Project acronym:	WellCo
Project full title:	Wellbeing and Health Virtual Coach
Call identifier:	H2020-SC1-2017-CNECT-1
Type of action:	RIA- Research and Innovation Action
Start date:	1 December 2017
End date:	30 November 2020
Grant agreement no. :	769765

D3.2 – Interfaces and user manual

WP3	WellCo prototyping and architecture
Due Date:	31/01/2019 (M14)
Submission Date:	27/05/2019
Responsible Partner:	ConnectedCare
Version:	1.1
Status:	Draft
Author(s):	Laurens Kemp (CON), Jan Keijzer (CON)
Reviewer(s):	MONSENSO, JSI
Deliverable Type:	R: Report
Dissemination Level:	PU: Public

Version history

Version	Date	Author	Partner	Description
0.1	16-01-2019	**Laurens Kemp**	CON	Setup of structure
0.2	21-01-2019	**Laurens Kemp**	CON	Added app structure and prototype 1 scope designs
0.3	05-02-2019	**Laurens Kemp**	CON	Modified social network section according to updated design
0.4	14-03-2019	**Laurens Kemp**	CON	Modified various sections according to updated designs and scope
0.5	11-04-2019	**Laurens Kemp**	CON	Updated entire structure to current state of development
0.6	19-04-2019	**Jan Keijzer**	CON	Added software architecture
0.7	01-05-2019	**Laurens Kemp**	CON	Finalized first draft for reviewing
0.8	02-05-2019	**Mads Frost**	MONSENSO	Review of first draft
0.9	15-05-2019	**Mitja Lustrek**	JSI	Review of first draft
1.0	16-05-2019	**Laurens Kemp**	CON	Processed all points from review
1.1	27-05-2019	**Inmaculada Luengo/ Paloma Jimeno**	HIB	Final Review and Upload to ECAS

Table of contents

List of figures

Executive summary

This document is the deliverable "D3.2 – Interfaces and user manual" of the European project "WellCo - Wellbeing and Health Virtual Coach" (hereinafter also referred to as "WellCo", project reference: 769765).

The first part of this document deals with the design process that preceded the design and development of the interfaces for the first prototype of the WellCo platform. The initial user test, user requirements, personas, scenarios, interviews and wireframes all contributed towards the state of user interface design and software development as described in this document. Since this deliverable is tightly coupled with the first prototype of the WellCo Senior app and the WellCo Coordinator app, this document describes only those applications, and within those applications, only the functionalities that will be realized in prototype #1. For both applications, the generic UI structure is discussed and each of the functionalities is explained, sometimes accompanied by the final high-fidelity mock-up design, envisioned for a later prototype. For all these functionalities, the technical architecture and functional design of the two applications are discussed on various levels. Since the goal of the first prototype is to test it during the first trial, a user manual, intended for the researchers in this project has been written as part of this deliverable, which can be used as practical guideline to initiate the first trial.

1 Introduction

This deliverable describes the user interface design and the frontend development processes of the various WellCo applications. In this version of this deliverable, the focus is on how the user interface designs for the senior app are created, mainly based on the results of previous design processes as described in D2.4 and D2.5. The design process is discussed in Chapter 2. Next to this, Chapter 3 deals with choosing a frontend framework for the senior app and the coordinator app and setting the technical foundations for development of the frontends. In Chapter 4, the user manual for getting the senior app running during the first trials is described.

In the end, the entire WellCo platform will consist of 4 different applications. The senior app will be the most important one and will contain the core functionalities of the platform, including the virtual coach. The coordinator app is a support app for program coordinators, which will only provide event-content for the senior app and not provide actual interaction with the users of the senior app. In a similar way, the expert app will be used to validate recommendations generated by the virtual coach, but also from that app, no actual interaction with the end-users will be done. The caregiver app however, will provide functionalities to interact with the senior app users and is therefore a more complex application. The interactions between the various apps is depicted in Figure 1. This version of this deliverable will deal with the initial versions of the senior app and the coordinator app. A future version of this deliverable will deal with the other two apps and their respective frontends and will provide both updated designs of the senior app and an updated user manual.

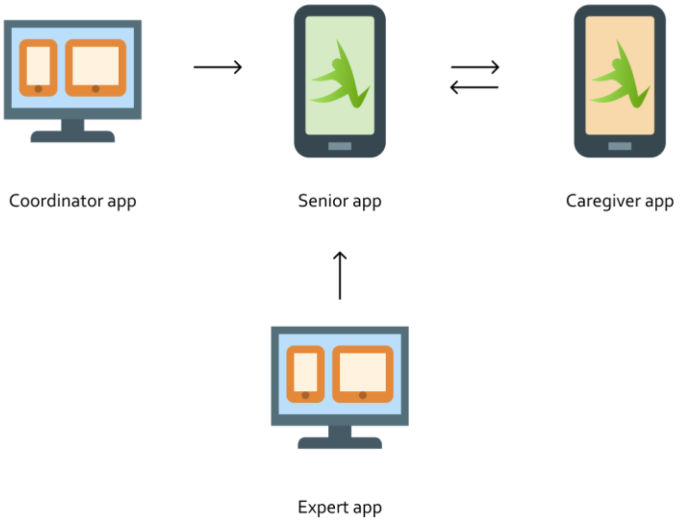


Figure 1 The four different user interfaces of the WellCo platform, showing the senior app as the core of the platform and indicating the other apps interact with it

2 Design and development process

This deliverable builds upon D2.4, where the initial wireframes for the WellCo applications were presented. The evaluations that took place with these wireframes were discussed in D2.5, resulting in an improved list of requirements and a set of end-user-based recommendations for changes or adaptations in the UI of the senior application.

While designing the high-fidelity mock-ups, the consortium has chosen to not execute another evaluation with these mock-ups, but rather directly move on to the first functional prototype and do the evaluations on this. This approach of skipping an intermediate round of evaluations using the high-fidelity mock-ups has been chosen due to the interactive nature of the WellCo platform and the expected effort it would take to mock-up all the interactions with the virtual coach. The high-fidelity mock-ups where nevertheless used as the design guidelines for the functional prototype and provided the developers with a clear view of what had to be built. The entire co-design process up till the current state is shown in Figure 2.

