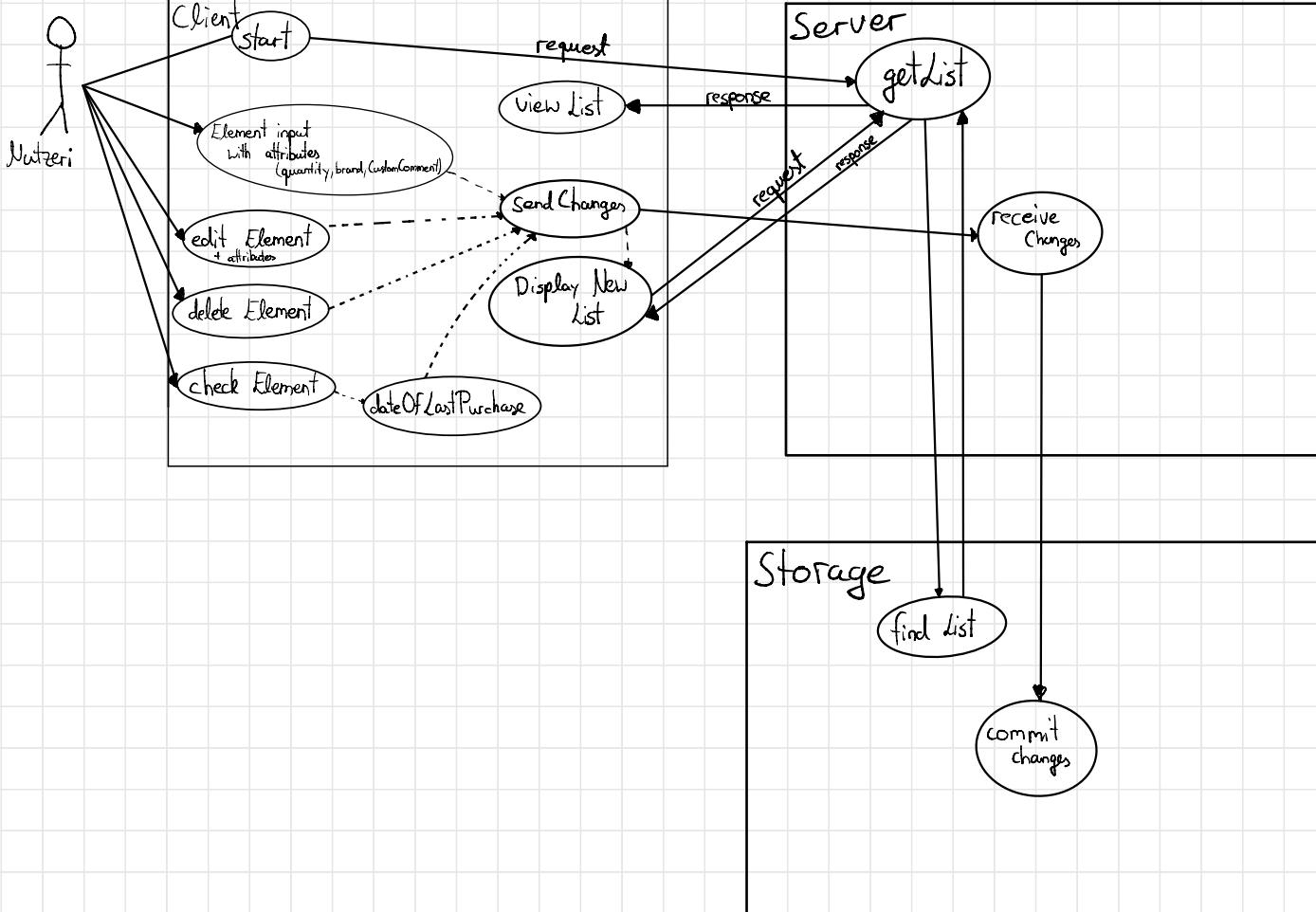


# Use-Case - Diagramm: shopping list



# UI-Scribble : shopping list

```
<input>  
type="text"  
id="product"  
required
```

```
<div>  
id="input" > change
```

```
<h1>
```

```
<div>  
id="display"
```

```
<input>  
type="number"  
name="Quantity"
```

## Shopping List

Product

Enter Product here

Quantity

How much?

Buy next time?

