aal-2014-171

SENIOR-TV

PROVIDING ICT-BASED FORMAL AND INFORMAL CARE AT HOME

Deliverable D2.2
SENIOR-TV V2

Document information	
Due date of deliverable	31/01/2018
Actual submission date	15/06/2016
Organization name of lead contractor for this deliverable	IMATIA
Revision	Final Version

Dissemination Level		
PU	Public	X
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Authors list	
Author	**Partner**
Iulian Anghelache	CPX
Giorgos Kostopoulos	GULK
Sira López	IMATIA
Carlos Rivas	IMATIA/UVigo
Marcos Mouriño	IMATIA/UVigo
Iztok Žilavec	RC-IKTS

Peer Reviewers	
Reviewer	**Partner**
Catalina Anghelache -Tutulan	COMPEXIN SA
Luis Anido-Rifón	Advisory Board Member

Versioning	
Version	**Summary**
V_1.0	SENIOR-TV platform from M1 to M25
V_2.0	Includes Catalina Anghelache -Tutulan (reviewer) comments

This project has been funded with support from the European Commission.

This document reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Table of Contents

1. Introduction

This document summarizes the work performed from month M1 to M25 in WP2, which final result is the second version of SENIOR-TV platform.

The main objective of WP2 is produce a platform based on interactive TV and mobile technologies according to the requirements of openness and low-cost. The obtained platform has to cover the general project aims: providing formal and informal caregiving services for older adults, focusing on the active prevention and the maintenance of relationships with their friends, their relatives and, the community.

These are the main tasks developed during the first and second cycle and the objectives have been achieved in each of them:

- T2.1.1. Technology Analysis (M1-M6) and T2.1.2. SMART-TV technology development (M7-M27): the analysis of existing smart-TV software and hardware technologies.
- T2.2.1: First wave (M3-M12): the initial version of most relevant informal services (according the result of WP1 workshops).
- T2.2.2: Second wave (M13-M25): the second version of informal services (according the result of WP2 workshops).
- T2.3.1: Formal services first wave (M3-M12): the analysis of potential formal services for the platform.
- T2.3.2: Formal services second wave (M13-M25): the initial version of Health service.
- T2.4.1: Ergonomic Design. First wave (M4-M12): the analysis of senior necessities regarding graphical interfaces and the design of implemented services according these restrictions.
- T2.4.2: Ergonomic Design. Second wave (M13-M24): interface designs improvements and remote-control selection according to first pilot feedback.
- T2.5.1. PD&SI first wave (M4-M12): internationalization and external providers integration.
- T2.5.2. PD&SI second wave (M13-M25): initial version of authentication, notification and device manager services.

The following sections include a detailed description of performed work in each of these tasks. Sections 2 and 3 include the details of technology analysis. Sections 4, 5, 6 and 7 presents the developed services the system architecture and the graphical interface advances. Section 8 includes a review of the required steps to advance on the platform development. Finally, the conclusions are included on section 9.

2. Technology analysis

During first annuity, a study of TV platforms and development technologies was done. As a result of this research, the Android operating system and a Minix STB were selected to build the SENIOR-TV platform. In addition, the first services were developed using web hybrid technologies.

Considering that decisions, the services of the second pilot have been developed with hybrid technologies or with Android native and, Android devices have also been used to test the second version apps.

2.1. Smart-TV software

In this section, we introduce the main environments for application development in the smart TV world. While early approaches were based on proprietary systems, present-day, more elaborated solutions are based on Linux, and in some cases, follow the-web-as-a-platform paradigm based on technologies such as HTML5, CSS or JavaScript. The most relevant proposals available at the time of this writing are introduced below, namely AndroidTV, WebOS, FireOS, Firefox OS, Tizen. A final summary is also included to compare the characteristics of the approaches discussed.

1. **Android**[1] is a Linux-based operating system developed by the Open Handset Alliance, a consortium of developers led by Google. In June 2014, Google introduced AndroidTV[2] as an evolution of Android version 5.0 Lollipop to be deployed in television environments. Currently, manufacturers like Sony, Sharp and Philips have shown interest in including this operating system in their appliances, being also possible to acquire distributions in USB sticks to be connected directly to a media input (e.g., HDMI).

   Application programming for smart televisions in Android follows the same model as smartphone programming. It is based on the Java programming language and supports user interfaces definition through XML files. Frameworks are also available to support the development of web applications based on HTML5, although these applications do not have unlimited access to operating system services as native applications do. AndroidTV developers must explicitly indicate that their applications are designed for a TV environment. Typical features of a mobile phone that are not available on a TV set (e.g., touch screen, GPS receiver, camera, etc.) may be removed from the interface requirements in the application manifest to facilitate application migration to a TV environment. AndroidTV users will access the Google application repository (i.e., Google play) to download new applications to their TV sets in a similar way to Android smartphones.

[1] Android official site: https://www.android.com/
[2] AndroidTV official site: https://www.android.com/tv/

2. **WebOS**[3] is an operating system developed by Palm also based on the Linux operating system. This platform was acquired by Hewlett-Packard to be integrated into its devices. With the advent of Android, Hewlett-Packard halted WebOS development and eventually sold it to LG, which included it into its smart TV sets.

   Thus, WebOS is an operating system designed for integration in smartphones and other devices such as smart TVs. Application development is based on web standards like HTML5, JavaScript and CSS. Its design allows web applications to access the native functionality of the host system. Applications are executed through the webkit renderer, and the system is a multitasking one able to run up to 4 simultaneous applications. As a general rule, a WebOS application is composed of HTML5 code and functional blocks written in JavaScript. While there is an open source distribution named Open WebOS, the version integrated in LG appliances is closed. For application development in this environment, LG offers the Enyo Framework to the developer's community.

   TV sets based on WebOS include a remote-control equipped with a motion sensor that sets a pointer on the television screen in the style of the Nintendo Wii. Additionally, it implements a voice recognition system that allows voice interaction using certain commands. Users access the LG store to download applications.

3. **FireOS**[4] is a Linux-based operating system developed by Amazon for mobile devices (e.g., Kindle) and smart TV devices like FireTV. It is a version of the Android operating system, specifically the Fire OS 3.0 version based on Android Jelly Bean (API level 17). It can be installed in Amazon devices only.

   Applications are written in Java and can be downloaded from the Amazon app store. From the point of view of the application developer, this platform is analogous to Android TV, with the particularity that the applications are available exclusively from the Amazon store. While the operating system is based on an Android distribution, Amazon limits its installation to its own hardware. This, together with the difficulty of downloading apps from other repositories, dramatically reduces the target audience of this platform.

4. **Firefox** OS[5] is an open source operating system based on Linux and the Mozilla Gecko technology. While initially designed for mobile devices, some smart TV manufacturers like Panasonic have already shown their interest in this platform. At the time of booting, the Gecko runtime environment is launched, thus allowing the execution of web applications. Developers may use HTML5, CSS

[3] WebOS official site: http://www.lg.com/uk/smarttv/index.html
[4] FireOS official site: https://developer.amazon.com/android-fireos
[5] Firefox OS official site: https://www.mozilla.org/en-US/firefox/os/

and JavaScript standards for their developments. Each application installed on this operating system behaves like a web page. Firefox OS also introduced the WebAPI concept, through which a selection of system calls can be invoked from the web interface. This reduces the gap between current web applications and native applications that can be installed on this operating system.

5. **Tizen**[6] is a Linux-based open source operating system supported by the Linux Foundation and companies such as Intel, Samsung, Huawei and Fujitsu, among others. While Tizen was initially conceived as a platform for developing web-based applications, from version 2.0 on it also supports native applications. With this enhancement, support for hybrid applications was also included. These latter applications include native components and web application components. Native application development is done in C++. For web applications, HTML5, CSS and JavaScript technologies may be used. Through a WebAPI web applications may access certain core features (e.g., Bluetooth support). The execution environment is based on webkit rendering. Samsung provides an SDK for application design. The ability to run native applications offers many advantages when it comes to access and interact with both the operating system and with other devices that can be connected to the TV set. Similarly, support of standards-based web applications greatly facilitates application design and deployment. The blend of these two approaches in hybrid applications allows programmers to take advantage of both paradigms.
6. **Google TV** created high expectations due to the relevance of the developing company (Google Inc.). However, Google abandoned this line in favour of Android TV. In our case, a Google TV device was evaluated and discarded due to the fact that interactivity is achieved through web browsers, which in turn poses significant restrictions on the access to operating system resources.
7. **Apple TV** is the TV software created by Apple Inc. In the case of Apple TV, application distribution is based on the Apple Store concept. Convenient support is already available to develop applications for it, but there are also very significant restrictions when connecting external devices. Also, in accordance with the user requirements, in order to provide them with apps with a great interaction experience and a rich set of features, this technology was disregarded.

Android is the base operating system for AndroidTV and FireOS. As a consequence, they can benefit from a rich set of development applications and an active developer's community (in the case of FireOS applications are limited to those available through the Amazon App Store).

Another trend identified is the adoption of web-as-a-platform paradigm by WebOS, FirefoxOS and Tizen. As in the previous case, many tools are ready available to support development. This, together with the

[6] Tizen official site: https://www.tizen.org/

Web APIs offered by these systems to provide access to the internals of the operating system, supports the development of complete and rich interactive applications.

All platforms discussed, either Android-based or web-as-a-platform, offer development environments similar to those found in the smartphone world. Thus, support is provided to access peripheral devices that will enhance the user experience (e.g., webcams, voice control, gesture control, etc.). In some cases, the integration of external applications with direct access to native functionalities is also possible to access devices like webcams or microphones.

In the smart TV world, there are application areas that require a more direct control of the operating system and the TV itself. For example, interaction with some applications has to take place while watching a broadcast channel. While this is not a dramatic limitation in the smartphone world, it is a relevant feature in the case of interactive TV (e.g. contests / games applications based on a TV show).

2.2. Smart-TV devices

Smart-TV is a special type of television with an operating system which adds to the TV a set of typical computer characteristics. Most of the smart-TVs on the today's market have one of the operating systems reviewed in the previous section.

During the last years, each television manufacturer has bet for a particular operating system, even has supported its development economically. Therefore, there is a clear relationship between the TV brand and its software. Next table shows the environments of the main European television brands:

TV Brand	Operating system
LG	webOS
Panasonic	Firefox OS
SAMSUNG	TIZEN
SONY	androidtv

The main risk to buying a smart-TV is the market volatility, because it is evolving continuously and, there is not a clearly dominant technology or brand. Android is the operating system that presents more facilities to developing services for it. However, the best-selling television brands in Europe are Samsung and LG[7] which have their own software - Tizen and WebOS respectively.

Furthermore smart-TVs, there are other devices on the market that allow you to convert any television with an HDMI port, into a smart TV. Most of these devices have an Android operating system, but there are devices with Apple-TV or even Tizen. In general, these devices are cheaper than smart-TVs, therefore they are a great alternative to enjoy the advantages of smart TVs at a low cost.

Here are different software and hardware parameters to consider when selecting the device to covers the formal and informal service requirement. The formal services are the most demanding because they use special sensors to communicate with devices. The main important issues to review are:

- Operating systems.
- Openness.
- Easiness develops.
- Supports modify the start.
- In/out ports (Bluetooth, HDMI, Wi-Fi USB, etc.).
- Internal and SD storage.
- Memory size and processing capacity.
- External peripherals support (webcam, microphone, remote-control, etc.).
- Tv antenna.

The tables below present the general characteristics of the HDMI devices analysed during months M1 to M6.

Hardware characteristics										
Device name	**Memory GB**	**Storage GB**	**Wi-Fi**	**BT**	**HDMI**	**SD**	**USB**	**Webcam**	**External peripherals**	**RF input**
Chromebit CS10	2	16	yes	yes	yes	no	yes	yes	yes	no
Rikomagic v5	2	16	yes	yes	yes	yes	yes	yes	yes	no
Guleek A8	2	8	yes	no	yes	yes	yes	no	yes	no
Nvidia Shield	3	500	yes	yes	yes	yes	yes	yes	yes	no
Amazon Fire TV	2	8	yes	yes	yes	yes	yes	no	yes	no
Samsung HomeSync		1 TB	yes	yes	yes	no	yes	no	no	no
Apple TV		32/64	yes	yes	yes	no	yes	no	yes	no

[7] https://www.statista.com/statistics/267095/global-market-share-of-lcd-tv-manufacturers/

Nexus player	1	8	yes	yes	yes	no	yes	no	yes	no
Raspberry pi 3	1		yes	yes	yes	yes	yes	yes	yes	no
Google TV Box MINIX NEO	2	16	yes	yes	yes	yes	yes	yes	yes	no
MINIX Neo X8-H Plus	2	16	yes	yes	yes	yes	yes	yes	yes	no
Roku4	512Mb	0	yes		yes	yes	yes	no	no	no
Asus Cube v2			yes	no	yes	no	yes	no		no
Boxee Box	1		yes	no	yes	yes	yes	no		no
Udoo	1	SD	yes	yes	yes	yes	yes	yes		no

Software characteristics									
Device name	**OS**	**Cordova**	**SDK**	**app store**	**3rd party libs**	**Openness**	**Easy dev**	**Community**	**Modify start**
Chromebit CS10	Chrome OS	yes	yes	yes	no	yes	yes	no	no
Rikomagic v5	Android, Ubuntu	yes	yes	yes	no	yes	yes	no	yes
Guleek A8	Android	yes	yes	yes	no	yes	yes	no	no
Nvidia Shield	Android TV	yes	yes	yes	yes	yes	yes	yes	no
Amazon Fire TV	Fire OS	yes	yes	yes	yes	yes	yes	yes	yes
Samsung HomeSync	Android	yes	yes	yes	no	yes	yes	no	no
Apple TV	tvOS	no	yes	yes	yes	no	no	yes	no
Nexus player	Android	yes	yes	yes	no	yes	yes	yes	yes
Raspberry pi 3	Windows, Linux	yes/no	no	yes	yes	no/yes	yes	yes	yes
Google TV Box MINIX NEO	Android	yes	yes	yes	yes	yes	yes	yes	yes
MINIX Neo X8-H Plus	Android	yes	yes	yes	yes	yes	yes	yes	yes
Roku4	RokuOS	no			yes	no	yes	yes	no
Asus Cube v2	Android TV	yes	yes	yes	no	yes	yes	no	no

Boxee Box	Linux	yes	no	no	yes	yes	yes	yes	yes
Udoo	Android, Linux, Arduino	yes	yes	yes		yes	yes		yes

After the market analysis, we have decided to acquire HDMI devices with Android operating system for first piloting cycle. This decision was founded in three main reasons:

- They are low-cost devices whose software can be upgraded. This property reduces the risk of investing a large amount of money in devices that could be obsolete at the end of the project.
- Android supports applications developed using standard web technologies (see section 3) which allows reusing not only the code but also the development experience for other operating systems and devices.
- It is possible to modify the Android boot to run apps when devices start.

2.3 Platform interaction

Elderly people will be the main users of SENIOR-TV services. The majority of time, they will access the platform through smart-TV. However, certain applications will allow access to the platform using other smart media such as mobile or tablets. However, secondary users (relatives and caregivers) will normally use their smartphones or tablets access to SENIOR-TV (to communicate with the elders or to access any other particular services). The next diagram shows the principal users of SENIOR-TV and how each kind of user will access to the platform.

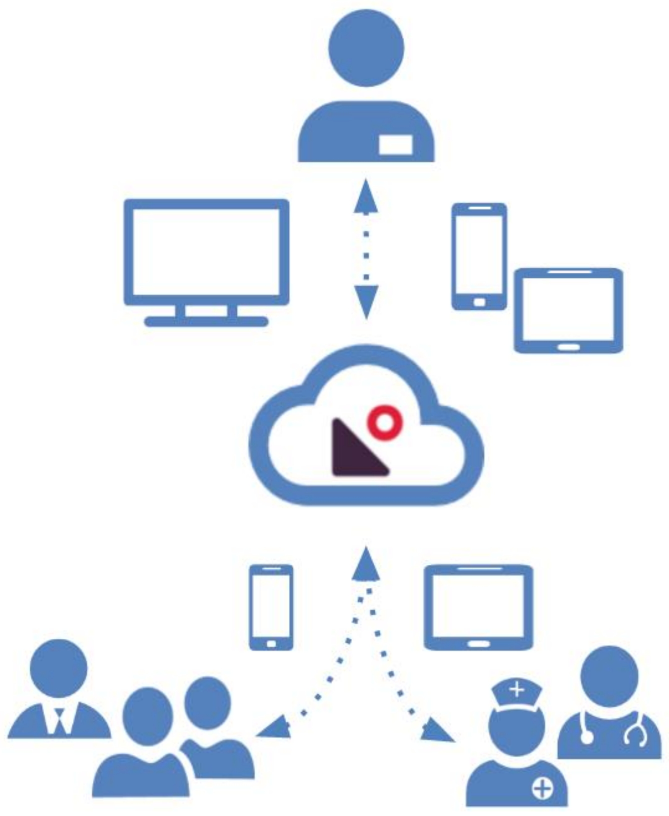


FIGURE 1. PLATFORM INTERACTION

3. Development technologies

This section includes a general overview of the different technologies used to develop the initial version of the services includes in SENIOR-TV V1.

As we have seen in the previous section, there are many different technologies in the smart-TV market. Because of that, in this first version, we have decided to use technologies which allow to create different versions of the apps for several operating system with reduced effort.

The following subsections review the characteristics of the different approaches to implement smart-TV apps and, show the characteristics of the selected technologies for this initial version. They also include a detailed description of the implemented services:

- Ionic and Apache Cordova to develop Weather, News and Events services.
- Unity to implement Attentix and Episodix games.

3.1. Smart-TV development paradigms

The Smart-TV market is changing every day. As we have seen in section 2, there is a great variety of Smart-TV technologies which are progressing continuously. Therefore, there is not a unique approach to develop apps for Smart-TV and, the current approaches are based in to reused or extend the mobile app development. This is a valid approach because, as we have seen, the Smart-TV operating systems are almost the same of smartphone software: Android, iOS, Tizen, etc. with some special characteristic oriented to run in televisions.

There are three typical approaches to implement mobile apps and we can use all of these approaches to develop Smart-TV apps:

1. **Native apps**: are developed in a programming language native to the device and operating system and require one specific app to be created for one target platform.
2. **Cross-platform apps**: use an intermediate language that is not native to the device's operating system, such as JavaScript, to implement the apps. This allows share some code across different platforms – for instance, across iOS and Android.
3. **Hybrid web apps**: are cross-platform apps which rendering the user interface using an embedded web browser and, which are developed using well-known technologies - HTML, CSS and JavaScript.

The choice of one of these options depending on different aspects such us the performance and hardware requirements, supporting devices and operating systems, available resources or economic restrictions.

1. **Native** apps are appropriate if you need a high level performance and speed, for instance graphics intensive apps like games. Or to use advanced features offered by the device and operating system (data storage, memory access or complex networking).
   However, this approach is not indicated to support multiple devices, because needs to create different versions for each operating system. So, it requires a bigger investment for development, maintenance and testing.
2. **Cross-platform** is indicated for create apps which must run in different platforms and have high interface performance restrictions. Reduces the development time because the code can be shared between different versions of the apps across devices. However, delegates the access to the device and operating system features to the framework, so, in the end, the performance could be reduced.
3. **HTML5 Hybrid** is the best choice when apps have to run on different devices and operating systems. The base of this approach is the use of basic web technologies such as HTML, CSS and JavaScript, but you can use more advanced libraries such as AngularJS or Bootstrap. Therefore, this paradigm allows us to create advanced interfaces with little effort. In addition, you can reuse most of the code, therefore the development time is greatly reduced.
   However, as in the case of Cross-platform, the access to the device and operating system is delegating to the framework, which reduces the response time. So, it is not recommended for implementing services which need a high level of performance.

The great variety of devices and operating systems presents in smart-TV market makes necessary using technologies which cover as many brands as possible. Neither HTML5 Hybrid nor Cross-platform approaches cover all the operating systems in the market. However, we decided to use these technologies because they allow us to achieve a greater number of systems.

We have used some hybrid technologies like Cordova and Ionic to develop the Weather, News and Events services and, Unity - a cross-platform technology - to implement the Attentix and Episodix games.

3.2. Apache Cordova

Apache Cordova[8] is an open-source mobile development framework created by Nitobi. Originally called PhoneGap it was bought by Adobe Systems in 2011, and later released as an open source version called Cordova. Contributors to the Apache Cordova project include, among others, Mozilla, BlackBerry, Google, IBM, Intel and, Microsoft.

[8] Apache Cordova official site: https://cordova.apache.org/

Apache Cordova allows software programmers to use standard web technologies (HTML5, CSS3, and JavaScript) for cross-platform development. Instead of relying on platform-specific APIs like those in Android, iOS, etc. The resulting applications are hybrid because use webviews to render the layouts and they are packaged as apps for distribution and have access to native device APIs.

Applications execute within wrappers targeted to each platform, and rely on standards-compliant API bindings to access each device's capabilities such as accelerometer, camera, compass, file system, microphone, network status, etc.

