Global

Members

awaitFeedback

Array contains all the task objects with the value 'taskStatus': 1

Source: board.js, line 8

currentDraggedElementID

Array of all the task objects from the server

Source: board.js, line 2

currentDraggedElementINDEX

Variable contains the ID of the current dragged element

Source: board.js, line 3

done

Array contains all the task objects with the value 'taskStatus': 2

Source: board.js, line 9

inProgress

Array contains all the task objects with the value 'taskStatus': 0

Source: board.js, line 7

toDo

Variable contains the Index position of the current dragged task object in the allTasksFromStorage[] array

Source: board.js, line 6

Methods

SubtasksDone(subtasks)

This function filters all subtasks with the status "done" in an extra array.

Parameters:

Name	Type	Description
subtasks	Array	An array containing all the subtasks of the task object.

Source: board.js, line 375

Returns:

Returns the length of an array containing all subtasks with the status "done" of the current selected task object

(async) addTaskInit() → {Promise.<void>}

Initializes the add task functionality. Loads templates, contacts, sets current page link active, and renders the add task form.

Source: task_add.js, line 44

Returns:

A promise that resolves when the initialization is complete.

Type
Promise.<void>

allowDrop(ev)

This function loads the preventDefault() method when a element is dragged over a dragable area which cancels the event if it is cancelable, meaning that the default action that belongs to the event will not occur.

Parameters:

Name	Type	Description
ev	*	

Source: board.js, line 148

checkboxRevertToggle(userId)

Reverts the toggle state of a checkbox for a given user ID.

Parameters:

Name	Type	Description
userId	string	The ID of the user.

Source: task_users.js, line 66

checkboxToggle(userId) → {HTMLInputElement}

Toggles the checkbox for a user.

Parameters:

Name	Type	Description
userId	number	The ID of the user.

Source: task_users.js, line 28

Returns:

- The checkbox element for the user.

Type
HTMLInputElement

closeAddContactForm()

This function closes the Add Contact form

Source: contacts.js, line 216

closeAddTaskPopup()

This function closes the Html container containing the "add task" form.

Source: board.js, line 420

closeEditContactForm()

This function closes the Edit Contact Form

Source: contacts.js, line 344

closeEditSavedSubtask(subtaskText, subtaskId)

Closes the edit mode for a subtask.

updateSavedSubtask
userDropDownClick

Parameters:

Name	Type	Description
subtaskText	string	The updated text of the subtask.
subtaskId	string	The ID of the subtask.

Source: task_subtasks.js, line 103

closeEditSubtask()

Closes the edit subtask section and resets the subtask input.

Source: task_subtasks.js, line 43

closeTaskCard()

Function to close the previous rendered task card template by removing the class "show-task-card" from the current selected Html element.

Source: board.js, line 290

closeThumbnailSubmenu(taskID, taskStatus)

This function closes the submenu of a thumbnail card by removing the class "show_task_card_thumbnail_submenu" to the Html element related to the current selected task object. It then empties the Html element in which the submenu links were loaded.

Parameters:

Name	Type	Description
taskID	string	ID of the current selected task object.
taskStatus	number	the taskStatus value of the current selected task object.

Source: board.js, line 235

convertDueDates()

This function parses through the tasks array and converts the dueDates from string to new Date() format

Source: summary.js, line 33

(async) createNewContact()

This function creates a new contact objects, pushes the object to the contacts array and then re-render the page

Source: contacts.js, line 192

createSubtaskListItem(subtaskText) → {void}

Creates a subtask list item and adds it to the select-subtask list.

Parameters:

Name	Type	Description
subtaskText	string	The text of the subtask.

Source: task_subtasks.js, line 147

Returns:

Type
void

(async) deleteContact(id)

This function deletes a object from the contacts array

Parameters:

Name	Type	Description
id	number	index of the object to be deleted

Source: contacts.js, line 272

deleteSubtask(event, subtaskId)

Deletes a subtask from the subtasks array and removes it from the DOM.

Parameters:

Name	Type	Description
event	Event	The event object triggered by the user action.
subtaskId	string	The ID of the subtask to be deleted.

