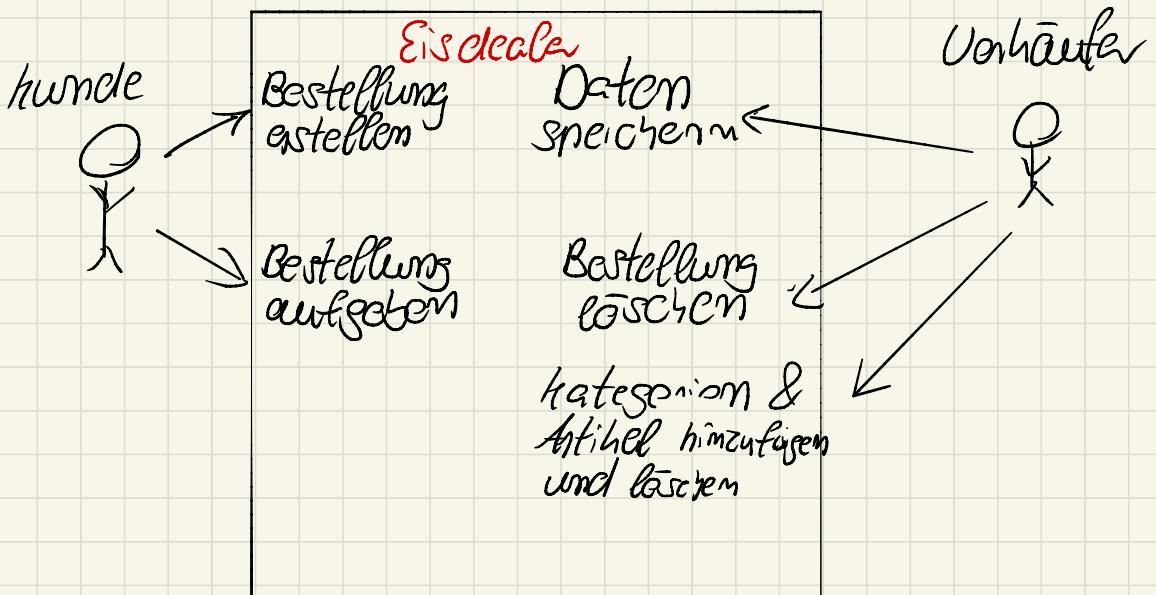


# Konzept EIA

Use Case Diagramm:



# User Experience (Aktion / Reaktion des Nutzen)

Kunde:

- kann Eissorten und andere Optionen auswählen
- kann die Bestellung aufgeben und verwenden

⇒ Warenkorb wird generiert und er bekommt eine Rückmeldung nach Abschluss der Bestellung.

## Verkäufer:

- kann Essorten und Toppings hinzufügen ändern.
- Bestellung speichern und löschen
- Datensatz speichern

⇒

- Der Datensatz ändert sich
- Datensatz wird an den Server geschickt und dort gespeichert
- Aufgegebene Bestellungen werden aus der DB gelöscht

Gerät: Auf dem PC

Grund: Für den Kunden wäre Mobile oder eine App vermutlich am besten. Ich habe mich dennoch für PC entschieden, da die Eingabe neuer Produkte durch den Verkäufer um einiges einfacher am PC ist als auf dem Handy. Bessere Darstellung wird durch den größeren Bildschirm ermöglicht.