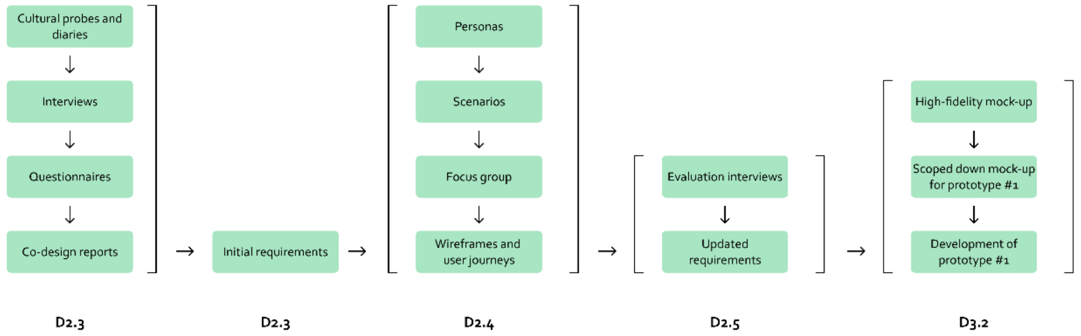


Figure 2 The co-design process that has been followed, from the start of the project up till the current phase. The green blocks indicate the activities that have been done, below it, the corresponding deliverables are mentioned.

In the following chapter, the updated requirements as presented in D2.5, Chapter 5 are integrated, as well as some of the specific comments from the user-testing, presented in D2.5, Chapter 4. The processing of these comments into tangible guidelines for the user interface design is added in Appendix 6.1.

3 Software architecture and functional design

3.1 General

As described in D3.1 WellCo Prototyping and Architecture, the WellCo Interfaces component acts as the link between the users in the system and the WellCo application backend. It will have at least 4 appearances: The senior application, the coordinator application, the caregiver application, and the expert application. The caregiver application and the expert application will not be part of prototype 1, so these two applications will not be described in this document. The software architecture and detailed design of other two applications (Senior and Coordinator) will be described in the next paragraphs.

3.2 Senior application

3.2.1 General

The main requirements for the senior application are described in D2.3 End Users' Requirement Report. Based on these general requirements, the requirements that are directly related to the WellCo Interfaces component are used to create the software architecture and the detailed design.

One of the main requirements is that the application should run on Android smartphones. This poses the question what kind of technology to use. One possibility is to directly build a solution on top of the Android SDK. The advantage of this solution is that it probably has the best performance. It also makes it easier to use the features like camera, microphone, location services, etc. in the application and it supports push notifications from the server side. You will also have full support in the Google Play Store. The disadvantages are that it is not very flexible, it will lead to an expensive and time-consuming development cycle and there are no cross-platform possibilities if in the future we want to extend the application to other mobile phone architectures like iOS.

The next option is to use a hybrid application framework like Cordova, PhoneGap, Ionic/Angular or Framework 7. The advantages are that you can write your code once and run your code everywhere. The disadvantages are that you will get lower performance for features like imaging and it can be an issue to get access to the smartphone features like camera, accelerometer etc. Also push notifications from the server side can be tricky, or not be possible at all.

The third option is to use near-native application frameworks like React Native, Xamarin or Flutter. It has the advantages of native application development as well as the possibility to have cross platform development of large parts of the application. Only real specific requirements on the target platform must be developed specifically for that architecture.

The requirements as described in D2.3, especially the requirements T5 (the ability to have push notifications), IN1 (support touch and speech), U2 (Support virtual rooms and video conferences), U11 (Sharing videos and pictures) and A4 till A9 (provide location dependent information) led us to choose the third option.

We have chosen React Native since it is JavaScript-based (which means the average Frontend developer is able to work with it), open source and free to use. It has a huge user base, since it is the framework used, maintained and developed by Facebook, and it is relatively mature. The alternative platform Flutter is only barely out of the alpha phase, and Xamarin is targeted towards the Windows ecosystem and proprietary.

3.2.2 Layered view

The layered view (Figure 3) considers that we use React Native as framework to build the application on.

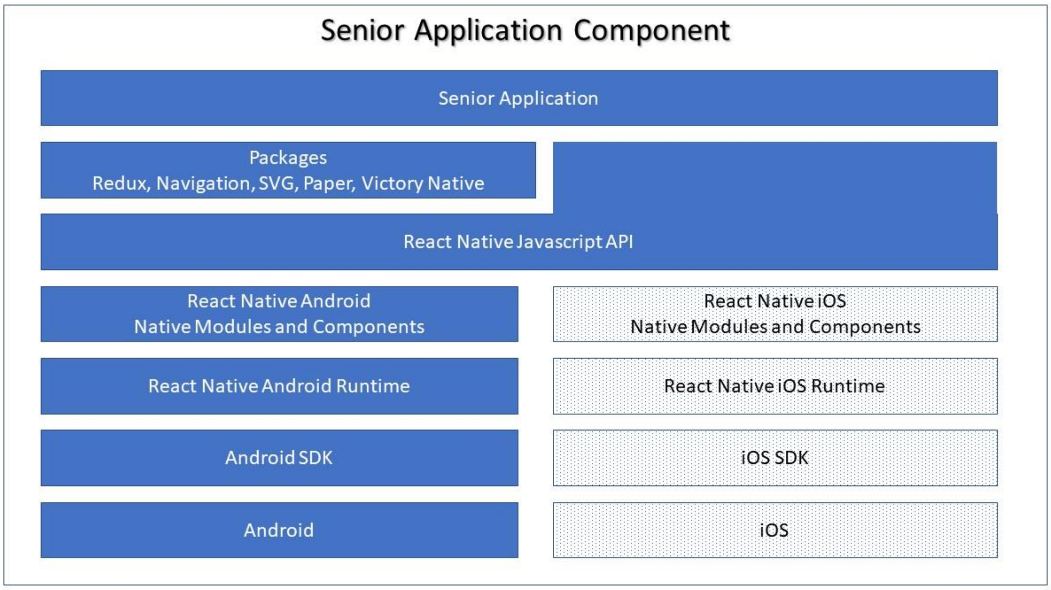

Senior Application Component
Senior Application
Packages
Redux, Navigation, SVG, Paper, Victory Native
React Native Javascript API
React Native Android
Native Modules and Components
React Native iOS
Native Modules and Components
React Native Android Runtime
React Native iOS Runtime
Android SDK
iOS SDK
Android
iOS

Figure 3: Layered view of the senior application

The basis of the application is either the Android operating system or the iOS operating system. Since our application is focused on Android, we will not go into details on the iOS layers. The Android OS layer is the bridge to the device hardware. On top of the Android operating system sits the Android SDK, which is the set of tools written in Java to use the components and libraries of the Android system. React Native Android Runtime layer is the bridge between the Android Java environment and the React Native environment. It is used by the React Native Android Native Modules and Components layer to define the modules and component that React Native exposes to it JavaScript API. This JavaScript API contains the endpoints that can be used by the application to define its application specific components and modules. Since it uses plain JavaScript, JavaScript packages can be added that provide extra functionality which can be exploited by the application. In the senior application we use several packages, which are, amongst others,

1. React Native Redux, a toolset used for state management services
2. React Navigation, a toolset used to navigate between screens
3. React Native SVG, a toolset to display and manipulate SVG images
4. React Native Paper, a toolset that provides Material Design UI components
5. Victory Native, a toolset with components for modular charting and data visualisation

3.2.3 Logical view

The Senior application component is the interface between the senior user and the WellCo backend as can be seen in Figure 4.

