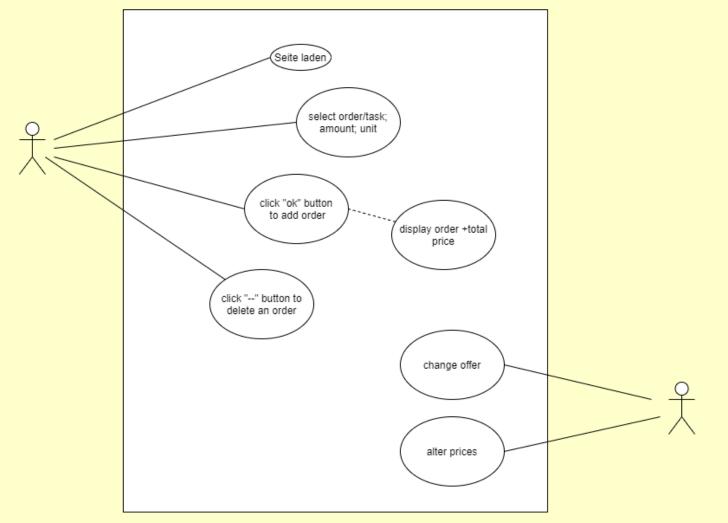
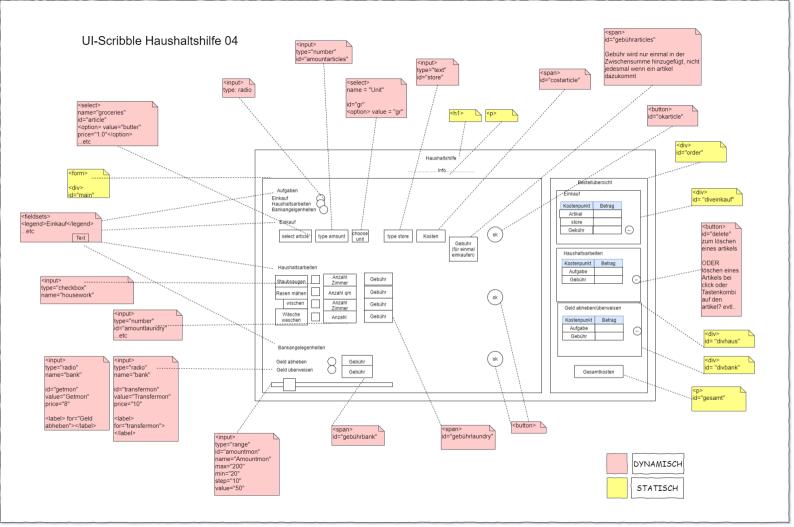
Haushaltshilfe: Use-Case Diagram





Data Structures: Haushaltshilfe

Category	name: string	items: string[]
fieldsets	"Tasks" 	["Einkauf"]
fieldeinkauf	"Groceries" 	["Butter", "Stück", "1.0"]
	"Amount"	["amountarticle"]
	"Store" 	["Edeka"]
fieldhaus	"Householdhelp"	["Staubsaugen", "Zimmer", "1.0"]
	"Amount"	["amounthousehold"]
fieldbank	"Bank"	["GeldÜberweisen", "Euro", "10.0"]

"Amount"

["amountmon"]

Category beinhaltet je nachdem [name, unit, price]

category

data

--> property in generateContent.ts

Beispiel:

let data: Data = { fieldeinkauf: [{name: "Groceries", items: ["Butter", "Stück", "1.0"]}, {name......},

{name: "Amount", items: ["amountarticle"]},

Haushaltshilfe: Avtivity Diagram - generateContent Date none: string items: string[] [category: string] Catgeory[] _items: Category[] _data: bata generateContent let group : create new div create fieldsets with id * disablet true (outer fieldsets "Tosks") return group [_items ovailable] let property : next property append fieldsets to form and legends to fieldsets type = radio name : property.name idv property.items[0] value = property.items[0] oppend group to fieldest (category ovaliable) let fieldset = get fieldset with id = category let items: Cotegory() : _date(category) let group: HTMLElement let lobel = create new lobel element htmlfor = redistosks id sitextContent = property items (O) [category is fieldeste] | group : createTasks (items) | | [category is fieldhaus] [category is fieldbook] group is createBank (items) _items Cotegory[] createEinkauf let group : create new div let selectorticle : create select element let amount : create input element let selectation : create select element let obbutton : create button element _items: Cotegory[] _items: Category[] createHouse let group i create new div let amount : create input element let akbutton i create button elemen create Book ekbuttox.id x "buttonhous" type: "buttox" textContent : "OK" akbutten.id = "buttoneinkauf" type= "butten" textCartent = "OK" okbutton.id = "buttonbonkunge type= "button" textContent = "OK" return group [_items ovailable] neturn group return group * let property = next property Litera moilchiel Litera contribie | let optionators = create option element | [property.nome is "Store"] | [property.nome is "Groceries"] | let optionarticle = create option element let property a next property let property a next property [property.nome is "Householdhelp"] let checkbaxen i create input element let lobel : create label element [property.name is "Bank"] let radiobanks create input element let label = create label element [property.name is "Amount"] selectanticle nome = property nome optionomicle setAttribute("price", property.items[2]) optionomicle setAttribute("uni", property.items[3]) optionomicle setAttribute unit ephceutrical texth property.items[0] optionomicle.sekle = property.items[0] selectatore.nome = property.name optionstare.value = property.items(0) optionstore.text = property.items(0) [property.none is "Amount"] omount, type z "number" name i property.name set min, max, step id s property.items(0) radiobank.type z "radio" set name, id, volue checkboxen.type z "checkbox set name, id, volue amount, type : "number" name i property.name set nin, max, step id : property.itame[0] append optionators to selectators oppond selectators to group checkbexen.setAttribute --- unit and price radiobank.setAttribute --- unit and price append amount to group append aptionarticle to selectarticle oppend selectarticle to group oppend amount to group append amount to group label.htmlfor a checkbasen.id label.textContent append label and checkboxen to group append label and rediabenk to group append akbutton to group append akbutton to group append akbutton to group

L04 Haushaltshilfe: Activity Diagram