Yes

CustomComment

Add more Details, like Unit

Commit List-Element

List

Milk , 1 , Pack, 17.10.22

Eggs 6 , Pieces

```
<input>  
type="checkbox"  
value="true"
```

```
<fieldset>
```

```
<textarea>  
id="comment"
```

```
<input>  
type="hidden"  
id="check"  
value=""
```

```
<input>  
type="submit"  
id="commit"  
value=""  
> click
```

```
<ul>
```

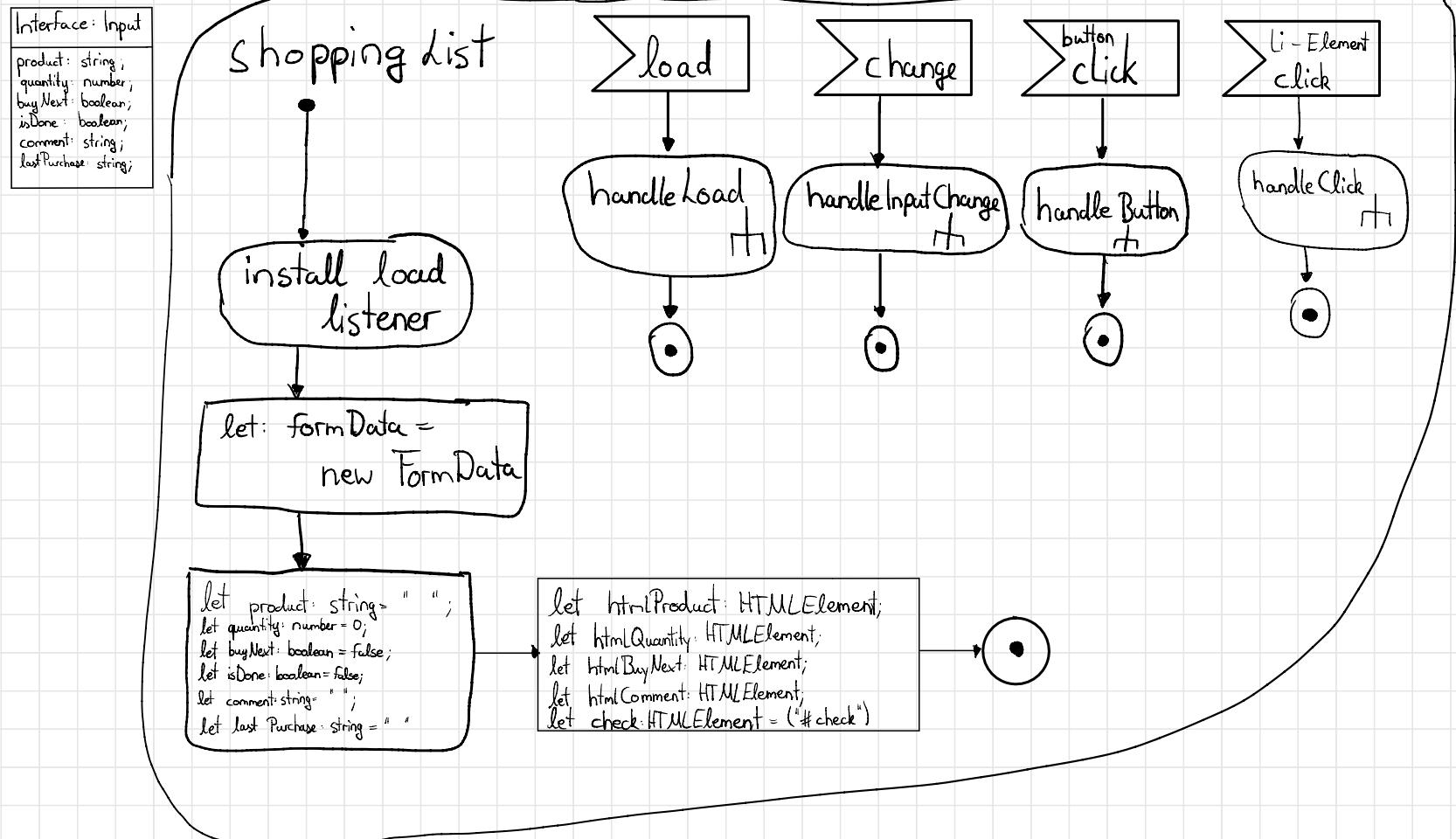
```
<img>  
> click
```

```
<li> > click  
id="listElement" + index  
class="true" → if buyNextTime = Yes  
class="false" → if buyNextTime = No
```

```
<img> > click
```

```
<img> > click
```

# Activity Diagramm



handleLoad

install change  
listener on div  
with id="input"

install click listener  
on button with id="commit"

requestList it!



