan Button

update Lahmaan fn

click on onion Button

update Onion fn

click on finishorder Button

computeOrder fn

click on corn Button

update Corn fn

click on tomato Button

update Tomato fn

click on Ketchup button

updateKetchup fn

click on Salad Button

updateSalad fn

click on refill Button

refillContainer fn

click on yufka button

updateYufka fn

click on cabbage Button

updateCabbage fn

click on render Button

renderIngredients fn

handleLoad event: Event

let startButton: HTMLButtonElement

add EventListener "click" on startButton

hide all Divs and Buttons with document.getElementById.hidden = true

prepareGame fn

prepareGame event: Event

let formData = new FormData(document.forms[0]

let form: HTMLFormElement.document.querySelector("form")
let body: HTMLBodyElement.document.querySelector("body")

remove form from body

save values from FormData as variables with formData.get()

storageLeft = {
salad: containerCapacity
cabbage: containerCapacity
onion: containerCapacity
corn: containerCapacity
tomato: containerCapacity

ingredientsLeft = {
salad: storageCapacity
cabbage: storageCapacity
onion: storageCapacity
corn: storageCapacity
tomato: storageCapacity

buildGameScreen()

buildGameScreen

show all Div's and Buttons with document.getElementById.hidden = false

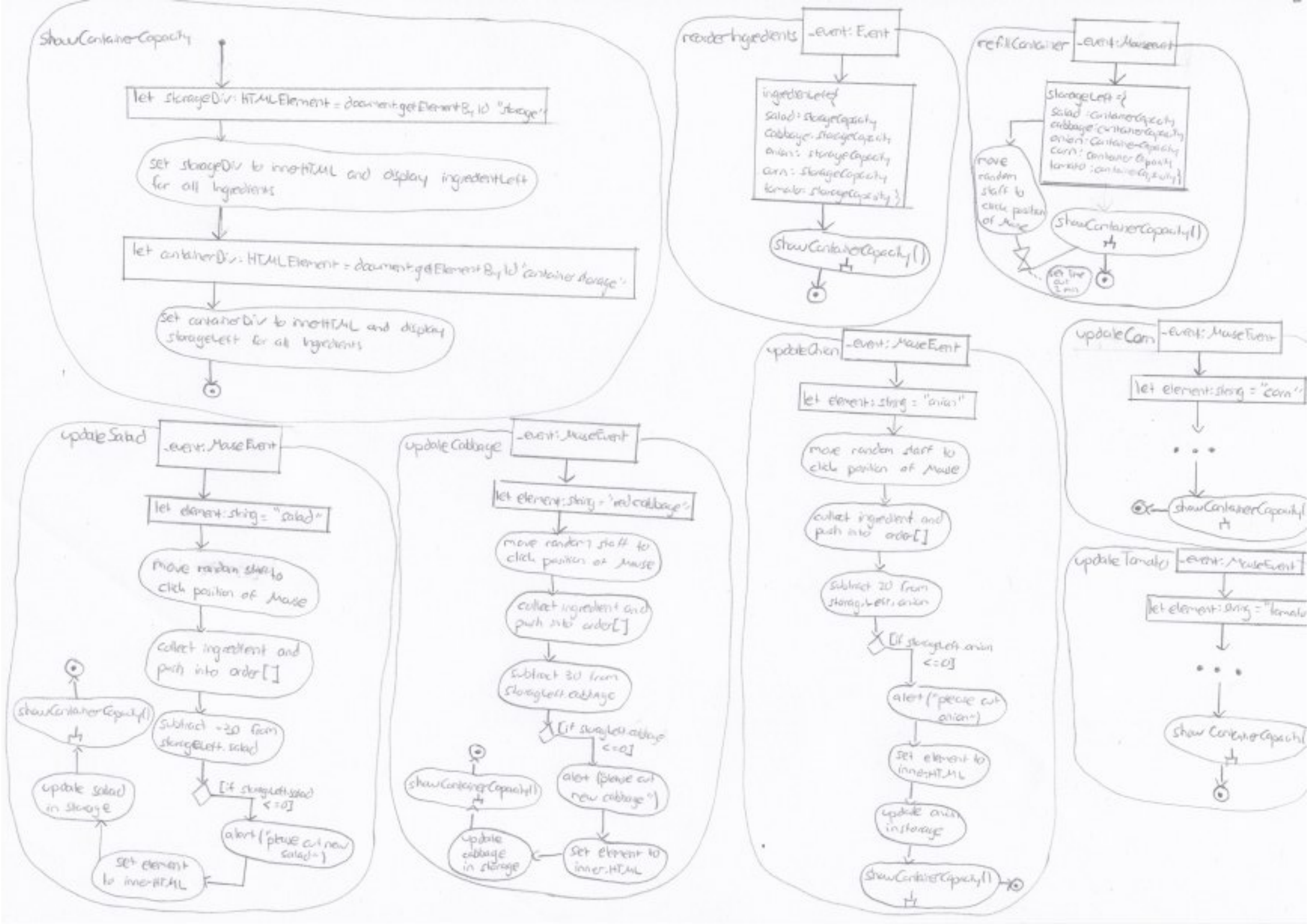
let canvas: HTMLCanvasElement = document.querySelector("canvas"
ctx2 = canvas.getContext("2d")

