Functional Analysis

Functional Analysis Mobile Clients

Introduction 2

Concepts 2

Navigation 2

DashBoard 2

Punch-Bar 3

Widget 3

Action Dialog 4

Notifications 4

Time Registration 4

Backup/Restore 4

Modules 4

Projects and Tasks 4

Table 4

Views 5

Project Deletion 8

Task Deletion 8

Time Registrations 9

Table 9

Views 9

Reporting 12

Preferences 12

Others 15

About Screen 15

Database 15

Database Versioning 16

Preferences 16

Out of Scope 16

Android 16

iOS 16

Projects List 16

Project Details 16

Preferences 16

Functional Analysis – Mobile Clients

# Introduction

The main goal of the application is to keep a list of time registrations, where a time registration has a start and end time. A registration also needs to be linked to a project and a task. This way users of the application can easily track their time spent on one specific task and generate reports out of this data.  
The general idea through the entire application is that only one time registration can be going on at a time.

# Concepts

## Navigation

Navigating through the different screens of the application will be done according to the platform specific guidelines.

Android: On Android we have a top-bar (Action Bar) that includes navigation on the left (with the application icon that is navigable), in the center of the bar the name of the currently active screen/window and the right side is filled with action that are possible on that screen. The overflow of actions that cannot be placed in the action bar will go into an overflow menu.

iOS: On iOS we have a top-bar (Navigation Bar) in which on the left-hand side we will have the back button to the previous screen (with the name of that screen), in the center the name of the currently visible/active screen and on the right the one button can be placed for an action (edit on a list for example)

## DashBoard

The dashboard is the main entrance point of the application. Whenever launching the application the dashboard will be visible. From the dashboard the user can navigate to the different modules available in the application.

On the dashboard there will be a punch-bar available. The punch-bar is explained in the next point.

## Punch-Bar

The punch-bar of the application is a bar at that will be placed at the bottom of the screen and that allows the user to quickly start or stop a time registration for a certain project and task. If a time registration is ongoing it will also display this in the bar together with some details for which project and task it is. This punch-bar will always be visible on the dashboard.

When starting a new time registration from the punch-bar a popup will ask the user to choose a project, when done it will pop up a second dialog to ask for a task for the project. In case that only one task is available for the project and the user has set the preference ‘Ask for a task if only one task is available’ to true the task-chooser popup will be skipped. If both have been selected (or in case no task needs to be selected and the project is selected correctly) a new time registration will be created an persisted in the local db. During the creation of a new time registration the preference ‘Auto close 60s gap’ should be taken into account. If it is set to true then the current time and the end time of the last time registration needs to be compared in seconds. If the difference is less than 60s than the new time registration needs to reuse the end-date of the last time registration.

When ending a time registration the current time should be put in the end time of the last (and only one ongoing) time registration. This should be updated in the local db. But if the setting ‘Enter comment on punch-out’ has been set to true the user should be able to first enter a comment that will also be persisted in the local db.  
After updating the time registration the user should be asked if he wants to mark the task as *finished* if and only if he has enabled the setting ‘Ask to finish task’.

Android: On the Android OS the punch-bar can be configured to be visible. The user will be able to choose if the punch-bar will be visible in the dashboard and if it should be visible in some other screens than the dashboard (such as in the time registration list, in the project and tasks list and the time registration details). This is configurable through the preferences. The user thus has the option to entirely disable the punch-bar as he can use the widget (explained in the next part) to quickly start and stop his time registration.

iOS: Because iOS doesn’t know the concept of widgets it will only be possible to start a time registration from within the application and thus the punch-bar will always be there in the dashboard. It cannot be turned off except for the time registration list and the project list (by default it’s off for these views).

## Widget

The concept of widgets is only known on Android, so this part can be skipped for iOS.  
The widgets allow users to quickly start and stop the a time registration for a certain project and task.

Different widgets are available:

1. A general widget: this widget has a header and a “punch in” or “punch out” button.   
   The header is used to select a project when no time registration is ongoing. If a time registration is ongoing the header will launch the application.  
   The punch-in button is used to start a new time registration based on the selected project. First the user will have to choose a task for the selected project and then the time registration will be started for that task and project.
2. A project/task widget: when adding this widget to your home-screen you can choose a project and if you want also a task.  
   If only a project is selected we are talking about a ‘project-widget’ and when the user now presses the punch-in button he will have to choose a task before starting his time registration.  
   If a project and task where selected the punch-in button will immediately start a new time registration for the selected project and task.

