FAVOR

System Specification

Philipp Auinger, Alexander Walter
HTBLA Leonding

Document Information

Project Name	FAVOR
Project Manager	Philipp Auinger, Alexander Walter
Document Owner	Philipp Auinger, Alexander Walter
Created on	21/12/2017 10:11
Last modified	04/02/2018 17:27
State	[ ] in work [X] submitted [ ] released
Dokumentablage	SystemSpecificationFavor.doc

Document History

Revision Nr.	Date	Version	Chapter	Modification	Author
1	21.12.2017	0.1	All	Improved and organized the given template and made it more useable	Philipp
2	05.01.2018	0.2	2. | 2.1. | 2.2.	Working on the use-case descriptions	Philipp
3	07.01.2018	0.3	2.1 | 2.1.1 | …	Finishing the use case details 1-8	Philipp
4	25.01.2018	0.4	3	Working on the Non-Functional-Requirements	Alex
5	25.01.2018	0.5	1.1.4.2 | 3 | 4	Business processes, chapter 3 and chapter 4 finished	Philipp
6	27.01.2018	0.6	1	Working on the chapter 1	Philipp
7	28.01.2018	0.7	1	Finishing chapter 1 and the sub headings.	Philipp
8	03.02.2018	0.8	2	Adding Activity Diagrams and pictures of the GUI.	Philipp
9	04.02.2018	0.9	2	Improved some sentences and added more pictures of the GUI.	Philipp
10	04.02.2018	1	All	Proofread the whole document.	Philipp

Content

1 Initial Situation and Goal

1.1 Initial Situation

Every family got their own routine and special repetitive tasks they all need to fulfil, remembering these task is really hard, often they need to write it down somewhere and these little pieces of paper can get lost or misunderstood by other family members.

There are many applications on the market which are pretty good for organizing a family. But these often got a lack of the ability that everybody, from young to old, can use them without any barriers or problems, and often also the younger parts of the family need some motivation, an online currency they can work for and use it to make some dreams come true, would be great.

That means that our goal is to make it possible that anybody that is part of the family can use our application, and the users just need a little bit of technical understanding to organize their family perfectly.

1.1.1 Application Domain

Every family member, divided into children or parents is allowed to create tasks which should be fulfilled by other members of the group. These tasks will be stored in lists, which are personalized for each member because there can also be tasks just for one specific member.

Another list stored in the family group is the reward list, this list goes hand in hand with the wish list of each person. This list will be used for storing the individual wishes, these wishes can be converted into rewards, by assigning a certain price of coins to it and then adding it to the reward list of the group.

These lists need to get synchronized automatically in case something gets edited every member needs to stay up to date. Therefore, it is important that every person got its personalized view on each list and the whole family group, and these different user information needs to be secured and it should be also possible locking it with a password to deny other members peeking into other accounts.

1.1.2 Glossary

1.1.2.1 List

This application stores three different types of lists, the wish list which is stored for every individual person, the reward list and the task list those are accessible for every member.

1.1.2.2 Synchronization

Synchronisation is important to provide all the members of the family–group the same information even if it got just changed. The database will help with this functionality by providing the updated data.

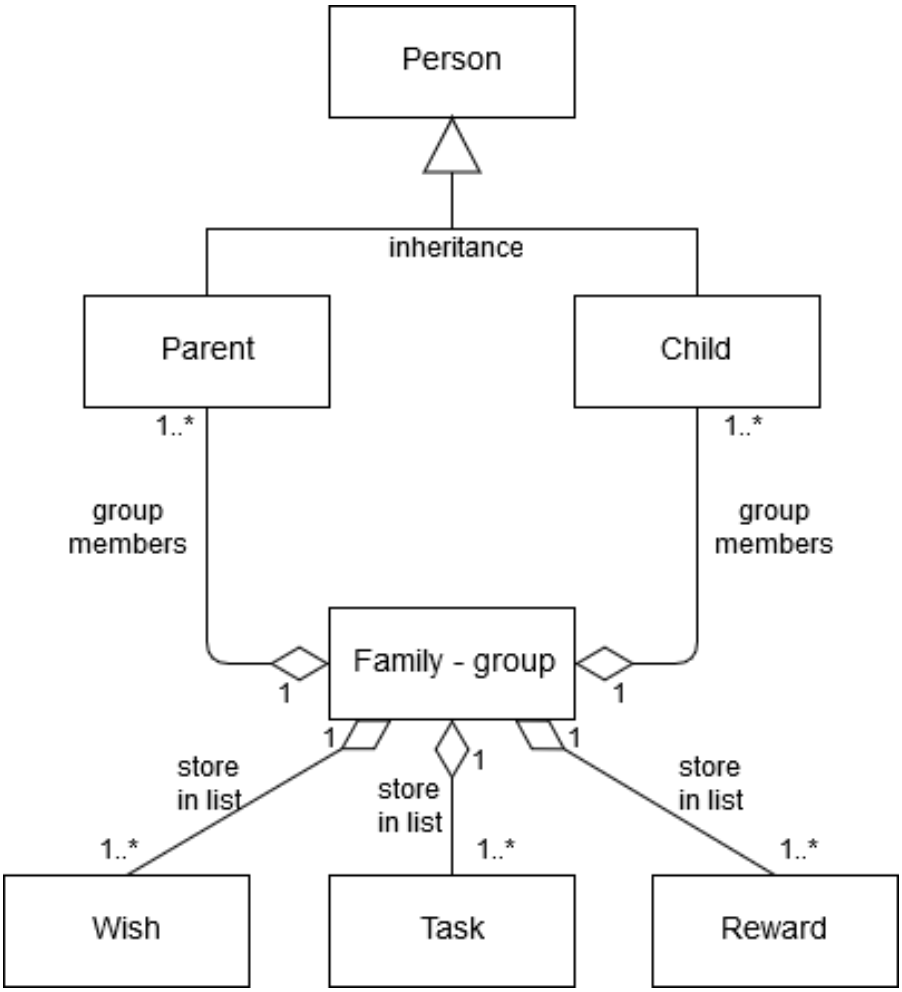
1.1.2.3 Password

The password that should secure the different member-information inside a family-group is planned to be something like a four-number pin which should be saved for every individual person.

1.1.3 Model of the Application Domain

The following low-detailed class diagram should show the basic relations between the different objects involved in the project. It needs to be considered that the whole functionalities and lists are implemented in the family-group but because we are not sure how these should be implemented.

Therefore, we will remain with this low-detailed class diagram.



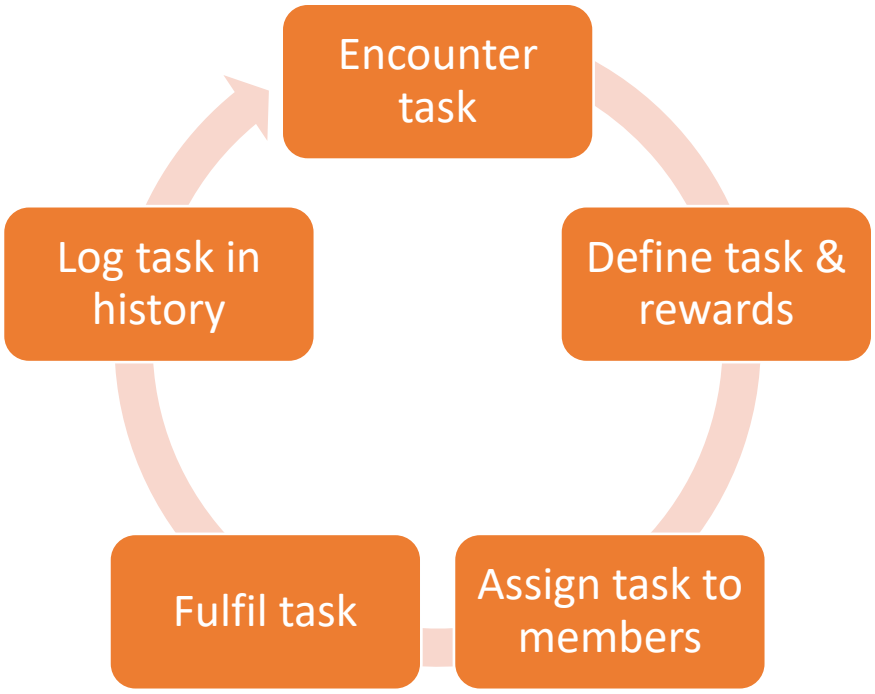
From	To	Description
Person	Person / Child	Inheritance of a basic person to a parent or child which got assigned the role.
Parent	Family - group	At least 1 parent needs to be part of the family group if there is also at least 1 child, if not there needs to be at least 2 parents.
Child	Family - group	At least 1 child needs to be part of the family group if there is also at least 1 parent.
Family - group	Task	The group manages and stores a list of different tasks.

1.1.4 Business Processes

1.1.4.1 Overview

There is just one main business process which should be fulfilled by this software, the following business process describes the routine.