Skizze

Name



Warenhaus

Kategorie 1

Col 4

Item 1

Item 2

Col 3

Menge

Menge

Col 3

Preis

Preis

Col 1

€

€

Del  
Item

X

X

Kategorie 2

Item 1

Meng

Preis

€

X

Item 2

Menge

Preis

€

X

Bestellungen für das Management

Benutzen

Client

Server

DB



Zusammenstellung der Zutaten

Bestellung wird entgegengenommen

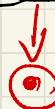
Query-String

Verbindung zum Server Request abschicken

Response empfangen

Aufbereiten

Ausgabe anzeigen



Request empfangen

Bestellung extrahieren

Query für DB

DBAbfrage starten

Ergebnisse aufbereiten

Response erstellen

Response verschicken

Nach Bestellung suchen

Datensätze

Callback abrufen

# Main.ts

DomContentLoaded → function

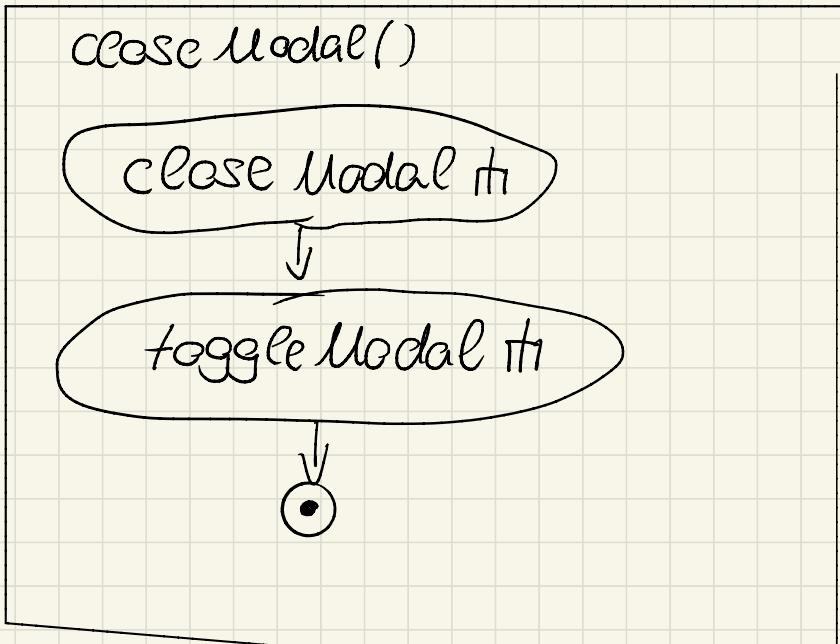
„click“

closeModal()

closeModal()

closeModal()

toggleModal()



toggle modal *th*

ConfirmButton  
"click"

close modal()

- typeOfElement: string  
- placeholderText: string  
- elementToRemove: HTMLElement  
- isHidden: boolean

document.URL.includes  
„orders”

deleteSingleOrder()

# newElements()

newElements()

- element: string
- classes: string
- appendTo: HTMLElement

- element

Amt der Elemente  
wird bestimmt

"input"  
"option"  
"div"  
"button"