It is possible to have multiple widgets of different types. You can for example have 3 general widgets (there’s not really any use case why you should have different general widgets, but you can) and have 5 project widgets and 2 task widgets.  
However only one time registration can be going on at a time, so if one widget (or the punch-bar) starts a time registration, no other widget (or even the punch-bar) can start another registration.   
The procedure for starting a new time registration and stopping a time registration should take into account the rules described in the punch-bar.

## Action Dialog

## Notifications

### Time Registration

### Backup/Restore

# Modules

## Projects and Tasks

The projects and tasks module comes in two parts. The first part with the projects, the second part with the tasks for the selected project.

### Table

#### Project

The project table should have following fields:

* id - INTEGER
* name – VARCHAR
* comment – VARCHAR
* order – INTEGER
* defaultValue – SMALLINT
* externalId – BIGINT
* externalSystem – VARCHAR
* flags – TEXT
* finished – SMALLINT

This is the create statement:  
CREATE TABLE 'PROJECT' ('id' INTEGER PRIMARY KEY AUTOINCREMENT, 'name' VARCHAR, 'comment' VARCHAR, 'order' INTEGER DEFAULT 0, 'defaultValue' SMALLINT DEFAULT false, 'externalId' BIGINT, 'externalSystem' VARCHAR, 'flags' TEXT, 'finished' SMALLINT DEFAULT 0);

There should always be a default value (one default record) available in the database. This is the insert for the required record (the name and comment should be localized!):  
INSERT INTO 'PROJECT' (id, name, comment, defaultValue, finished, flags) values (1, “name”, “comment”, 1, 0, "”);

#### Task

The task table should have following fields:

* id – INTEGER
* name – VARCHAR
* comment – VARCHAR
* projectId – INTEGER
* order – INTEGER
* externalId – BIGINT
* externalSystem – VARCHAR
* flags – TEXT
* finished – SMALLINT

This is the create statement:  
CREATE TABLE 'TASK' ('id' INTEGER PRIMARY KEY AUTOINCREMENT, 'name' VARCHAR, 'comment', VARCHAR, 'projectId' INTEGER, 'order' INTEGER DEFAULT 0, 'externalId' BIGINT, 'externalSystem' VARCHAR, 'flags' TEXT, 'finished' SMALLINT default 0);

There should always be a default value (one default record) available in the database. This is the insert for the required record (the name and comment should be localized!):  
INSERT INTO 'TASK' (id, name, comment, defaultValue, finished, flags, projectId) values (1, “name”, “comment”, 1, 0, "”, 1);

### Views

#### Project List

The list of projects is a list sorted on the name of the project and only displaying the name of each project. Using the top-bar the user can add an extra project. By selecting one project the detail-view opens. The detail view is in fact the task list defined later on. Depending on the preferences the project list should contain all projects or only the non-finished projects.

From the project list a few actions on the existing projects are possible. Which are depending of the operating system.

Android: On Android we use the action bar overflow to add an option to show or hide the finished projects. Following actions are possible on an existing project in the list by long-pressing the item:

1. View details
2. Edit project
3. Copy project and tasks
4. Add new project

iOS: On iOS we don’t implement the hide/show finished projects feature in this screen because of the limits of the top-bar.   
For the actions possible on existing projects the user will only be able to delete a project right from the list using the edit button in the navigation bar or by swiping the project out of the list. When deleting the same logic should be applied as in the edit-screen. Other options will not be available in this screen. The view details will be opened when pressing an item in the list, the add new project feature can be launched with the add button in the navigation bar. Other options will be made available from the project details screen aka the task list.

#### Add/Edit Project

When adding a project the user can enter two things:

1. The name of the project
2. A comment for the project

The name of the project is mandatory (but limited to a maximum size of 30 characters).  
The comment input field should be single-line but unlimited in size.

When saving a project some checks need to be done:

1. Check if the project name is already in use, this is not allowed!
2. Check if the project name is filled in.

When the project is saved in the database it should be marked with defaultValue set to false, finished set to false, order set to 0 and all other fields should be empty. When this is saved a new task should be created for this project. This way a project has always a task by default! The task should have no comment, the name should be localized to something like ‘Default Task’, the project-id should be set, flags should be empty, order should be 0 and finished should be false. After saving to the DB the add-view should close and the list-view should be updated.

The screen can also be used in ‘edit’ mode. In this mode the project will be preloaded and when updating the project the same logic and checks needs to be applied as when creating a new one.

iOS: On iOS there’s also a ‘delete’ button in the navigation bar. The details for deleting a project are described later in the document.