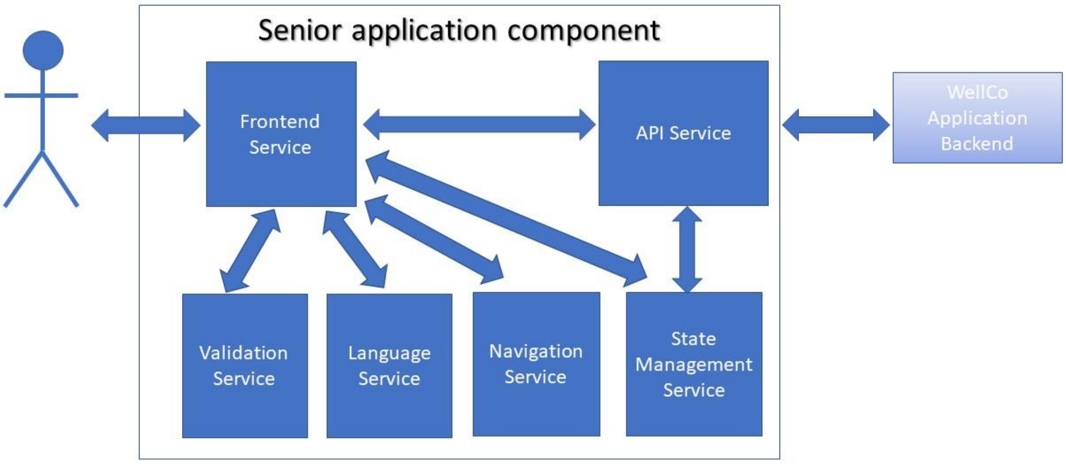


Figure 4: Logical view of the senior application

The user interacts with the Frontend service, which provides the screen and the input fields. When validation of input of the user is needed, the validation service is used. The language setting of the user's device is chosen as the language of the senior application. Translations of the default language (English) to the user's language is done with the Language service. The Navigation service controls the navigation between the screens. Global state management is provided by the State Management service. Storing data to and retrieving data from the backend is done through the API service.

3.2.4 Process view

The global activity diagram for the senior application component is shown in Figure 5. When a user opens the application the splash screen is shown. In parallel the local authentication is retrieved. If a local authentication is found, the dashboard is displayed. If no local authentication can be found, the login page is shown. The user can login and is authenticated. If the user is known the dashboard is displayed. If the user is not known or any other error happens during authentication, the login page is shown again. The user can also decide to close the application from that point.

When the user is on the dashboard, he/she can select any of 6 features: handling activities, handling notifications, handling his/her profile, handling his/her questionnaires, handling his/her social network or handling his/her video groups. After one of these features has been handled, the dashboard is shown again. On the dashboard he/she can also choose to log out, in which case the application shows the login page again.

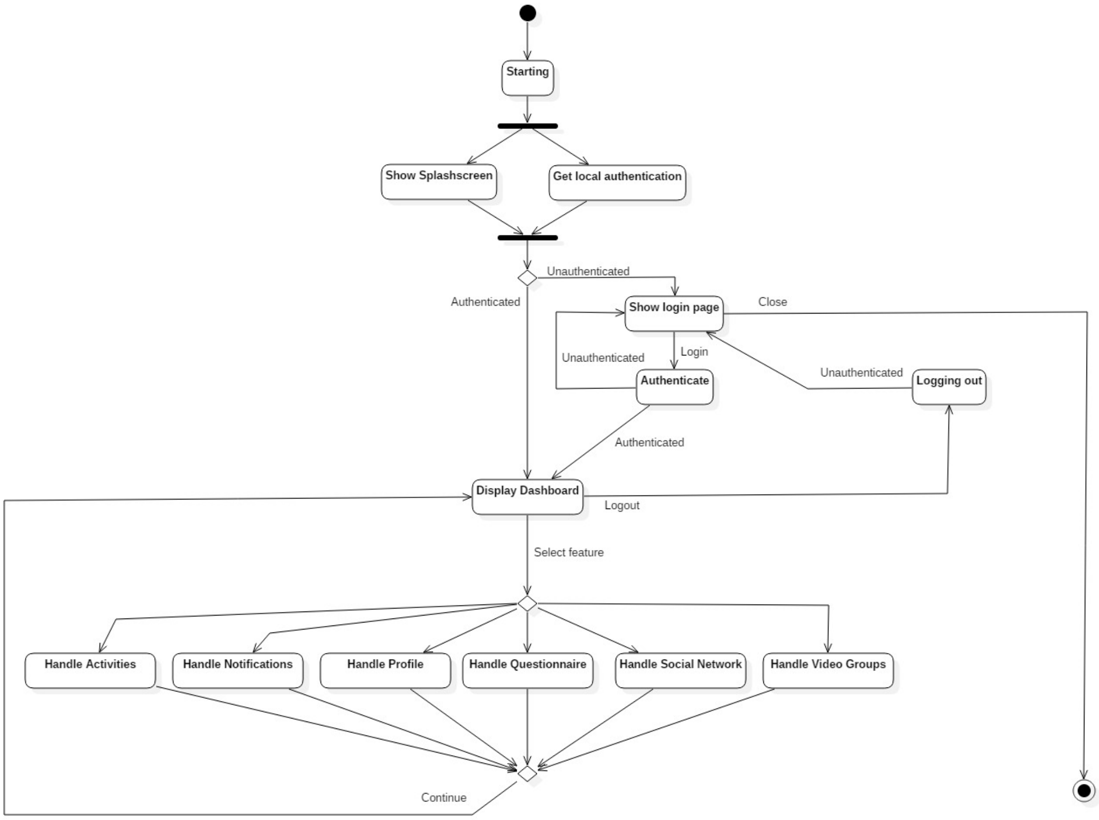


Figure 5: Activity diagram of the senior application

3.2.5 Development view

Figure 6 illustrates the development view of the senior application in the form of a UML component diagram. The senior application component consists of 4 subcomponents:

1. The features.
   a. This subcomponent contains all features in the senior application:
      i. The activities – Shows the activities like Steps, Weight, Heart rate and Sleep.
      ii. The authentication – Contains all authentication and authorization items like log in and log out
      iii. The dashboard – Shows the dashboard, in which the user can switch between features.
      iv. The notifications – Shows the notifications
      v. The profile – The place where the user can edit his/her profile
      vi. The questionnaire – Shows the user specific questionnaire
      vii. The social network – The part in which the user can manage his/her network, the posts and the comments
      viii. The video groups – Shows the video chat groups.
   b. All Features subcomponents have their own parts of the navigation and global state management (redux) subsystems. The subcomponent navigation controls the navigation to and within the subcomponent and the subcomponent redux processes the global state related to the events of the subcomponent. Every subcomponent also has a section of screens, which are pages shown to the user for the given subcomponent. The social network subcomponent also has a

section of reusable UI components (used in multiple screens of the social network subcomponent).