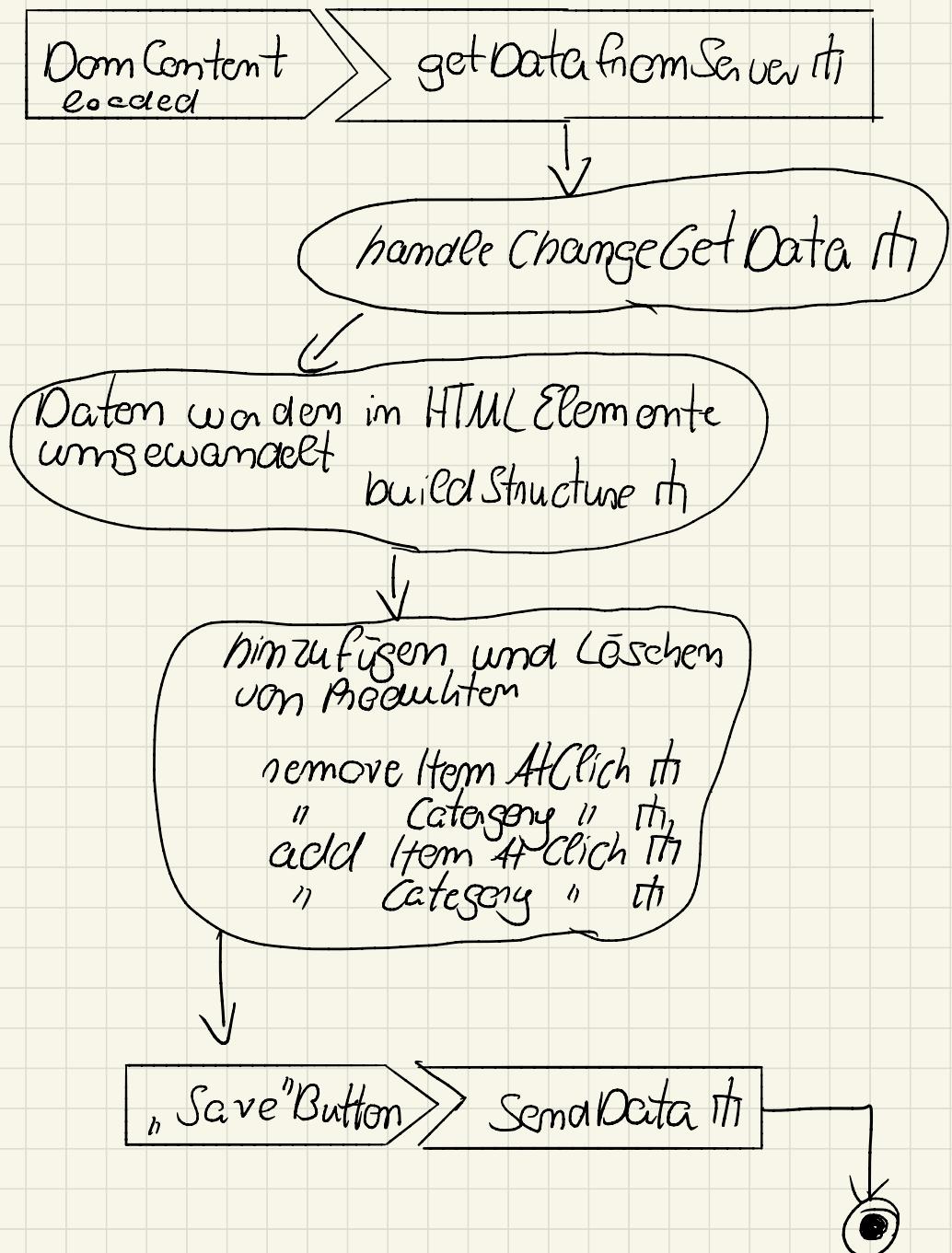
Attribute werden  
übergeben

- classes

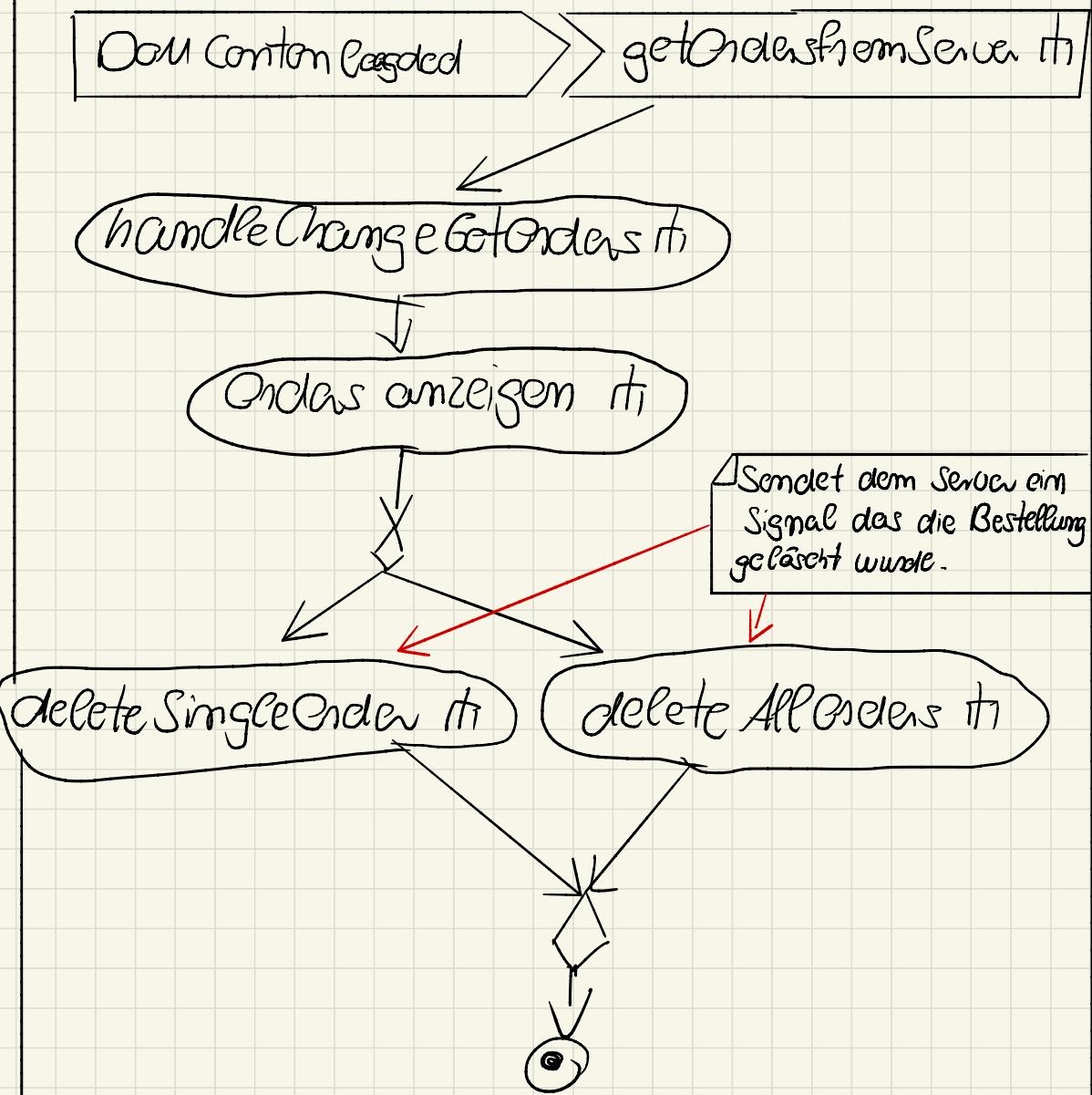
- appendTo

Der Ort wo die  
Elemente  
stehen soll  
wird  
festgelegt

# Configurator.ts



# Orders.ts



buyer.ts