1.1.4.2 Main Business Process: Routine



1. Encounter task
   A person bumps into a task that he wants to be fulfilled by other members of the family.

2. Define task & rewards
   The person thinks about what have to be done exactly and even what rewards he can offer the members which fulfilled the task.

3. Assign task to members
   The person describes his/her task and assigns it to other members of the family-group.

4. Fulfil task
   The members are working on the task until it gets fulfilled and also gets approved by the “creator” of the task.

5. Log task in history
   When the task got fulfilled by all members which got assigned to it, it will be theoretically saved into the memory of the different persons.

1.2 Goal Definition

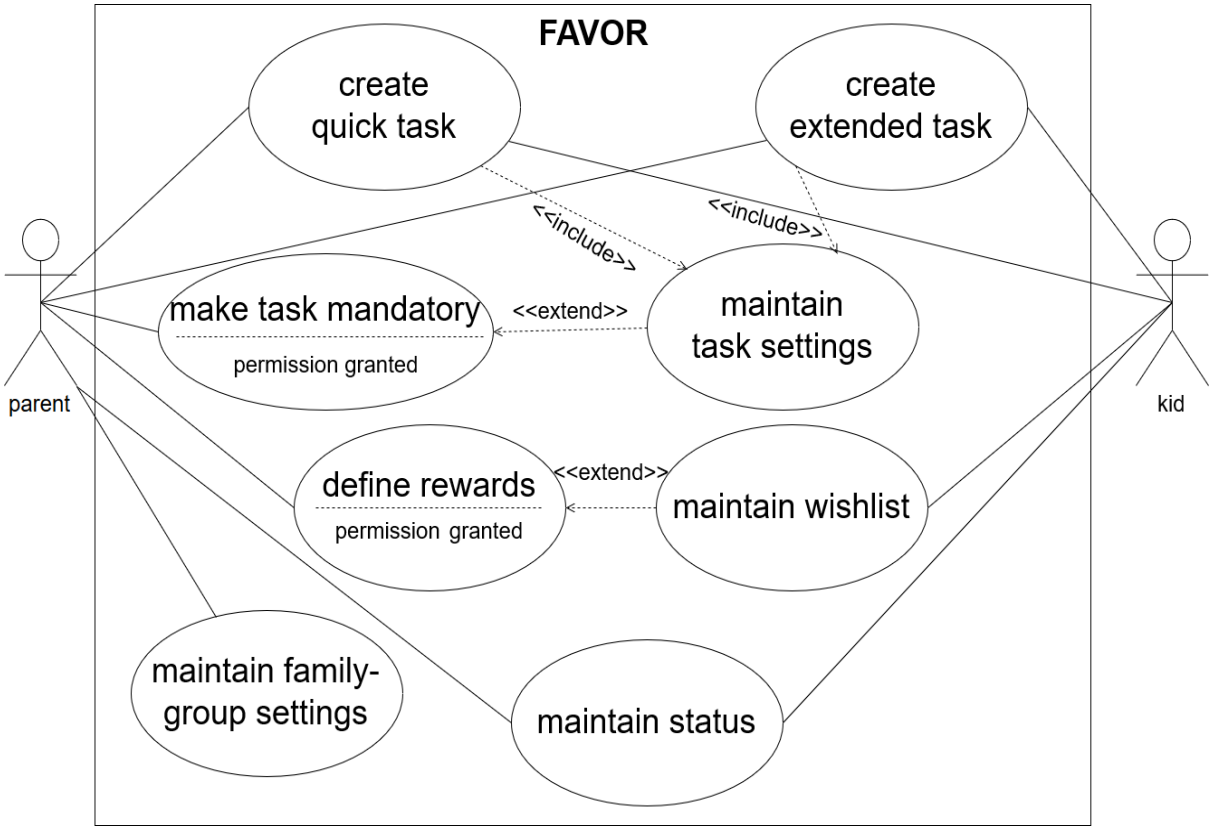
Our goal and project idea is to help families get some things done, giving them a good overview what they have to do for each other and motivate them a little bit.

To successfully finish the project the whole application needs to be solid and easy to understand for young and old people. Therefore, we need to add a little tutorial, descriptions and a help menu.

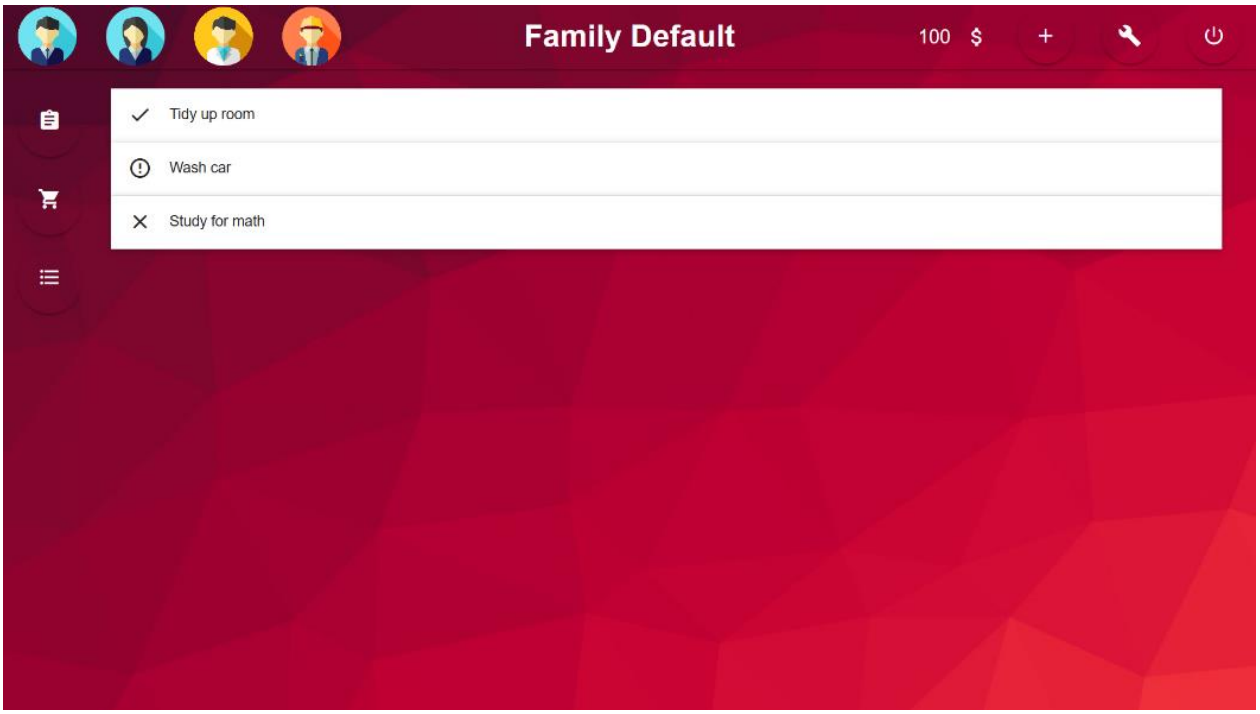
The final application will be a web–application/website, because a good optimized and clearly structured website could be used on every device, anywhere.