2. The API
    a. The subcomponent deals with storing data to the WellCo backend and retrieving data from the WellCo backend. General storing and retrieval logic are captured in the Abstract API Service. Specific data exchange is done in the following 5 services:
        i. Activities API service. All API services related to the activities
        ii. Authentication API services. All API services related to authentication and authorization
        iii. Questionnaire API services. All API services related to the questionnaires
        iv. Social network API services. All API services related to the social network, posts and comments
        v. User API services. All API services related to the user information.
3. The components
    a. The subcomponent contains all reusable UI components of the senior application.
        i. Drawer Navigator Component: Used to display the hamburger menu
        ii. Input groups: Button, Calendar, Radio, Text, Verification Code, Header. UI controls for input on forms containing the controls themselves and helper text
        iii. User Avatar, User Name and User Row: UI components to show the avatar, the name or a row of avatar and name of a user.
        iv. Post list: UI component to display a list of posts.
        v. Wizard: UI component to display a wizard of steps.
        vi. App Text Input: Common text input styled for the senior application
        vii. Time Ago: UI component to display the create, update or delete time of an item.
        viii. Page Dot Indicator: UI component to indicate the position of a page in a series of pages
        ix. Message Header: UI component to display the header of messages (posts or comments)
        x. Card Action Group: UI component to display a group of actions in card.
4. Generic services
    a. All services used throughout the application
        i. Lang: The language services
        ii. Navigation: The navigation backbone
        iii. Reducers: The state management backbone
        iv. Validation: The validation services
        v. Assets: All assets in the applications
        vi. Bootstrap: All components used for bootstrap purposes.
        vii. Styles: All global styling components
        viii. Utils: All utility functions.

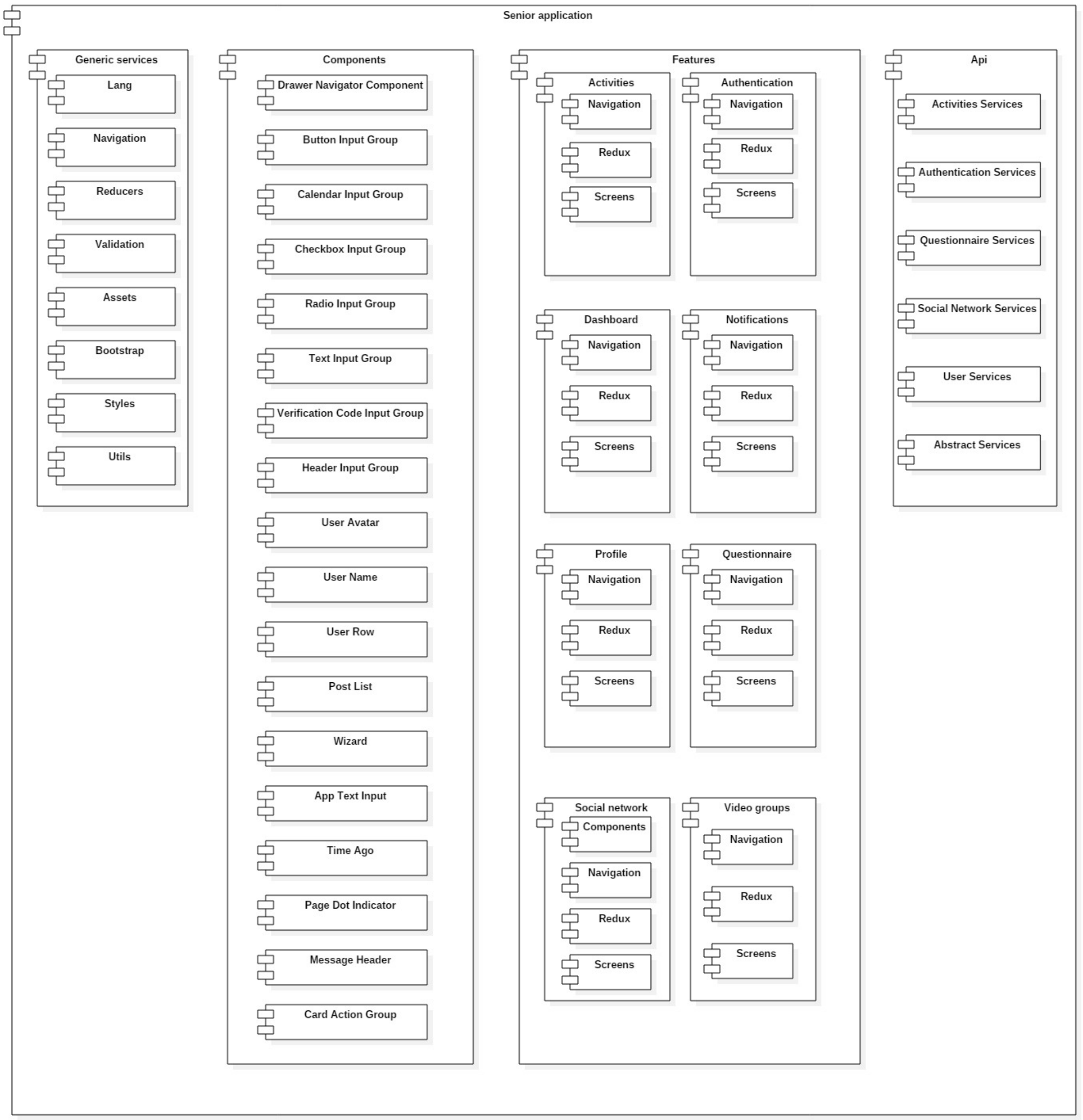


Figure 6: The component diagram of the senior application

3.2.6 Physical view

The full application runs on Android based smartphones. The target for prototype 1 is the Motorola Moto g[6] play. In future iterations, also other Android based smartphones will be targeted.

3.3 Coordinator application

3.3.1 General

The requirements for the coordinator application came from the existence of the Video Chat Groups functionality in the senior application. The events for these video chat groups had to be defined by a new actor: the program coordinator. Since it should be an application accessible from everywhere and we decided that a web application would be the best choice. To be in the same ecosystem as the senior application, we decided to use ReactJS.

3.3.2 Layered view

ReactJS is the starting point of the coordinator application as can be seen in the layered view of Figure 7. It lies on top of the Document Object Model (DOM) and the built-in functions of the browser, which is the fundament of the application. The React Native Web modules and component together with various JavaScript packages provide the interface to the Coordinator application specific code.