#### Project Details

The project details screen contains quite a lot of information.  
First of all the title of the screen should be the name of the project. Secondly it will also display the comment, or a label telling the user that no comment has been entered for the project.

##### Time Registration Info

The second part is the information about the time registrations linked to this project. The total time that has been spent on this project so far should be list there and total number of punches (punch-in) for this project.  
The total time spent on the project so far should be in a short hour-minute-second format (something like 1h 35s 20m) depending on the time precision (minutes or seconds) the user has set in the settings.

##### Tasks

The tasks-part of the project details is list of all tasks available in the database for the selected (current loaded) project.

##### Actions

The actions in the screen depend on the platform in use.

Android: For Android all project actions (and two task actions) will be available from the action bar. The actions are:

1. Edit: This will open the project add-screen with the selected project’s details already loaded. When pressing save no new instance will be created but a database update query will be launched and upon returning to the detail view the fields (including the window title) will be updated.
2. Delete project: After asking for confirmation the project will be deleted and the user will be returned to the project list, which gets reloaded (the details for deleting a project are described later in the document).
3. Add task: This launches the task-adding screen. When returning from this screen the task-list needs to be reloaded.
4. Show only unfinished tasks/Show all tasks: just as for projects this switches between all tasks or only the ones that are not yet finished.
5. Reporting: This opens the reporting module with the project pre-selected.

From the list of tasks the user will be able to able to long-press a task and have some task-specific actions:

1. Edit task: Will launch the add screen with the task-name preloaded for editing. The list of tasks needs to be reloaded afterwards.
2. Move task: This will move the task to another project that can be chosen from a list of projects (the list should not include the current selected project). The tasks needs to be reloaded afterwards.
3. Mark finished/unfinished: this will switch the finished flag of the task and reload the task-list.
4. Reporting: this will open the reporting module with the selected project and task preloaded.
5. Delete task: this will delete the current task with a confirmation box and reload the task-list (The details for deleting a task are described later in the document).
6. Add new task: This launches the task-adding screen. When returning from this screen the task-list needs to be reloaded.

iOS: In the navigation bar exactly one action will be available: edit. This action will open up the add-screen but with the currently selected project preloaded. When saving changes it will update the database and return to the detail view. The detail view however needs to be updated. The edit-screen will have an extra action (other than save) being ‘delete’. The delete option will remove the selected project from the database after confirmation of the user. After this the user needs to be delegated back to the project-list via the details view.

Above the task list there will be a sub-navigation-bar with an internationalized title like ‘Tasks’. The bar will also contain a button for adding a new task and an ‘edit’ button for enabling modifications in the task-list. Pressing the add-button will load the add-task-screen and when a new task is saved the list of tasks is reloaded.  
From the task-list the user can also select the ‘edit’ button to enable the delete tasks function on the list or he can simply wipe over row to show the delete button in the row. When the delete button is pressed the same logic should be applied as described later in this document.  
The switch between finished and unfinished tasks will be done exclusively in the preferences of the application (as in the projects list).

#### Add/Edit Task

### Project Deletion

When a projects gets deleted some rules need to be applied:

1. At least one project is required for the application to work. So if the global project count is equal to one it may not be possible to delete the project.
2. When a project has the flag *defaultValue* set to true (or in database 1) then another project should be marked with the *defaultValue* to true. To determine which project you can just load a list of all projects and select the first project.
3. When deleting a project the user is asked for confirmation, this should also mention that all related tasks and time registrations would also be removed.

### Task Deletion

When a task gets deleted some rules need to be applied:

1. At least one task is required per project for the application to work. So if the global task count for a project is equal to one it may not be possible to delete the task.
2. When deleting a task the user is asked for confirmation, this should mention that all related time registrations would also be removed.

## Time Registrations

### Table

The table TimeRegistration must have following fields:

* id – INTEGER
* startTime – DATE
* endTime – DATE
* comment – VARCHAR
* taskId – INTEGER
* externalId – BIGINT
* externalSystem – VARCHAR
* flags – TEXT

This is the create statement:  
CREATE TABLE 'TIMEREGISTRATION' ('id' INTEGER PRIMARY KEY AUTOINCREMENT, 'startTime' DATE, 'endTime' DATE, 'comment' VARCHAR, 'taskId' INTEGER, 'externalId' BIGINT, 'externalSystem' VARCHAR, 'flags' TEXT);

### Views

#### Time Registration List

The list of time registrations should always load time registrations in batches of 10. If more time registrations are available in the database a final (11th) list item should be shown called the load-more-item. Pressing this should load another batch of 10 items, add them to the original list and if there are still more again add a last list item to load more. Important when loading more items is that the position in the list should be maintained. This means that when the user presses the load-more-button the user cannot jump back up to the beginning of the list.