Source: task_subtasks.js, line 186

(async) deleteTask(id) → {Promise.<void>}

Deletes a task with the specified ID.

Parameters:

Name	Type	Description
id	string	The ID of the task to be deleted.

Source: task_add.js, line 301

Returns:

- A promise that resolves when the task is deleted.

Type
Promise.<void>

disableBtnsDefault(button)

Disables the default behavior of a button.

Parameters:

Name	Type	Description
button	string	The ID of the button to disable.

Source: task_add.js, line 156

disablePrioBtns()

Disables the default behavior of buttons.

Source: task_add.js, line 165

(async) editContact(id)

This function saves the edited contact object into the contacts array

Parameters:

Name	Type	Description
id	number	index of the object that was edited

Source: contacts.js, line 316

editModeSavedSubtask(subtaskText, subtaskId)

Entering edit sutask mode.

Parameters:

Name	Type	Description
`subtaskText`	string	The text of the subtask.
`subtaskId`	string	The ID of the subtask.

Source: task_subtasks.js, line 88

`editModeSubtask()`

Adds the first subtask to the list if the input is not empty. Generates a unique ID for the subtask, creates a list item, and resets the subtask input field.

Source: tasks.js, line 377

`editModeSubtask()`

Adds the first subtask to the list if the input is not empty. Generates a unique ID for the subtask, creates a list item, and resets the subtask input field.

Source: task_subtasks.js, line 12

`(async) filterUsers()`

Filters the users based on the search input and renders the filtered list. If the users list is empty, it fetches the contacts remotely.

Source: task_users.js, line 195

`focusEditSavedSubtask(elementId)`

Sets focus on the input element with the specified ID.

Parameters:

Name	Type	Description
`elementId`	string	The ID of the input element.

Source: task_subtasks.js, line 114

`generateUniqueID() → {number}`

Generates a unique ID based on the current timestamp.

Source: task_add.js, line 262

Returns:

The generated unique ID.

Type
number

getSelectedUser()

Retrieves the selected user from the user-select element and adds their ID to the assignUserList array.

Source: task_users.js, line 7

(async) getTaskById(id) → {Object}

Retrieves a task by its ID.

Parameters:

Name	Type	Description
id	string	The ID of the task.

Source: task_add.js, line 290

Returns:

- The task object matching the provided ID.

Type
Object

(async) getTaskCategorieList() → {Promise.<Array>}

Retrieves the task category list.

Source: task_add.js, line 218

Returns:

The task category list.

Type
Promise.<Array>

getTaskStatusByIndex(idx) → {string}

Retrieves the task status by index.

Parameters:

Name	Type	Description
idx	number	The index of the task status.

Source: task_add.js, line 281

Returns:

The task status.

Type
string

(async) getTasks()

This asynchronous function loads all the task objects from the server into a local array, after formated into JSON format.

Source: board.js, line 36

greetUser()

This function greets the user based on the logged in user and time of the day

Source: summary.js, line 124

hideContactDetails()

This function hides the Contact details

Source: contacts.js, line 172

hideMore()

This function hides the showMore Menu which contains edit and delete buttons on mobile design

Source: contacts.js, line 368

highlight(id)

Function to add the class "drag_area_highlight" to a Html element by ID.

Parameters:

Name	Type	Description
`id`	string	ID of the Html element to which the class is added.

Source: board.js, line 183

highlightActiveContact()

This funtion removes the .contact-entry-active css class from all elements and then adds the class to the element who triggered the function

Source: contacts.js, line 101

highlightInvalid(element)

This function adds the .input-invalid css class when input is not valid

Parameters:

Name	Type	Description
`element`	string	

Source: contacts.js, line 384

highlightLatestContact()

This function highlights the last created contact in the contact list

Source: contacts.js, line 239

incertSubtask()

Inserts a subtask into the task list.

Source: task_subtasks.js, line 26

(async) init()

This asynchronous function loads required functions for the board.html page to load.

Source: board.js, line 15

(async) init()

This function loads several functions required on page load

Source: contacts.js, line 6

(async) init()

This function loads several functions required on page load

Source: summary.js, line 7

loadBoard()

Function to load the board columns with the attached task objects on the board.html.