The following diagram shows a high-level view of an Apache Cordova application architecture:

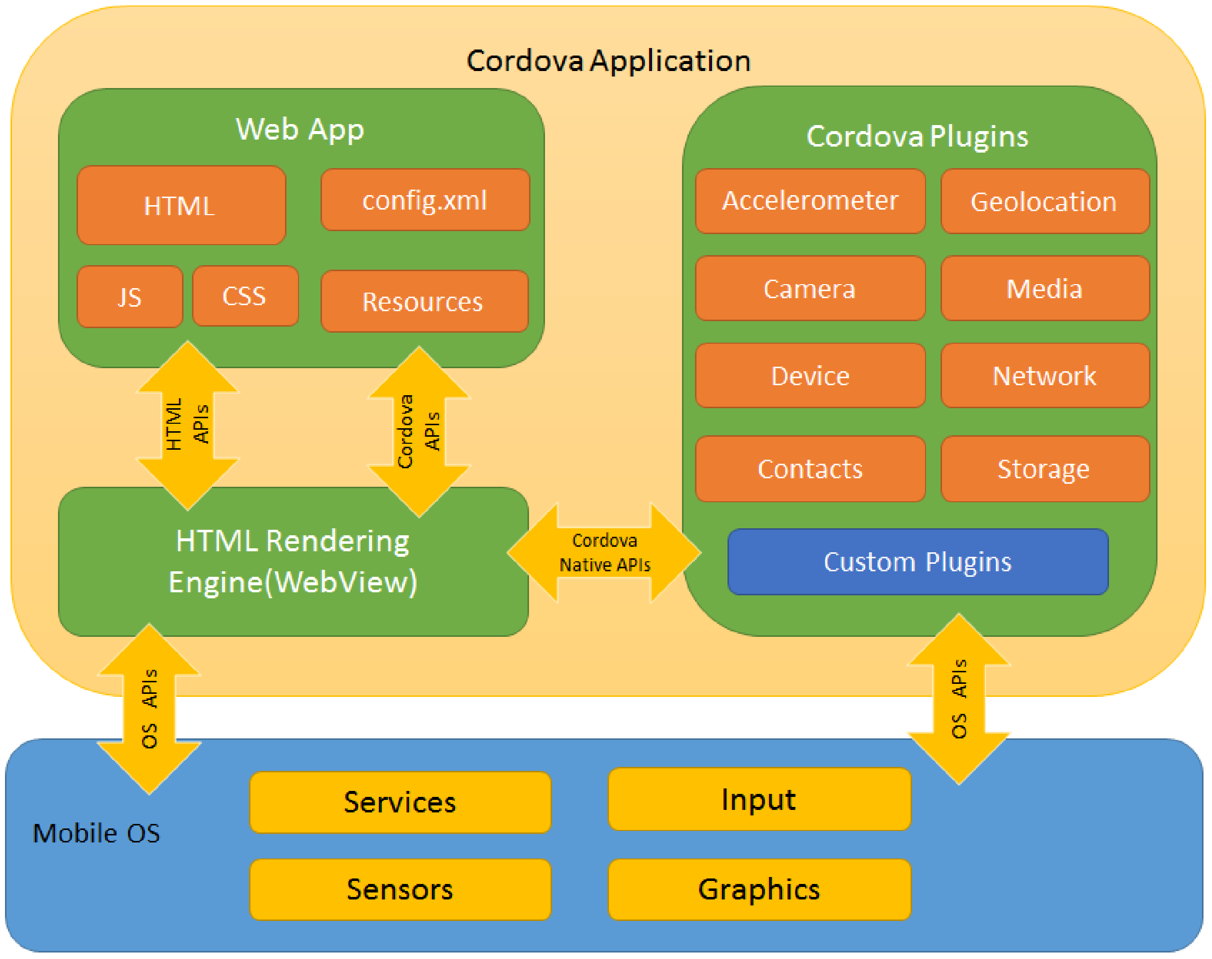


FIGURE 2. APACHE CORDOVA ARCHITECTURE

These are the Cordova architecture main components:

- Web App: the application code. Uses CSS3 and HTML5 to create the user interfaces and JavaScript implement the application logic.
- WebView: minimal browser that delivers web content and rendering the apps. HTML5 provides access to underlying hardware such as sensors, data or GPS. However, browsers' support is not

consistent across mobile browsers of the oldest versions of operating systems. To overcome these limitations, Apache Cordova embeds the HTML5 code inside a native webview on the device.

- Plugins: provide an interface for Cordova and native components to communicate with each other, allowing invoke native code from JavaScript. Apache Cordova allows developers to extend its functionalities using native plugins that can be called from JavaScript[9].

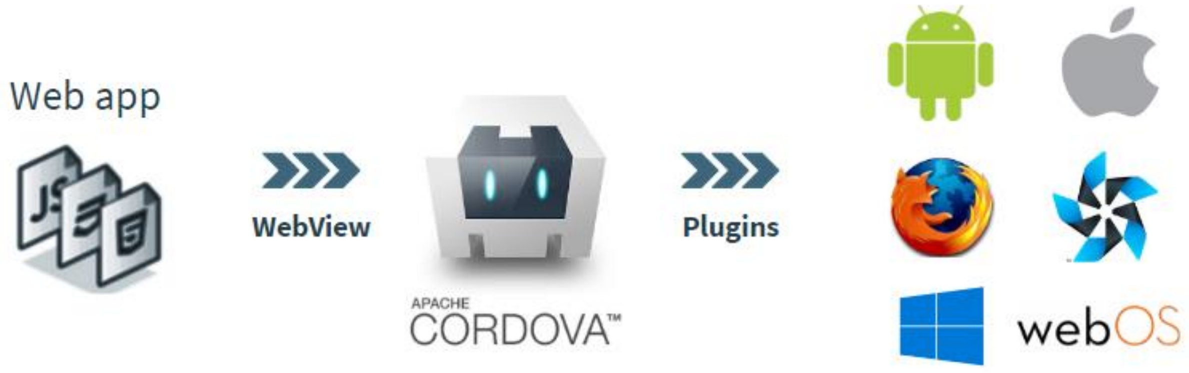


FIGURE 3. APACHE CORDOVA PLUGINS

The supported platforms by Cordova have been changing over the last few years. Previous versions have supported Android, iOS, Samsung Tizen, WebOS, Firefox OS, etc. However, only Android, Blackberry, iOS, OS X, Ubuntu and Windows 8 and 10 are supported in the last version. These limitations are not a deep problem, because Firefox OS is now discontinued. Also, Tizen has similar Cordova tools to develop hybrid apps. So, we can reuse the HTML5, CSS and most of JavaScript code to create Tizen apps.

However, the new Apple tvOS doesn't use webviews, so it is not possible develop apps using web resources, only native code is allowed. Therefore, this could be a potential problem to analyse in future versions of SENIOR-TV platform.

3.3. Ionic Framework

Ionic[10] is a powerful HTML5 SDK that is focused mainly on the look and feel, and UI app interaction. It helps you build native-feeling mobile apps using web technologies like HTML, CSS, and JavaScript. It doesn't replace Cordova but fits in well with Cordova projects in order to simplify one big part of the app: the front-end.

Ionic was created by Drifty Co. in 2013. As Ionic is based on Angular, currently exists two versions of the framework in parallel: a stable version upon first generation of AngularJS (first release released in May

[9] Cordova Plugin development Guide: https://cordova.apache.org/docs/en/latest/guide/hybrid/plugins/
[10] Ionic official site: https://ionic.io/

2015) and, a beta version based on Angular 2 (a new version that has not been released as stable until a few months ago).

Ionic includes mobile components, typography, interactive paradigms, and an extensible base theme. It also provides Angular custom components such as collection repeat, scroll-view, etc. Users can build their apps and customize for their favourite operating system and then deploy the apps using Cordova.

The last version supports several systems like Android 4.1 and up, iOS 7 and up, Windows 10 and Blackberry 10.

3.4. Unity

Unity 3D[11] engine is a tool to create games, applications and 3D animations. It is a multiplatform engine that allows you to deploy across major mobile, desktop, console, and TV platforms plus the Web.

The Unity editor is a complex visual editor to develop games. A game in Unity is structured in scenes that can be any part of the game, from a start menu to a level of your game or an area of al level. Unity engine also includes a terrain editor where you can create the terrain of your game using visual tools, painting, etc.

In Unity editor you can create your own objects or you can import them from different 3D platforms like Blender, 3ds Max, SketchUp, Maya and many more. Unity can read 3D formats like ".FBX", ".OBJ" or ".dae" files so you can import 3D objects from any platforms that can export to these formats. Unity can also read proprietary 3D application files like ".Max", ".Blend" or ".skp".

The behaviour of a Unity game object is defined by his scripts -one or more- that you can develop in JavaScript or C#. All the physics and many other behaviours are also provided from the Unity engine.

Although you can also create your own methods and classes all Unity scripts derive from the base class "MonoBehavior". This class provides some facilities to interact objects with unity engine including same methods to initialize whatever you want even before the scene of your game starts. The behaviour of objects is mainly based on frames, that is, the script is continuously repeated frame by frame. Same methods are called every frame or even more per frame. After initialization you can control physics, input events, game logic or even the rendering of the objects.

Unity 3D is a 3D game engine officially launched on June 2005. This engine allows to create games and other interactive content or 3D animations in real time. Many people interested in development find the difficulty of learning programming languages and the engines that use them. The creation of video games become very difficult without programming studies or computer animation skills.

[11] Unity 3D official site: https://unity3d.com/

Unity Technologies is one of the companies that has decided to change this situation. The Unity development team has decided to maintain the source code but offering the user a complete graphical interface so that users can control the source code without creating new elements in the code. This is what has made Unity so popular for video game developers.

Unity is a 3D real time and multimedia application besides being a 3D and physics engine to create games, animations in real time with audio, video and 3d objects content.

This engine does not allow complex modelling but you can create scenes with illumination, terrains, cameras, textures, etc. Unity allows to deploy video games to Windows, Mac OS, iOS, Android, Wii, PlayStation, Xbox 360, Nintendo, Web and same TV platforms.

Unity has several advantages that make it one of the most used video game engines at this moment. Some of these advantages are:

1. Allows to import several 3D formats like 3ds Max, Maya, Blender, SketchUp, FBX, etc. and you also can import some resources like Photoshop textures, PNG, TIFF, audios and videos.
2. It is compatible with graphic Direct3D, OpenGL and Wii graphic APIs. It is also compatible with QuickTime and use internally Ogg Vorbis format.
3. In Unity, the game is built using the editor and a scripting language so user does not have to be a programming expert to create a video game. Scripts can be created in JavaScript or in C# language.
4. The Unity game structure is defined by scenes that represent some part of the game.
5. Includes a terrain editor that allows you to create a terrain from the beginning, making his geometry, texture and including 3D elements that you can import from other 3D applications or elements already predefined in Unity.
6. There are several Unity versions but the simplest is free for personal use and let you create commercial games till you get $100k per fiscal year.
7. As we said Unity is multiplatform, so you can easily deploy on multiple mobile platforms, desktop, etc.

4. Informal services

The following subsections show the main characteristics and the informal services state after the development cycles (months 1 to 25).

4.1. First pilot

The platform obtained after the first cycle only includes services which run in the smart-TV devices, as the following chart is showing:

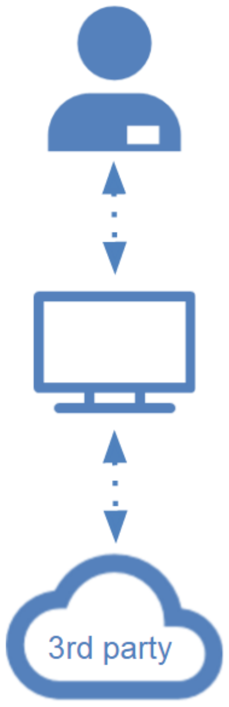


FIGURE 4. CURRENT APPS INTERACTION

The aim of the first version of SENIOR-TV was developed the initial version of a set of informal services to be tested during the first cycle of WP3 pilots by seniors from the three pilot countries (Cyprus, Slovenia and Romania).

This first version does not include complete functional services, but it includes initial versions like clickable mock-ups, which are going to improve in next versions using the feedback obtained from seniors after the pilots. They will evaluate issues such as the look and feel colours, the type and size of fonts, usability, the interaction devices, the contents, etc.

The WP1 workshops provided valuable input from the users, based on which we selected the services to be implemented in the first iteration. According to those results, the services most demanded by the elderly, regardless of their sex and age, are: the weather, the news and events, apart from games and entertainment services in general.

4.1.1. Events 1.0

The scope of this service is to enable elderly users to find and participate to events that they are interested in by just selecting an event on the TV screen.

The app is also localized, meaning that it will only show events accessible to the elderly user and in his own language. During the writing of this report we are also investigating the possibility of enabling users to participate to online events.

The following chart shows the architecture of the Events service:

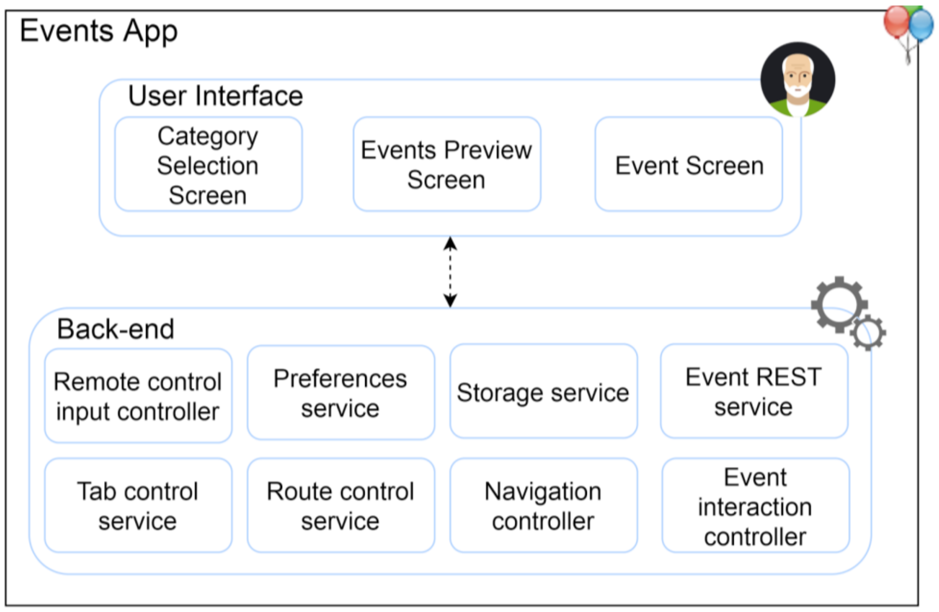


FIGURE 5. EVENTS ARCHITECTURE

The Back-end of the Events service has the following components:

1. Remote-control input controller: is the service that handles the user input and transform the input into actions inside the app.
2. Preferences service: is responsible for managing the preferences of the user (e.g. language).
3. Storage service: is responsible for storing information, locally on the device, about the interaction with the app and also about each session the user has with the app.
4. Tab control service: is responsible for managing the switching between different types of news categories.
5. Route control service and Navigation Controller: allow the user to navigate from one page of the app to another page of the app with ease.

6. Event Interaction Controller: is responsible for enabling the user to interact with a specific event. The user will be able to send a notification for Participation to the Event Organisers.
7. Event REST service will be responsible for fetching and parsing different event metadata (e.g. location, date, availability, price, etc.).

The main screens of the TV interface are as follows:

1. The Category Selection Screen

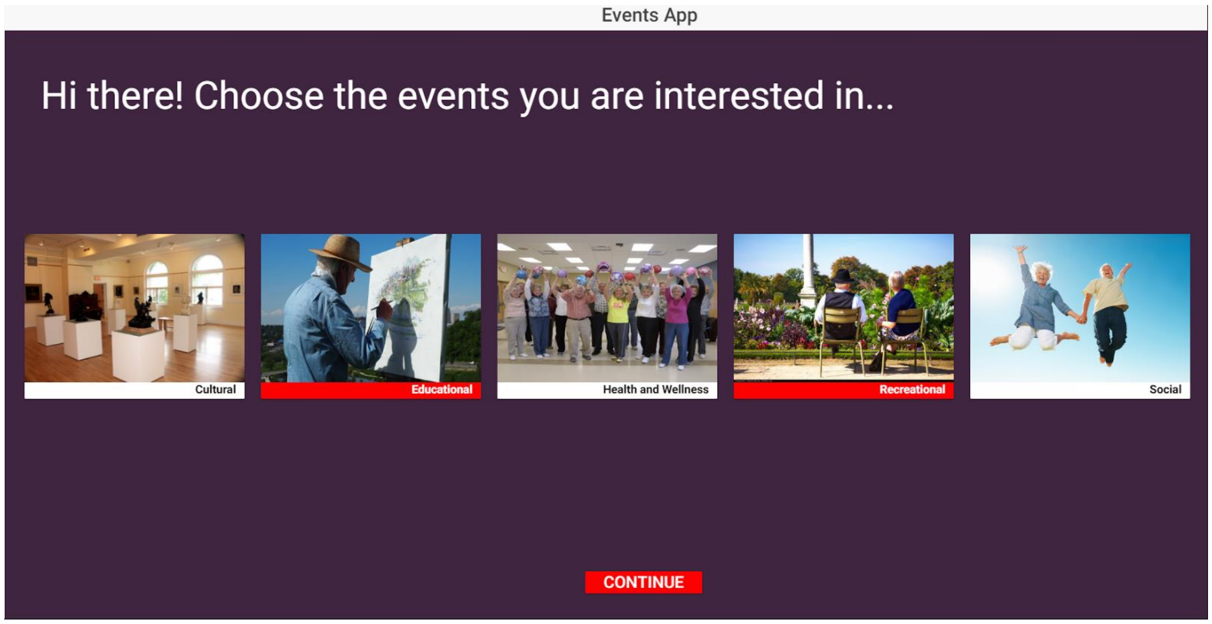
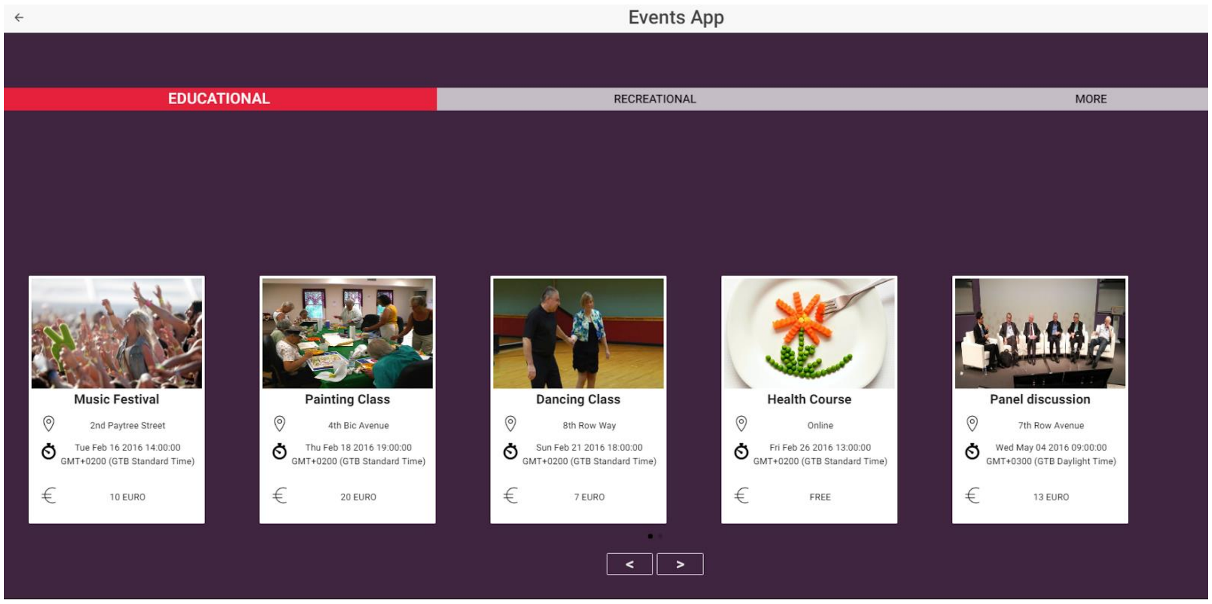


FIGURE 6. EVENTS APP SELECTION SCREEN

2. The Events Preview Screen



Music Festival
2nd Paytree Street
Tue Feb 16 2016 14:00:00 GMT+0200 (GTB Standard Time)
10 EURO

Painting Class
4th Bic Avenue
Thu Feb 18 2016 19:00:00 GMT+0200 (GTB Standard Time)
20 EURO

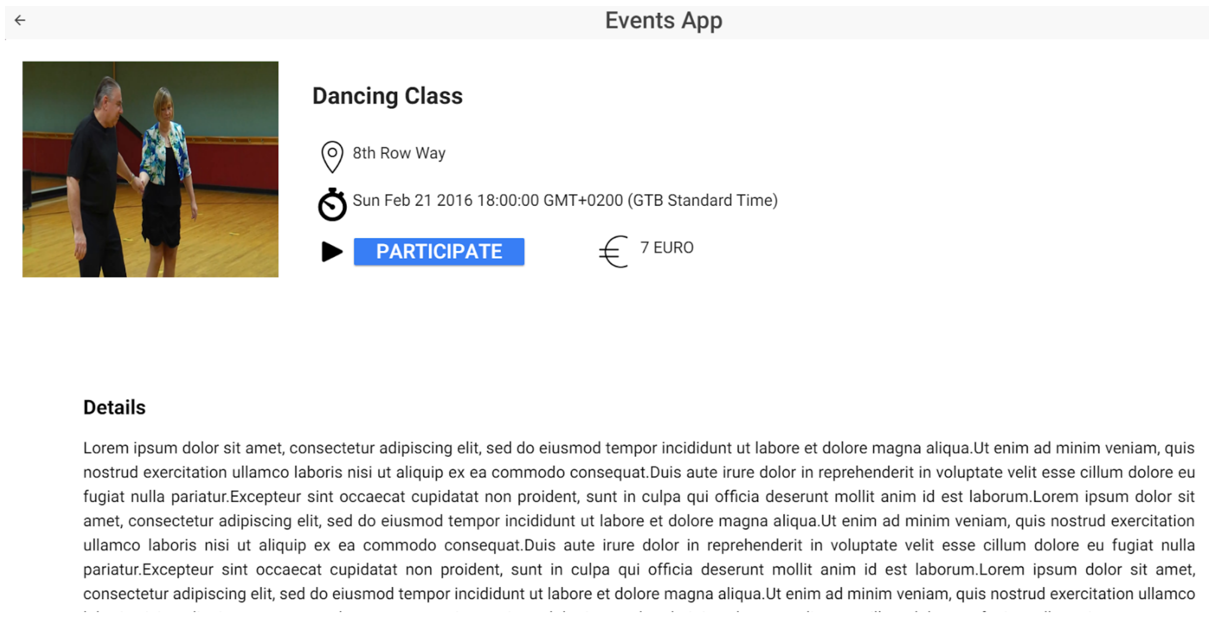
Dancing Class
8th Row Way
Sun Feb 21 2016 18:00:00 GMT+0200 (GTB Standard Time)
7 EURO

Health Course
Online
Fri Feb 26 2016 13:00:00 GMT+0200 (GTB Standard Time)
FREE

Panel discussion
7th Row Avenue
Wed May 04 2016 09:00:00 GMT+0300 (GTB Daylight Time)
13 EURO

FIGURE 7. EVENTS APP PREVIEW SCREEN

3. The Events View Screen


Events App

Dancing Class

8th Row Way

Sun Feb 21 2016 18:00:00 GMT+0200 (GTB Standard Time)

PARTICIPATE

7 EURO

Details

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur.Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur.Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.Ut enim ad minim veniam, quis nostrud exercitation ullamco

FIGURE 8. EVENTS VIEW SCREEN

4.1.2. News 1.0

The News service scope is to facilitate the viewing of news in an easily manner on TV, while also taking into consideration personal preferences and interests. The elderly user is able to choose which category of news he wants to follow and which news to read.