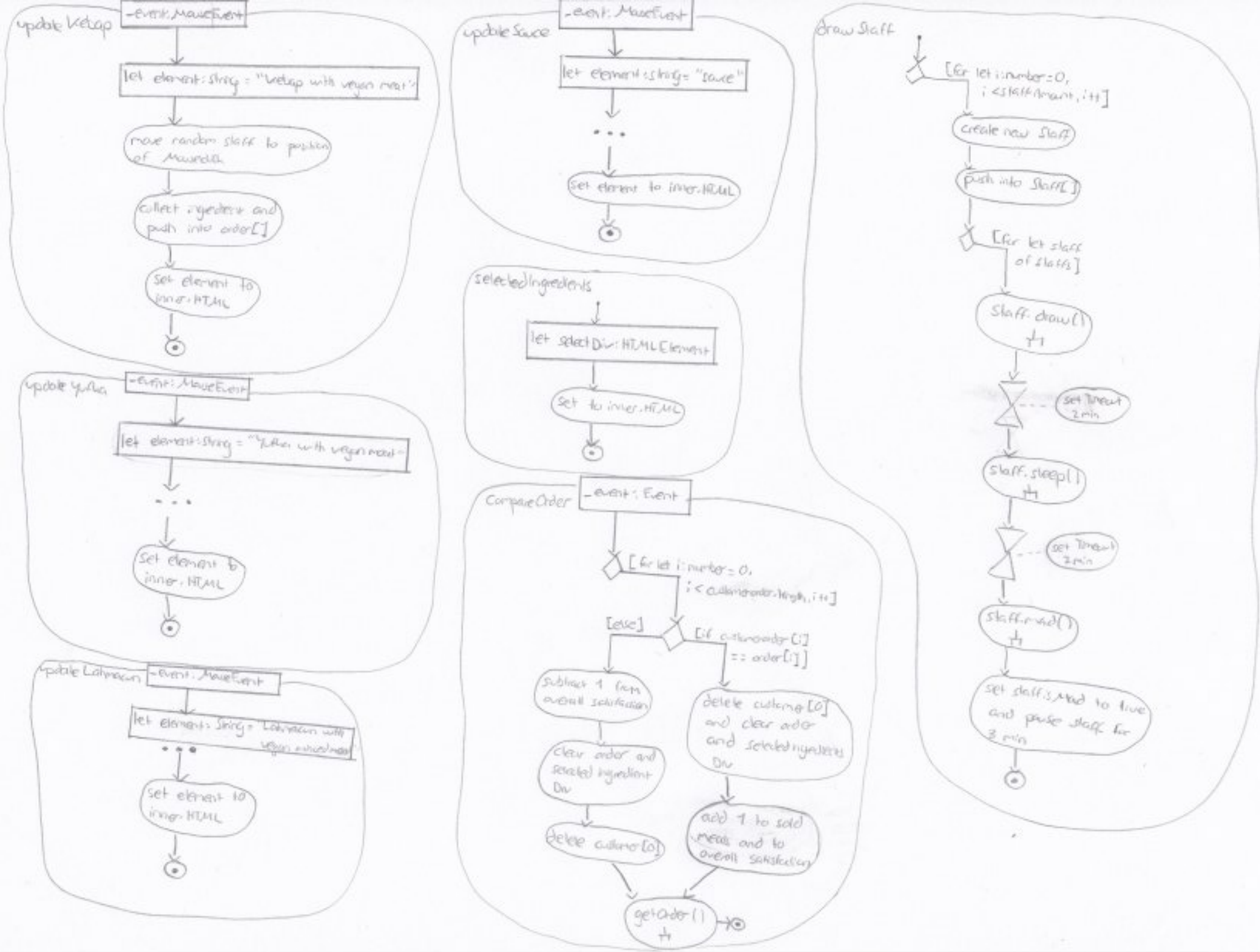
drawCounter()
drawCuttingboard()
drawSalad()
drawTomato()
drawCorn()
drawOnion()
drawCabbage()
drawSauce()
drawKetchup()
drawYufka()
drawLahmacun()
showContainerCapacity()
drawStaff()
getOrder()
selectedIngredients()