Source: board.js, line 26

(async) loadContacts()

This function loads all contact objects from the remote storage - using getItem function - and saves them to the contacts array

Source: contacts.js, line 17

loadEditContactValues(id)

This function loads the contact objects property values into the Edit Contact Form

Parameters:

Name	Type	Description
id	number	

Source: contacts.js, line 302

(async) loadEditTaskForm(elementId, taskID)

This function loads the Html task form to editing an existing task object.

Parameters:

Name	Type	Description
`elementId`	string	The ID of the Html element in which the task form is loaded into.
`taskID`	string	ID of the current selected task object.

Source: board.js, line 388

`loadMoveTo(taskID, taskStatus)`

This function loads the "Move To" category links to the submenu of the thumbnail, of the current selected task object.

Parameters:

Name	Type	Description
`taskID`	string	ID of the current selected task object.
`taskStatus`	number	the taskStatus value of the current selected task object.

Source: board.js, line 247

`loadNoSubtasksInThumbnail(element)`

This function loads all the subtasks of the current selected task object in the Html thumbnail card.

Parameters:

Name	Type	Description
`element`	object	The current selected task object containing all task data.

Source: board.js, line 351

`loadPrioTasks()`

This funtion parses through the tasks array to push all objects with high priority to the prioTasks array

Source: summary.js, line 68

`loadSearchResult()`

This function loads all the filtered task objects from the allTasksFromStorage[] array, related to the value of the "find task" inputfield.

Source: board.js, line 44

loadSubtasksInCard(taskID, subtasks)

This function loads all the subtasks of the current selected task object in the Html task card.

Parameters:

Name	Type	Description
taskID	string	The ID of the current selected task object.
subtasks	Array	An array containing all the subtasks of the task object.

Source: board.js, line 337

Returns:

It returns the subtasks of the task object. If there are no subtasks in the task object it returns a string "No Subtasks".

(async) loadTasks()

This function loads all tasks objects from the remote storage - using getItem function - and saves them to the tasks array

Source: summary.js, line 21

moveTo(task_status)

The function changes the taskStatus value of the current task object after it is dropped.

Parameters:

Name	Type	Description
task_status	number	The status value of the task object.

Source: board.js, line 170

noTaskDoneHTML()

Html Template to render a Notification in the board column.

Source: board_HTML_Templates.js, line 119

Returns:

Returns the Html code for the notification, that the Board Column titled "Done" is empty of task objects.

noTaskToDoHTML()

Html Template to render a Notification in the board column.

Source: board_HTML_Templates.js, line 108

Returns:

Returns the Html code for the notification, that the Board Columns titled "Todo, In progress, Await Feedback" are empty of task objects.

normalDate(date) → {string}

Converts a given date to a normal date format (dd/mm/yy).

Parameters:

Name	Type	Description
date	Date	The date to be converted.

Source: task_render.js, line 121

Returns:

The date in the format dd/mm/yy.

Type
string

normalDateEditTask(date) → {string}

Converts a date string to a formatted date string in the format "YYYY-MM-DD".

Parameters:

Name	Type	Description
date	Date	The date to be converted.

Source: task_render.js, line 138

Returns:

The formatted date string.

Type
string

openAddTaskPopup(taskstatus)

This function loads the Html container containing the "add task" form.

Parameters:

Name	Type	Description
taskstatus	number	Status value of the new added task. By default the value is set to 0 (= to do).

Source: board.js, line 410

openTaskCard(elementByID, cardID)

Function to load the Html template of the task card of the current selected task object containing all data of the task object.

Parameters:

Name	Type	Description
elementByID	string	The ID of the Html element in which the task card template, containing all data of the current selected task object, should be loaded.
cardID	string	The ID of the current selected task object.

Source: board.js, line 281

openThumbnailSubmenu(taskID, taskStatus)

This function opens the submenu of a thumbnail card by adding the class "show_task_card_thumbnail_submenu" to the Html element related to the current selected task object.

Parameters:

Name	Type	Description
taskID	string	ID of the current selected task object.
taskStatus	number	the taskStatus value of the current selected task object.