cutID

\_id: string  
-length: number

```
let newID: string = _id.slice(-length);
```

```
return parseInt(newID);
```

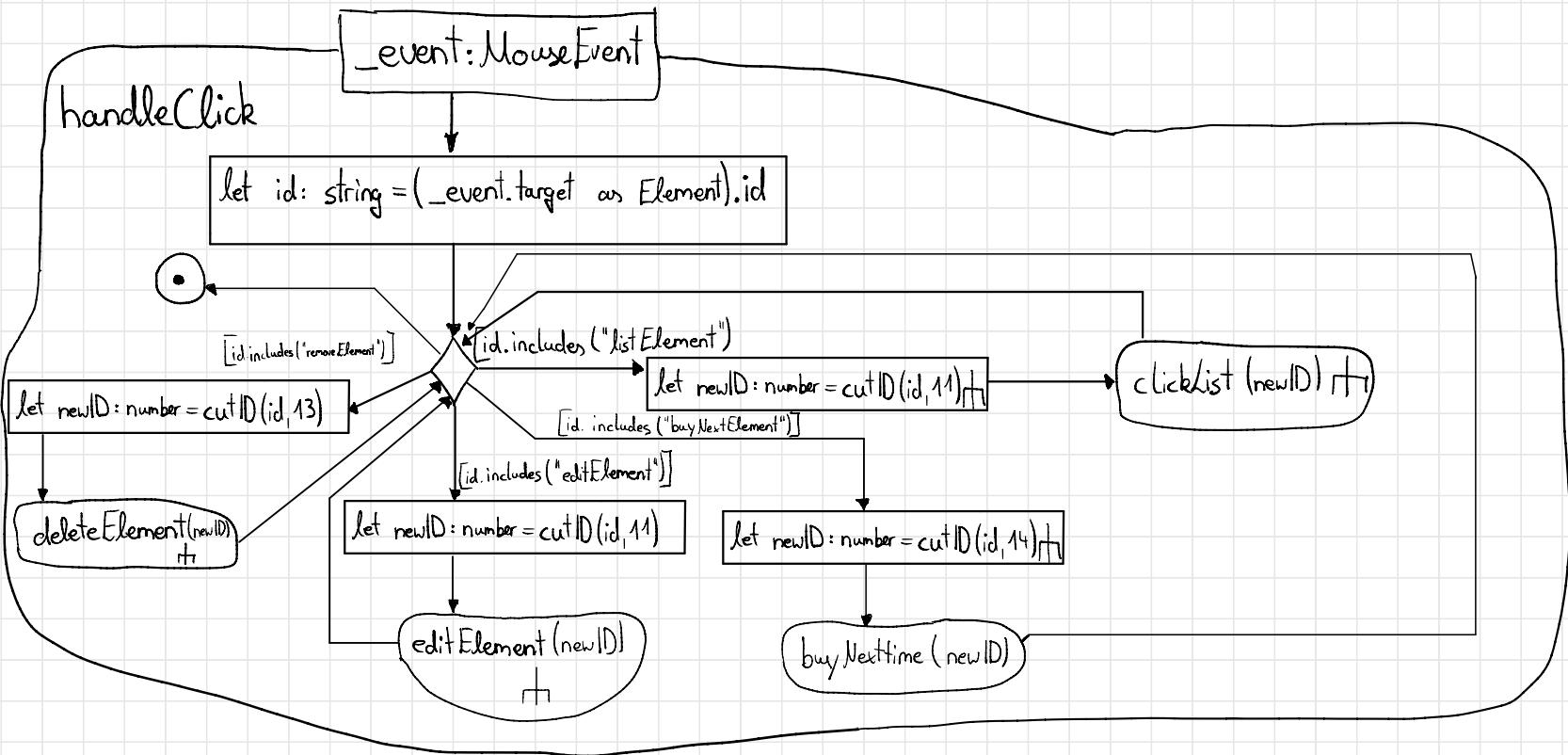
clickList

\_bought: number

```
inputs[_bought].isDone = !inputs[_bought].isDone;  
inputs[_bought].lastPurchase = "current Date";
```

sendListElement (\_bought, update)





delete Element

-element: number

delete inputs[-element]

sendListElement (-element, delete)



buyNexttime

-element: number

inputs[-elements].buyNext =  
!inputs[-elements].buyNext

sendListElement (-elements, update)



handleInputChange

-event: Event

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

let buy: string | undefined =  
formData.get("buyNext")?.toString();