The app runs directly on the Set Top Box and fetches the News from different RSS News Websites. The app and news feeds are localized, meaning that the users from different countries see the news from their own countries and in their own language.

The next diagram shows the architecture of the News app:

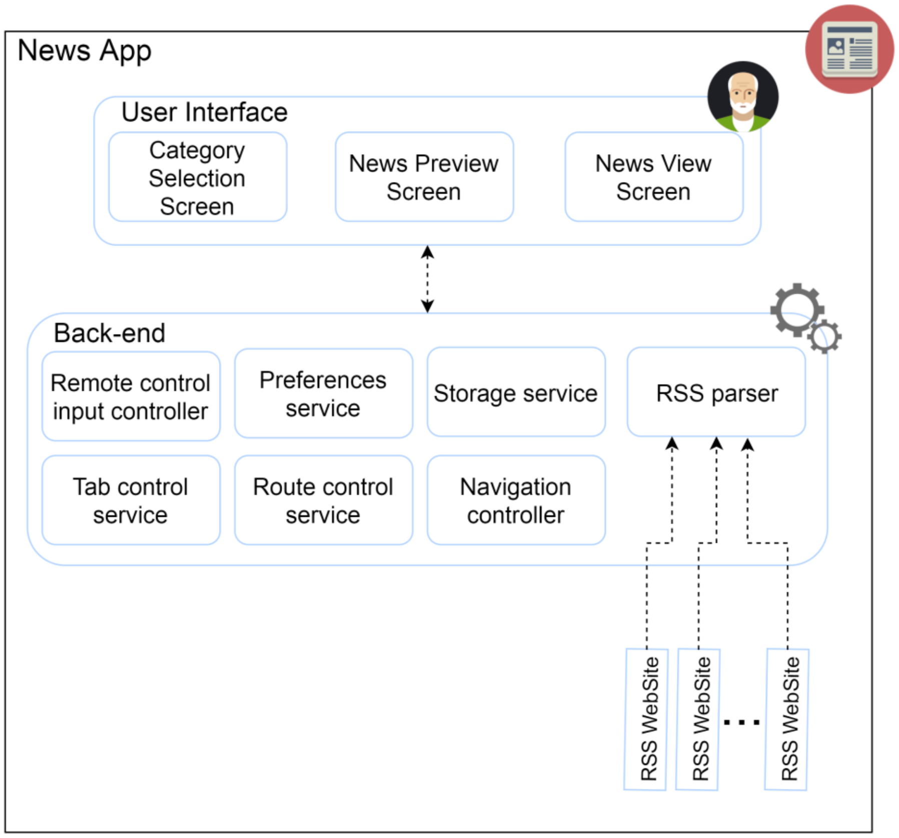


FIGURE 9. NEWS ARCHITECTURE

The back-end of the News app has the following components:

1. Remote-control input controller: is the service that handles the user input and transform the input into actions inside the app.
2. Preferences service: is responsible for managing the preferences of the user (e.g. language).
3. Storage service: is responsible for storing information, locally on the device, about the interaction with the app and also about each session the user has with the app.
4. RSS parser: is in charge of fetching news from RSS feeds and parsing them for presentation on the user interface.
5. Tab control service: is responsible for managing the switching between different types of news categories.
6. Route control service and Navigation Controller: allow the user to navigate from one page of the app to another page of the app with ease.

The application has one screen for viewing the news information and two main screens for browsing the news and the preferences:

1. News Central Screen

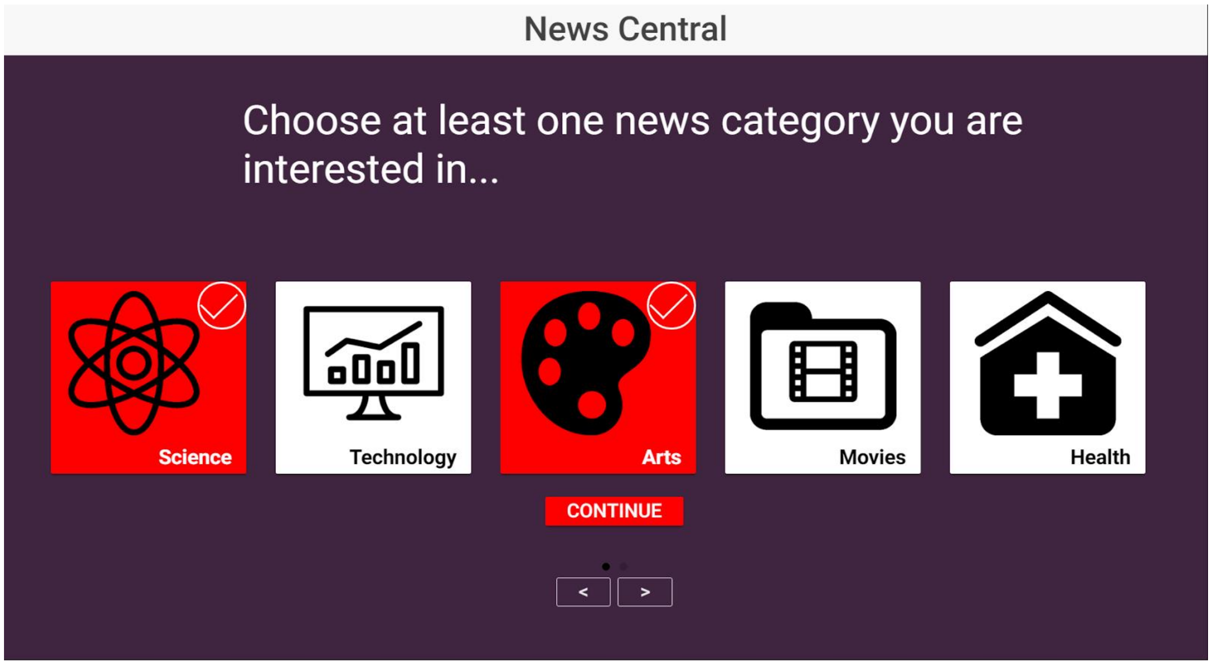
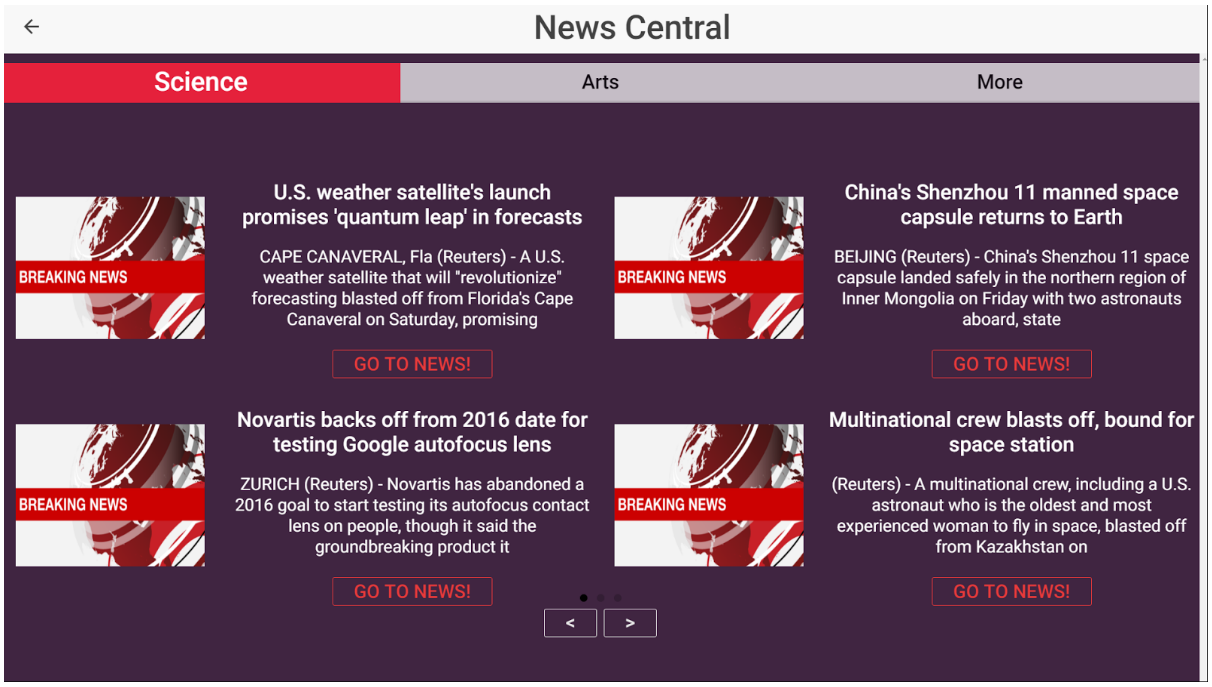


FIGURE 10. NEWS APP CENTRAL SCREEN

2. Category List News Screen



U.S. weather satellite's launch promises 'quantum leap' in forecasts

CAPE CANAVERAL, Fla (Reuters) - A U.S. weather satellite that will "revolutionize" forecasting blasted off from Florida's Cape Canaveral on Saturday, promising

China's Shenzhou 11 manned space capsule returns to Earth

BEIJING (Reuters) - China's Shenzhou 11 space capsule landed safely in the northern region of Inner Mongolia on Friday with two astronauts aboard, state

Novartis backs off from 2016 date for testing Google autofocus lens

ZURICH (Reuters) - Novartis has abandoned a 2016 goal to start testing its autofocus contact lens on people, though it said the groundbreaking product it

Multinational crew blasts off, bound for space station

(Reuters) - A multinational crew, including a U.S. astronaut who is the oldest and most experienced woman to fly in space, blasted off from Kazakhstan on

FIGURE 11. CATEGORY LIST NEWS SCREEN

4.1.3. Weather 1.0

The Weather service allows seniors to check the weather conditions for their village or city. The elders don't need to select their location, the app is able to get the users' position and ask the forecast automatically.

The app runs directly on the Set Top Box and fetches the weather data via HTTP requests to an open weather API free service, called *OpenWeatherMap*[12]. The Weather app are localized, meaning that the users from different countries see the information in their own language.

The next chart shows the architecture of the Weather service:

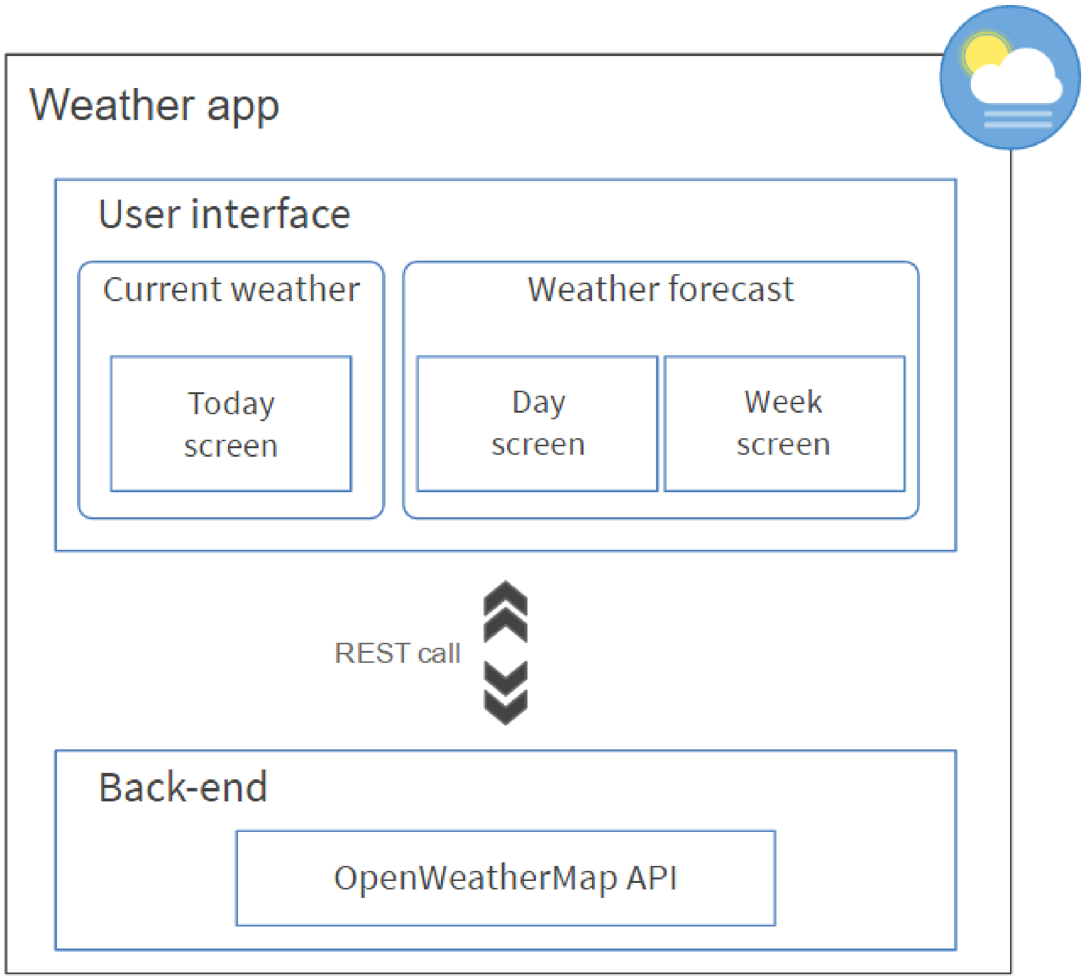


FIGURE 12. WHEATEAR APP ARCHITECTURE

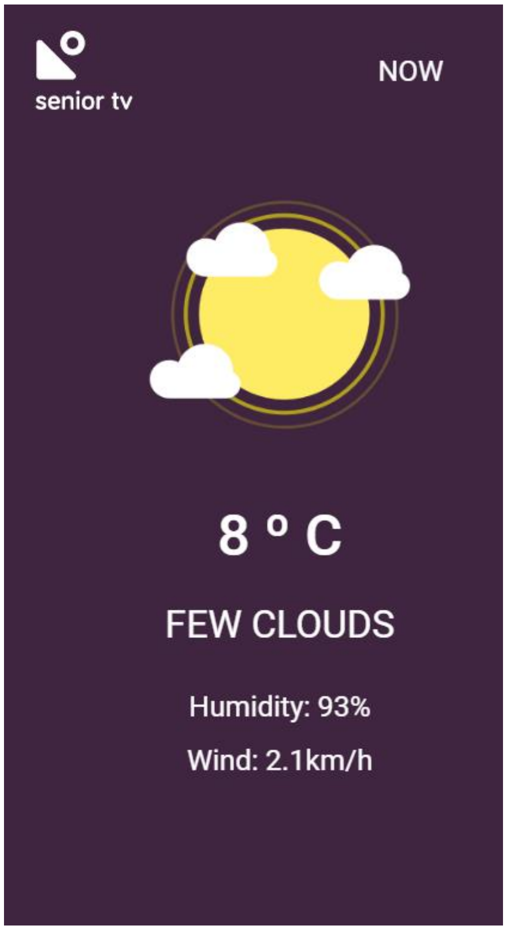
The user interface is divided in two sections:

1. Left side shows the current weather conditions.
2. Right side shows the forecast for the current day or for the next three days.

[12] OpenWeatherMap official site: https://openweathermap.org/

1. Current weather screen

Shows the temperature, humidity and wind forecast for the senior location, in this moment.



8 ° C

FEW CLOUDS

Humidity: 93%

Wind: 2.1km/h

FIGURE 13. WEATHER APP CURRENT SCREEN

2. Weather forecast screen

Shows the weather forecast for the current day, or for the next three days if we choose so. The user can navigate between these options using the top of the screen buttons: “Day” and “Week”.

The user can switch the screens pressing the corresponding button or using the right and left key arrows of the remote-control.

3. Day screen

In the case of the current day state, the information shown consists in a line chart with the temperature forecast for the next hours.

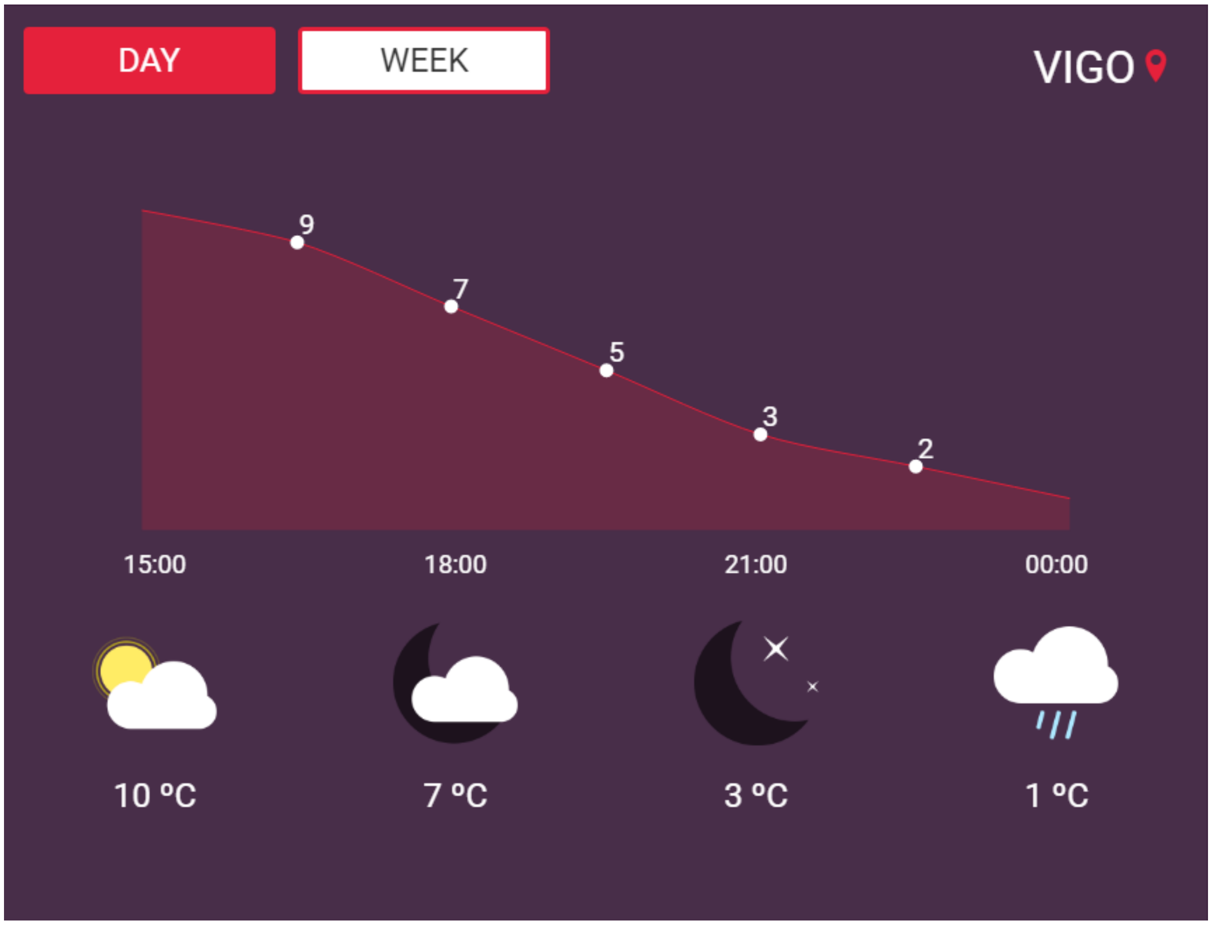
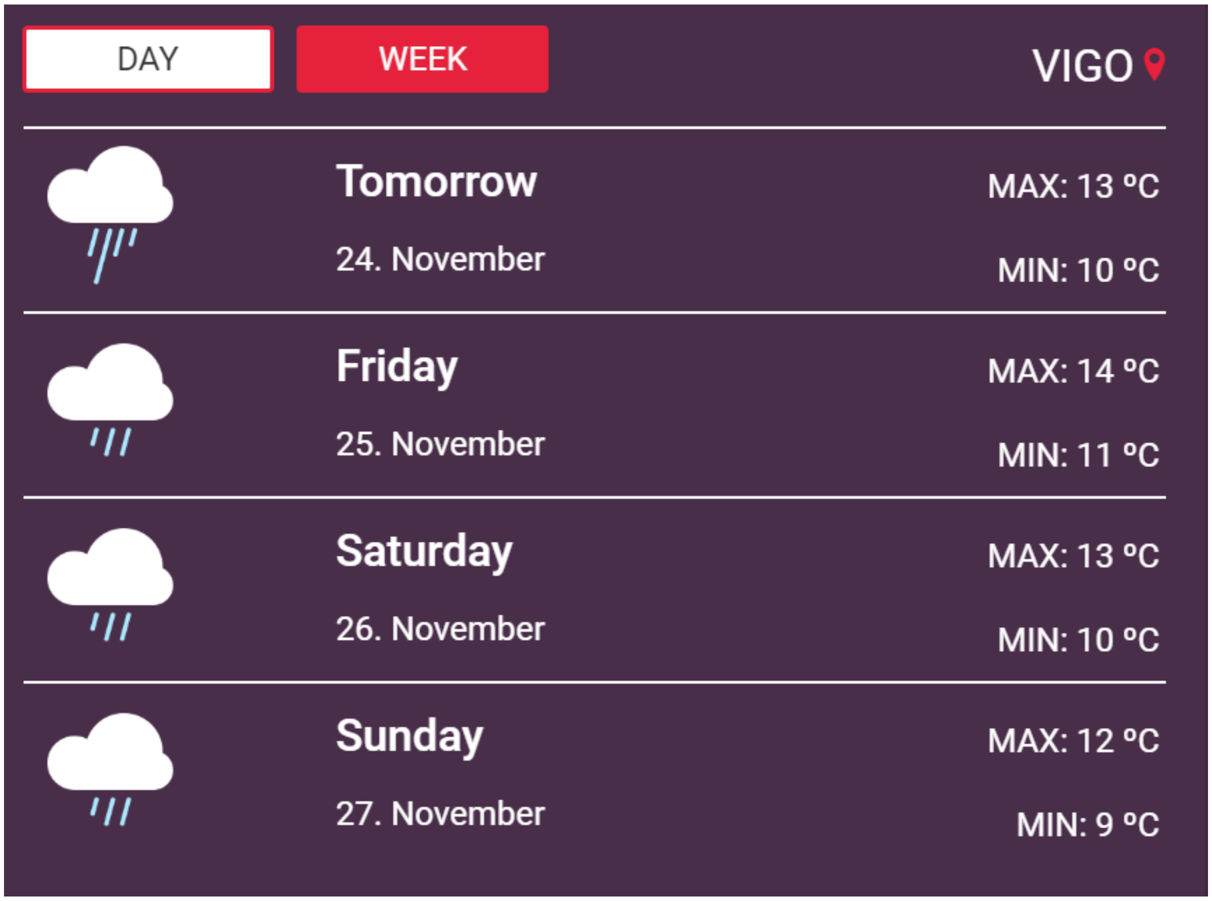


FIGURE 14. WEATHER APP DAY SCREEN

4. Week screen

Instead, for the week state the application shows the weather forecast and the max and min temperatures forecast.