Source: board.js, line 222

overwriteAddTaskFormCSS()

This function overwrites the css rules of the "add task" form

Source: board.js, line 397

prioButtonLow()

Sets the task priority to low and updates the priority buttons accordingly.

Source: task_add.js, line 145

prioButtonMedium()

Sets the task priority to medium and updates the priority buttons accordingly.

Source: task_add.js, line 135

prioButtonUpdate(pressed, btn)

Updates the priority button based on the pressed state.

Parameters:

Name	Type	Description
pressed	boolean	The state of the button (true if pressed, false otherwise).
btn	string	The ID of the button element.

Source: task_add.js, line 112

prioButtonUrgent()

Sets the task priority to urgent and updates the priority buttons accordingly.

Source: task_add.js, line 125

removeAssignedUser(id)

Removes the assigned user with the specified ID from the list.

Parameters:

Name	Type	Description
id	number	The ID of the user to be removed.

Source: task_users.js, line 125

removeHighlight(id)

Function to remove the class "drag_area_highlight" to a Html element by ID.

Parameters:

Name	Type	Description
id	string	ID of the Html element from which the class is removed.

Source: board.js, line 192

renderAddTaskForm(elementId, setTaskStatus*opt*)

Renders the add task form in the specified element.

Parameters:

Name	Type	Attributes	Default	Description
elementId	string			The ID of the element where the form will be rendered.
setTaskStatus	string	<optional>	TASK_STATUS_TODO	The status of the task to be set by default.

Source: task_render.js, line 155

renderAssignedUserIcons(assignedUserList) → {string}

Renders the assigned user icons.

Parameters:

Name	Type	Description
assignedUserList	Array	The list of assigned users.

Source: task_render.js, line 71

Returns:

The HTML representation of the assigned user icons.

Type
string

renderAssignedUsers(element)

This function renders all the assigned users of the current selected task object into the task card template.

Parameters:

Name	Type	Description
element	object	The current selected task object.

Source: board.js, line 314

Returns:

- If a curtain number of assigned users is rendered, it returns a Html template, containing the remaining assigned users as a summarized number.

renderAwaitFeedback()

Function to render all filtered task objects with the value 'taskStatus': 2 from the allTasksFromStorage[] array into the "Await feedback" column.

Source: board.js, line 93

renderCategoriesList(name, colour, index) → {string}

Renders a category list item with the given name, colour, and index.

Parameters:

Name	Type	Description
name	string	The name of the category.
colour	string	The colour of the category.
index	number	The index of the category.

Source: task_add.js, line 185

Returns:

The HTML representation of the category list item.

Type
string

renderContactDetails(id)

This function renders the contact details

Parameters:

Name	Type	Description
id	number	index of the object within the contacts array

Source: contacts.js, line 134

renderContactDetailsMobile(id)

This function is a sublement to renderContactDetails(id) to optimize design on mobile devices

Parameters:

Name	Type	Description
id	number	index of the object within the contacts array

Source: contacts.js, line 156

renderContactInitials(initials, id)

This funtion returns the html to render a contact picture showing the initials of the contacts first and last name

Parameters:

Name	Type	Description
initials	*	used to hand over the initials
id	*	used to hand over the index of the contacts array of the related contact to load the background color

Source: contacts.js, line 89

Returns:

the html to render a contact picture showing the initials of the contacts first and last name

renderContactList()

This function renders the contact list with all objects found in the constacts array Entries are sorted alphabetically using the sortContactsAtoZ() function

Source: contacts.js, line 29

renderDone()

Function to render all filtered task objects with the value 'taskStatus': 3 from the allTasksFromStorage[] array into the "Done" column.

Source: board.js, line 112

renderEditModeSavedSubtask(subtaskText, subtaskId) → {string}

Renders the HTML for the subtask edit mode.

Parameters:

Name	Type	Description
subtaskText	string	The text of the subtask.
subtaskId	string	The ID of the subtask.

Source: task_subtasks.js, line 63

Returns:

The HTML markup for the subtask edit mode.

Type
string

(async) renderEditTaskForm(elementId, taskId) → {Promise.<void>}

Renders the edit task form with the provided task details.

Parameters:

Name	Type	Description
elementId	string	The ID of the HTML element where the form will be rendered.
taskId	string	The ID of the task to be edited.