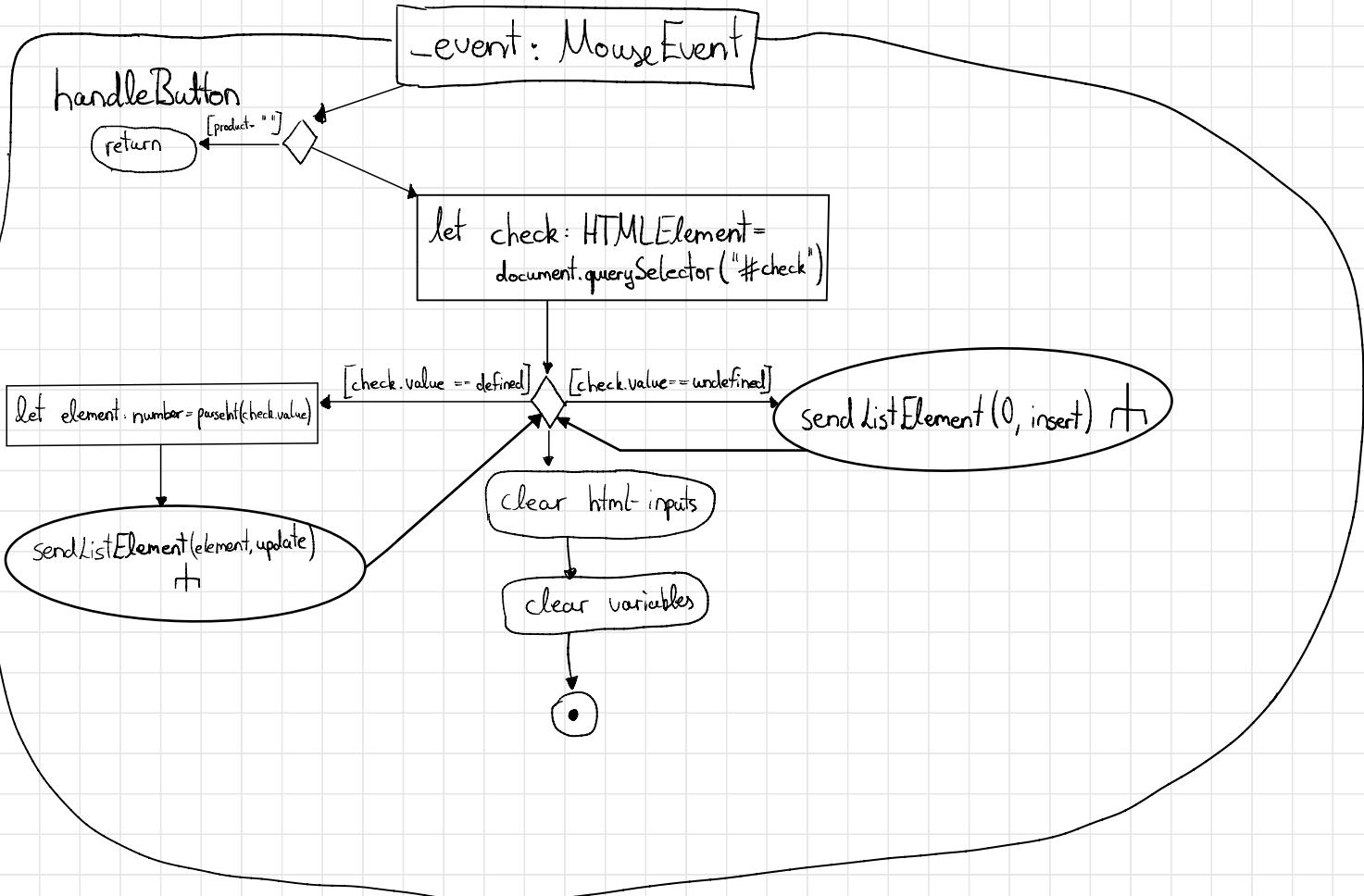
product = ]  
quantity = } form Data  
comment = ]

[buy = "true"]

[buy = "false"]

[buy = "true"]





edit Element

-element: number

htmlProduct → ("# product")  
htmlQuantity → ("# quantity")  
htmlBuyNext → ("# buyNext")  
htmlComment → ("# comment")

htmlProduct.value = inputs[-element].product  
htmlQuantity.value = inputs[-elements].quantity  
htmlBuyNext.value = inputs[-elements].buyNext  
htmlComment.value = inputs[-elements].comment  
check.value = -elements

lastPurchase = inputs[-element].lastPurchase

# write list