DAY WEEK VIGO

Tomorrow	24. November	MAX: 13 ºC MIN: 10 ºC
Friday	25. November	MAX: 14 ºC MIN: 11 ºC
Saturday	26. November	MAX: 13 ºC MIN: 10 ºC
Sunday	27. November	MAX: 12 ºC MIN: 9 ºC

FIGURE 15. WEATHER APP WEEK SCREEN

4.1.4. Attentix

Attentix[13] is a game for improving memory skills. The game creates sequence of lights and asks the user to repeat it. If user do well, the game will increase number of lights in the sequence one by one. After that, asks a new sequence to the user. At the end, when the series generated. Users can always reach their maximum skill level in the game. You can select different levels. Each level adds new colour box in the screen. The next diagram shows the architecture of the Attentix game:

[13] The Attentix game is co-financed within the SENIOR-TV project.

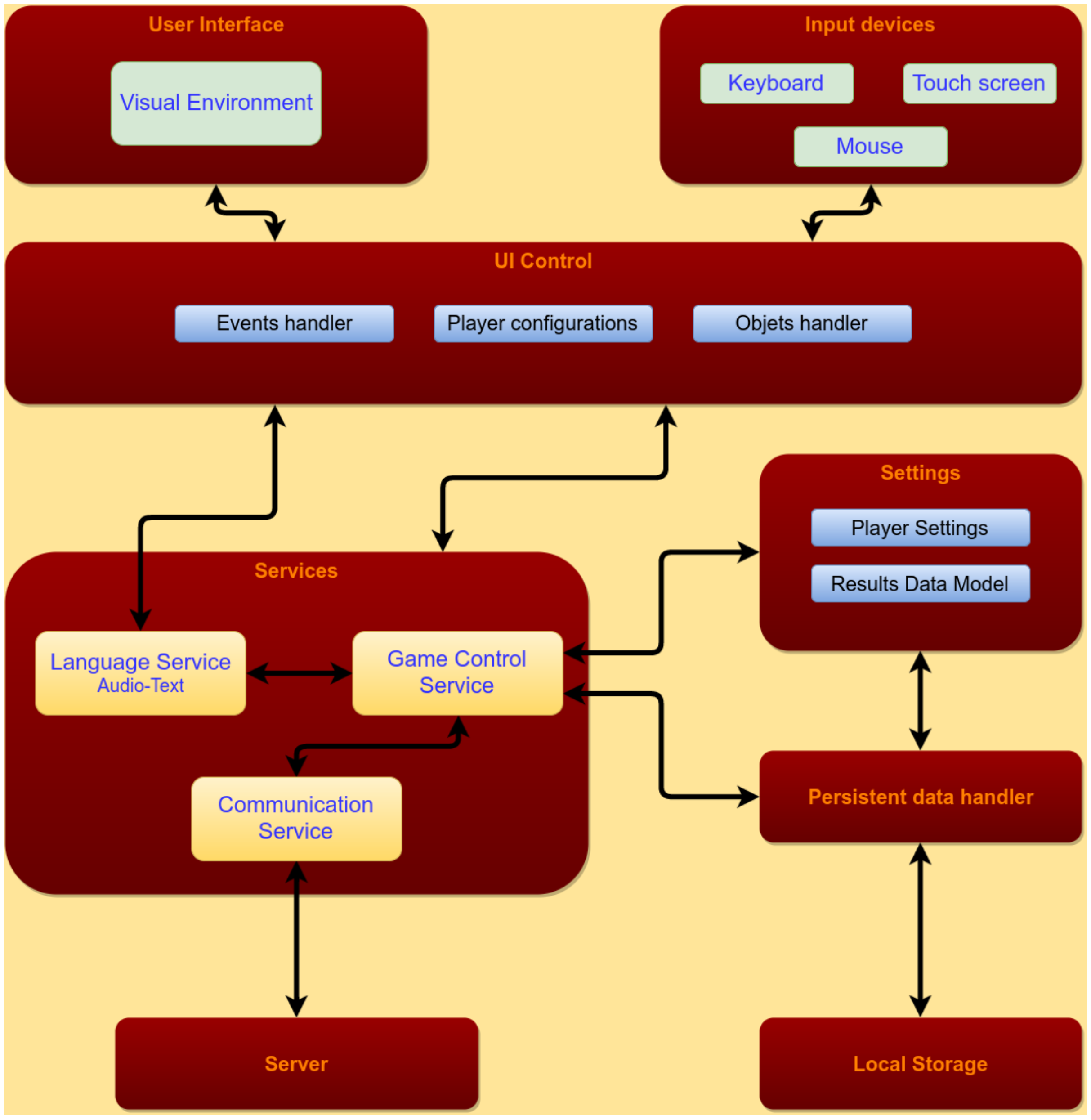


FIGURE 16. ATTENTIX GAME ARCHITECTURE

The previous chart shows the modules of the game and the relationships between them.

1. The "Game Control Service" module is the core of the game. It is used by other services or controllers to get current game information or to set it. "Game Control Service" has some different objectives like read current player setting to set the language or to prepare the results to send to the server.
2. The "User interface" block are those parts of the game responsible for the rendering of the game objects. This block is connected to the "Game Control Service" through the "UI Control" block to get object's positions for example.

3. The “UI Control” is used to interact between user interface and the input. Also, it is responsible for connecting with the “Language Service” to get the current translation for the words or sentences to show on screen. This block has to create the sequence to represent and control if is correctly pressed.
4. The “Settings” block is to represent the data model to save player setting or how to send results to the server.
5. The “Communication Service” is responsible for sending the game results to the server.
6. The “Persistent data handler” is the block responsible for storing results or player settings into the device. The data is saved in a persistent data folder with the package name.

The seniors have to use the pointer to interact with the game, clicking different squares on the screen. The next image shows the main screen of Attentix game:

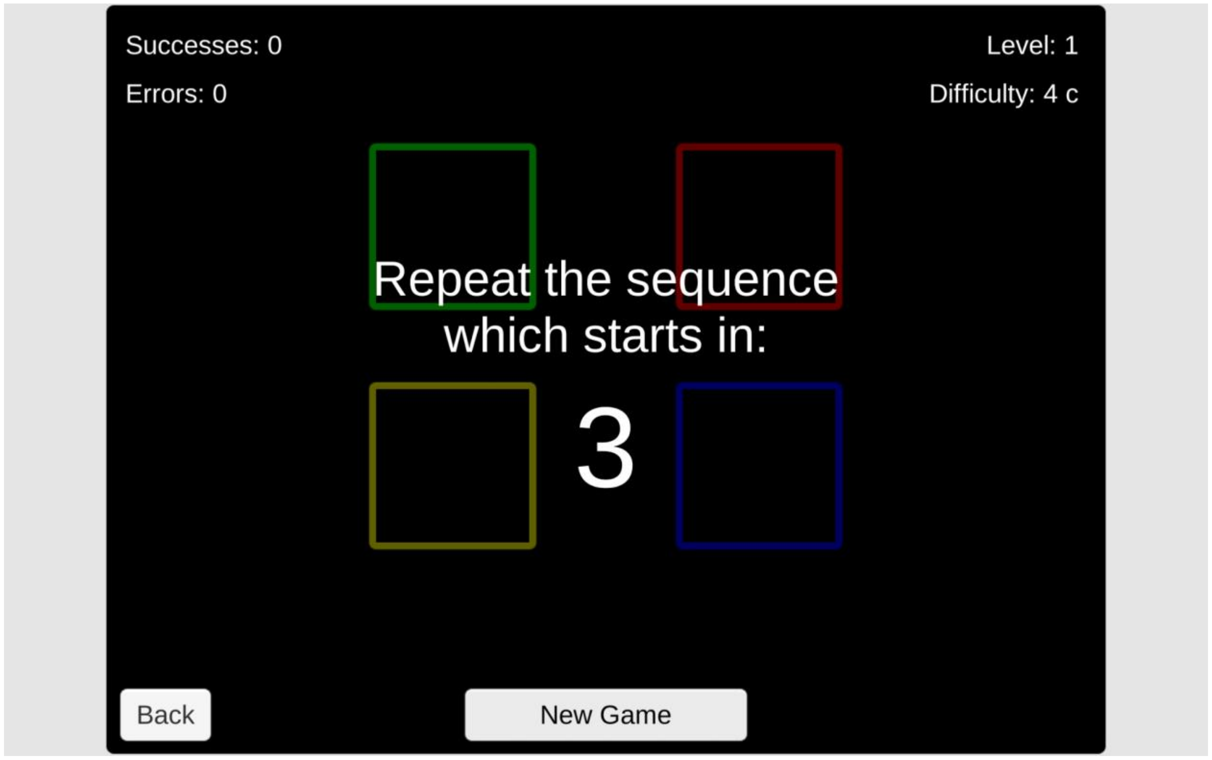


FIGURE 17. ATTENTIX GAME SCREEN

4.1.5. Episodix

Playing to Episodix[14] game, seniors can see a virtual city and walk around their streets. During their walk, some objects could appear and the senior have to remember which objects have appeared and which not. This game stimulates the elders' memory. The next chart shows the architecture of the Episodix game:

[14] The Episodix game is co-financed within the SENIOR-TV project.

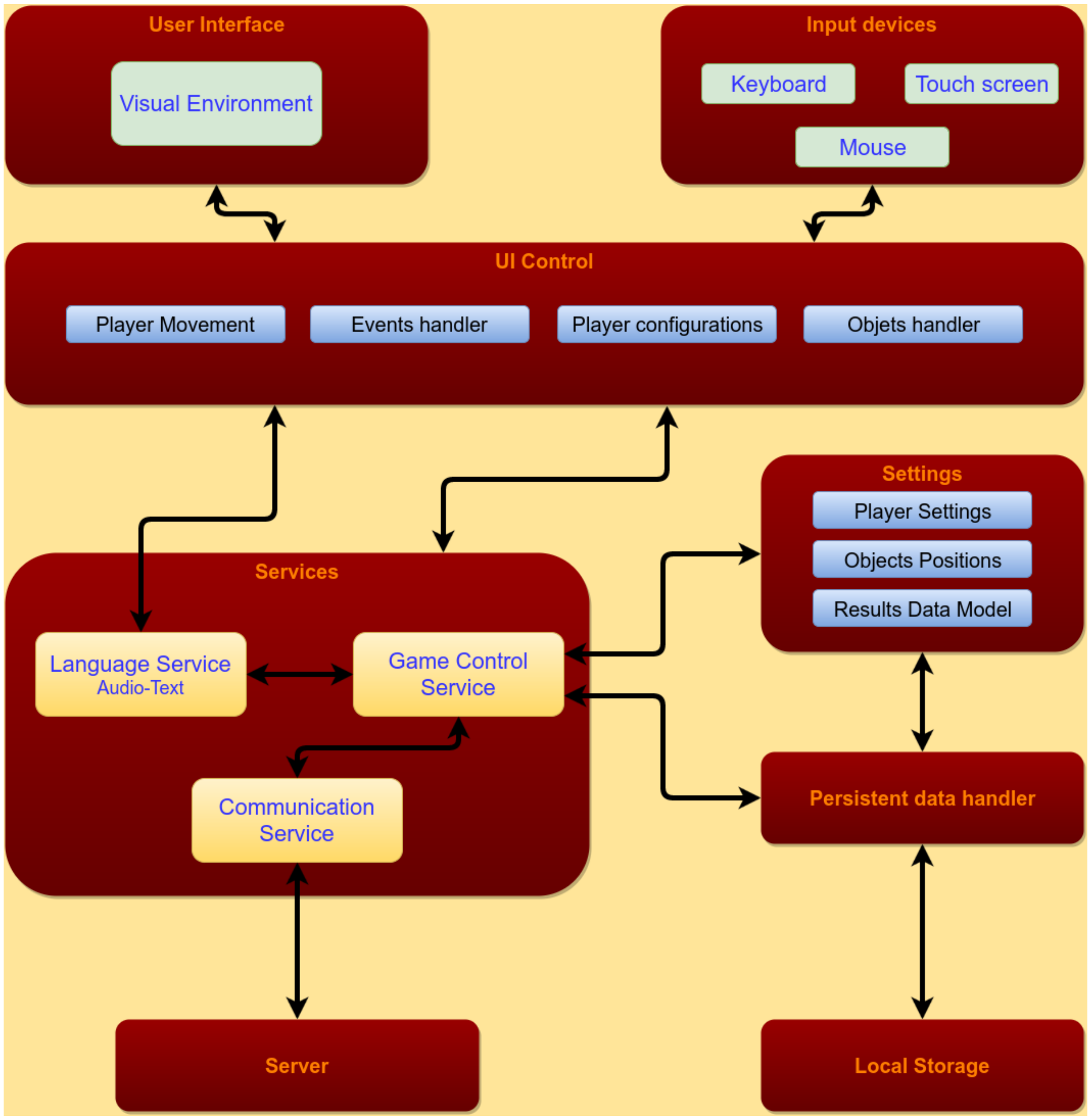


FIGURE 18. EPISODIX GAME ARCHITECTURE

1. The "Game Control Service" is the core of the game. It is used by other services or controllers to get current game information or to set it. "Game Control Service" has some different objectives like read current player setting to set the language or to prepare the results to send to the server.
2. The "User interface" block includes the parts of the game which are responsible for rendering the game objects. This block is connected to the "Game Control Service" using the "UI Control" block to get object's positions.

3. The “UI Control” is used to interact between user interface and the input. Also, it is responsible for connecting with the “Language Service” to get the current translation for the words or sentences to show on screen. The player movement is controlled in this block.
4. The “Settings” block represents the data model for saving the player settings.
5. The “Communication Service” is responsible for sending the game results to the server.
6. The “Persistent data handler” is responsible for storing results and the player settings into the device. The data is saved in a persistent data folder with the package name in the devices SD-card

The seniors have to use the pointer to interact with the game, selecting the up, down, left and right with arrows that appear on the screen. Following you can see the main screen of Episodix game:

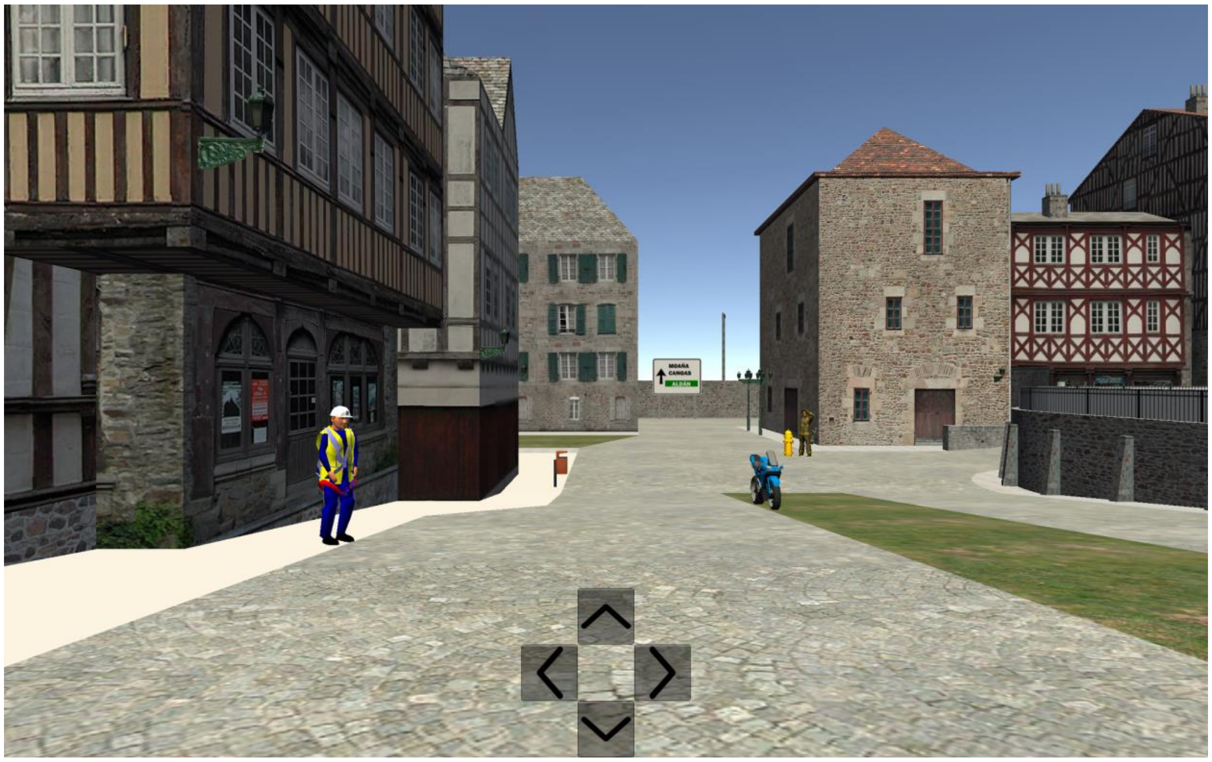


FIGURE 19. EPISODIX GAME SCREEN

4.2. Second pilot

After first pilot finishing some important feedback were obtained from seniors. All that information was included in the informal services of the second platform version.

Second version includes complete versions of previous services and four more services which objective is promote active lifestyle, seniors’ autonomy and keep them inform. In addition, some of the services were extended to use other platforms: mobile, tablet and web.

4.2.1 Agenda

The Agenda service allows senior, family members and caregivers to keep a record of seniors' appointments and events. Seniors may have trouble remembering all appointments on the day. This service can help them to organize their daily tasks, keeping them informed about the most relevant events (e.g. medicines intake, appointments with the doctor). Family and caregivers can also add events and reminders to seniors' agendas.

The Agenda service is composed by two different apps: a TV and a Web app. The TV app is focussed on seniors and the Web app is oriented to secondary users. The following chart shows the architecture of the service:

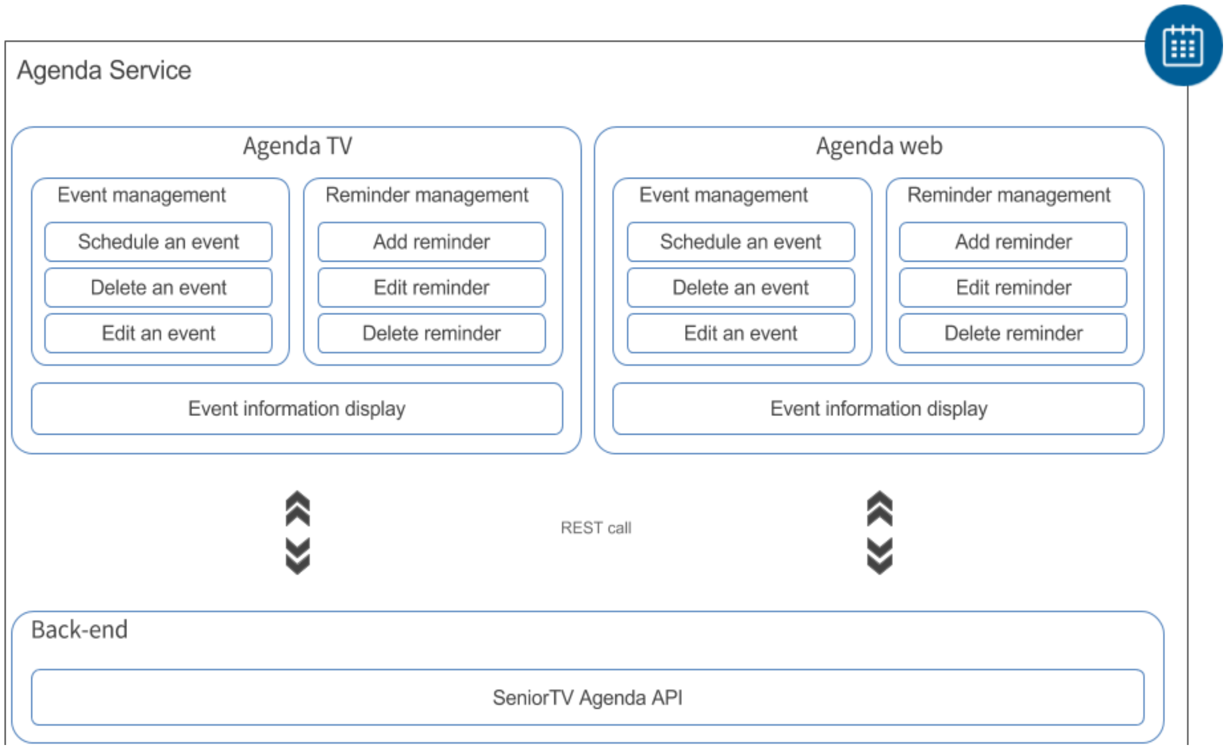


FIGURE 20. AGENDA ARCHITECTURE

Both, Web and TV applications have the same functionalities described below. The functionalities are arranged into two groups:

1. Events management: these functionalities allow users to read, create, update and remove Agenda events.
    1. *Display an event information* allows users to read the information of a scheduled event: the Name or Title of the event, the Starting and Ending time, the Location or Address, the associated Reminder and the Reminder time.