2 Functional Requirements

2.1 Use Case Diagram

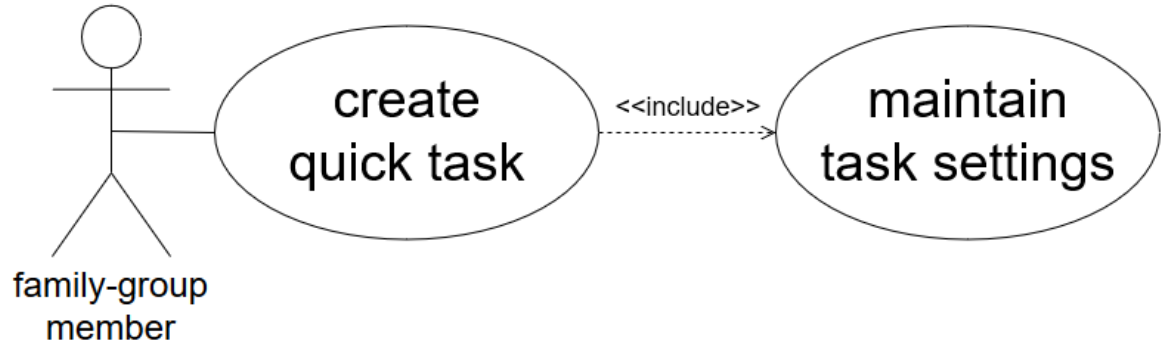


2.1.1 General GUI to call the use cases



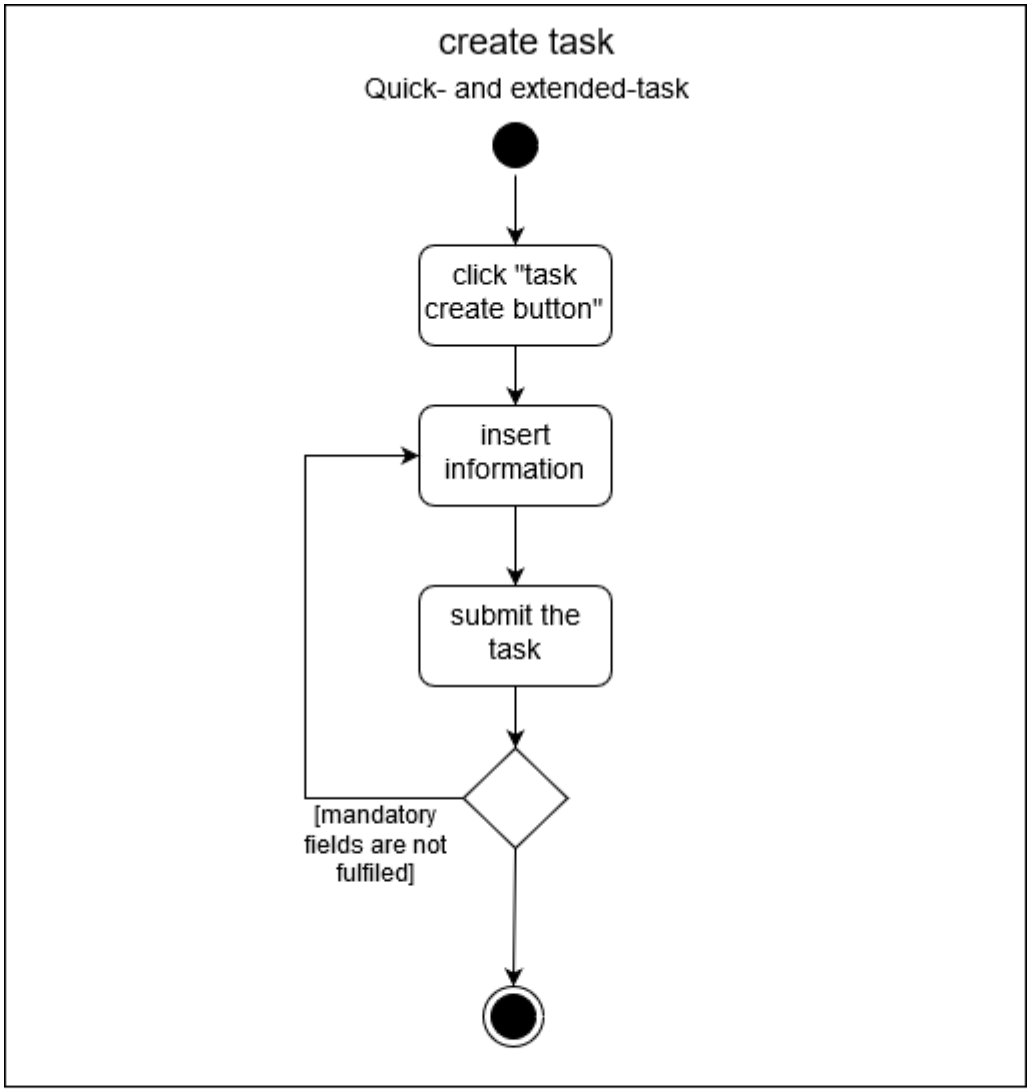
Different Menus can be distinguished by tags which describe the buttons.

2.1.2 Use case Details 1: Create quick task



The main function of this use case is to create a task with less different setting possibilities, the member should be able to create a task with the least effort.

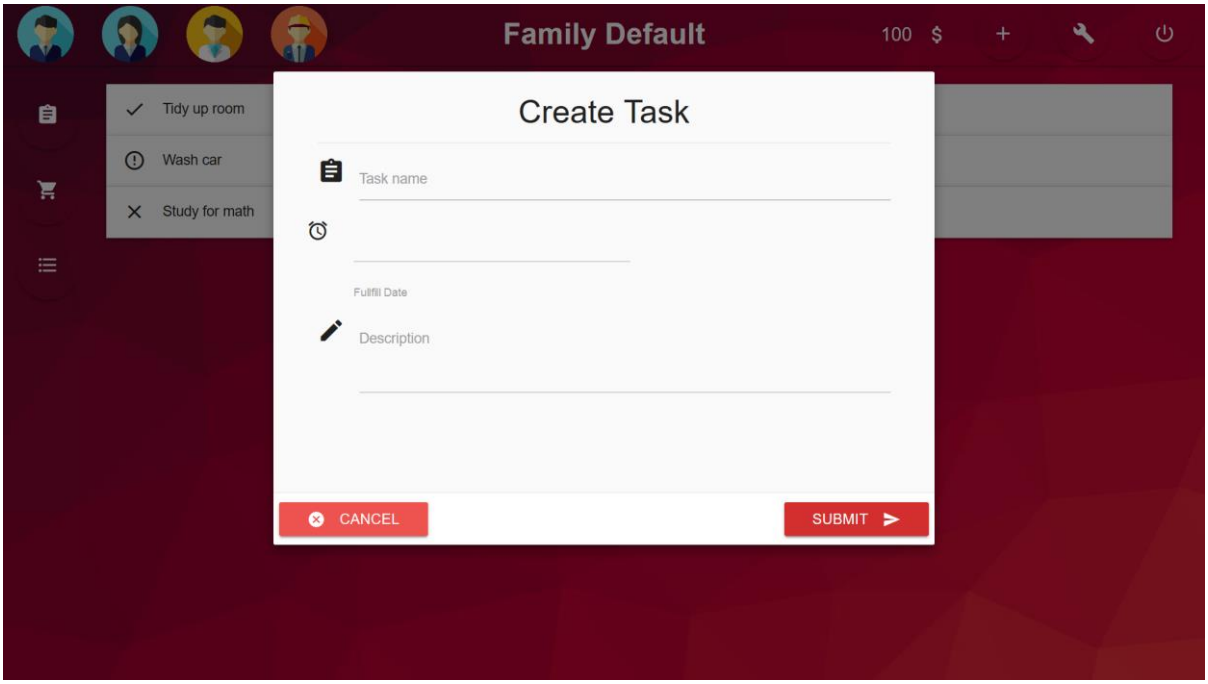
See “Project Proposal” for an overview of the different setting possibilities and which fields are mandatory.



2.1.2.1 Characteristic Information

Goal:	The family-group member can create a task in the fastest possible way and less setting possibilities.
Precondition:	The user is logged in.
Postcondition:	The member created a task.
Involved User:	All family-group members.
Triggering Event:	A member wants to create a task by clicking a button.

2.1.2.2 GUI to call the use case



GUI Description: **First Draft** of the Creation Menu

2.1.2.3 Scenario for the standard use

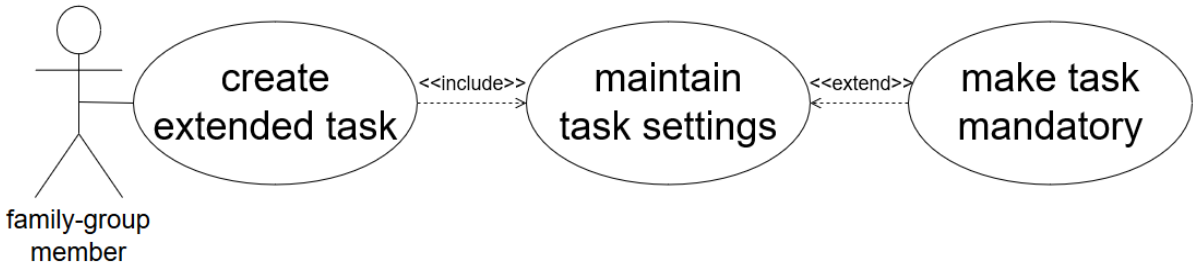
Step	User	Activity
1	Family-group member	Open the creation menu of a quick task
2	Family-group member	Insert information and adjust the task
3	Family-group member	Finish to create the task
4	Application	Add the task to the task list

2.1.2.4 Open Points

2.1.2.4.1 Limit

- Limited number of created tasks
- Limited number of accepted tasks in a certain time period.

2.1.3 Use case Details 2: Create extended task



The main function of this use case is also to create a task, but with the chance to use all setting possibilities.

The user is able to create an extended task by “extending” the create-menu of the quick task.
See: Use case Details 1: Create quick task (Activity Diagram)

2.1.3.1 Characteristic Information

Goal:	The family-group member can create a task with more different setting possibilities.
Precondition:	The user is logged in.
Postcondition:	The member created a task.
Involved User:	All family-group members.
Triggering Event:	A member wants to create a task by clicking a button.