The order of the list is also important. The list should always be ordered descending based on the start time of the time registration.

The data to display is as follows:

1. The start time formatted using the device locale for the date (with a rather small format, only the month needs to be an abbreviation of the month), for the time a short format is used the and the preference for time precision (h:m:s or h:m) and the AM/PM preference should be taken into account when generating this piece of text. The format of the text should be simple, not to big, noting bold.
2. The end time again formatted using the same rules as for the start time. If there is no start time available (so it’s an ongoing time registration) then the label “Until now…” will be shown. The format of the text should be simple, not to big, noting bold.
3. Next is the duration of the time registration that will be shown. When calculating the duration it should take into account the time precision preference (h:m:s or h:m). If the greatest amount of time is 0 it will not be show. For example “0 hours 0 minutes 10 seconds” needs to become just “10 seconds”. The time indications should be of a long format (e.g.: seconds instead of just ‘s’). The format of this piece of text should be light-grey and bold.
4. Also a concatenation of the project and the task for this project should be shown in the form of ‘A Project – A Task’. The format is again light grey and bold.
5. If the time registration has a comment it needs to be displayed. It will display a label on line (‘Comment:’) and the comment itself on the next line. If no comment is available the comment (including the label) will be made invisible.

When single pressing a time registration the details screen will open for the selected time registration.

Android: On Android when long-pressing a time registration an edit-action will be made available. When selecting the edit-action the ‘Action Dialog’ opens from which all different kind of operations are available for the user. What the content of the action dialog will be is explained in the concepts area at the beginning of this document.

iOS: On iOS no actions are possible for a time registration by long pressing it. All actions will only be possible from the detail view.

#### Time Registration Detail

The time registrations detail screen (title: ‘Registration Details’) should contain the same information that was already displayed in the list of time registration. Only difference here is that the information is a little more spread:

1. The same logic should be applied as for displaying the list
2. For the end time again the same logic.
3. For the duration the same as in the list view.
4. The project and task are now split:
   1. There’s one line displaying the project
   2. And another line displaying the task
5. For the comments the same rules apply as in the list view to display or not display the comment.

In the top-bar an edit button will be available which will act different depending on the platform.

Android: On android the button will have a pencil icon and will launch the action dialog that is explained in the beginning of this document. From that dialog the user will be able to do all kinds of time registration actions. All the actions available in the action dialog are described below.

iOS: On iOS the user will have the edit button in the navigation bar that will launch a popup with different buttons for the possible actions on this time registration. Each button will launch a new window in which the edit-action can be performed and finished (and thus saved to the local DB). The different edit-actions are described below.

#### Add Time Registration

Time registrations can be added manually to the system using this feature. When launching this feature a wizard-like setup will have to be walked through by the user. In just a few steps he can define all the needed properties for the new time registrations.

All different steps are explained right now:

1. Info: this first step shows the user some info about this feature and what they have to take into account.
2. Start Time: The second step lets the user choose a start date and time.  
   Initial Value: this should be the current date and time.  
   Validation: after selecting a start time and navigating to the next step some validation is required. First of all a check needs to be done to see if the selected date-time is not in the future. The second check is to see if the start time interferes with another time registration. This check will perform a query like this:  
   *select \* from timeregistration where startTime <= DATE and (endTime is null or endTime > DATE);*If one validation fails the next is not checked anymore and an error message is displayed.
3. End Time: This step lets the user choose an end time for the time registration.  
   Iniital Value: this should be the start time selected in the previous step.  
   Validation: after the end time has been selected it needs to be validated to check if greater than the start time. The second validation is to see if it’s not in the future. And the third validation requires a lookup of the ‘next’ time registration (by setting the selected start time from step 1 to the end time of dummy lookup time registration). The start time of the result of the lookup is the maximum value for the date and time that is selected in this step. If one validation fails the next is not checked anymore and an error message is displayed.
4. Project and Task: The fourth step is to select a project and a task. The project can be chosen out of all projects. The task can be chosen only if a project has been chosen and from a list that is filtered based on the preference ‘Hide finished tasks’. If no task and/or project is selected a validation error needs to be shown.
5. Comment: this is the last step for the user to enter data. The comment can be entered but is not required. It should be a multiline text-input field.
6. Confirmation/Validation: In this last step all the time registration info is shown to the user in order for him to confirm or start making some changes to it (by going back in the wizard). After this step the user can press the save button. This will trigger a database save of the newly created time registration and will check if the notifications need to be updated.