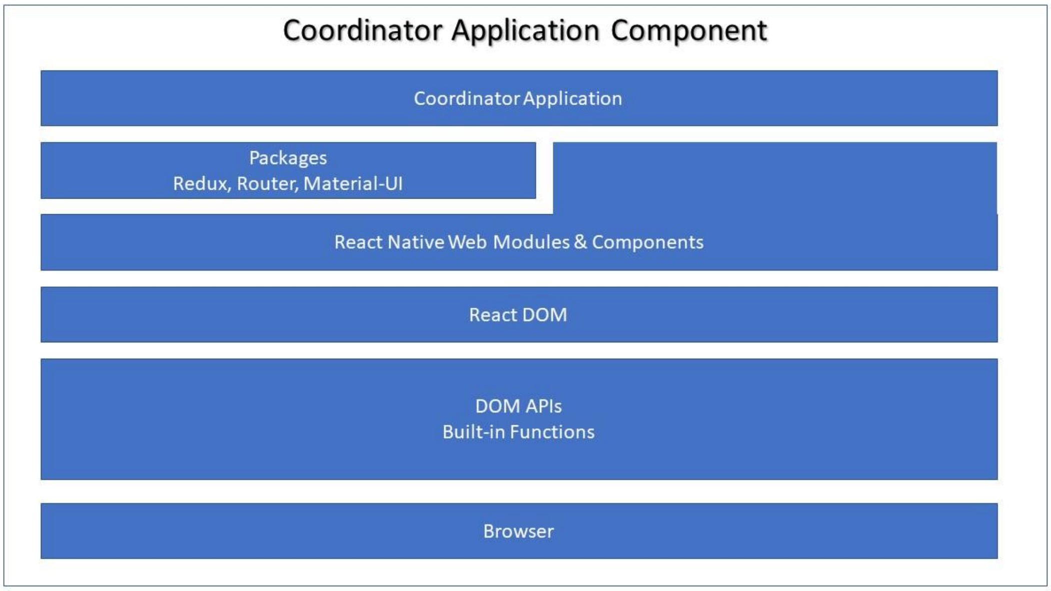

Coordinator Application Component
Coordinator Application
Packages
Redux, Router, Material-UI
React Native Web Modules & Components
React DOM
DOM APIs
Built-in Functions
Browser

Figure 7: Layered view of the coordinator application

3.3.3 Logical view

The Coordinator application component is the interface between the program coordinator user and the WellCo backend as can be seen in Figure 8

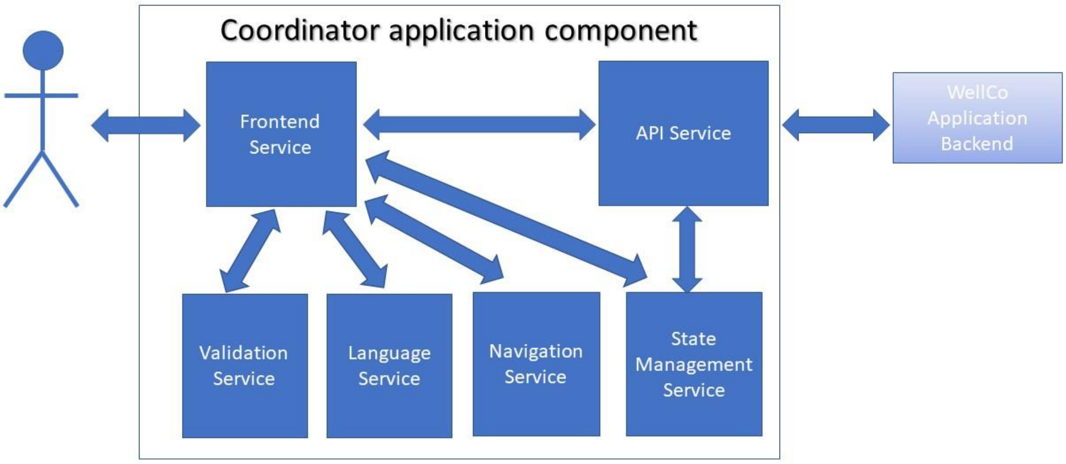


Figure 8: Logical view of the coordinator application

The user interacts with the Frontend service, which provides the screen and the input fields. When validation of input of the user is needed, the validation service is used. The language setting of the user's device is chosen as the language of the senior application. Translations of the default language (English) to the user's language is done with the Language service. The Navigation service controls the navigation between the screens. Global state management is provided by the State Management service. Storing data to and retrieving data from the backend is done through the API service.

3.3.4 Process view

The process view of the coordinator application is illustrated in an UML activity diagram (see Figure 9)