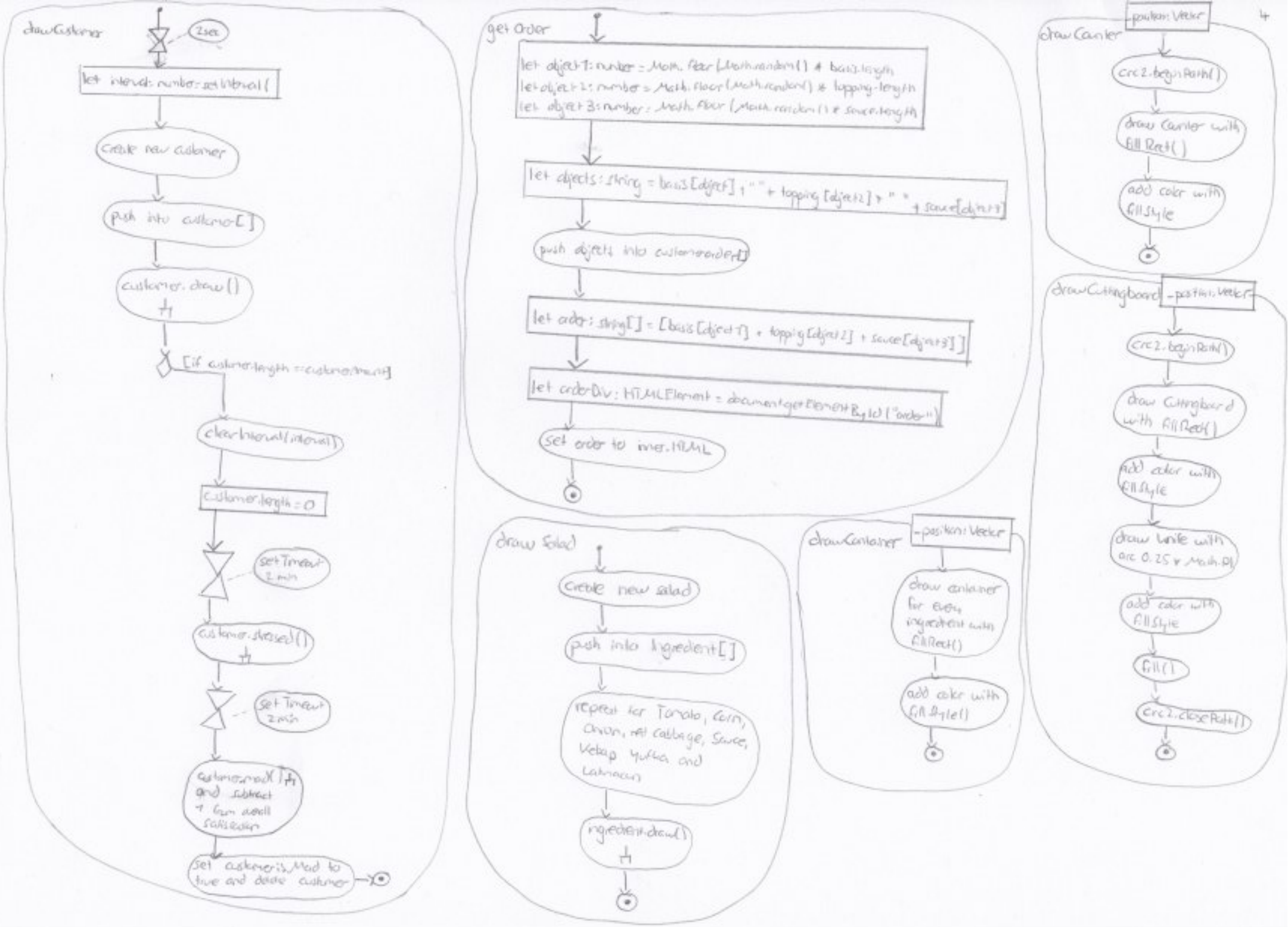
create for all ingredients Button HTMLButtonElement.document.querySelector and add EventListener "click"

window.setInterval(update, 20)

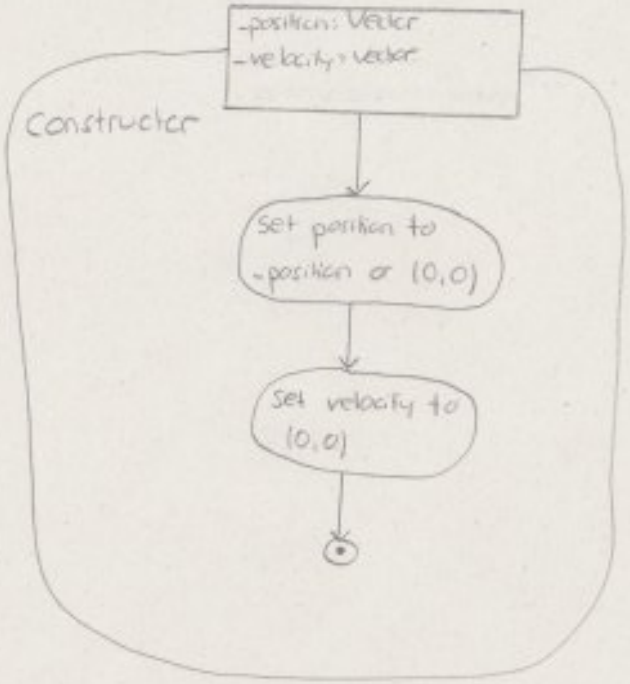
setInterval(drawCustomer)

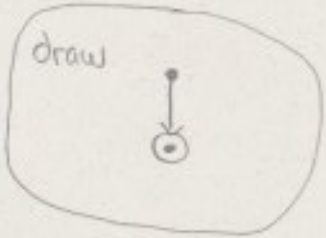

showContainerCapacity
let storageDiv: HTMLElement = document.getElementById("storage")
set storageDiv to innerHTML and display ingredientLeft for all ingredients
let containerDiv: HTMLElement = document.getElementById("containerStorage")
set containerDiv to innerHTML and display storageLeft for all ingredients
reorderIngredients
_event: Event
ingredientLeft{
salad: storageCapacity
cabbage: storageCapacity
onion: storageCapacity
corn: storageCapacity
tomato: storageCapacity}
showContainerCapacity()
refillContainer
_event: MouseEvent
storageLeft ={
salad: containerCapacity
cabbage: containerCapacity
onion: containerCapacity
corn: containerCapacity
tomato: containerCapacity}
move random staff to click position of Mouse
showContainerCapacity()
set time out 3 min
updateSalad
_event: MouseEvent
let element: string = "salad"
move random staff to click position of Mouse
collect ingredient and push into order[]
Subtract -20 from storageLeft.salad
[if storageLeft.salad <=0]
alert("please cut new salad")
set element to innerHTML
update salad in storage
showContainerCapacity()
updateCabbage
_event: MouseEvent
let element: string = "red cabbage"
move random staff to click position of Mouse
collect ingredient and push into order[]
Subtract 30 from storageLeft.cabbage
[if storageLeft.cabbage <=0]
alert("please cut new cabbage")
Set element to innerHTML
update cabbage in storage
showContainerCapacity()
updateOnion
_event: MouseEvent
let element: string = "onion"
move random staff to click position of Mouse
collect ingredient and push into order[]
Subtract 20 from storageLeft.onion
[if storageLeft.onion <=0]
alert("please cut onion")
set element to innerHTML
update onion in storage
showContainerCapacity()
updateCorn
_event: MouseEvent
let element: string = "corn"
showContainerCapacity()
updateTomato
_event: MouseEvent
let element: string = "tomato"
showContainerCapacity()