2. *Schedule an event* allows users to schedule an event. In particular, users can schedule an event with the following information: Name or Title of the event, Starting and Ending time, Location or address and, Reminder.
3. *Delete an event* allows users to remove a scheduled event. To remove the information a user confirmation is mandatory.
4. *Edit an event* allows users to modify the information of a scheduled event. In particular, the user can update the following information: Name or Title of the event, Starting and Ending time, Location or address and, Reminder.

2. Reminders management: these functionalities allow users to manage the reminder associated with an event.

    1. *Add a reminder* allows users to set a reminder to an event. There are two ways of adding a reminder: when scheduling a new event or when modifying an event. Possible reminder values are: Never, 30 minutes before the starting time, 1 hour before the starting time or 1 day before the starting time.
    2. *Edit the reminder* allows users to modify the event associated reminder. This functionality is only accessible when the user is updating an event.
    3. *Delete the reminder* allows users to remove the event associated reminder selecting the option "Never".

The Agenda TV and Web have been implemented using Angular and Apache Cordova frameworks. The cloud backend has been developed using Java EE[15] technology. The two apps use HTTP request to communicate with the backend to obtain the seniors' data. The next paragraphs describe the TV and Web applications architecture and the main interface screens.

4.2.1.1 Agenda TV

The following charts show the Agenda TV app architecture and some interface screens:

[15] Java EE reference: https://www.oracle.com/technetwork/java/javaee/documentation/index.html

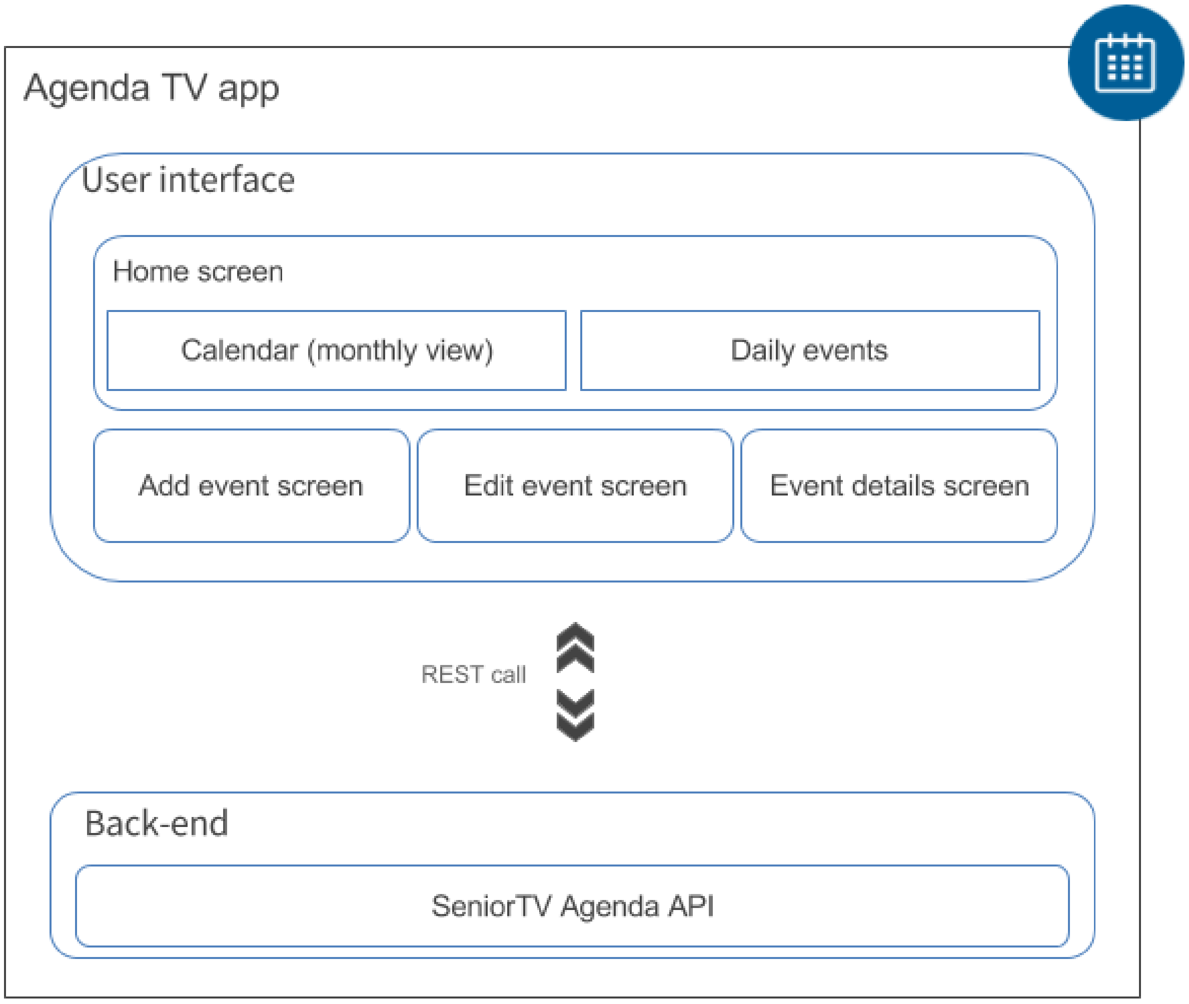


FIGURE 21. AGENDA TV ARCHITECTURE

1. Home Screen: The home screen is the first screen shown after the Agenda TV application starts. This screen is divided in two sections:
    - On the left side, the monthly view of the calendar: offers a general overview of the scheduled events for the current month. Days with scheduled events are highlighted in grey color, the current day is highlighted in red and the selected day is highlighted in orange.
    - On the right side, the scheduled events for a particular day are shown. Also, a "Add New Event" button is included to move to the "Add Event" screen to schedule a new event for the selected day.

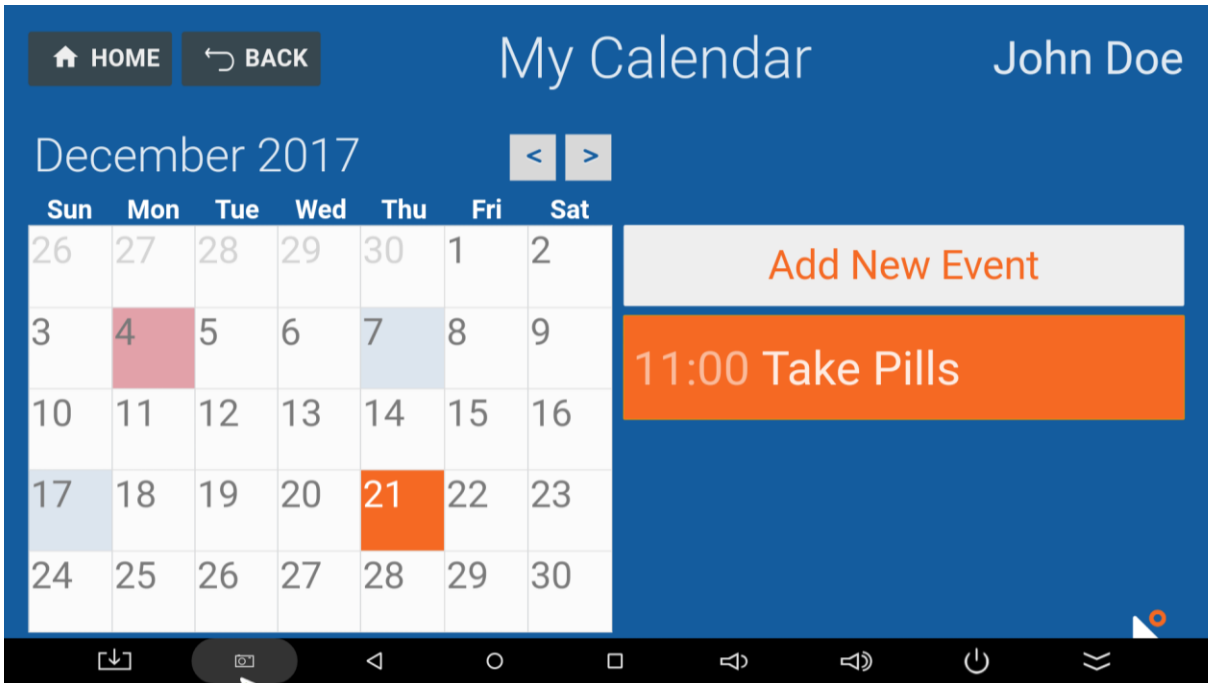


FIGURE 22. AGENDA TV HOME SCREEN

2. Event Details Screen

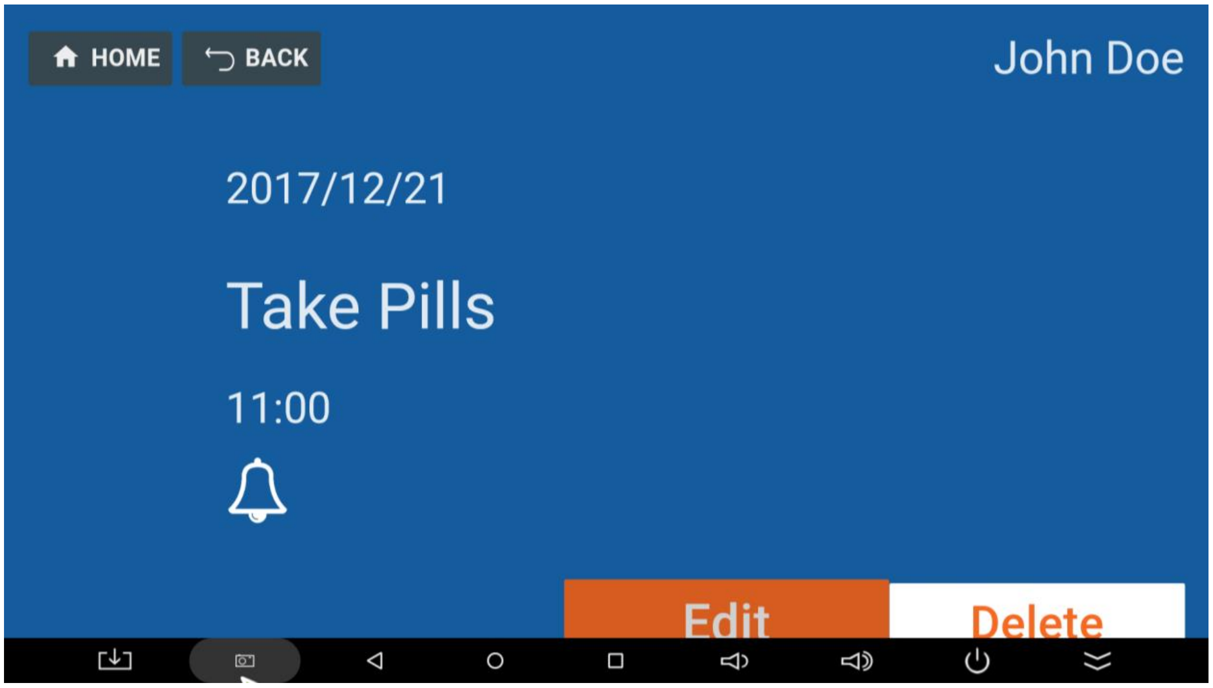


FIGURE 23. AGENDA TV EVENT DETAILS SCREEN

3. Add Event Screen

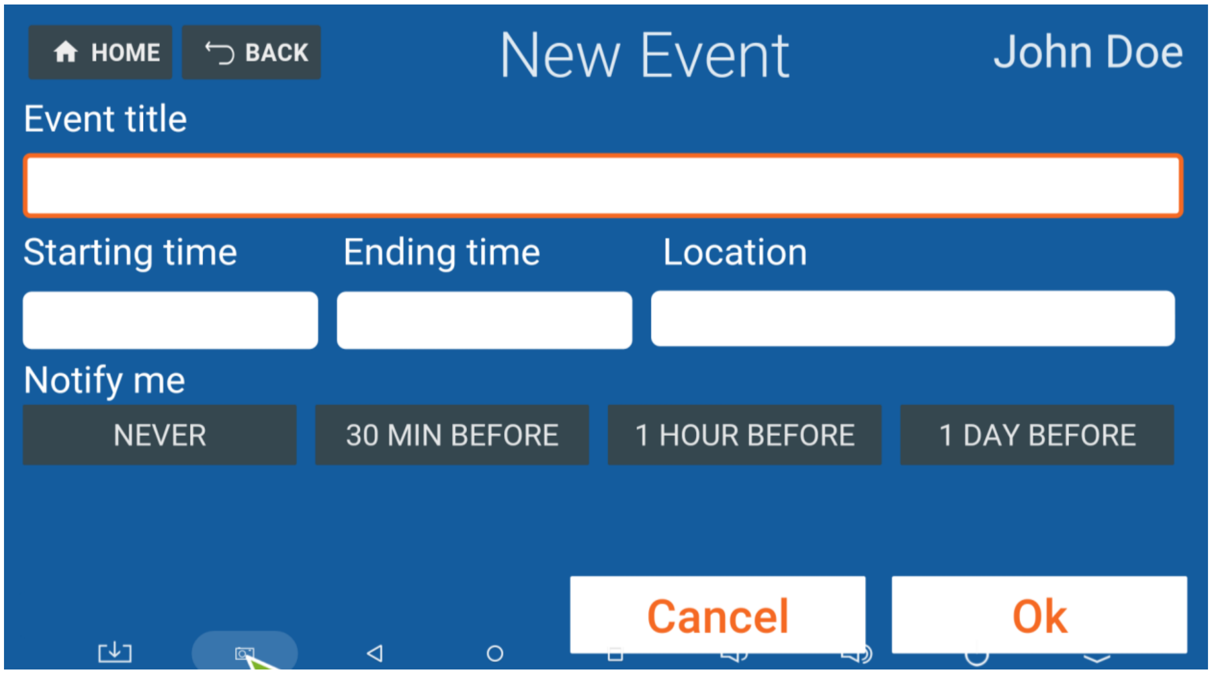
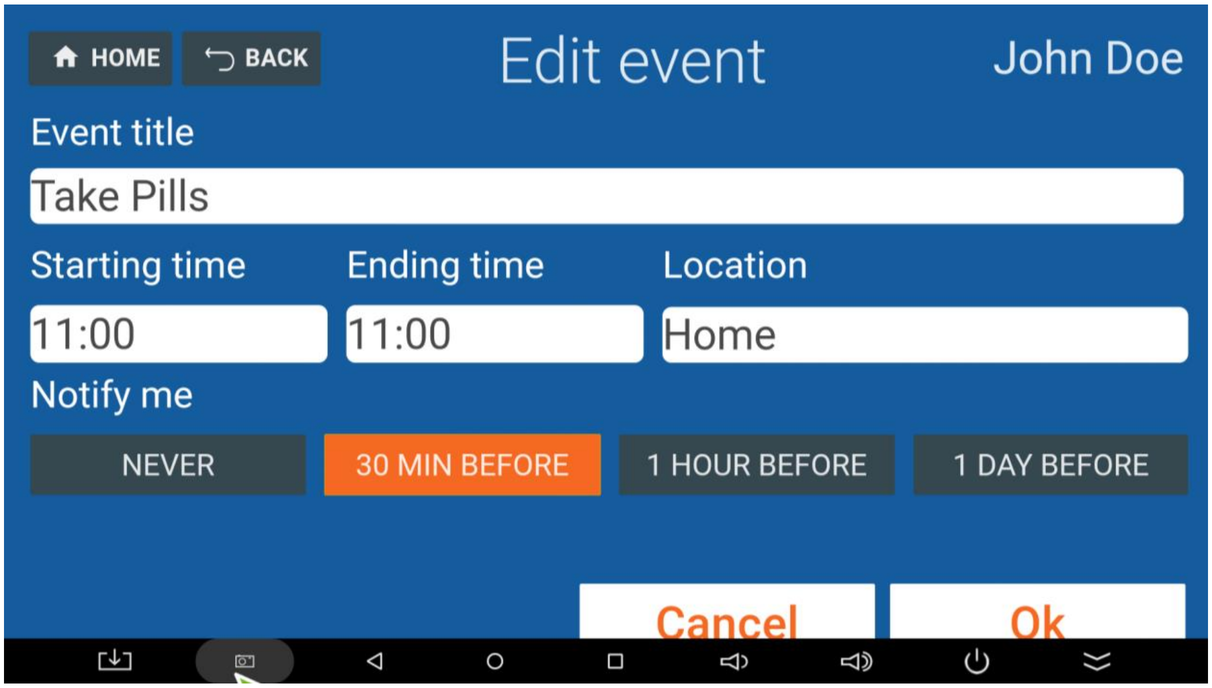


FIGURE 24. AGENDA TV ADD EVENT SCREEN

4. Edit event Screen



Event title: Take Pills
Starting time: 11:00
Ending time: 11:00
Location: Home
Notify me: 30 MIN BEFORE

FIGURE 25. AGENDA TV EDIT EVENT SCREEN

4.2.1.2 Agenda Web

The following chart shows the architecture of the user interface of the Agenda Web app.

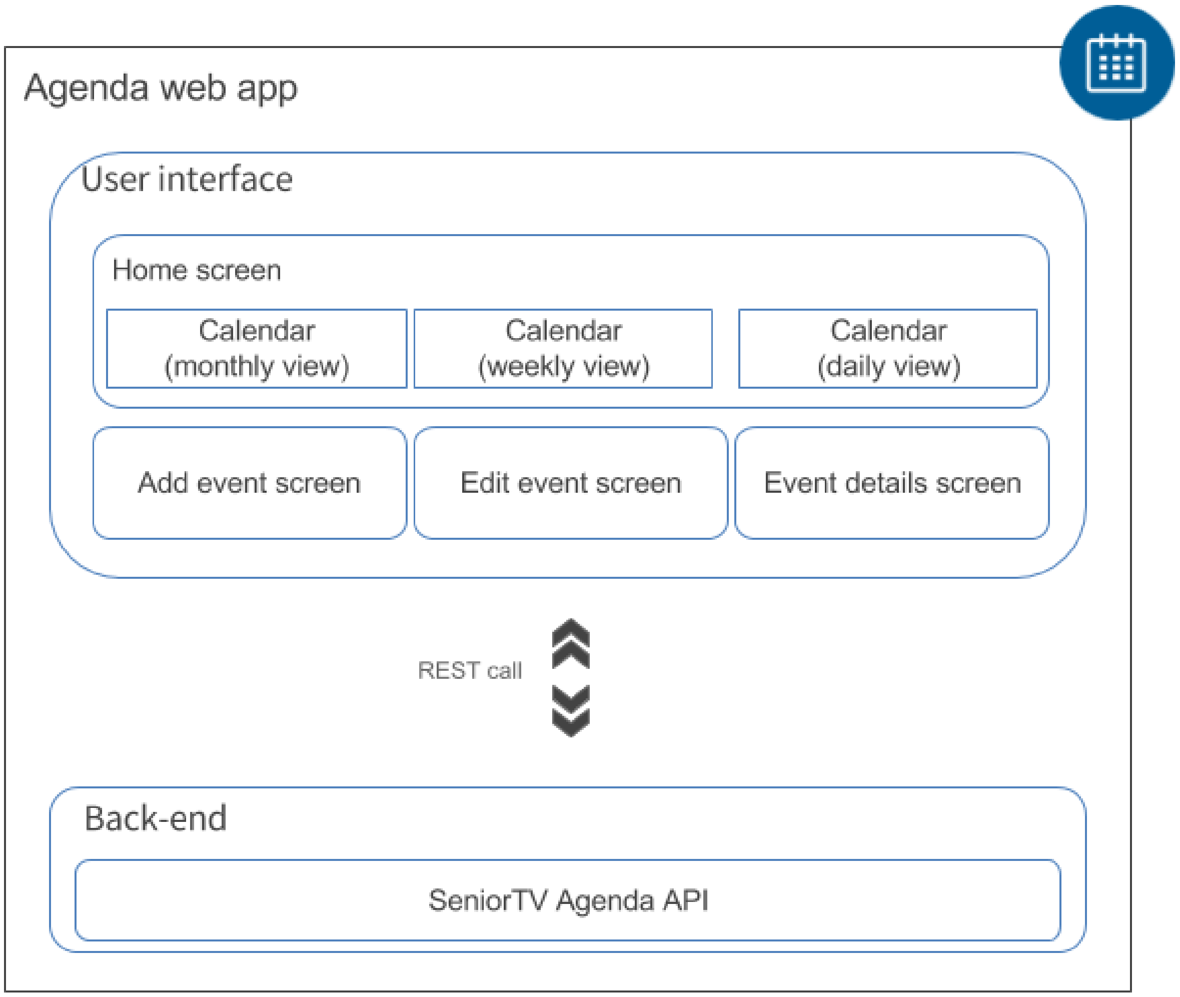


FIGURE 26. AGENDA WEB ARCHITECTURE

1. Home Screen is the first screen shown after the Agenda web application starts. By default, the application displays the calendar monthly view, which offers a general overview of the scheduled events for current month. The app also offers a Weekly and Daily view to filter the events.

