pkg new_overall (only Edible)

persistence

1 persistence

Persistence

- folder : str
- name : str
- data : dict

+ register(data : dict) : void
+ find(filter : dict) : list[dict]
+ delete(data : dict) : void
+ load() : void
+ save() : void
+ filename() : str
+ create() : void
- dict() : dict
- getitem(item : dict) : dict
+ set_database_folder(folder_path : str) : void

EdiblePersistence

- persistence
1   - persistence   1

- persistence

control

EdibleController

+ search_by_name(name : str) : dict
+ delete_by_name(name : str) : void
+ *build_object(data : dict) : Edible*

- controller   1   - controller   1

Controller

- cls : type

+ save(obj : object) : void
+ search(data : dict) : dict
+ delete(data : dict) : void
+ *build_object(data : dict) : void*

view

EdibleRegister

- error_label : ttk.Label()
- confirm_button : ttk.Button()
- frame : ttk.Frame()

+ confirm() : void

+ register   1

EdibleSearch

- selected_name : str
- right : ttk.Frame()
- left : ttk.Frame()
- confirm_button : ttk.Button()
- response_list : tk.Listbox()
- response_scroll : tk.Scrollbar()

+ confirm() : void
+ delete() : void
+ selection_callback() : callable

1

- search

EdiblePanel

+ title : ttk.Label()
+ sep : ttk.Separator()
+ tab_controller : ttk.Notebook()
+ tab_names : list[str]
+ tabs : dict
+ search_frame : tk.Frame()
+ register_frame : tk.Frame()

1   name_entry 1   calories_entry   1   type_selector

Entry

- content : tk.StringVar()
+ label : ttk.Label()

+ clear() : void

Selector

- content : int = tk.StringVar()
- box : ttk.Combobox()
+ label : ttk.Label()

+ clear() : void

Card

- delete_callback : callable
+ title : ttk.Label()
+ delete_button : ttk.Button()

- delete() : void

View

+ title : str
- panels : dict
- selected_panel : dict
- sidebar : ttk.Frame()

+ set_panel(panel_id : str) : void

model

<<enumeration>>
EdibleType

- MAIN_COURSE : str = "Prato principal"
- SIDE_DISH : str = "Acompanhamento"
- MEAT : str = "Carne"
- VEGETARIAN : str = "Vegetariano"
- JUICE : str = "Suco"
- DESSERT : str = "Sobremesa"

Edible

- name : str
- type : str
- calories : int

+ to_dict() : dict
- str() : str
+ attr_labels() : list

pkg model

Restaurant

- price : float
- operating_time : tuple[str,str]
- name : str
- instance : Restaurant

+ to_dict() : dict
+ attr_labels() : tuple[str,str,str]

<<enumeration>>
EdibleType

- MAIN_COURSE : str = "Prato principal"
- SIDE_DISH : str = "Acompanhamento"
- MEAT : str = "Carne"
- VEGETARIAN : str = "Vegetariano"
- JUICE : str = "Suco"
- DESSERT : str = "Sobremesa"

DailyMenu

- price : float
- date : date

+ add_edible(target_list : list[Edible]) : void
+ lunch_add(edible : Edible) : void
+ dinner_add(edible : Edible) : void
+ lunch_erase() : void
+ dinner_erase() : void
+ to_dict() : dict
- str() : str
+ attr_labels() : list

0..*

- lunch

0..*

- dinner

Edible

- name : str
- type : str
- calories : int

+ to_dict() : dict
- str() : str
+ attr_labels() : list

uc use_case

sd Communication diagram

1: add data

2:click register

3.2: save(edible) : void

3.2.1: register(edible) : void

: Admin

: EdibleRegister

: EdibleController

: EdiblePersistence

3.2.1.1: load() : void

3: confirm() : void

3.2.1.1.1: create() : void

3.1: edible = Edible(name_entry, calories_entry, type_selector) : Edible

3.2.1.2: save() : void