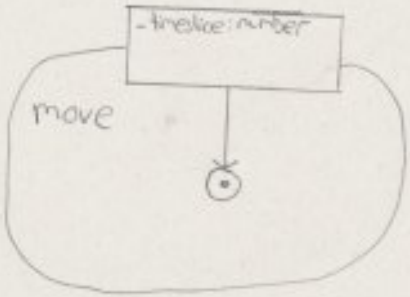

updateKetchup
-event: MouseEvent
let element: String = "ketchup with vegan meat"
move random staff to position of MouseEvent
collect ingredient and push into order[]
set element to inner.HTML
updateSauce
-event: MouseEvent
let element: String = "sauce"
...
set element to inner.HTML
drawStaff
[for let i:number=0, i<staffAmount, i++]
create new Staff
push into Staff[]
[for let staff of Staff]
staff.draw()
set Timeout 2 min
staff.sleep()
set Timeout 2 min
staff.mad()
set staff.isMad to true and pause staff for 3 min
selectedIngredients
let selectDiv: HTMLElement
Set to inner.HTML
updateMustard
-event: MouseEvent
let element: String = "mustard with vegan meat"
...
Set element to inner.HTML
updateLahmacun
-event: MouseEvent
let element: String = Lahmacun with vegan ingredients
...
set element to inner.HTML
compareOrder
-event: Event
[for let i:number=0, i<customerOrder.length, i++]
[else]
[if customerOrder[i] == order[i]]
Subtract 1 from overall satisfaction
clear order and selected ingredient Div
delete customer[0]
delete customer[0] and clear order and selectedIngredients Div
add 1 to sold meals and to overall satisfaction
getOrder()


drawCustomer
2sec
let interval: number: setInterval()
create new customer
push into customer[]
customer.draw()
[if customer.length == customerMax]
clearInterval(interval)
customer.length = 0
setTimeout 2 min
customer.dressed()
setTimeout 2min
customerMax ++ and subtract 1 from deskill skillLevel
set customerIsMad to true and delete customer
getOrder
let object1: number = Math.floor(Math.random() * basis.length
let object2: number = Math.floor(Math.random() * topping.length
let object3: number = Math.floor(Math.random() * sauce.length
let objects: string = basis[object1] + " " + topping[object2] + " " + sauce[object3]
push objects into customerorder[]
let order: string[] = [basis[object1] + topping[object2] + sauce[object3]]
let orderDiv: HTMLElement = document.getElementById("order")
set order to inner.HTML
drawSalad
create new salad
push into ingredient[]
repeat for Tomato, Corn, Onion, Hot cabbage, Sauce, Kebap Yufka and Lahmacun
ingredient.draw()
drawCanter
-position: Vector
crc2.beginPath()
draw Canter with fillRect()
add color with fillStyle
drawCuttingboard
-position: Vector
crc2.beginPath()
draw Cuttingboard with fillRect()
add color with fillStyle
draw knife with arc 0.25 * Math.PI
add color with fillStyle
fill()
crc2.closePath()
drawContainer
-position: Vector
draw container for every ingredient with fillRect()
add color with fillStyle()

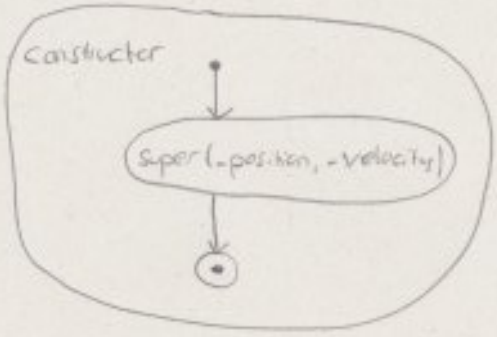
Vegan Döner Trainer: Activity Diagram: Superclass Person