2.1.3.2 GUI to call the use case

See: GUI to call the use case (Quick-task)

2.1.3.3 Scenario for the standard use

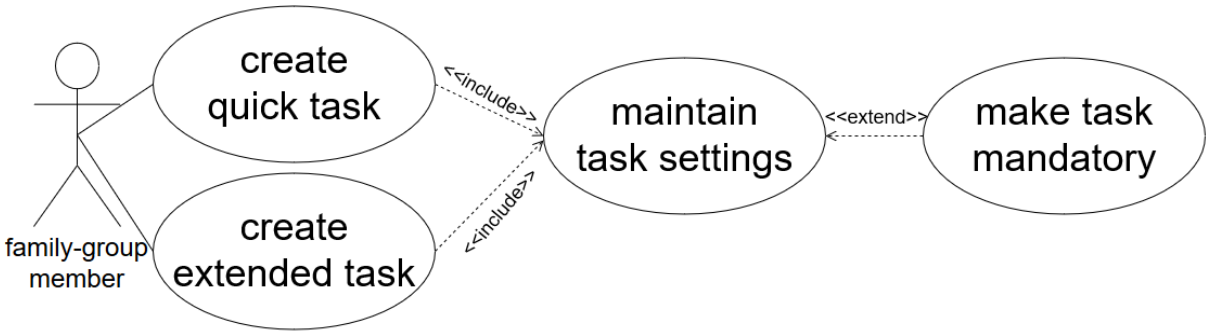
Step	User	Activity
1	Family-group member	Open the creation menu of a quick task
2	Family-group member	Extend the creation menu
3	Family-group member	Insert information and adjust the task
4	Family-group member	Finish to create the task
5	Application	Add the task to the task list

2.1.3.4 Open Points

2.1.3.4.1 Limit

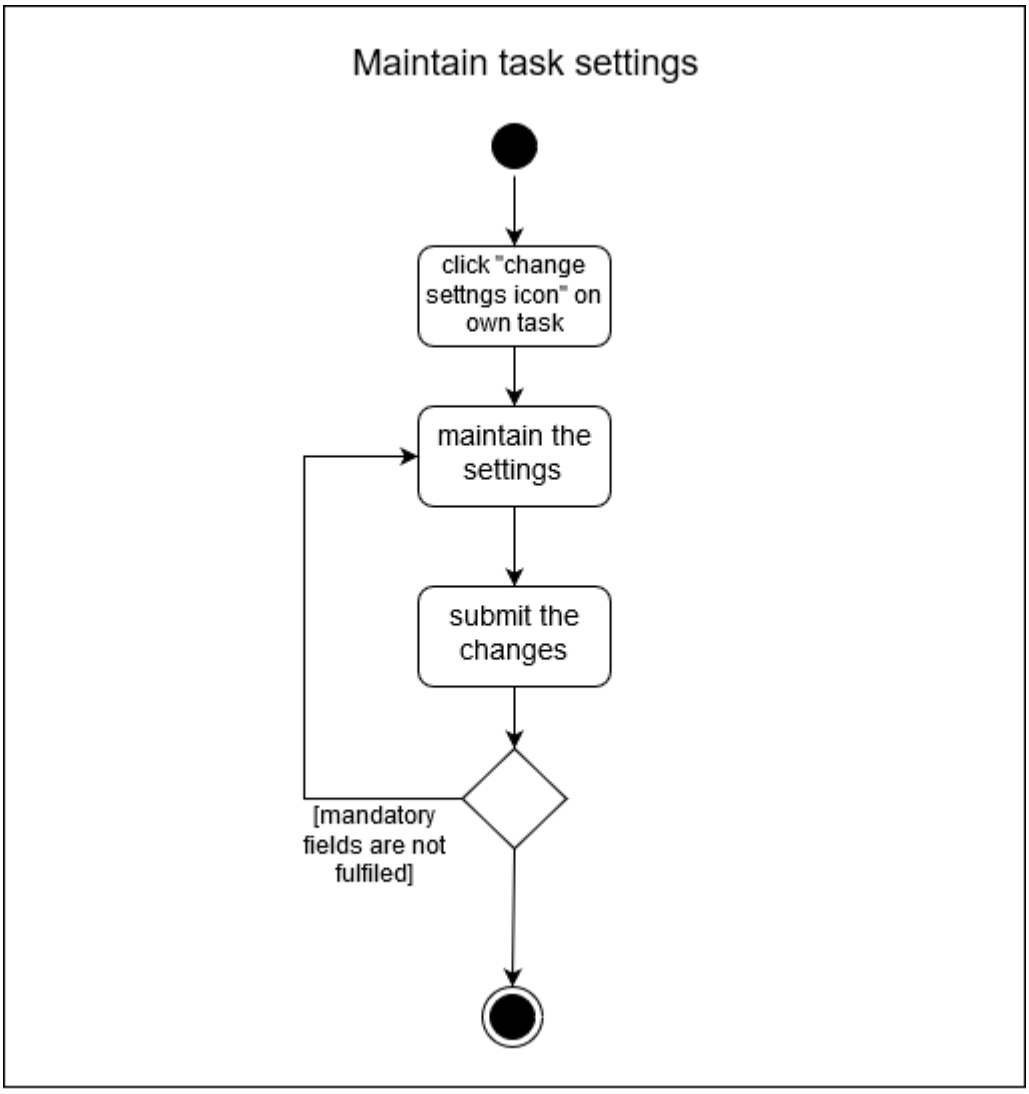
- Limited number of created tasks
- Limited number of accepted tasks in a certain time period.

2.1.4 Use case Details 3: Maintain task settings



The main function of this use case is that the family-group members can maintain the settings of an already created task or maintain the settings in the process of the creation of the task.

This use case also includes the process of deleting the task.



2.1.4.1 Characteristic Information

Goal:	The family-group member can maintain the settings of a task he created.
Precondition:	The user is logged in and he already created a task.
Postcondition:	The member changed the settings of a task.
Involved User:	All family-group members that created a task.
Triggering Event:	A member wants to maintain the settings of a task by clicking a button.

2.1.4.2 Scenario for the standard use

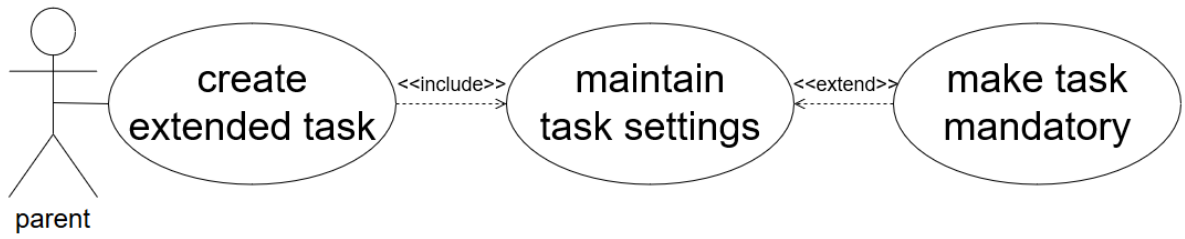
Step	User	Activity
1	Family-group member	Open the task list
2	Family-group member	Open the task menu
3	Family-group member	Maintain the task
4	Family-group member	Finish to maintain the task
5	Application	Save the changes to the task

2.1.4.3 Open Points

2.1.4.3.1 Restrictions

- Which settings can get changed afterwards?

2.1.5 Use case Details 4: Make task mandatory



The main function of this use case is that the parents of the family-group can make their created tasks, mandatory for their children.

This setting is possible in the extended task-create-menu.
See: Use case Details 2: Create extended task

A mandatory task needs to be fullfilled in the given timespan, if not, there will be penalties.

2.1.5.1 Characteristic Information

Goal:	The parents of the family-group can make tasks mandatory for their children.
Precondition:	The user is logged in and a parent of the family-group.
Postcondition:	The parent made a task mandatory.
Involved User:	Parents
Triggering Event:	A parent wants to make a task mandatory by checking a checkbox in the process of creating or maintaining a task.

2.1.5.2 GUI to call the use case

Checkbox which is part of the “Create Extended Task” environment.