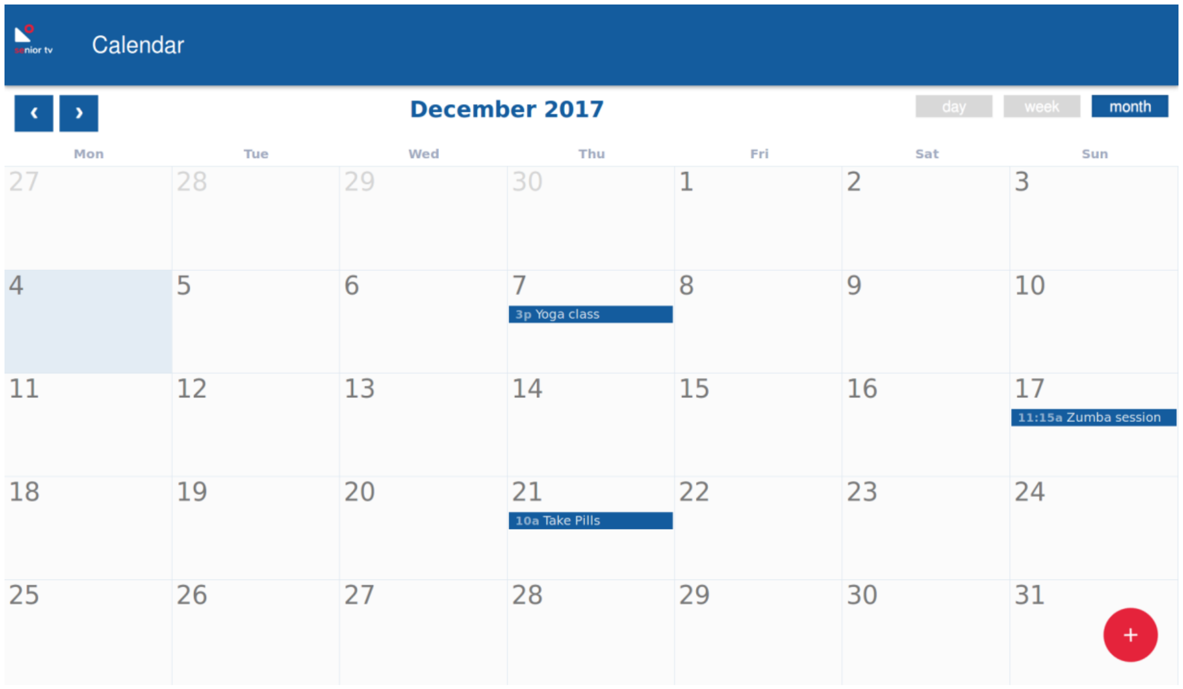


FIGURE 27. AGENDA WEB HOME SCREEN MONTHLY VIEW

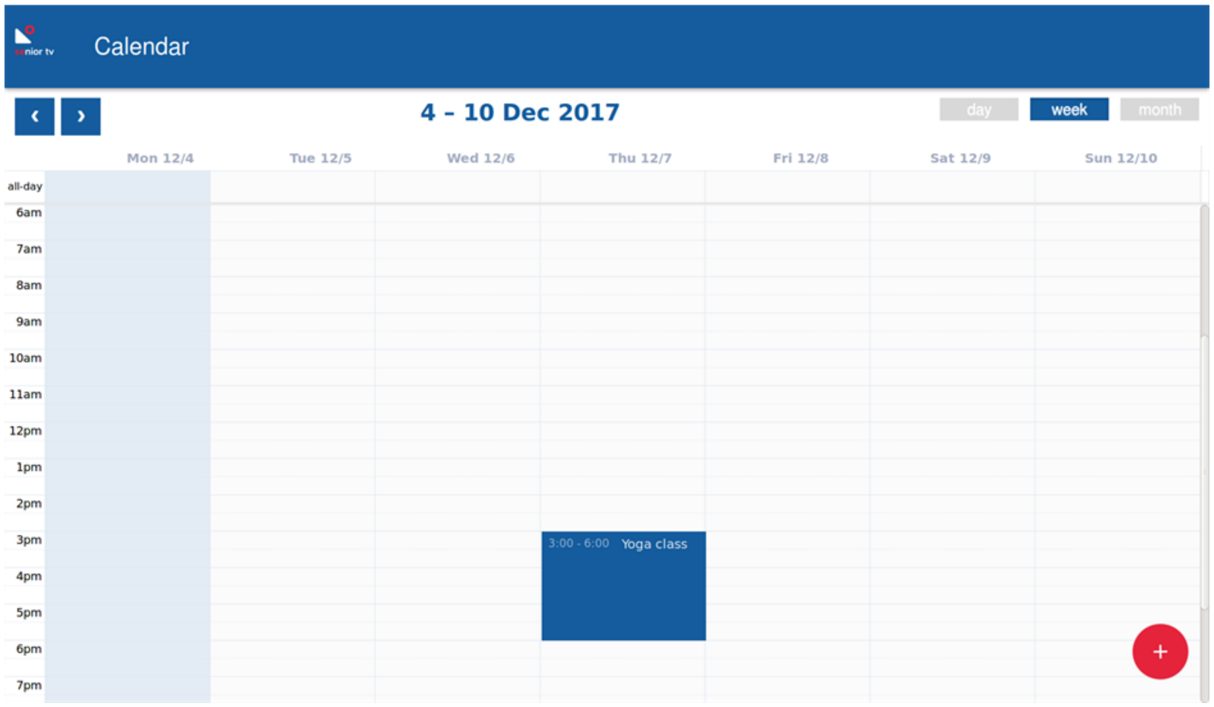


FIGURE 28. AGENDA WEB HOME SCREEN WEEKLY VIEW

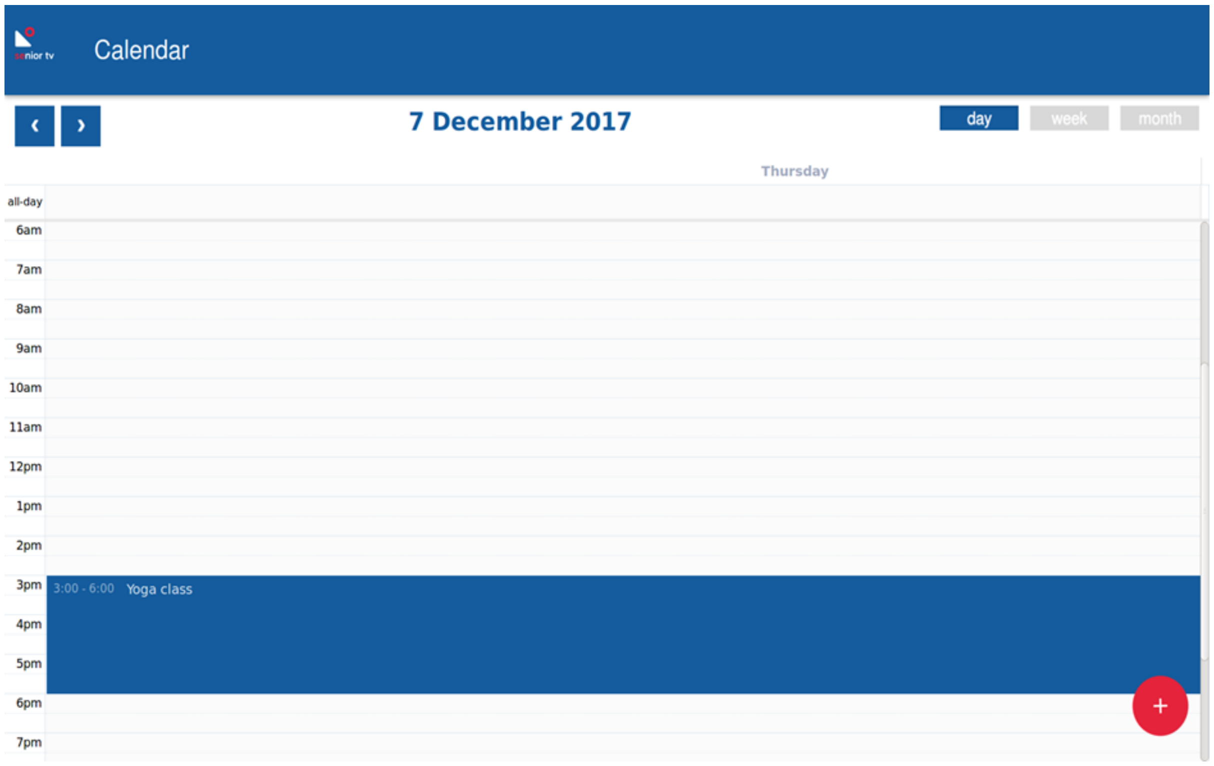
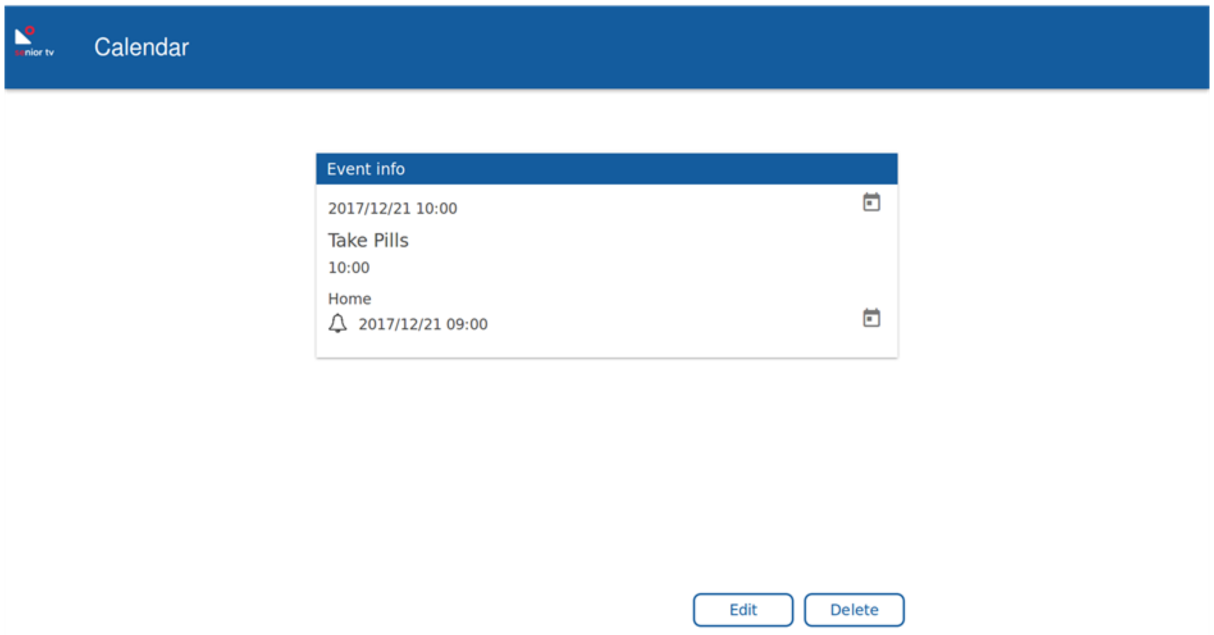


FIGURE 29. AGENDA WEB HOME SCREEN DAILY VIEW

2. Event Details Screen


Event info

2017/12/21 10:00

Take Pills

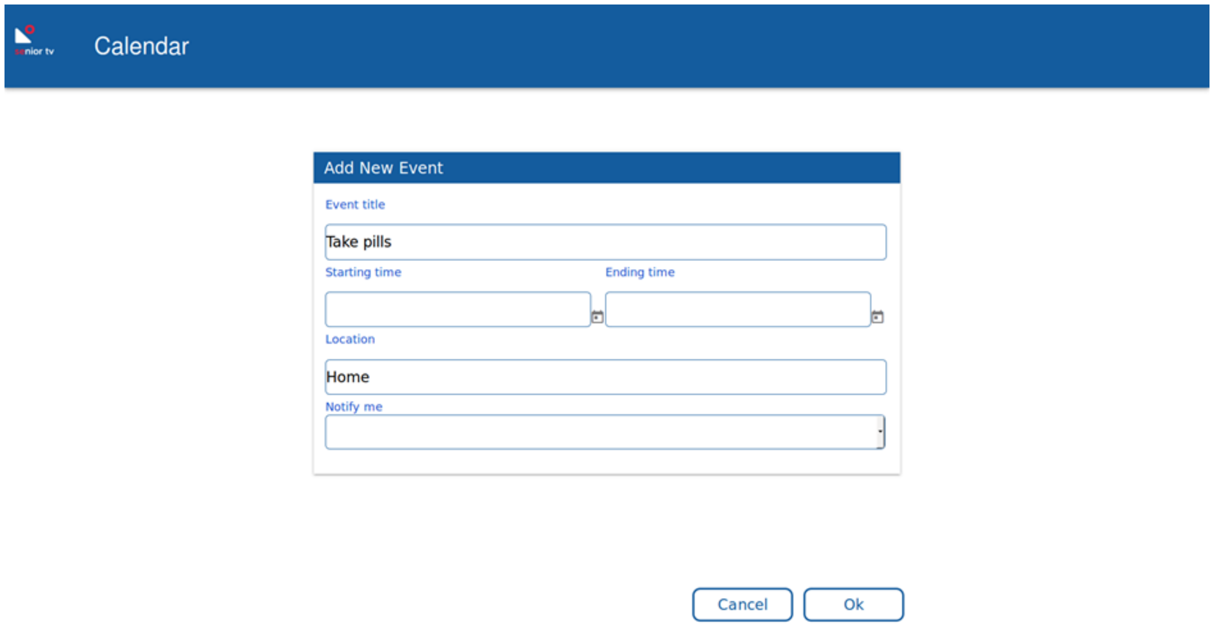
10:00

Home

2017/12/21 09:00

FIGURE 30. AGENDA WEB EVENT DETAILS SCREEN

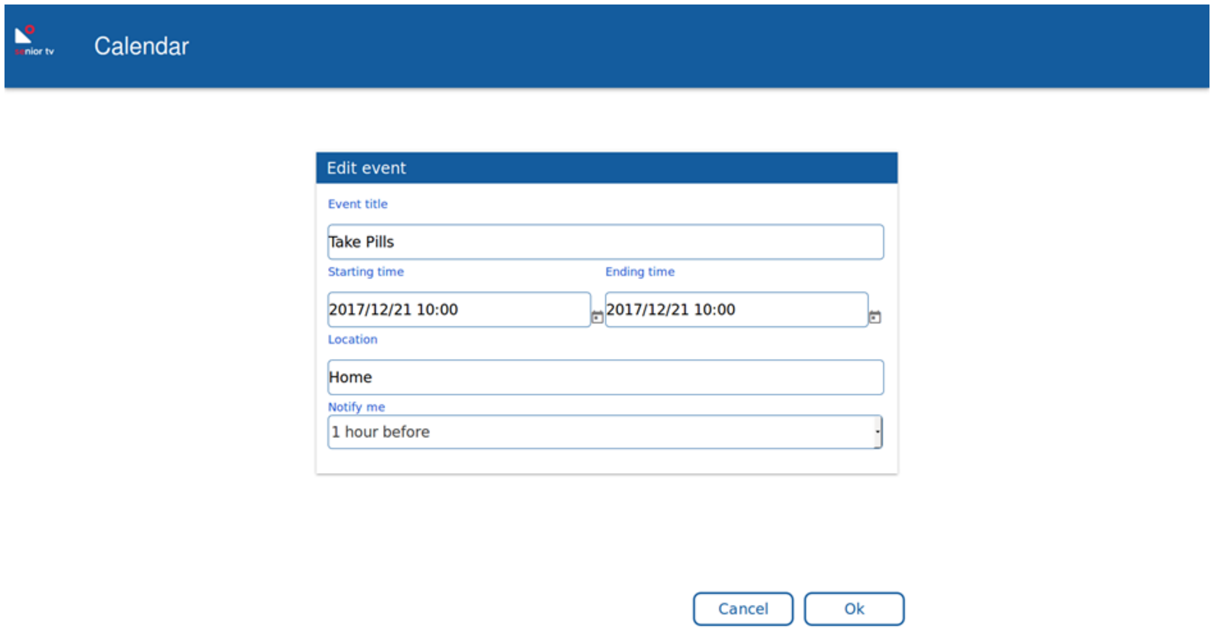
3. Add Event Screen



Event title
Take pills
Starting time
Ending time
Location
Home
Notify me

FIGURE 31. AGENDA WEB ADD EVENT SCREEN

4. Edit event Screen



Event title
Take Pills
Starting time
2017/12/21 10:00
Ending time
2017/12/21 10:00
Location
Home
Notify me
1 hour before

FIGURE 32. AGENDA WEB EDIT EVENT SCREEN

4.2.2 Events 2.0

This service is an upgrade of Events service developed to first pilot. The initial version was only used to get the seniors feedback about the interface usability the remote-control interaction and also the service itself.

The new version of this service is composed by two different modules: a TV and a Web app. The TV app is focussed on seniors and the Web app is oriented to secondary users.

4.2.2.1 Events TV

The scope of the TV app has not change from the initial version: use the TV screen to find and participate to events that seniors are interested in. The backend functionalities and the architecture of upgraded version have not changed from the initial one (see section 4.1.1). Although new developments were done in the version 2.0 in order to complete all mock-up functionalities shown in the initial version and to include the System Integration services. From the previous version the following improvements were achieved:

- Integration with the Auth app and it's services.
- Integration with the Events Web app and Cloud.
- Integration with the Agenda service.
- Implementation of the new design guidelines.
- The app is completely controllable from only the D-PAD, Home, OK and Back Buttons.
- A preference screen was implemented for personalizing the text size and the preferred categories.

For developing, the same hybrid technologies have been used than for initial version: Ionic 3, HTML, TypeScript, Cordova, and various open source third party plugins. The main screens of the TV interface are as follows:

1. The Category Selection Screen

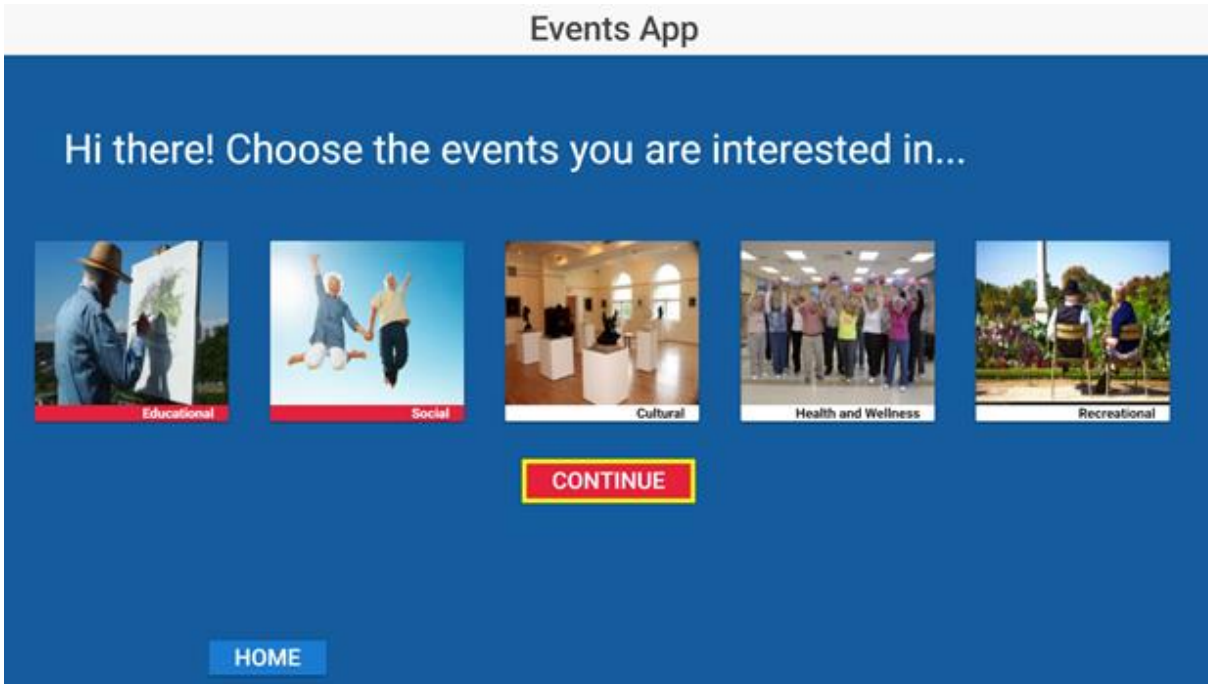
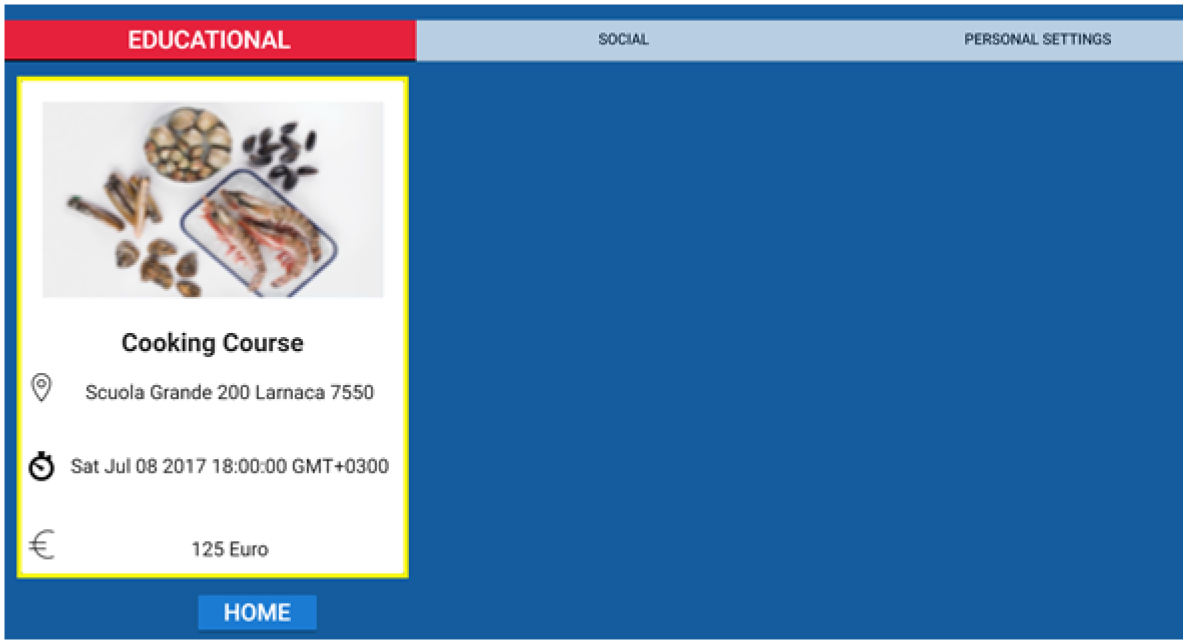