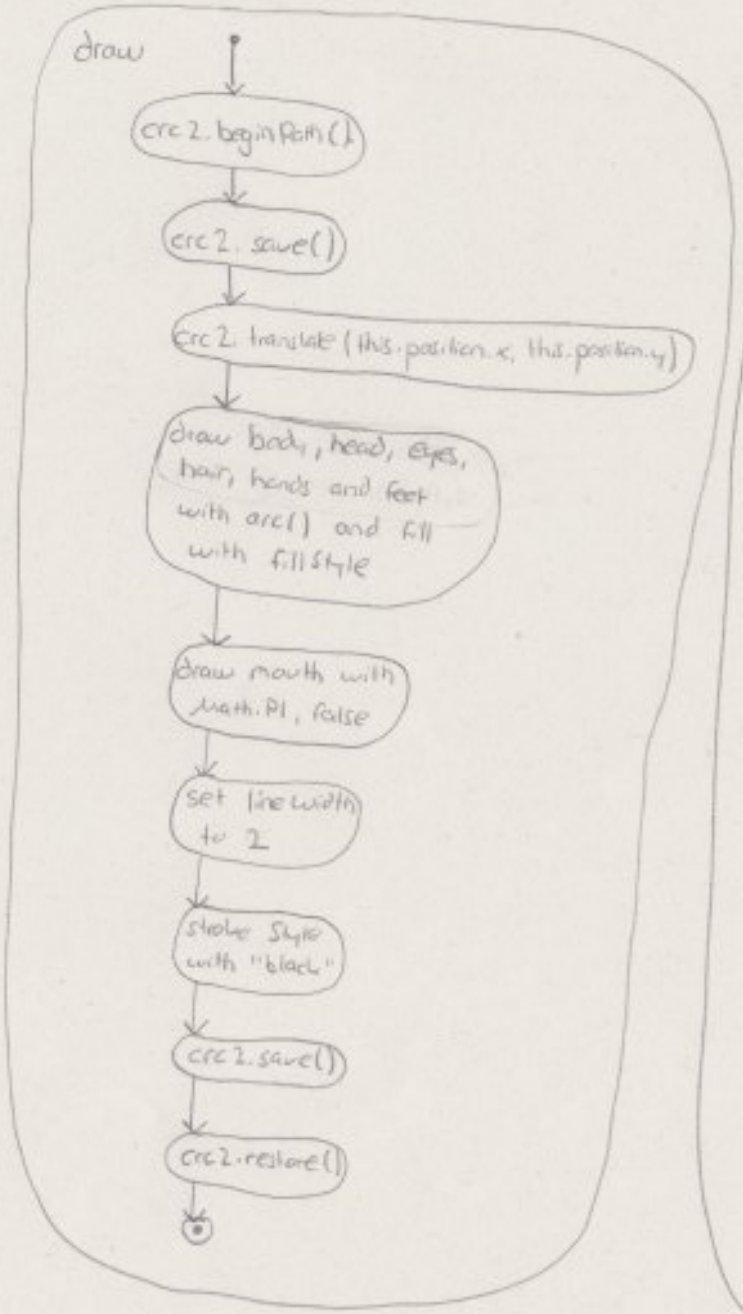


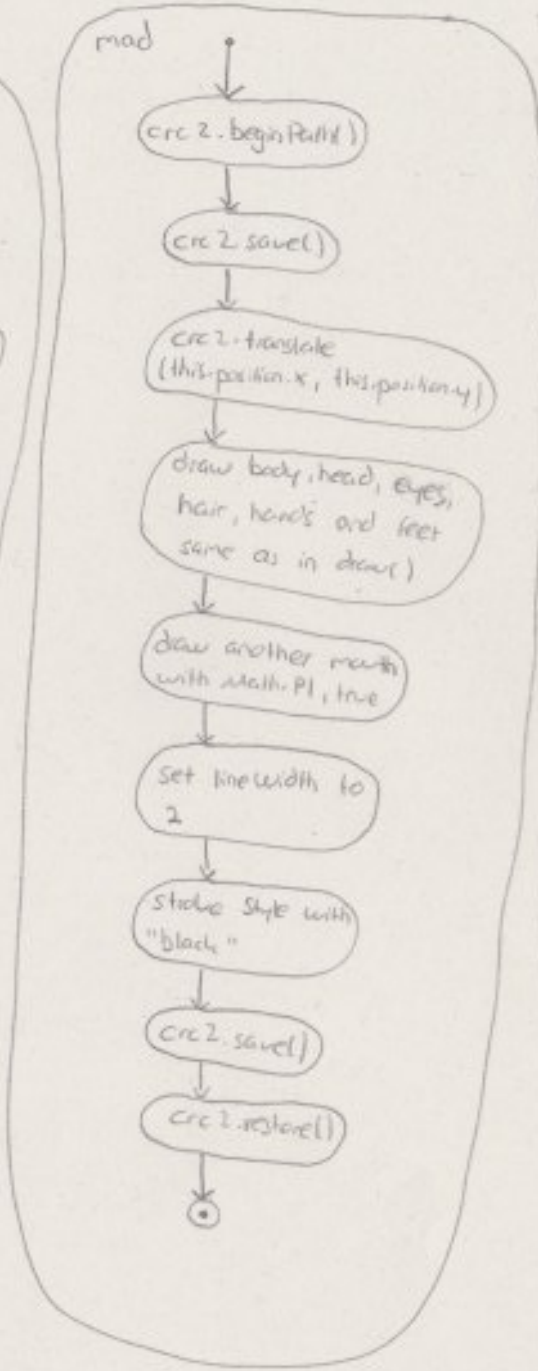


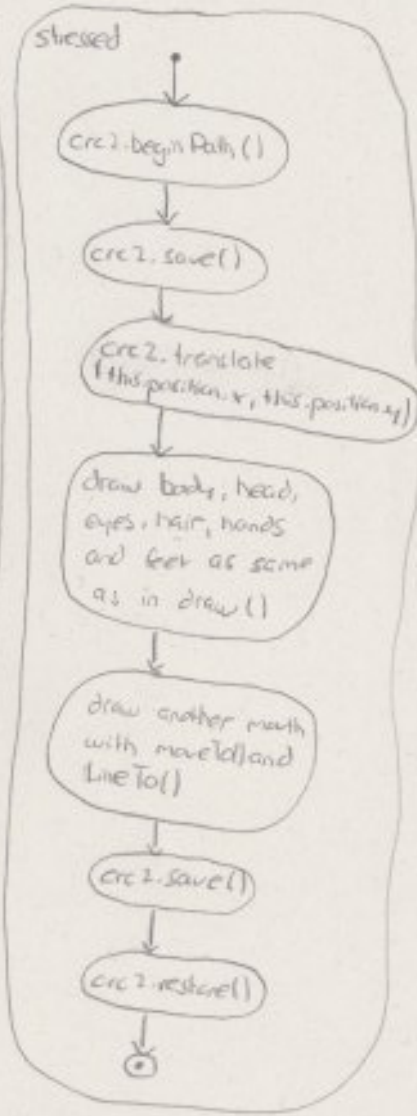


Vegan Döner Trainer: Activity Diagram: Subclass Customer

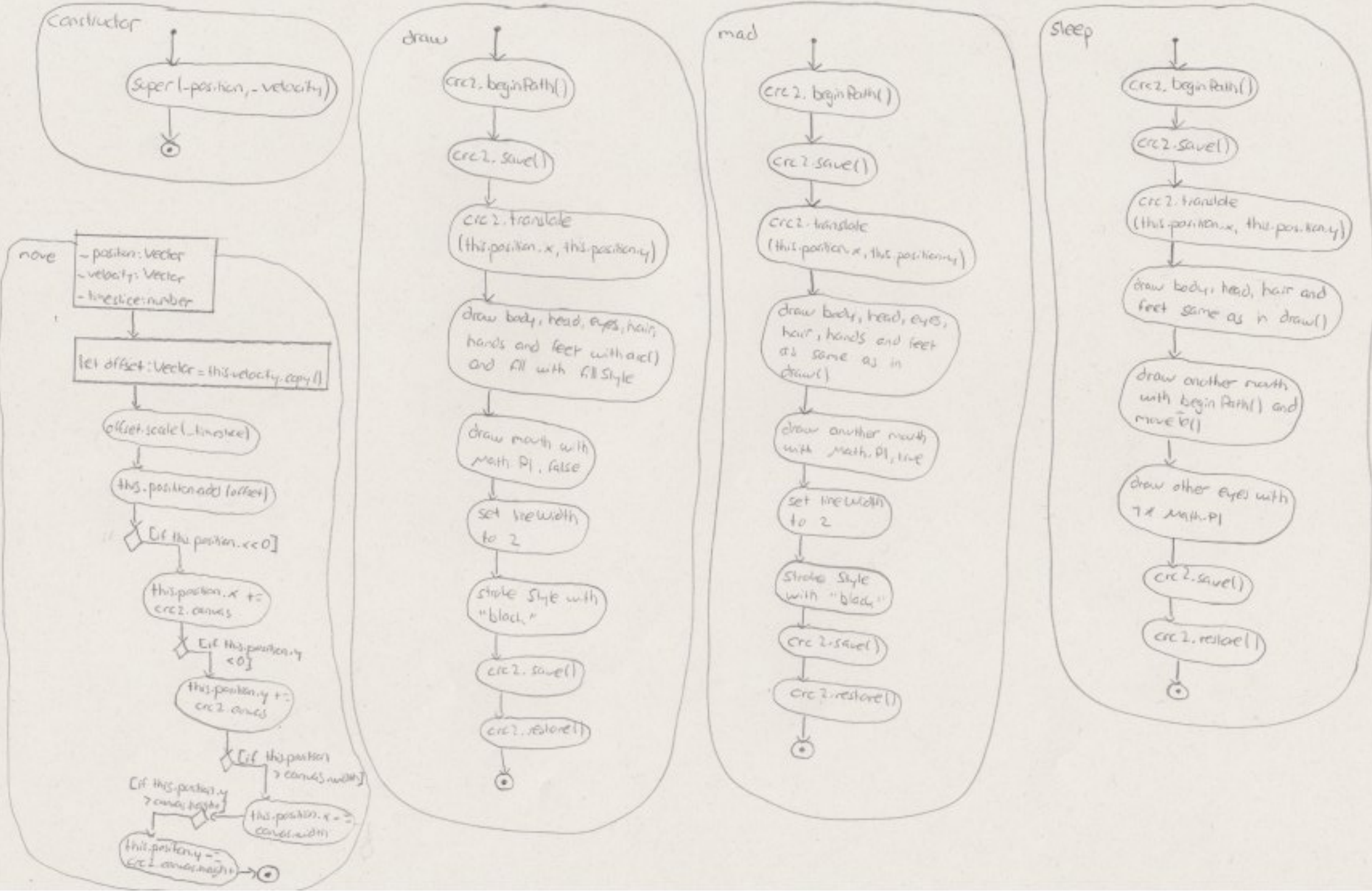




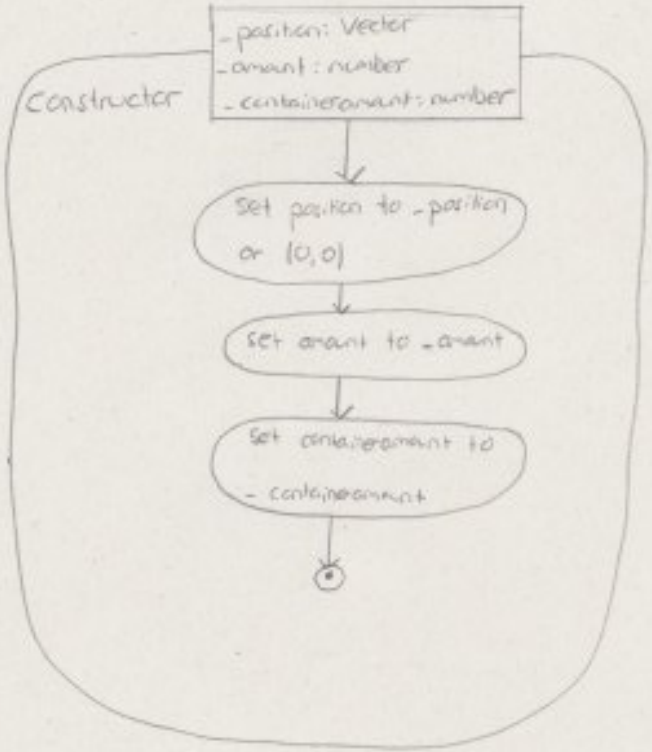