2.1.5.3 Scenario for the standard use

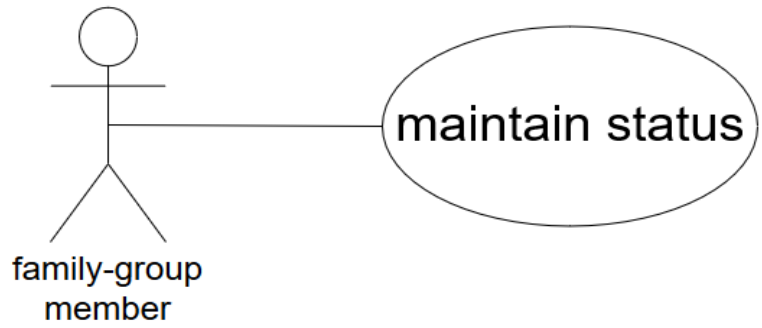
Step	User	Activity
1	Parent	Open the creation menu of a quick task
2	Parent	Extend the creation menu
3	Parent	Adjust the task and make it mandatory
4	Parent	Finish to create the task
5	Application	Add the task to the task list

2.1.5.4 Open Points

2.1.5.4.1 Penalties

- Algorithm for calculating the penalty

2.1.6 Use case Details 5: Maintain status



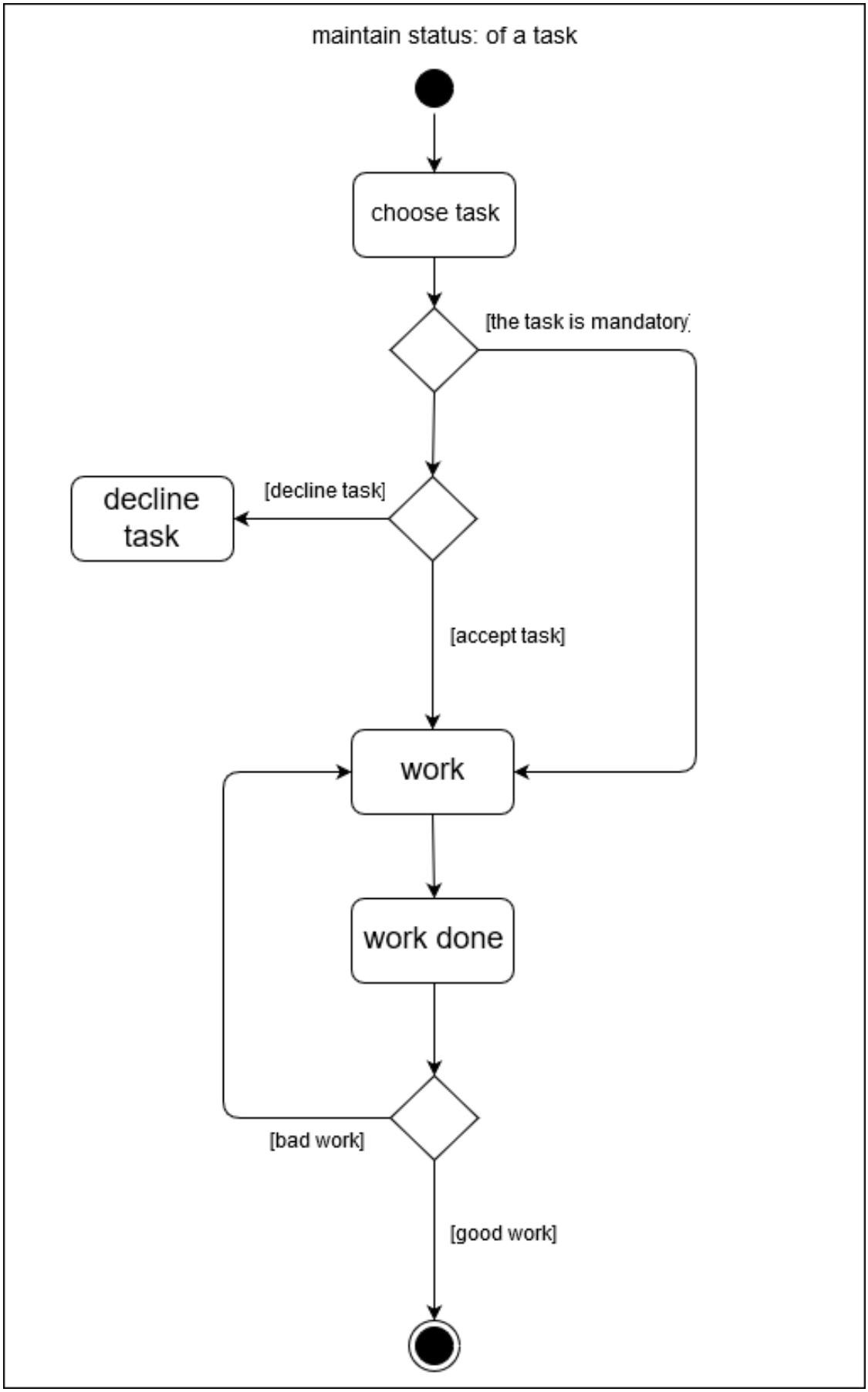
The main function of this use case is that all family-group members can maintain the status of a task they need to fulfil or of a task they created.

2.1.6.1 Maintain Status: Own task

The creator of a task can see in which status the different family members are in relation to his task.

He got the power to throwback a member if he/she fulfilled the task not good enough, back to the “working” status.

2.1.6.2 Maintain status: Not own task



2.1.6.3 Characteristic Information

Goal:	The family-group member can maintain the status of a task he needs to fulfill or of tasks he created.
Precondition:	The user is logged in and either got a task assigned to him or created a task.
Postcondition:	The member maintained the status of a task.
Involved User:	All family-group members.
Triggering Event:	A member wants to maintain the status of a task.

2.1.6.4 GUI to call the use case

GUI First Draft: Green and Red button for accepting or declining a task.

2.1.6.5 Scenario for the standard use

2.1.6.5.1 Scenario: Own task

Step	User	Activity
1	Family-group member	Open the task list
2	Family-group member	Open the task info
3	Family-group member	Check and judge the work
4	Family-group member	Choose good or bad work
5	Family-group member	Approve the changes
6	Application	Save the changes

2.1.6.5.2 Not own task

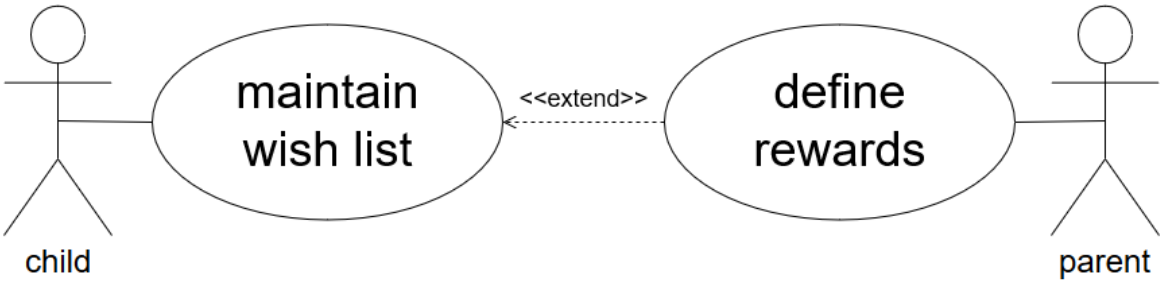
See: Maintain status: Not own task (This chart shows the Scenario for the standard use.)

2.1.6.6 Open Points

2.1.6.6.1 Coin penalties

- Less coins for every negative judgment (bad work)

2.1.7 Use case Details 6: Maintain wish list

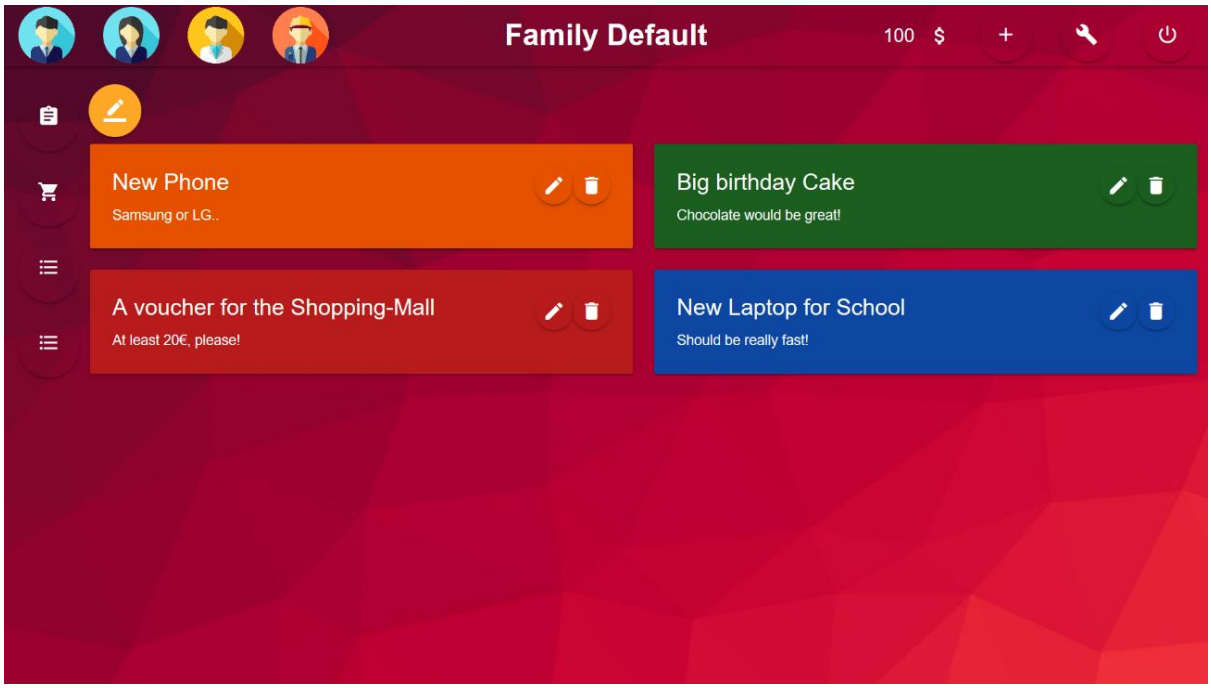


The main function of this use case is that the children of the family-group can maintain their wish list, by adding or deleting elements.