Source: task_render.js, line 182

Returns:

- A promise that resolves when the form is rendered.

Type
Promise.<void>

(async) renderFullUsersList() → {Promise.<void>}

Renders the full list of users by fetching data from the remote server and rendering it on the HTML page.

Source: task_users.js, line 17

Returns:

A promise that resolves when the rendering is complete.

Type
Promise.<void>

renderHTMLUsersList(usersList)

Renders the HTML for the users list dropdown.

Parameters:

Name	Type	Description
usersList	Array	The list of users.

Source: task_users.js, line 75

renderInProgress()

Function to render all filtered task objects with the value 'taskStatus': 1 from the allTasksFromStorage[] array into the "In progresso" column.

Source: board.js, line 74

renderMetricValues()

This function calls the returnNumberOfTasks() function to fill the innerHTML with the relevant metrics to be displayed

Source: summary.js, line 98

renderNoTaskDone(category)

This function loads the Html template for the notifacation "No Task Done" in the empty column of the category "Done".

Parameters:

Name	Type	Description
`category`	string	ID of the column "Done".

Source: board.js, line 210

`renderNoTaskToDo(category)`

This function loads the Html template for the notifacation "No Task To Do" in an empty column of category.

Parameters:

Name	Type	Description
`category`	string	ID of the column of category.

Source: board.js, line 201

`renderPriority(prio, renderName) → {string}`

Renders the priority of a task.

Parameters:

Name	Type	Description
`prio`	string	The priority of the task.
`renderName`	boolean	Indicates whether to render the priority name.

Source: task_render.js, line 86

Returns:

The HTML representation of the priority.

Type
string

`renderSubtask(taskID, subtask) → {string}`

Renders a subtask element.

Parameters:

Name	Type	Description
`taskID`	number	The ID of the parent task.
`subtask`	object	The subtask object.

Source: task_render.js, line 39

Returns:

The HTML representation of the subtask element.

Type
string

`renderSubtaskListItem(subtaskText, subtaskId) → {string}`

Renders a subtask list item with the given subtask text and ID.

Parameters:

Name	Type	Description
`subtaskText`	string	The text of the subtask.
`subtaskId`	string	The ID of the subtask.

Source: task_subtasks.js, line 160

Returns:

The HTML representation of the subtask list item.

Type
string

`renderSubtaskProgressBar(subtasks)`

This function renders the progressbar in the thumbnail card. It is related to the number of checked

Parameters:

Name	Type	Description
`subtasks`	Array	An array containing all the subtasks of the task object.

Source: board.js, line 363

Returns:

Returns the rounded number of all checked subtasks related to the total amount of subtasks as percentage. The number is used to set the width of the progressbar.

`renderSubtasks(taskID, subtasks) → {string}`

Renders the subtasks for a given task.

Parameters:

Name	Type	Description
taskID	string	The ID of the task.
subtasks	Array	An array of subtasks.

Source: task_render.js, line 57

Returns:

- The HTML representation of the subtasks.

Type
string

renderTaskCard(elementId, task)

Renders a task card with the given task data and appends it to the element with the specified ID.

Parameters:

Name	Type	Description
elementId	string	The ID of the element to which the task card will be appended.
task	object	The task object containing the task data.

Source: task_render_taskcard.js, line 7

(async) renderTaskCardBoard(elementId, cardID)

This function renders the Html template of the task card of the current selected task object containing all data of the task object.

Parameters:

Name	Type	Description
elementId	string	The ID of the Html element containing the current selected task object.
cardID	string	The ID of the current selected task object.

Source: board.js, line 301

renderTaskForm(formTitle, title, description, date, prio, assignedUsers, categorySubmit, taskStatus, taskID, subtasksSubmit, elementId) → {string}

Renders a task form with the specified parameters.

Parameters:

Name	Type	Default	Description
`formTitle`	string	Untitled	The title of the form.
`title`	string		The title of the task.
`description`	string		The description of the task.
`date`	number		The due date of the task in milliseconds.
`prio`	string		The priority of the task.
`assignedUsers`	Array		The list of users assigned to the task.
`categorySubmit`	Array		The category for the task.
`taskStatus`	string		The status of the task.
`taskID`	number	0	The ID of the task.
`subtasksSubmit`	Array		The list of subtasks for the task.
`elementId`	string	board_popup_placeholder	The ID of the HTML element where the form will be rendered.