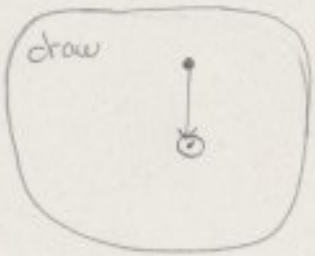


Vegan Döner Trainer: Activity Diagram: Subclass Staff

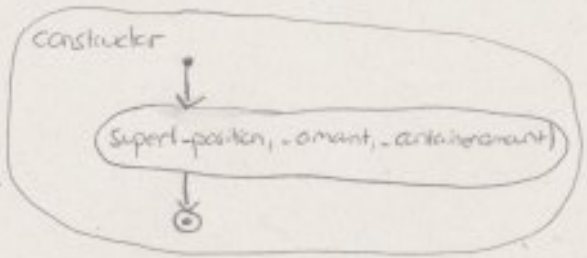


Vegan Döner Trainer : Activity Diagram : Superclass Ingredient

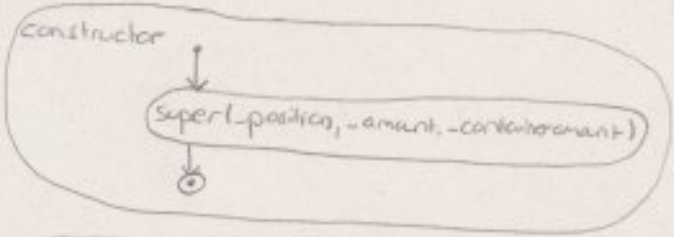




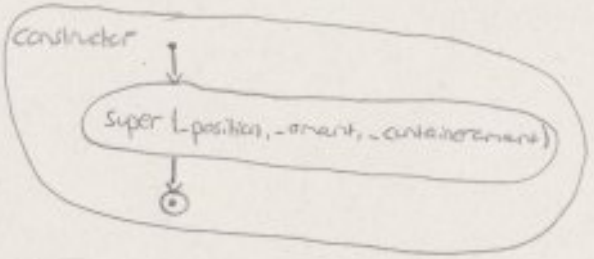