On the first screen you have a ‘Cancel’ and ‘Next’ button, on the last screen a ‘Previous’ and ‘Save’ button and on all other screens a ‘Previous’ and ‘Next’ button. Whenever going back or forward in the application after values have been modified/selected they should always be restored in the view so that the user always sees his entered data when moving back or forward.  
Whenever moving forward from the start time selection to the end time selection after an end time was already selected before a check needs to be done if the end time is still after the start time. If so no problem, otherwise the end time should be reset to the start time.

#### Edit Actions

##### Punch Out

This will simply punch out the time registration (with or without a comment).  
When: if a time registration is ongoing  
Platform: Android

##### Punch Out and start next

This will simply punch out the time registration (with or without a comment) and start the procedure to start a new time registration.  
When: if a time registration is ongoing   
Platform: Android

##### Split

This will start a wizard-like feature that will allow you to split a certain time registration in two parts. The preference ‘default gap when splitting’ will be used to present a default option to the user where the time registration is split exactly in the middle with the gap that has been defined in the preferences.  
When: if a time registration is finished   
Platform: Android, iOS

##### Open details

When: always   
Platform: Android

##### Edit starting time

When: always   
Platform: Android

##### Edit end time

When: if a time registration is finished   
Platform: Android

##### Restart

When: if a time registration is finished and it is the last one   
Platform: Android

##### Edit project and task

When: always   
Platform: Android

##### Set comment

When: always   
Platform: Android

##### Delete

When: always   
Platform: Android

## Reporting

## Preferences

A lot of customization of the application is possible through the preferences of the application. What follows is a list with categories and preferences in the categories that can be set by the user. The way this is done on each platform is described later.