> Source: task_render.js, line 245

Returns:

The HTML code for the task form.

Type
string

renderTaskUserIcon(userID, userColour, userName, marginRight)

Renders a task user icon with the specified user ID, user colour, user name, and margin right.

Parameters:

Name	Type	Description
userID	number	The ID of the user.
userColour	string	The colour of the user icon.
userName	string	The name of the user.
marginRight	number	The margin right value for the user icon.

Source: task_users.js, line 153

renderTaskUses(assignedUserList)

Renders the task users on the selected users container.

Parameters:

Name	Type	Description
assignedUserList	Array	The list of assigned users.

Source: task_users.js, line 137

renderThumbnailCard(category, index, element)

Function to render the Html template of the thumbnail card of the current selected task object containing previewed data of the task object.

Parameters:

Name	Type	Description
category	number	ID of the category column in which the Html template should be rendered.
index	number	Position of the current selected task object in the array allTasksFromStorage[].
element	object	The current selected task object.

Source: board.js, line 269

renderToDo()

This unction renders all filtered task objects with the value 'taskStatus': 0 from the allTasksFromStorage[] array into the "To Do" column.

Source: board.js, line 55

`renderUserIconDropdown(user) → {string}`

Renders the user icon dropdown.

Parameters:

Name	Type	Description
`user`	object	The user object.

Source: task_users.js, line 49

Returns:

The HTML string representing the user icon dropdown.

Type
string

`resetAddContactForm()`

This function resets the Add Contact form

Source: contacts.js, line 229

`resetEditContactForm()`

This function resets the Edit Contact Form

Source: contacts.js, line 334

`resetSubtaskInput(subtaskInput)`

Resets the value of a subtask input field.

Parameters:

Name	Type	Description
`subtaskInput`	HTMLInputElement	The input field to be reset.

Source: task_add.js, line 271

returnContactListCategory(character)

This function returns the html to render the contact list categories

Parameters:

Name	Type	Description
character	character	initial letter for the category

Source: contacts.js, line 54

Returns:

html to render the contact list categories

returnContactListEntry(id)

This Function return the html to render a contact list entry

Parameters:

Name	Type	Description
id	number	used to hand over the index of the contacts array of which the contact entry should be generated

Source: contacts.js, line 68

Returns:

the html to render the contact list entry

returnNextDueDate() → {string}

This function parses through the prioTasks array and

Source: summary.js, line 79

Returns:

the next upcoming due date

Type
string

returnNumberOfTasks(property, PropertyValue) → {number}

This function parses throuh the tasks array to return the number of objects matching the criteria set by the parameters

Parameters:

Name	Type	Description
`property`	string	parameter to specify the objects property
`PropertyValue`	string	parameter to specify the property value

Source: summary.js, line 47

Returns:

the number of objects matching the criteria set by the parameters

Type
number

`returnTotalTasks() → {number}`

This function parses throuh the tasks array

Source: summary.js, line 60

Returns:

- total number of objects

Type
number

`(async) searchTask()`

This asynchronous function compares the input value of the inputfield on the board page and filters all task objects from the allTasksFromStorage[] array which includes the inputfield value in their title and/or the description. The result is loaded in an array named filteredTasks which is then loaded back into the allTasksFromStorage[] array.

Source: board.js, line 133

`selectOption(checkbox, id)`

Handles user selection in the dropdown menu. If the checkbox is checked, the corresponding user icon is added to the list and removed from the dropdown.

Parameters:

Name	Type	Description
checkbox	HTMLInputElement	The checkbox element.
id	number	The ID of the selected option.

Source: task_users.js, line 172

(async) selectOptionCat(element, index) → {Promise.<void>}

Handles category selection in the categories dropdown and updates the category value.

Parameters:

Name	Type	Description
element	HTMLElement	The element representing the selected option.
index	number	The index of the selected option in the dropdown.

Source: task_add.js, line 228

Returns:

- A promise that resolves when the selected category is retrieved.