2.1.7.1 Characteristic Information

Goal:	The children of the family-group can maintain their wish lists.
Precondition:	The user is logged in and a child of the family-group.
Postcondition:	The child maintained its wish list.
Involved User:	Children
Triggering Event:	A child wants to maintain its wish list by clicking a button in order to add or delete elements to the wish list.

2.1.7.2 GUI to call the use case



GUI in the eyes of the children when they visit their wish list.

2.1.7.3 Scenario for the standard use

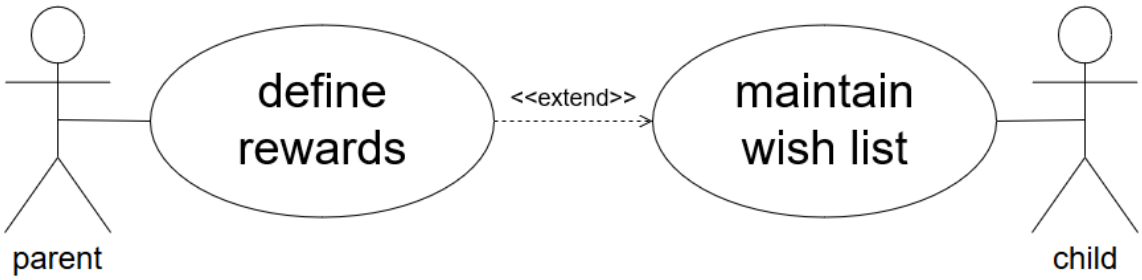
Step	User	Activity
1	Child	Open the wish list
2	Child	Maintain wish list
3	Child	Finish maintaining the wish list
4	Application	Save wishes

2.1.7.4 Open Points

2.1.7.4.1 Ad system

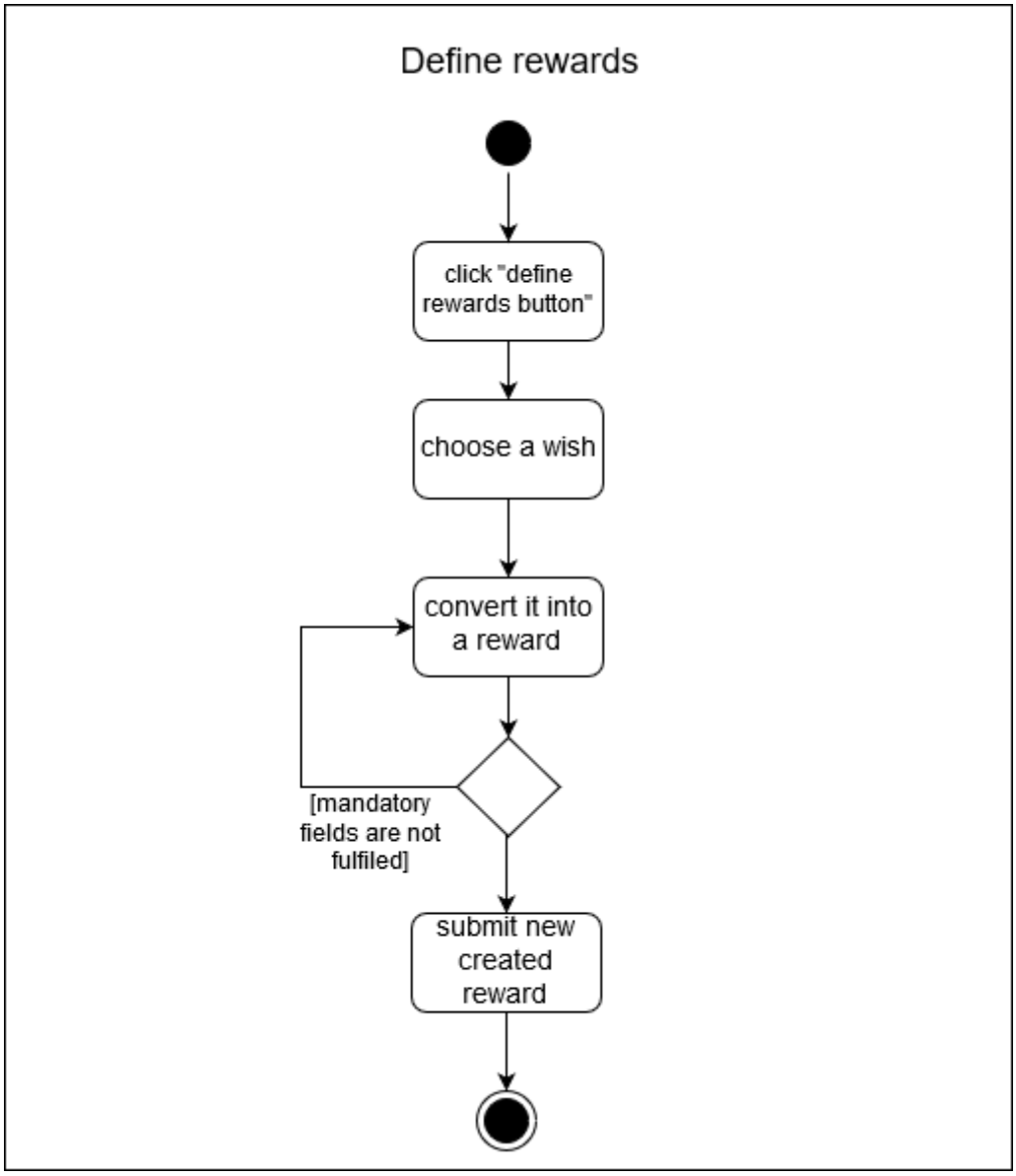
- Placing ads which help to maintain the wish list.
- Product recommendations

2.1.8 Use case Details 7: Define rewards



The main function of this use case is that the parents of the family-group can use the wish list of their children for defining rewards.

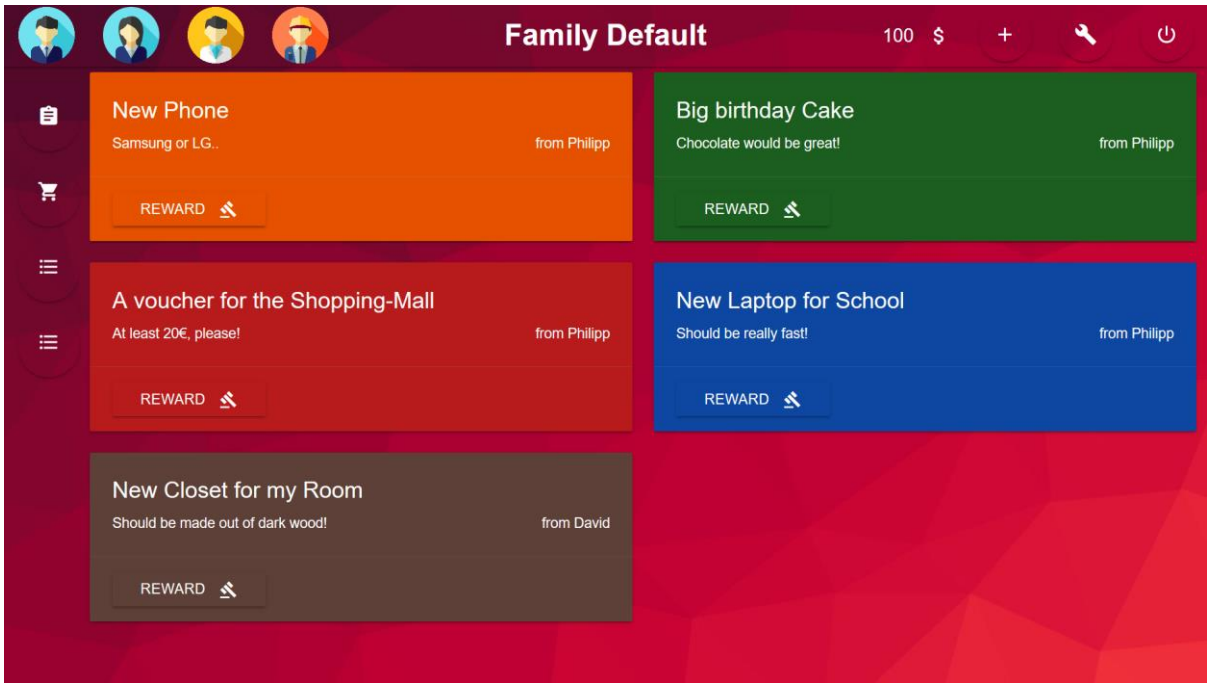
The parent chooses an element of the list, and adding a certain price to it, so the children can spend their coins on it.