Vegan Diner Trainer: Activity Diagram: Subclass: Cabbage

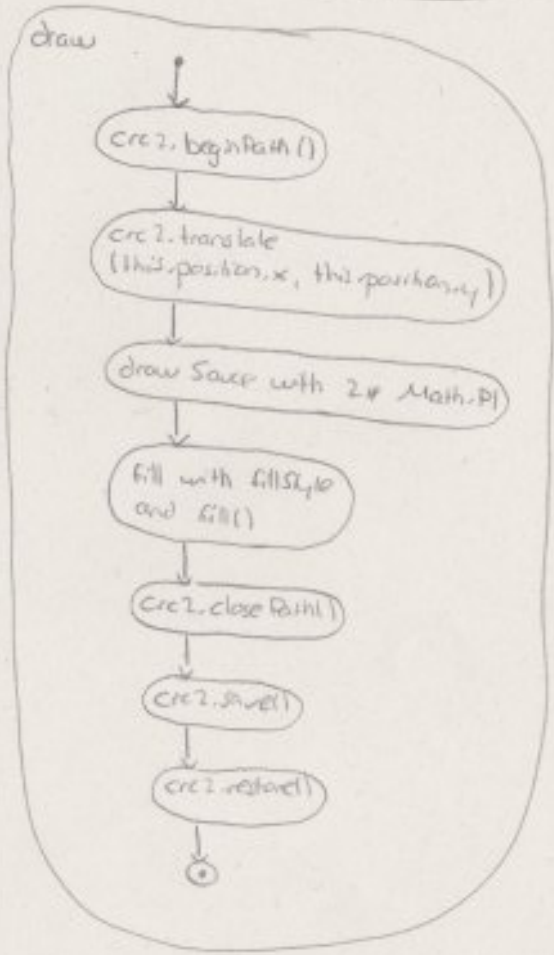


Subclass Corn

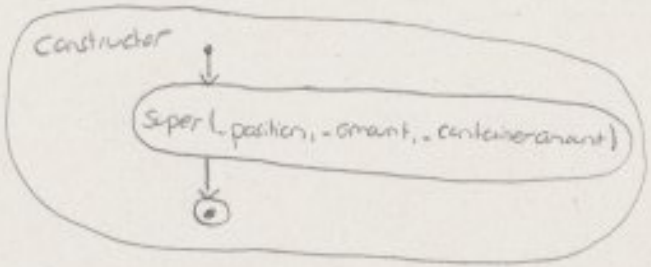


Subclass Sauce

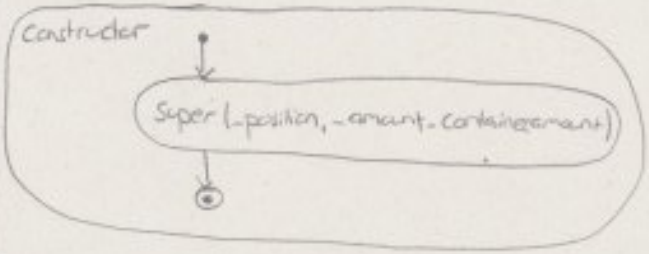




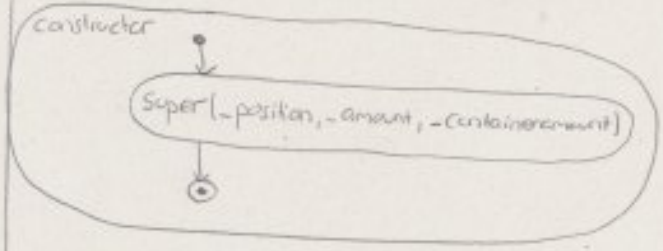
Vegan Döner Trainer: Activity Diagram: Subclass Salad