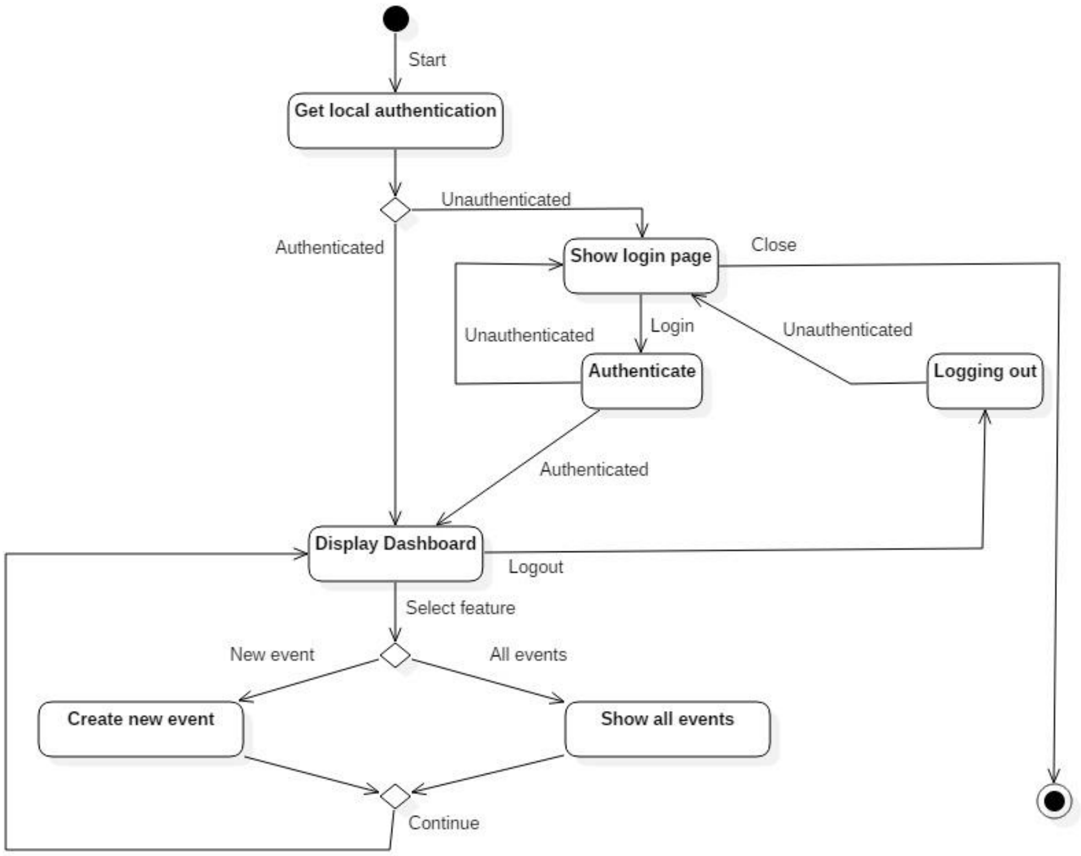


Figure 9: Activity diagram of the coordinator application

The coordinator application first tries to get the local authentication. If the user is known the dashboard is displayed. If the user is not authenticated, the login page is shown, and the user can authenticate himself. The user will see the dashboard if he/she is known. Otherwise the login page is displayed. From the login page he/she can also close the application.

Once on the dashboard, the user can select either to create new events, to show all events or to log out.

3.3.5 Development view

Figure 10 shows the component diagram of the coordinator application. Main features are the events and the authentication. Both subcomponents have local navigation control and local redux state management. The Pages subcomponent contains the pages to be shown to the user. Global state management (reducers) and global navigation control are two subcomponents of the generic services. The generic services also contain language service, validation, asset management, and common styling. Bootstrap functionality and general utilities are also part of the generic services. Common UI components like the text input group and the abstract navigation UI are separated in the “components” subcomponent. The connection to the WellCo backend is concentrated in the API subcomponent. It contains the abstract API services that contains generic API services, that are used by the two specific API services: Events API services and Authentication API services.

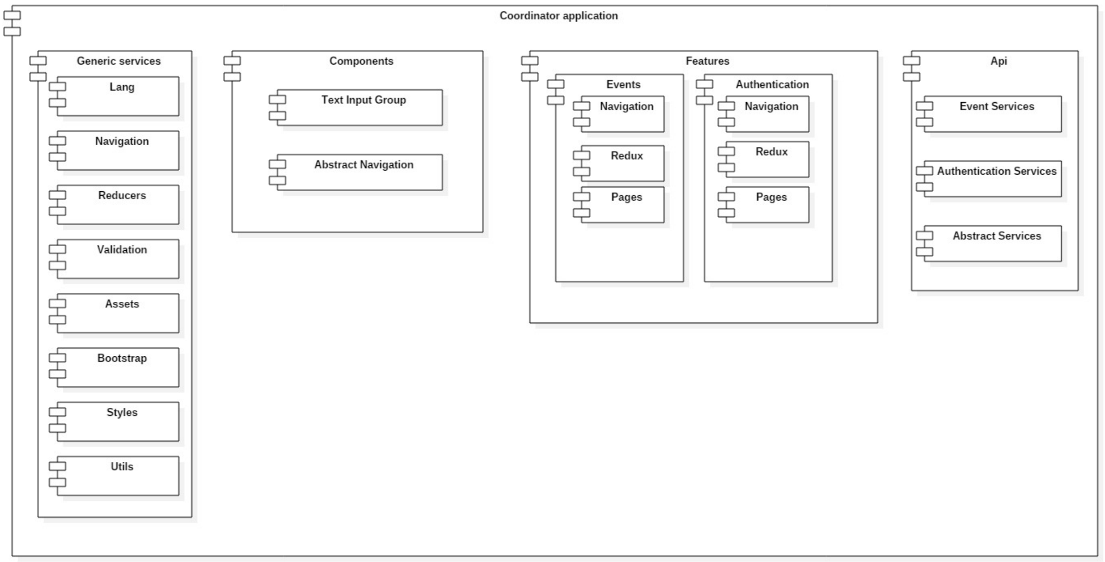


Figure 10: Component diagram of the coordinator application

3.3.6 Physical view

The coordinator application runs at any device that can handle browser functionality.

3.4 Caregiver application

Will be developed and designed in the next prototype.

3.5 Expert application

Will be developed and designed in the next prototype.

4 User manual

In order to get the senior application up and running for the first trials, it has been agreed that the researchers in the WellCo project are responsible for this setup, not the end-users. Nevertheless, the researchers also needed a user manual on how to get the app working. This user manual has not been included because of IP reasons. For future prototypes and trials, a user manual targeting the end-user might be developed, either integrated into the app or as a separate document that will be distributed via the WellCo platform.

5 Conclusions

This deliverable gives a first overview of the final user interfaces and technical realization of those interfaces within the WellCo project. As can be read throughout this report, this deliverable clearly deals with a first version; a first version of the design, of the developed software and of the user manual. In the future iterations of this report, more and more parts will get filled in and all missing functionalities or interfaces, such as the expert and caregiver application, will be described. Nevertheless, this deliverable gives both developers, designers and other project members a firm foundation of how all elements in the project come together and are presented to the end-user. The realization of the software as described in Chapter 3 can be explored and tested in the prototypes that are developed as deliverable D3.3.

6 Appendices

6.1 Processing of end-user comments on wireframes

6.1.1 User journey 1: Opening the app and goals

Scope

Opening the app in the “combined speech-type mode” and navigate the users to the goals section.

Screen / Action	**Questions**	**Comments**	**CON processing comments**
Open the app in speech-type mode		Some users want to type. Despite some want to speech, at least in the current development of the wireframes, they wander when should start to talk.	The fact that users can always speak should be clearly included in start-up flow. Also, as soon as the coach is able to speak, they will be more likely to speak back.
Dashboard	Does the user understand how to navigate to a section?	All users recognize the start menu icon and the one identified as "WellCO" to activate the application. But once they are in the main menu, mostly are not sure what they should do, thus does not understand the interface so easily. Almost all users know how to go to the dashboard, how to exit the application, navigate with the arrows forward and backward, etc.	The UI of the actual prototype should indicate that items are clickable cards, e.g. by using material design cards.
	Does the user want to speak/type?	Mostly feel more comfortable with typewrite	OK
	Does the user navigate with the hamburger menu of	The majority of the users navigates easily with the hamburger menu they explain that this is similar in all websites.	OK