Type
Promise.<void>

(async) selectSubtaskStatus(checkbox, taskID, subtaskID) → {Promise.<void>}

Selects or deselects a subtask status based on the checkbox state.

Parameters:

Name	Type	Description
checkbox	HTMLInputElement	The checkbox element representing the subtask status.
taskID	string	The ID of the task containing the subtask.
subtaskID	string	The ID of the subtask.

Source: task_render.js, line 10

Returns:

- A promise that resolves once the subtask status is updated.

Type
Promise.<void>

showAddContactForm()

This function shows the Add Contact Form

Source: contacts.js, line 180

showAddedTaskMsg()

Shows a message indicating that a task has been added.

Source: task_add.js, line 97

(async) showContactCreatedNotification()

This function shows a notification when a new contact is created

Source: contacts.js, line 258

showEditContactForm(id)

This function shows the Edit Contact Form

Parameters:

Name	Type	Description
id	number	index of the object in the contacts array which should be edited

Source: contacts.js, line 287

showMore()

This function opens the showMore Menu which contains edit and delete buttons on mobile design

Source: contacts.js, line 357

showPopUp(elementId, show)

Toggles the visibility of a popup element.

Parameters:

Name	Type	Description
`elementId`	string	The ID of the element to show or hide.
`show`	boolean	Determines whether to show or hide the element.

Source: task_render.js, line 415

`sortContactsAtoZ()`

This function copies the contacts array to the sortedContacts array and then sorts the array from a to z based on the first letter of the objects name

Source: contacts.js, line 114

`startDragging(index, element_taskID)`

When an Html element is dragged this function adds a class to it, to rotate the element about 5 degrees clockwise. It then saves the taskID and the index of the task object in the allTasksFromStorage[] array.

Parameters:

Name	Type	Description
`index`	number	Position of the task object in the array allTasksFromStorage[].
`element_taskID`	string	TaskID of the current task object.

Source: board.js, line 159

`(async) submitTask(taskStatus, submitTaskID`*opt*`) → {Promise.<void>}`

Submits a task with the given task status and optional task ID.

Parameters:

Name	Type	Attributes	Default	Description
`taskStatus`	string			The status of the task.
`submitTaskID`	number	<optional>	0	The ID of the task to be submitted (optional).

Source: task_add.js, line 57

Returns:

- A promise that resolves when the task is submitted.

Type
Promise.<void>

`taskCardBoard_HTML(task, elementId)`

HTML Template to render a single taskcard with more data of the task object, loaded on the board.html

Parameters:

Name	Type	Description
`task`	object	The task object containing the task data.
`elementId`	string	The ID of the selected Element to which the task card will be attached.

Source: board_HTML_Templates.js, line 6

Returns:

- Returns the Html code for the more detailed Card of the Task.

`taskFormClear()`

Clears the task form by removing assigned users and subtasks.

Source: task_add.js, line 314

`thumbnailCard_HTML(index, element)`

HTML Template to render a single preview taskcard with less data of the task object, loaded on the board.html

Parameters:

Name	Type	Description
`index`	number	Position of the current selected task object in the allTasksFromStorage[] Array.
`element`	object	The current selected task object containing the task data.

Source: board_HTML_Templates.js, line 61

Returns:

- Returns the Html code for the Thumbnail card of the Task.

`toggleCategoriesSelect()`

Toggles the visibility of the categories select dropdown and renders the categories list.

Source: task_add.js, line 200

`toggleCustomSelect()`

Toggles the visibility of a custom select dropdown.

Source: task_users.js, line 106

`toggleInputUsers()`

Removes the visibility of the input users dropdown.

Source: task_users.js, line 116

`updateSavedSubtask(subtaskText, subtaskId)`

Updates the saved subtask with the provided text.

Parameters:

Name	Type	Description
`subtaskText`	string	The updated text for the subtask.
`subtaskId`	string	The ID of the subtask.

Source: task_subtasks.js, line 126

`userDropDownClick(userId)`

Handles the click event on the user dropdown.

Parameters:

Name	Type	Description
`userId`	string	The ID of the user.

Source: task_users.js, line 38