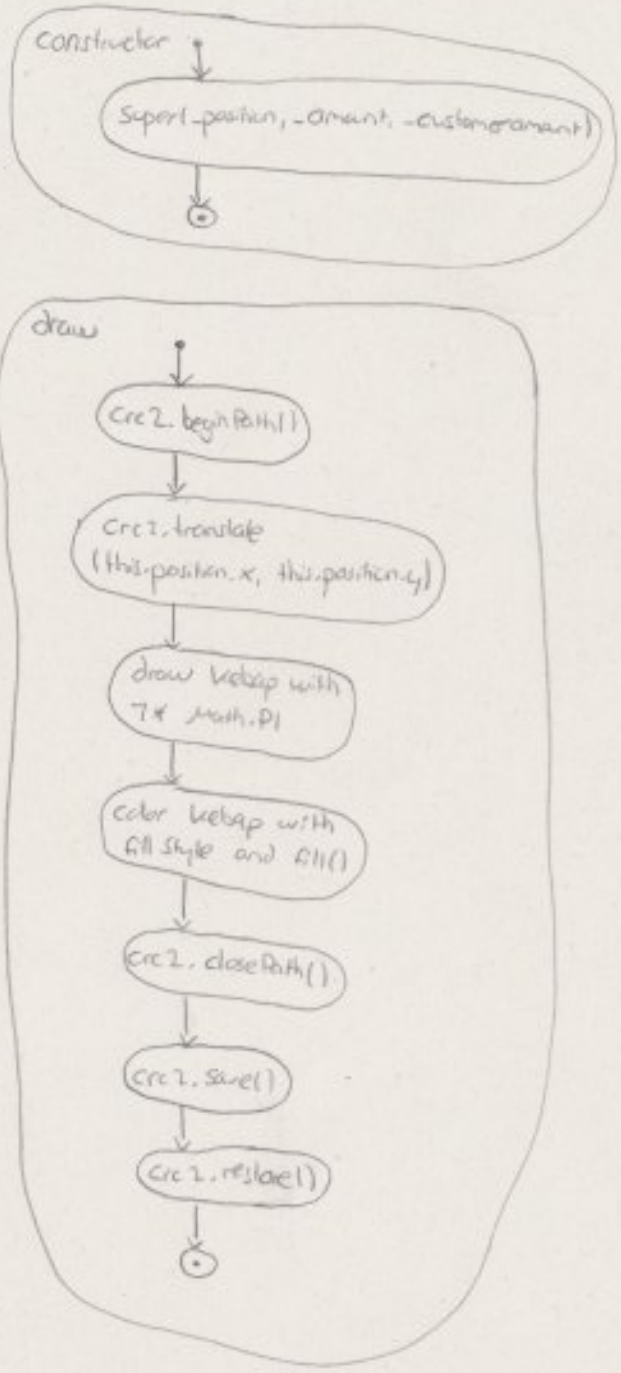
Subclass Tomato



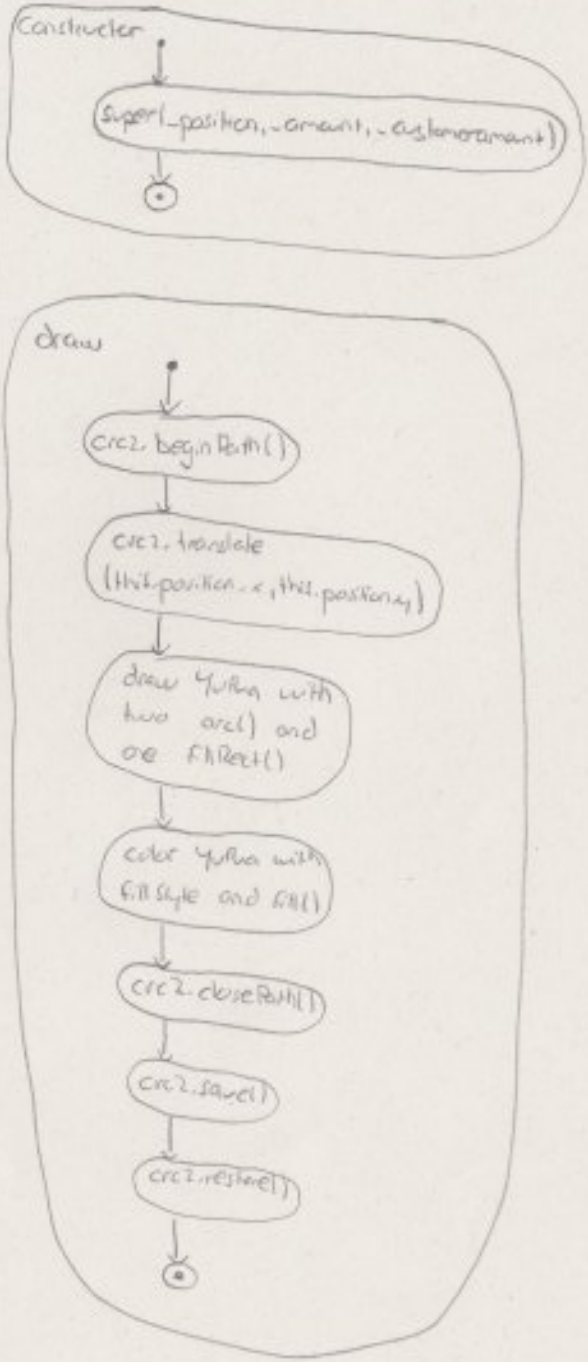
Subclass Onion



Vegan Diner Trainer: Activity Diagram: Subclass Kebap



Subclass Yufka



Subclass Lahmacun