FIGURE 33. EVENTS 2.0 APP SELECTION SCREEN

2. The Events Preview Screen



Cooking Course

Scuola Grande 200 Larnaca 7550

Sat Jul 08 2017 18:00:00 GMT+0300

125 Euro

FIGURE 34. EVENTS 2.0 APP PREVIEW SCREEN

3. The Event View Screen

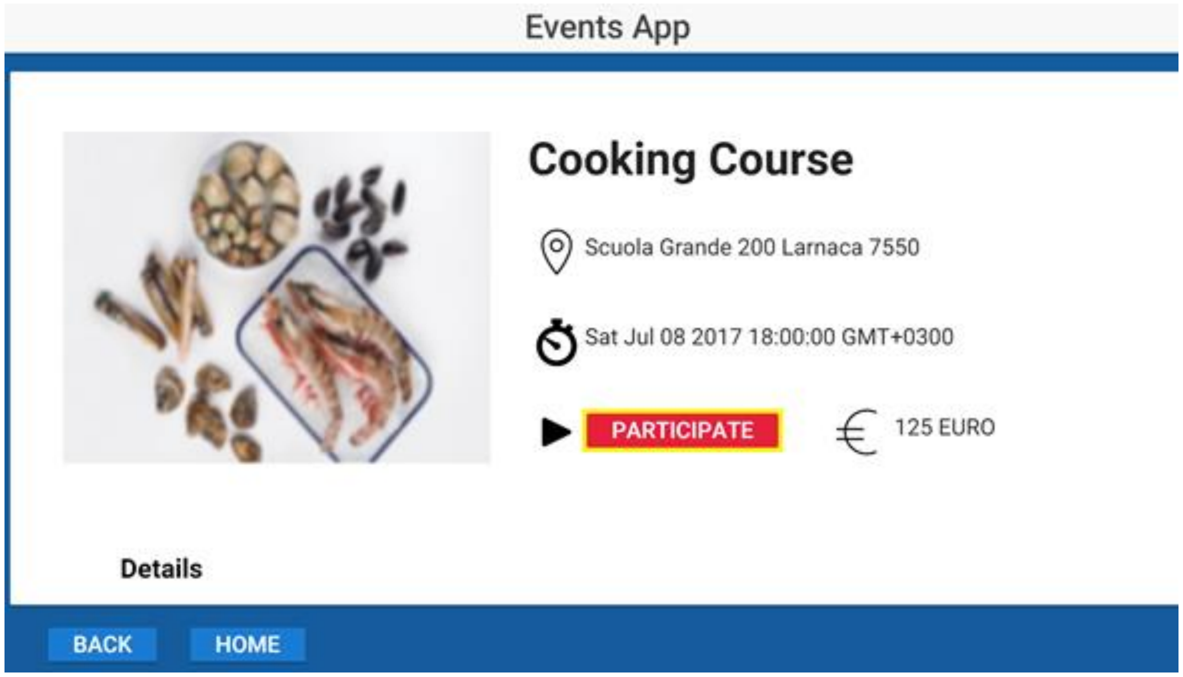

Events App
Cooking Course
Scuola Grande 200 Larnaca 7550
Sat Jul 08 2017 18:00:00 GMT+0300
PARTICIPATE
125 EURO
Details
BACK
HOME

FIGURE 35. EVENTS 2.0 VIEW SCREEN

4. The Customization Screen

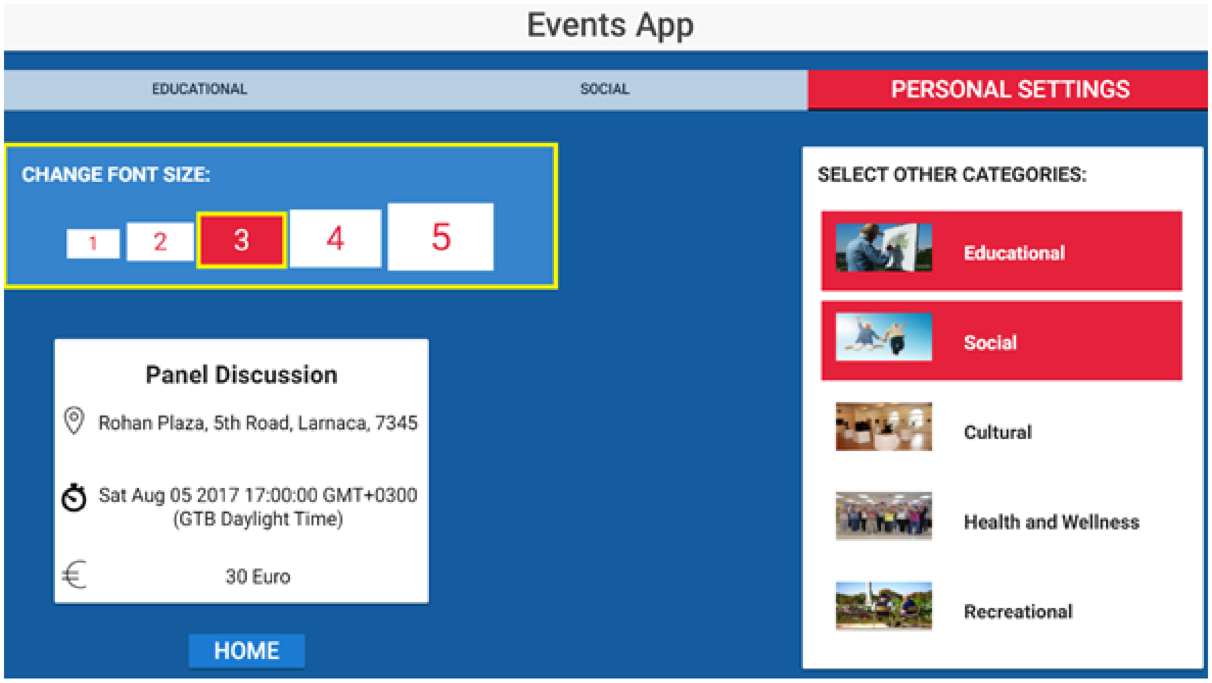


FIGURE 36. EVENTS 2.0 CUSTOMIZATION SCREEN

4.2.2.2 Events Web

The scope of this application is to enable different Event Providers for elderly to add events which will be shown through TV application. These events will be filtered when they are shown on the TV based on the preference for language for each SENIOR-TV system. The architecture of the Events Web app is shown below:

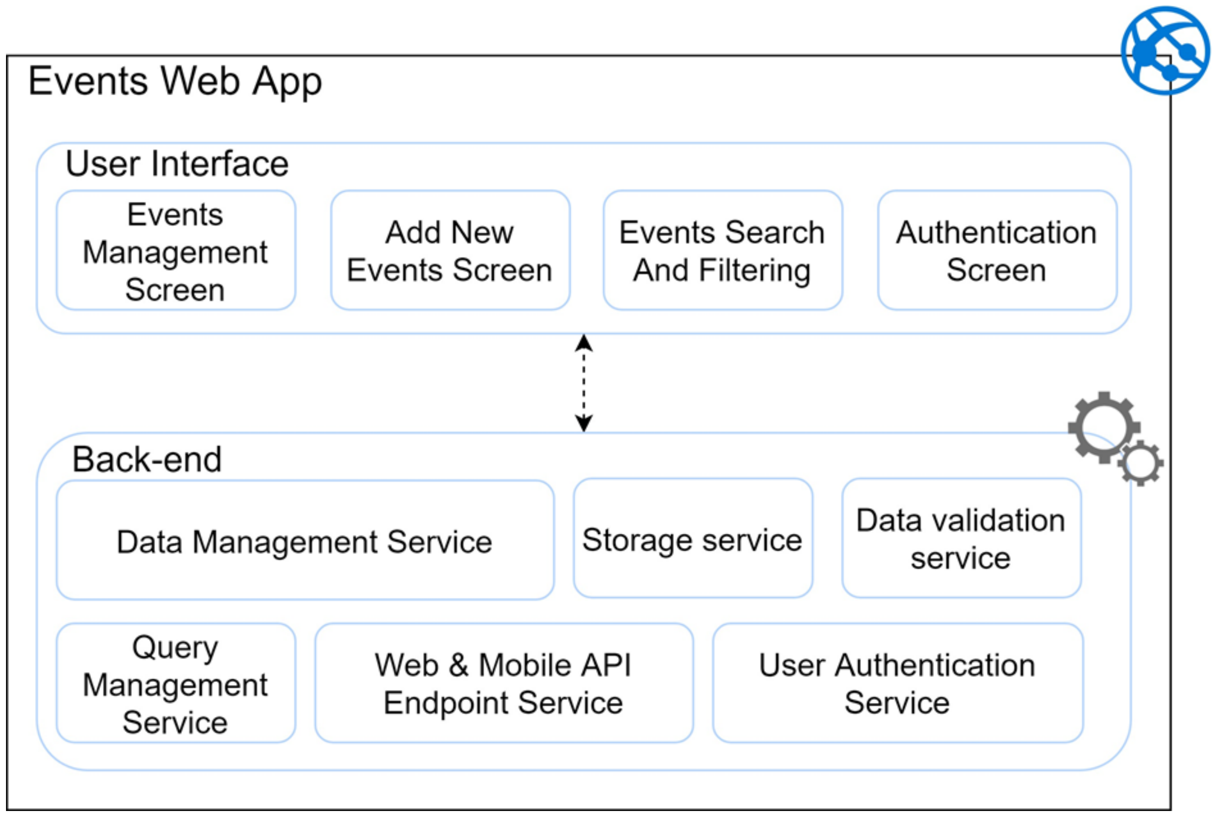


FIGURE 37. EVENTS 2.0 CUSTOMIZATION SCREEN

The components of the Events Web app are:

- The Events Management Screen enables the editors of the events to view, add, remove and edit events.
  - Editors can add new events by clicking on the Add New Button. In order to add a new event some detailed information is required: Group (Social, Cultural, Recreational, etc.), Language, Type, Address, Date, Price, Details and, Image.
  - The Editors can search for events using a variety of comparison operators.
  - The Authentication Screen allows the user to log in to the application.
- The Back-end of the Events service has the following components:
  - Data Management Service: enables the management of data transferred between the APIs and the Storage Service.

- ○ Storage Service: enables the persistence of Data in a SQL Database.
- ○ Data Validation service: verifies that the queries and commands sent to the cloud server are authorized and in accordance to the business logic.
- ○ Query management service: enables the dispatch of queries and commands from the API Controllers to the corresponding handlers.
- ○ Web & Mobile API Endpoint Service: enables the Web App and Mobile app to exchange REST messages with the Cloud Server.
- ○ User Authentication Service: enables the create, update and login/logout of users in the web services.

The technologies used for developing the Events Web app are: Angular4, HTML, TypeScript, .Net and various other open source and third-party libraries. The interface first version is shown below:

1. Add new event Screen

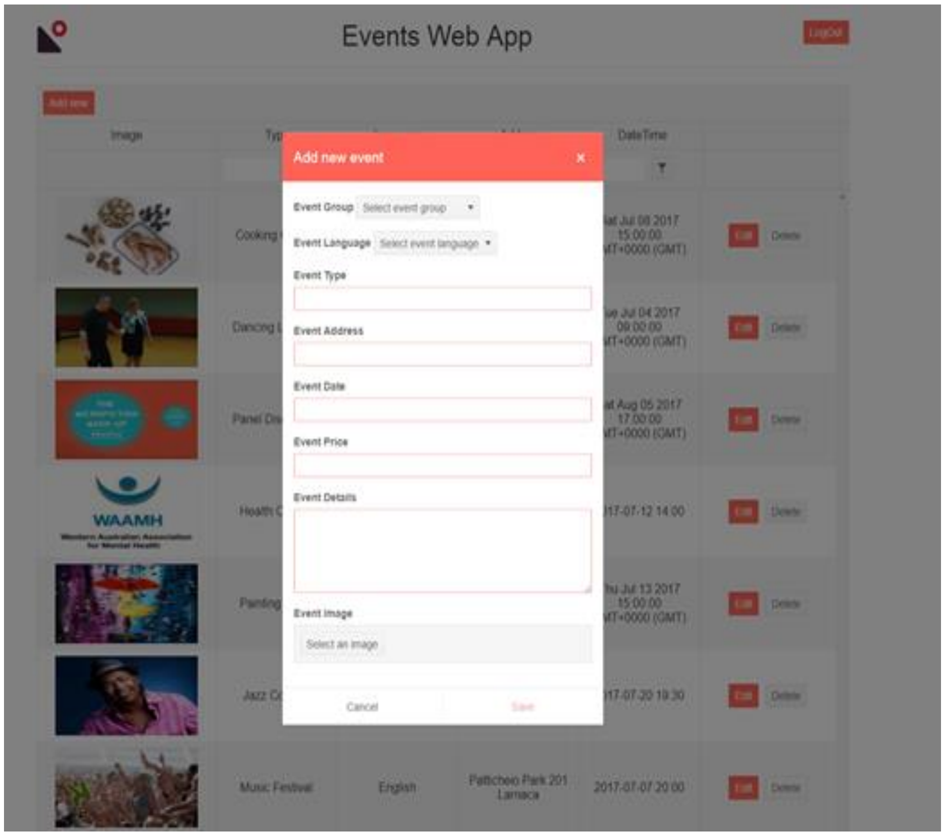


FIGURE 38. NEW EVENT SCREEN

2. List events Screen

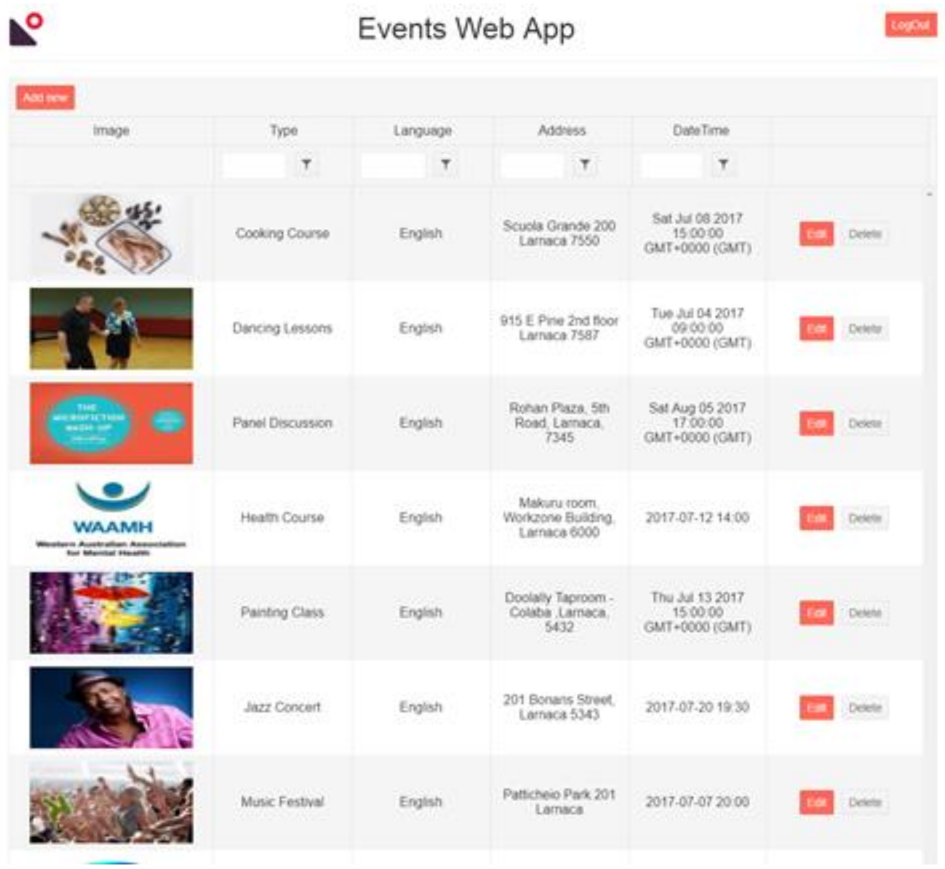


Image	Type	Language	Address	DateTime	
	Cooking Course	English	Scuola Grande 200 Larnaca 7550	Sat Jul 08 2017 15:00:00 GMT+0000 (GMT)	Edit Delete
	Dancing Lessons	English	915 E Pine 2nd floor Larnaca 7587	Tue Jul 04 2017 09:00:00 GMT+0000 (GMT)	Edit Delete
	Panel Discussion	English	Rohan Plaza, 5th Road, Larnaca, 7345	Sat Aug 05 2017 17:00:00 GMT+0000 (GMT)	Edit Delete
	Health Course	English	Makuru room, Workzone Building, Larnaca 6000	2017-07-12 14:00	Edit Delete
	Painting Class	English	Doolally Taproom - Colaba ,Larnaca, 5432	Thu Jul 13 2017 15:00:00 GMT+0000 (GMT)	Edit Delete
	Jazz Concert	English	201 Bonans Street, Larnaca 5343	2017-07-20 19:30	Edit Delete
	Music Festival	English	Patticheio Park 201 Larnaca	2017-07-07 20:00	Edit Delete

FIGURE 39. LISTS EVENTS SCREEN

4.2.3 News 2.0

This service is an upgrade of News service developed to first pilot. The initial version was only used to get the seniors feedback about the interface usability and the remote-control interaction. Although new developments were done in the version 2.0 in order to complete all mock-up functionalities shown in the initial version and to include the System Integration services.

From the previous version the following improvements were achieved:

- Integration with the Auth app and its services.
- Implementation of the new design guidelines.
- The app is completely controllable from only the D-PAD, Home, OK and Back Buttons.
- A preference screen was implemented for personalizing the text size and the preferred categories.