	the dashboard buttons?		
Goals	Does the user know how to navigate to a single goal? Does the user understand the graph or prefer the list?	Denmark: consensus on an easy navigation from the dashboard to goals and single goals; b) they like the visual graph and symbols. Spain: In many cases the navigation is not immediate, but finally they all get the goals site by intuition. All users are easily handled in some aspects of the objectives itinerary: a) They know how to navigate with the arrows forward and backward. Italy: Some sets the goal but does not realizes they have.	OK, The graph will be removed in the wireframes and should be re-introduced only if we have something that can be plotted in time. The UI should clearly show newly set goals after they are set (e.g. by a different colour) and we should allow the user to update/remove the set goals, in cases they have set a goal they didn't want to set.
Running 1 km goal		They like the overview and to see how far they are from the stablished goal. Most users recognize some icons, basic navigation commands between screens (arrows), where you must click to activate a step, etc. Many difficulties have been observed when interpreting the graphs of evolution in the achievement of the objectives (some users recognize the icons but do not know how to interpret their position in the graph as a temporary development); the majority does not understand the information that it contributes (it is interpreted as a representation of times but not of achievement, of challenges, etc.); They also do not understand well the difference between current objectives and objectives already	Graph should be secondary visualization, list as primary. The graph will be removed from now, as we need to know the details of all goals before individual pages can be finalized. Also removed achieved section, can be accessed by a filter now

		achieved (nor do they know how to locate them well in the graph).	
Achievements	Does the user understand how to navigate here?	Spain: They clearly identify where they have to click to go towards an objective. They recognize the "success" or "medal" icon in the achievement of an objective. Difficulties are observed in other aspects: a) They do not understand the objective progress bar and its meaning well. b) Does not distinguish well between messages of support from the social network and messages from the coach. c) They doubt, sometimes, if they must touch the icon or the arrow to access each objective.	Social aspect and messages from social network will become more clear when actual photos and names are used.
Back to dashboard		Denmark: The user wants to type - uses the hamburger menu to get back Spain: users understand how to go to the dashboard, how to exit the application, navigate with the arrows forward and backward, etc.	The voice control navigation should be implemented in a later prototype.

General comments about this user journey
Almost half of the users feel comfortable and considers that is easy to open, because they are accustomed to use other apps to search for information. The rest looks confused in some aspects.

6.1.2 User journey 2: Opening the app and tips/events

Scope

Opening the app in the “combined speech-type mode” and navigate the users to the tips and events section.

Screen / Action	**Questions**	**Comments**	**CON processing comments**
Open the app in speech-type mode			
Dashboard		Easy navigation to here	
Tips and events	What other categories would the user like to have?	Denmark: Work related topics. Events about crocheting. Training videos for help with back pain. To foster users to record videos and share. Others do not demand any further categories. Spain: Information about leisure groups, cultural, entertainment, accompaniment ... Italy: A category for taking notes about her own affections (e.g. family life, activities with her friends). Something about do-it-yourself. Information about policies and promotions dedicated to over 65. Humour, stories, anecdotes, information about volunteering. A daily planner.	Added some of the mentioned categories, others should be added in the actual prototype
Physical activities	What type of physical activity videos would the user like to see? Does the user want to get recommended videos or search for videos	Italy: Videos about low-impact exercises to do at home. Videos about walking paths, divided for level of difficulty. Stretching exercises. How to Run, running at different ages. Some prefer to search freely the videos, but the interaction has to be simple and easy. Some prefer to be recommended.	Added a search functionality
Articles	Does the user want to search for articles or get articles recommended?	Denmark: recommended articles with short text to be readied easily on the phone	Recommendation based on the goals that already have been set.

Events	What type of events does the user want to see?	Denmark: Football, Cooking, Movies, Theatre, Walking, cinema. Spain: cultural activities, leisure, entertainment, health talks, meetings, excursions and all kind of social activities. Italy: Clinical conferences about the clinical treatment of common healthcare issues, folk festival and cultural events (for example theatre). Non-competitive footrace. Food markets. Festivals, local markets and other free events.	Should be added in the final database
	Does the user want to search for events?	Denmark: yes Spain: They would like to be able to search for events, but they really prefer to receive recommendations. Italy: different opinions, but mainly yes	Added the search feature
	Does the user understand how to navigate through time?	Denmark: some do not understand did not understand the navigation in time and prefer to have a common appearance of a monthly calendar, instead of needing to move among weeks. Some recommend making available to add it to the regular calendar (outlook). Spain: Some users recognize how to navigate in time (go to past or future weeks), although some do not recognize where and how they should click. In fact, one of the users does not show interest in this section and understands that it is something that would not be useful.	Kept existing navigation, can be changed in the final prototype, based on the amount of available data per week or day.

		Italy: mainly yes	
	What details does the user want to see about an event?	Denmark: how to get there, link to the official website or ticket sales. Spain: Details about the events that are of interest: i. The price of the entrance (and if the event is free entry or not). ii. The means of transport available to get there (and whether or not it is included in the entrance). iii. The target audience for that event. Italy: For each event it could be useful to receive reference contacts (e.g. mail and/or phone number of organizers) Price of the events. Information about informal groups that share hobbies. Possibility to define a geographical area of interest	Details are added to a detail page when clicking on an event card.
Back to dashboard		Without problems	

General comments about this user journey
Spain: In general, navigation through the itinerary is not complicated, although some details are not easily recognizable (location, video playback, etc.). Many of the users are a little bewildered but interested; They show curiosity to understand the different aspects and utilities of this user journey/section "Tips and events". We understand that in the interaction with the application they can achieve a greater assimilation of the usefulness of the section "Tips and events". The majority shows difficulties to identify the meaning of the icon representing location/venue/map.

6.1.3 User journey 3: Getting a recommendation to set a goal

Scope

Opening the app in the “combined speech-type mode” and trigger the coach tip “Trigger recommendation about cycling”

Screen / Action	**Questions**	**Comments**	**CON processing comments**
Dashboard with avatar	Does the user talk back to the avatar directly? Does the user want to use the type/speak area in the bottom?	Denmark: not familiar with the symbol for speaking and all prefer to type Spain: Mainly without problems, but all prefer to write; It is not always easy to identify the icon to activate the interaction via voice. Italy: most of them does not talk back to the avatar directly	In the real prototype, we need an animation in the bottom area that shows clearly that the coach is always listening
Set goal cycling		Denmark: mainly easy navigate. Spain: Users, in general, encounter difficulties in the process of establishing objectives/goals; Italy: mainly easy to set	Interaction can be improved in the real prototype by using more clear sentences, aimed at the user itself
Set goal	Doe the user accepts such recommendations?	Mainly yes in the three countries. Answers depends a lot if the user do like the recommendation: the example shows cycling and if users are not interested in, then the answer is not as positive as expected.	
View goal	Does the user want to view its goals after accepting or not? Understand he is in the goals section now?	Some users visit the goals again but not sure it affected his understanding of the goal section. Users mainly want to view its goals and also understand where they are.	
Back to goals		Easy back navigation in the three countries	
Back to dashboard		Without problems	

General comments about this user journey
Spain: They do not understand that in WellCo users are in fact able to establish their own goals, and later the dynamics of interaction between the coach and the user will be generated.