* Time Registrations
  + General
    - Auto close 60s gap  
      when a time registration is stopped, and a new one started within a minute (60 seconds) then the start time of the new registration will be set to the same as the end time of the previous registration so no gaps appear. If set to false nothing will be done and a one minute gap can appear…  
      Default value: true  
      OS: Android, iOS
    - Punch-bar on main screen  
      if the punch-bar should be visible on the main screen or not.  
      Default value: true  
      OS: Android
    - Punch-bar on all screens  
      if the punch-bar should also be available in the project-list, the project details, the time registration list and the time registration details.  
      On the iOS platform this will only make the punch-bar available on the project- and time registration-list, not on the project details because of limited space.  
      Default value: false  
      OS: Android, iOS
    - Default gap when splitting  
      when splitting a time registration you can define a default gap (in minutes) that will be presented to you. So if you selected the end time of your first time registration, then the start time will be set (but can still be modified before saving) to the end time + the gap.  
      Default value: 30 minutes  
      Minimum value: 1 minute  
      Maximum value: 120 minutes  
      OS: Android, iOS
  + Punch In
    - Ask for a task if only one task is available  
      when starting a new time registration and only one task is available for the selected project the system will (if this option is enabled) still ask you to select a task. If the option is disabled the system will, after selecting the project, immediately start a new time registration for this project and task.  
      Default value: true  
      OS: Android, iOS
  + Punch Out
    - Immediate punch out  
      if the action dialog should be shown or not when you want to punch out.  
      Default value: false  
      OS: Android
    - Ask for comment  
      when punching out if a dialog should be shown to enter a comment for the time registration.  
      Default value: true  
      OS: iOS
    - Ask for finishing task  
      upon ending a time registration, if this option is enabled you will be asked if the task of the time registration should be marked as finished or not. If this option is disabled nothing is asked for and nothing is done.  
      Default value: false  
      OS: Android, iOS
  + Action Dialog
    - Default action for ongoing Time Registrations  
      Default Value: ‘Punch Out’  
      OS: Android
    - Default action for finished Time Registrations  
      Default value: ‘Punch Out’  
      OS: Android
    - Ask for comment  
      if the action dialog is shown and the option ‘punch out’ is selected then the comment input field will be shown to register a comment when this option is enabled. If disabled it will hidden and no comment will be registered.  
      Default value: true  
      OS: Android
* Projects and Tasks
  + Hide finished tasks  
    everywhere that tasks are listed the finished one will be in- or excluded from the list depending on this setting.  
    Default value: true  
    OS: Android, iOS
  + Hide finished projects  
    everywhere that projects are listed the finished one will be in- or excluded from the list depending on this setting.  
    Default value: true  
    OS: Android, iOS
* Date and Time
  + 12h/24h time configuration  
    defines if the user prefers a 12 or 24-hour display of times in the application.  
    Values: System Default (0), 12-hour (1), 24-hour(2)  
    Default value: System Default (0)  
    OS: Android, iOS
  + First Day of Week  
    defines what the user his first day of the week will be.  
    Values: Monday (0), Tuesday (1), Wednesday (2), Thursday (3), Friday (4), Saturday (5), Sunday (6)  
    Default value: Sunday (6)  
    OS: Android, iOS
  + Time Precision  
    how precise the times should be displayed (and calculated with).  
    Values: Seconds (0), Minutes (1)  
    Default value: Minutes (1)  
    OS: Android, iOS
* Notifications
  + Show status bar notifications?  
    if status bar notifications should be shown or not.  
    Default value: true  
    OS: Android
* Backup and Restore
  + Backup  
    this option will instantly create a backup of the SQLite database. The backup will be saved on the location defined in the ‘Location of backups’ preference. If no location has been defined a fallback location will be used. This fall back directory is platform dependent. The file that is stored will need to have the extension ‘bak’.  
    OS: Android
  + Restore  
    this option will restore a backup that has been created previously or even from another device. When restoring a list of files will be generated from the ‘Location of backups’ directory, if no location of backups is defined the fallback directory will be used. The list will be filtered on files with the extension ‘bak’. After a backup file is selected the existing SQLite database fill will be removed and replaced with the contents of the backup file.  
    OS: Android
  + Send backup…  
    a list of backup files will be shown the same way as in the restore-option. After selecting a backup file the user will be able to send it using a mail application (or any application that can handle an attachement). Which applications can be used is highly dependent on the platform.  
    OS: Android
  + Location of backups  
    this option will be used to select a directory where the backup files will be stored. If no location is selected here the fall back directory will be used.  
    OS: Android
  + Backup/Restore documentation  
    this option shows the ‘backup and restore’ documentation to the user so he knows how to use this feature.  
    OS: Android
* Reset Application  
  this option will remove all application data. It will clean all tables in the SQLite database and recreate the default data as if the application was just installed.  
  OS: Android
* Reset Preferences  
  this option will reset the preferences to their default values as if the application was just installed.  
  OS: Android

# Others

## About Screen

Technical Analysis

# Database

All the application data will be stored in a SQLite database that is available on the device.

The exact location of storing the database and accessing it will depend on the environment for which the application is developed. Both iOS and Android have different standards and mechanisms to store the database and interact with it. The only thing that is fixed over all platforms is name of the database: **worktime.db**.

## Database Versioning

In order to make it possible for the database to be updated when releasing a new version of the application we need to implement some kind of database versioning system.

When launching the application a check needs to be performed every time to check if the database is available or not. If not it should be created. The creation should contain all the initial create scripts but also all update scripts should be executed.  
If the database is available all update scripts need to be loaded and if one of the update scripts is for a newer DB version than that is currently installed the script should be executed.

In order for this system to work a DB version number of the current database should be available somewhere.

# Preferences

All preferences for the application should be stored using the platform specific standards.  
This means that for Android we use the preference views, keys and the preference fallback mechanism. TODO: THE IOS WAY!

# Out of Scope

Beneath is a list of features that are not (yet) implemented in certain screens on some OS’s. If a feature is missing it doesn’t mean it will be really out of scope, but for now it is. Sometimes this just means that it makes more sense to implement another screen/feature first and then start implementing these ones.

## Android

## iOS

### Projects List

1. Copy project and tasks “with the option to copy only the finished tasks”

### Project Details

1. Total time spent on a project (missing the time registration link)
2. Punch-In count (missing the time registration link)
3. Marking finished/unfinished of a task (missing the preferences)
4. Move task “to another project”
5. Reporting link for projects (missing the reporting module)
6. Reporting link for tasks (missing the reporting module)

### Preferences

1. Time Registrations -> punch-bar on main screen
2. Time Registrations -> punch-bar on all screens
3. Time Registrations -> immediate punch out
4. Time Registrations -> action dialog
5. Notifications
6. Backup and restore
7. Reset application
8. Reset preferences