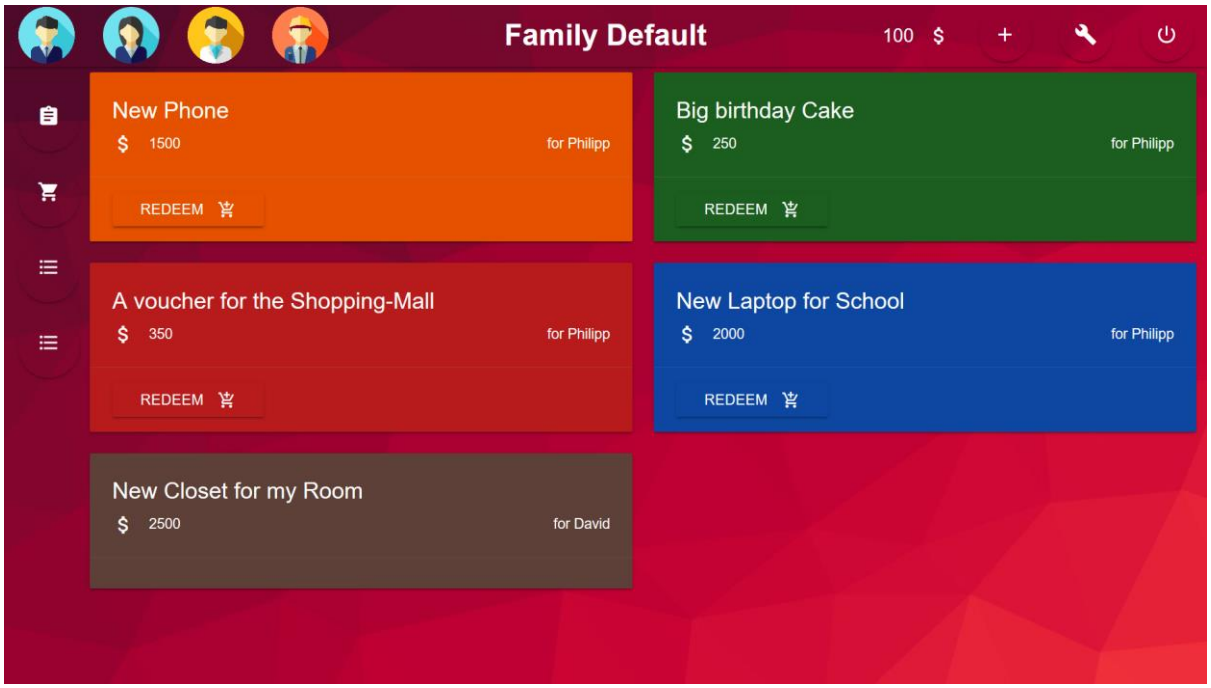
2.1.8.1 Characteristic Information

Goal:	The parents of a family-group can define rewards using the wishes written down in the wish lists.
Precondition:	The user is logged in and a parent of the family-group.
Postcondition:	The parent defined a reward.
Involved User:	Parents
Triggering Event:	A parent wants to define a reward by converting a wish into a reward by adding a “price” to it.

2.1.8.2 GUI to call the use case



GUI in the eyes of the parents, where they can transform wishes into rewards.



GUI in the eyes of the children, every child can redeem their wishes if they got transformed into a reward.

2.1.8.3 Scenarios for the standard use (good case)

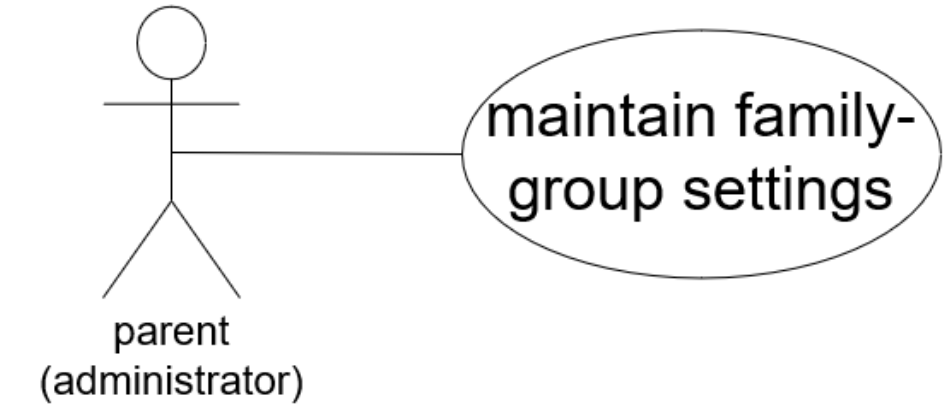
Step	User	Activity
1	Parent	Open the reward menu
2	Parent	Define a reward
3	Parent	Finish defining rewards
4	Application	Save the reward

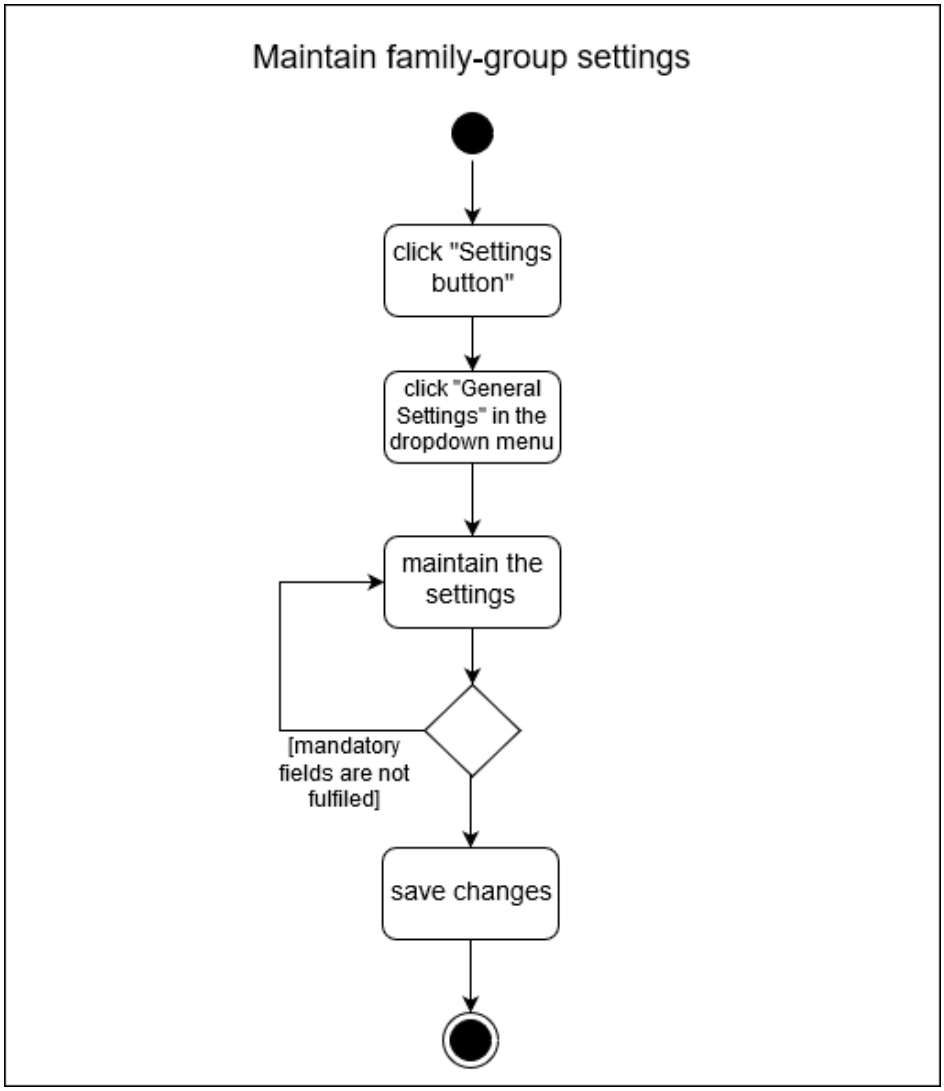
2.1.8.4 Open Points

2.1.8.4.1 Ad system

- Online shop recommendations
- Where to buy the products that are written down on the wish list?
- Price comparison website ads