6.1.4 User journey 4: Getting a recommendation to join a group

Scope

Opening the app in the "combined speech-type mode" and trigger the coach tip "Trigger recommendation about a social group"

Screen / Action	**Questions**	**Comments**	**CON processing comments**
Dashboard with avatar	Does the user talk back to the avatar directly? Does the user want to use the type/speak area in the bottom?	No, they mainly prefer to type	
Set goal cycling group	Can the user easily join the group?	Yes, mainly can do it easily. Most users understand well how they should proceed to join the group.	OK
Supporting group for cycling		Users mainly finds interesting this feature. They understand the functionality of getting a supporting group for an specific goal.	OK
Back to my network	Does the user understand he is now in the social network	Usually understand well that they have joined the group	

	section and he has joined the group?		
My Network	Does the user understand this interface e.g. from experience with Facebook?	The majority of the users does understand this interface	
Groups	Does the user want to search for supporting groups? Does the user want to receive recommendations for supporting groups?	The majority of the users does understand this interface and want to receive recommendations	
Back to dashboard		Easy navigation	

General comments about this user journey
Information and recommendations are useful only if they are focused on local events. Despite the answers were underlining easy navigation in this user journey, some general comments show that some users (mainly from Italy) were a bit confused. One of the users, male, finds it interesting to connect with new groups (his personal circumstances encourage him to look for spaces for meeting and relationships with other people).

6.1.5 User journey 5: Interacting with the coach Speech only

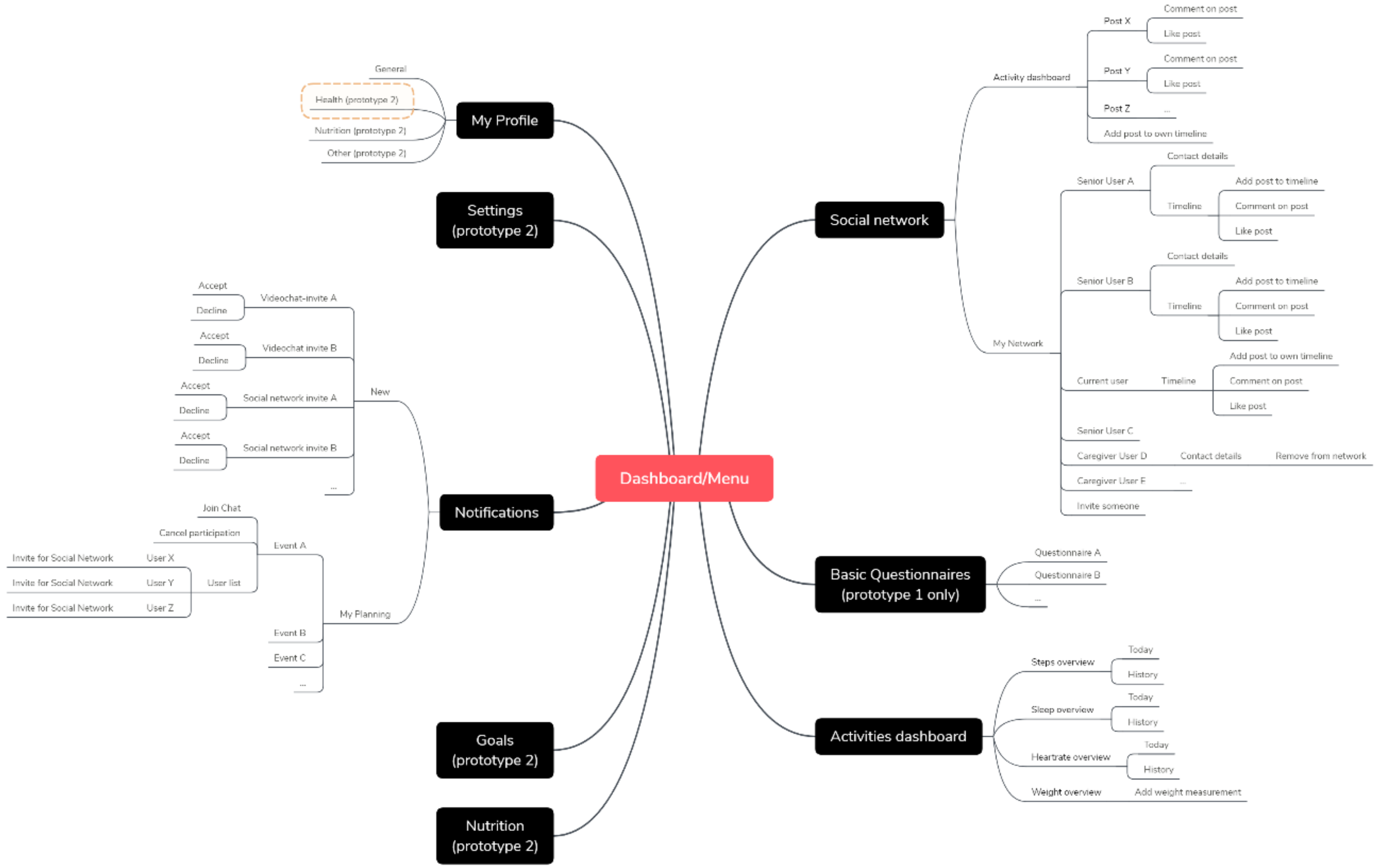
Scope

Opening the app in the “speech only mode” and trigger the coach tip “Trigger recommendation about a social group”

Screen / Action	Questions	Comments	CON processing comments
Follow interaction flow	How does the user react to a coach that communicates only by speech?	No, users mainly prefer to type, because speech does not feel natural	
Activity monitor	Does the user understand what he sees here?	In Denmark and Italy: Mainly yes. Some refers to its similarity with their health apps. In Spain only partially. Thy are wondering how app can monitor some specific issues.	ok
Steps activity	Does the user understand the graph? And does he want more/other recommendations about activities?	Yes, they mainly understand the graph and consider it useful. By contrast, the activity graph, the historical one, was not clear for the majority of the Spanish users. Users do not show a great interest on more/other recommendations.	Changed the graph
Back to activities		Easy navigation	
Back to dashboard		No problems to get back to the dashboard	

General comments about this user journey
In Spain some users show not understanding the usefulness of the monitoring.

6.2 Senior app navigation map



6.3 Researcher User Manual

Not included because of IPR. The manual contains links to some apks and web apps developed because of the project and that are not available publicly yet, due to the effect that the interference with people not participating in the project could have in the quality of the results we get to measure the impact of the project.