2.1.9 Use case Details 8: Maintain family-group settings





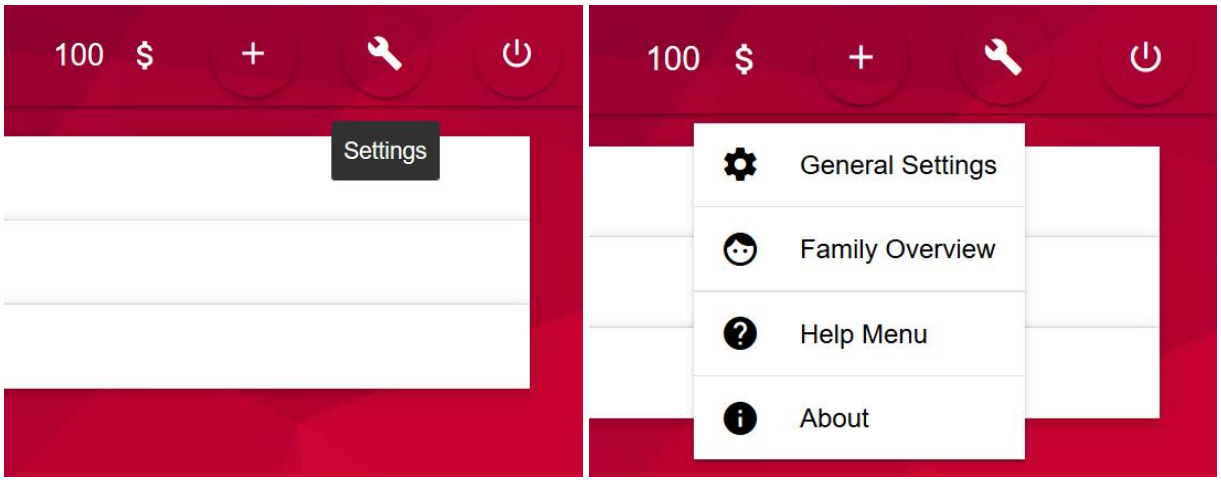
The main function of this use case is that the parents of the family-group can maintain the family-group settings.

To get a overview of the different setting possibilities, see: “Project Proposal”.

2.1.9.1 Characteristic Information

Goal:	The administrator of the family group can maintain the family-group settings.
Precondition:	The user is logged in and either he is a parent or the administrator status got assigned to him.
Postcondition:	The administrator maintained the family-group settings.
Involved User:	Administrator (default: parents)
Triggering Event:	An administrator wants to maintain family-group settings by opening the setting menu.

2.1.9.2 GUI to call the use case



Dropdown menu, each Element will open a site where different things can be adjusted.

2.1.9.3 Scenario for the standard use

Step	User	Activity
1	Parent	Open the family-group setting menu
2	Parent	Maintain the settings
3	Parent	Finish maintaining the settings
4	Application	Save the settings

2.1.9.4 Open Points

2.1.9.4.1 Coins

- Different coin pictures

2.1.9.4.2 Themes

- Different Background images
- Colour settings

3 Non-functional-Requirements

3.1 Type USERE: Usability requirement

To efficiently use “FAVOR” it has to fulfil some criteria like:

- The website should load quickly, so a good server is more or less necessary.
- The main criteria is that it should be nice to look at and easy to use.
- The website should be child—friendly because families are our main target.
- The website should be easy to access so you can take a quick look at your unfulfilled task while in a hurry.

3.2 Type EFFICRE: Efficiency requirements

We cannot affect the download speed of the user but we can write the code as efficient as possible to reduce loading times.

3.3 Type SECRE: Security requirements

Protecting the data of our customers is the most important thing for us, families should not fear of getting spam mails sent to their email addresses or getting their family photo leaked.
Also, in-app safety is important, like in the family, because maybe Mom, Dad and your sister want to plan a surprise birthday party for you and need to organise some materials, and you should not see it so they hide it from your FAVOR-account.

3.4 Type LEGALRE: Legal requirements

Just like in SECRE described, we want and must protect the data of our users.

4 Quantity Structure

The data of a whole family-group, the different members and the login data needs to be saved on a database.

4.1 Data structure

4.1.1 Member

Data ID	Description	Size
PDATA	Personal data (e.g.: Name, Birthdate, Money, …)	5KB
PPIC	Profile picture (max. 500x500)	10 - 100KB

Wish-lists will be stored for each member.

4.1.2 Family-group

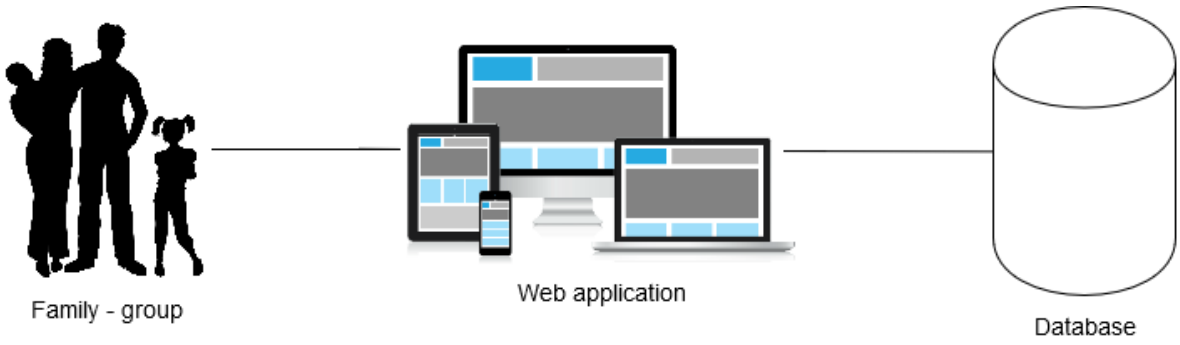
Data ID	Description	Size
GDATA	Group data (e.g.: Login, Members, Pictures …)	100KB
TDATA	Task data (different tasks and lists)	50KB

Reward-lists will be stored for the whole family – group.

4.2 Conclusion

We are not sure about the exact disk space we need for a specific number of users, but in conclusion there is a clear structure how the different data will be stored.

5 System Architecture and Interfaces



The members of a family group can get information from the database through the web application which is optimized for desktop and mobile devices.

One the database are all personal data saved, for example the different lists, settings and pictures.

See: Data structure

6 Acceptance Criteria

6.1 Use case 1: Create quick task

Test step	Expected Behaviour
Click the round shaped “plus button”.	A menu opens where the information can be tipped in, if every mandatory field is fulfilled, a new task which contains the information will appear in the list after pressing the “submit button”.
Click the round shaped plus button but don’t fulfil every mandatory field	The not fulfilled mandatory fields will be marked with a red colour and the “submit/create button” is disabled. It will be enabled after all fields are fulfilled.

6.2 Use case 2: Create extended task

Test step	Expected Behaviour
Click the “expand button” in the menu described in “Use case 1: Create quick task”	The menu will expand and new input fields will appear, after filling in information and enabling settings the task can be submitted and will appear in the list.
Check the checkbox in the expanded menu to set the priority to high and submit the task.	The task will appear on the top of the list and will be marked as an “High priority task”. (Colour and Icon)

6.3 Use case 3: Maintain task settings

Test step	Expected Behaviour
Click on a task which got created by the tester, and press the “change task settings button”.	The task create menu will appear but every information that got tipped in, in the process of creating the task will be written down in the input fields.
Uncheck the mandatory checkbox in the change menu and submit the task.	The other family-group members are now able to decline the task.
Change the allocation input field to “all” and submit the task.	All the other family-group members can now procced with the task.

6.4 Use case 4: Make task mandatory

Test step	Expected Behaviour
Check the mandatory checkbox in the, expanded create or change menu and submit the task.	The other family-group member must procced with the created task and can’t decline it.

6.5 Use case 5: Maintain wish list

Test step	Expected Behaviour
Open the wish list tab on the left navigation-bar and press on the “pencil button”.	For every wish there will appear a dustbin-icon and a pencil-icon, clicking the first one will delete the wish, and clicking the pencil-icon will open a menu where the wish can be maintained.
Pressing the “underlined pencil button” in the wish list tab.	A menu will open where a wish can be defined, after inserting the mandatory information it will be possible to submit the task. After submitting it, it will appear in the list.

6.6 Use case 6: Define rewards

Test step	Expected Behaviour
Click on one of the wishes and press on the “define reward button”.	A menu will open and the user needs to fulfil the mandatory field which is the “price field”. If the field is fulfilled the task can be submitted and will appear in the reward-list.
Click on a reward in the reward list and press the dustbin-icon.	The reward will disappear from the reward list and won’t be accessable anymore.

6.7 Use Case 7: Maintain family-group settings

Test step	Expected Behaviour
Click on the “wrench button” in the top right corner.	A dropdown menu will open which holds four different tabs.
Choose “General Settings” in the dropdown menu.	A menu will open and it will be possible to adjust the family-group-settings, if every mandatory field will be fulfilled after the maintenance the changes can be saved.

7 List of Abbreviations

Abbreviation	Description
USERE	Usability requirement
EFFICRE	Efficiency requirements
SECRE	Security requirements
LEGALRE	Legal requirements
PDATA	Personal Data
PPIC	Profile Picture
GDATA	Group Data
TDATA	Task Data