For developing, the same hybrid technologies have been used than for initial version: Ionic 3, HTML, TypeScript, Cordova, and various open source third party plugins. Below are screenshots of the improved app:

1. News Central Screen

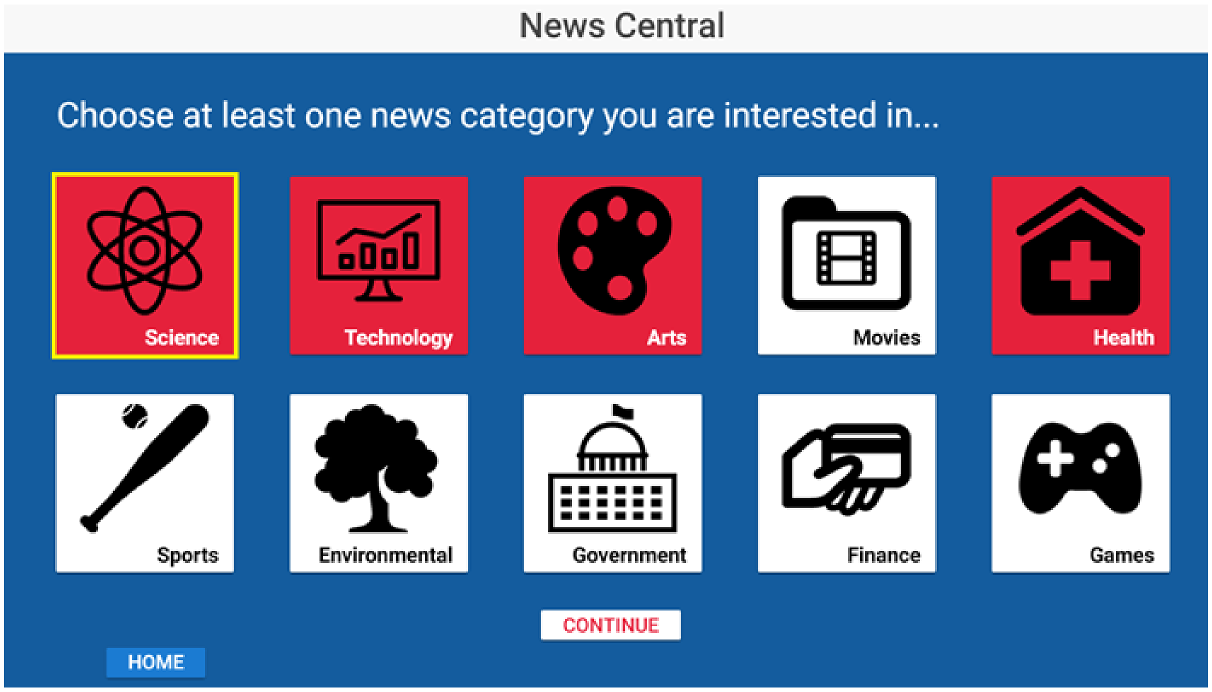
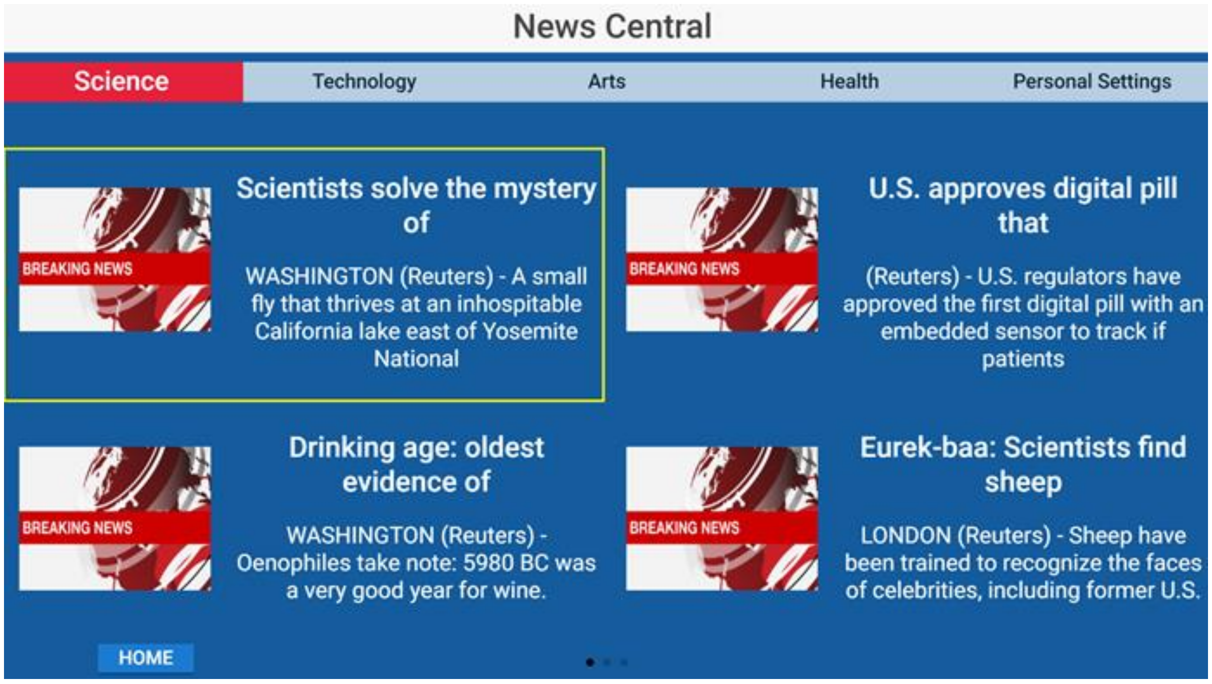


FIGURE 40. NEWS 2.0 CATEGORIES SELECTION SCREEN

2. Category List News Screen



Scientists solve the mystery of

WASHINGTON (Reuters) - A small fly that thrives at an inhospitable California lake east of Yosemite National

U.S. approves digital pill that

(Reuters) - U.S. regulators have approved the first digital pill with an embedded sensor to track if patients

Drinking age: oldest evidence of

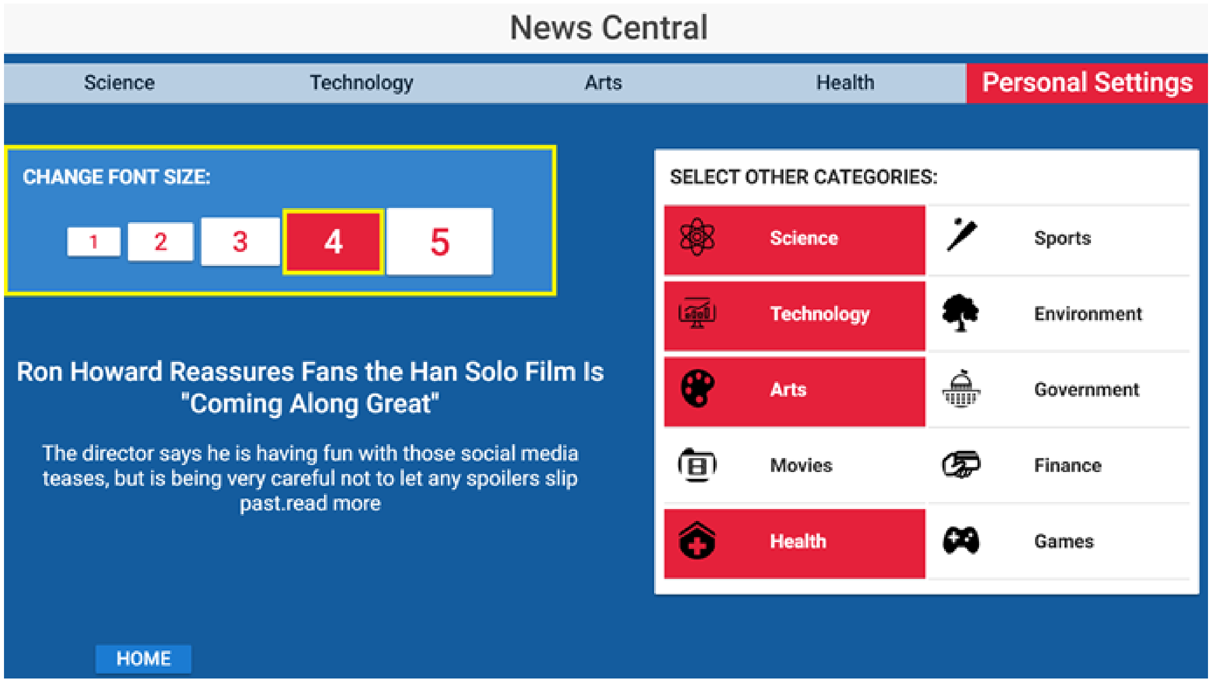
WASHINGTON (Reuters) - Oenophiles take note: 5980 BC was a very good year for wine.

Eurek-baa: Scientists find sheep

LONDON (Reuters) - Sheep have been trained to recognize the faces of celebrities, including former U.S.

FIGURE 41. NEWS 2.0 CATEGORY LIST NEWS SCREEN

3. The Customization Screen



Ron Howard Reassures Fans the Han Solo Film Is "Coming Along Great"

The director says he is having fun with those social media teases, but is being very careful not to let any spoilers slip past.read more

FIGURE 42. NEWS 2.0 CUSTOMIZATION SCREEN

4.2.4 Tracker

Keeping an active life brings a lot of benefits for seniors. Tracker service allows users to keep a record of their walks using a mobile phone and then, use the TV to see the walk metrics and results.

To encourage senior to do outdoor exercises the service includes a common gamified environment where friends can set up their own competitions: seniors not only can see their walks but also can see their “friends” results.

Besides, Tracker service provides mechanisms to reduce the seniors’ sense of insecurity when they walk. The seniors’ address can easily store in the mobile app in order to know the way home in case they felt lost or disoriented.

The Tracker service is composed by two different apps: a TV and a Mobile app. The Mobile app records the seniors’ walks and the TV app shows the walks summarizes. The following chart shows the architecture of the service:

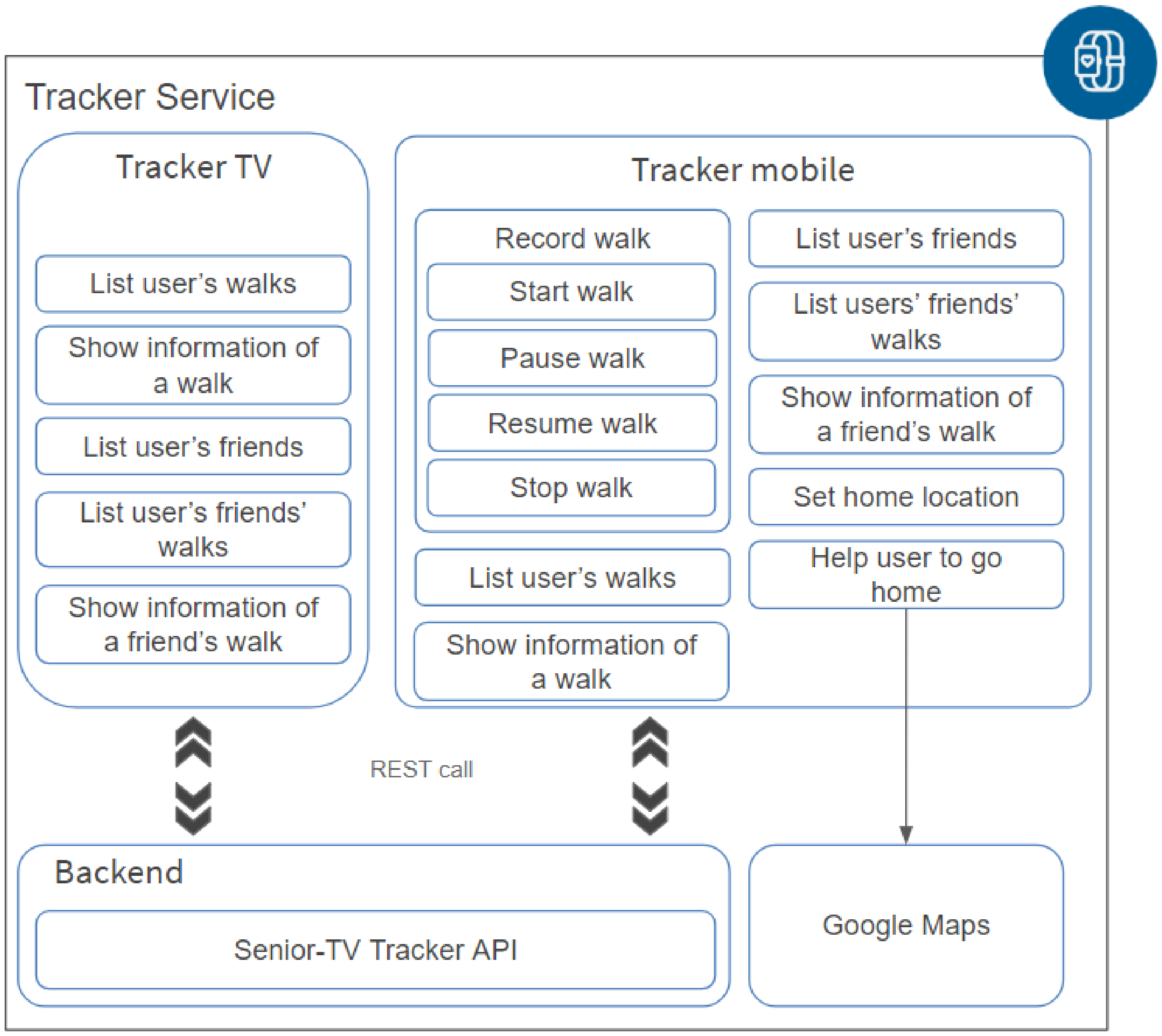


FIGURE 43. TRACKER ARCHITECTURE

Both, Mobile and TV apps have been implemented using Angular framework and Apache Cordova for TV app. The walk information is shown on a map in two apps. To paint this information the Leaflet library is used.

Leaflet[16] is a widely used open-source JavaScript library used to build interactive maps in applications. It supports most mobile and desktop platforms, supporting HTML5 and CSS3. Leaflet allows developers without a geographic information system to display tiled maps, with optional tiled overlays. It can load feature data from GeoJSON files, style it, and create interactive layers, such as markers with popups when clicked.

[16] Leaflet official site: https://leafletjs.com/

The cloud backend has been developed using Java EE[17] technology. The two apps use HTTP request to communicate with the backend to obtain the walks' data. The cloud backend is composed by two services:

- SENIOR-TV Tracker API: allows manage the favorite locations (add, remove or update the database information).
- Google Maps[18] is an external service used to calculate the route from the seniors' location to their home.

The next paragraphs describe the TV and Web applications architecture, their functionalities and the main interface screens.

4.2.4.1 Tracker TV

The scope of the TV app is to enable elderly users to see their walk metrics from the TV. The Tracker TV main functionalities are listed below:

1. List user's walks: allows users to see all their walks.
2. See user's friends' walks: users can see the walk details made by their friends. Before to access the walks' data, it is necessary to select one of their friends.
3. Show a walk information: allows users to see the details of a walk. In particular, users can see the following pieces of information:
    - Route on a map.
    - Walk Time.
    - Walk Total time.
    - Walked distance.
    - Average speed.

The following charts show the Tracker TV app architecture and some interface screens:

[17] Java EE documentation: https://www.oracle.com/technetwork/java/javaee/documentation/index.html
[18] Google Maps Platform: https://cloud.google.com/maps-platform/

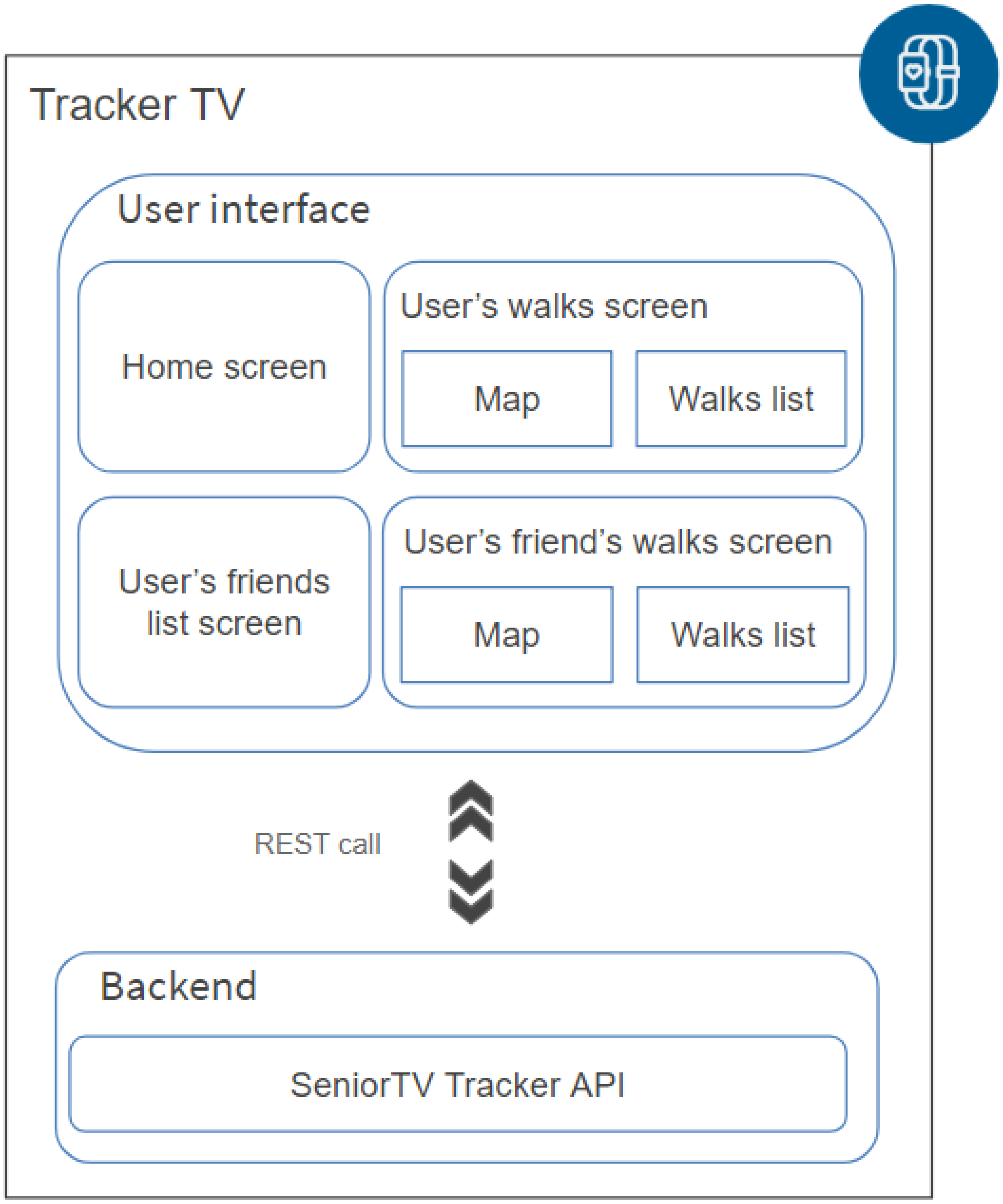


FIGURE 44. TRACKER TV ARCHITECTURE

1. Home screen

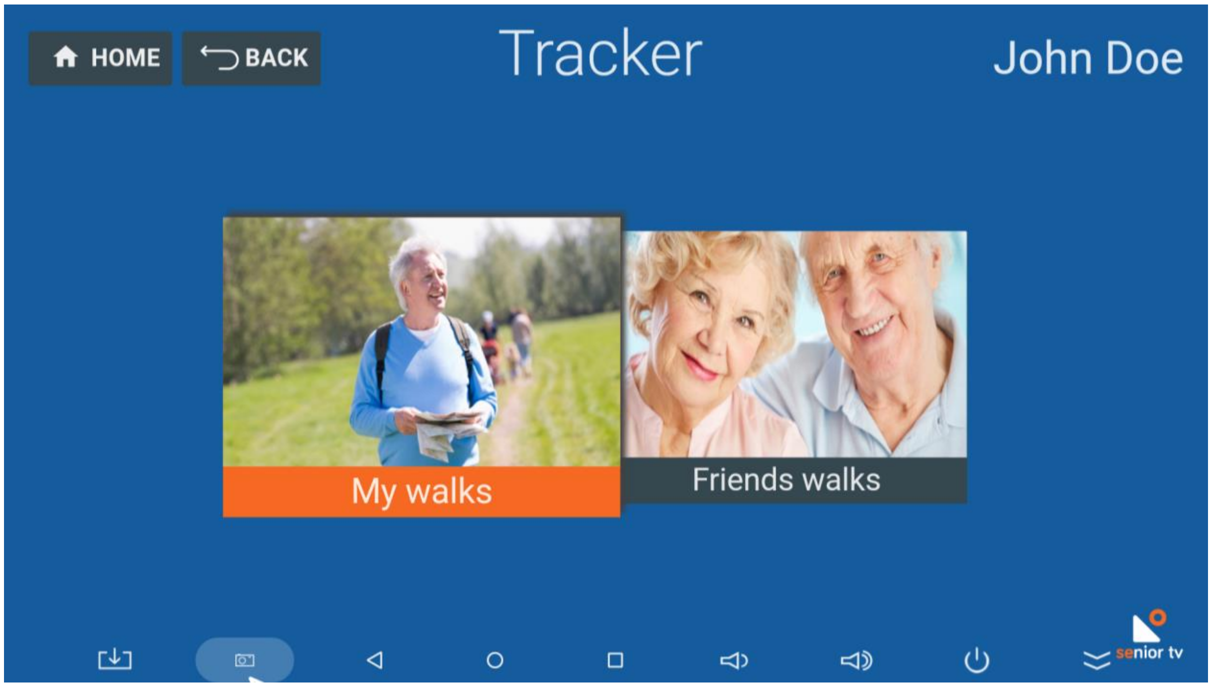


FIGURE 45. TRACKER TV HOME SCREEN

2. Walk list screen

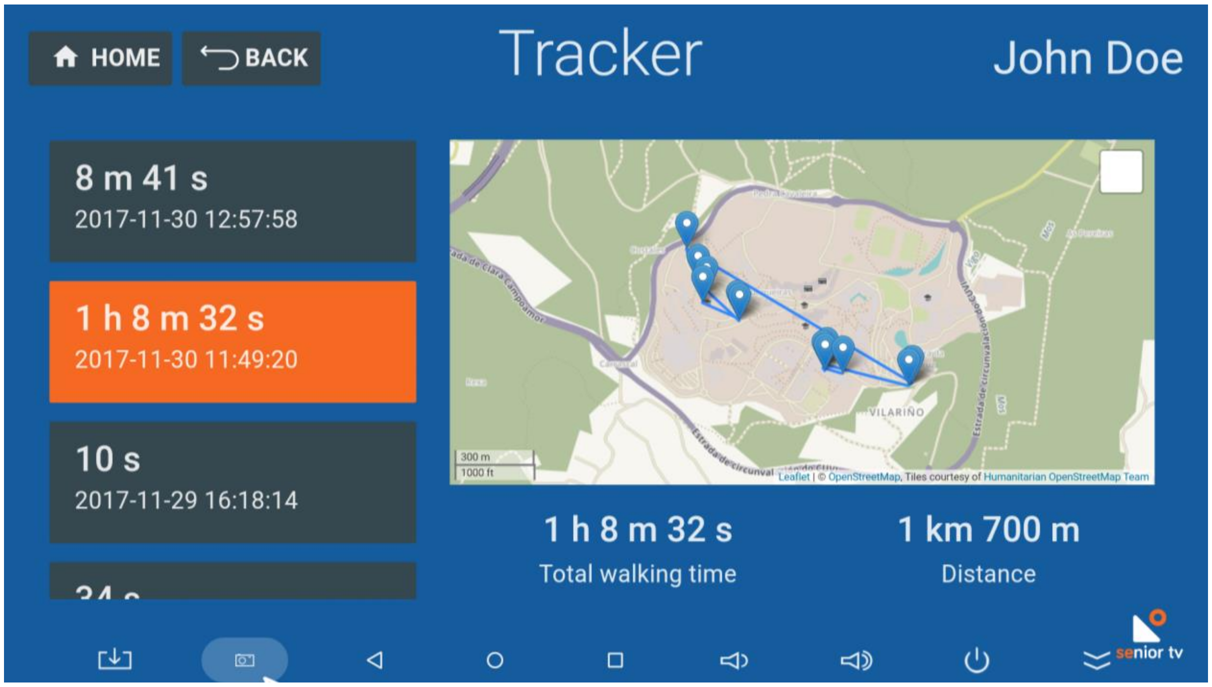


FIGURE 46. TRACKER TV USER'S WALKS SCREEN

3. Friend list screen

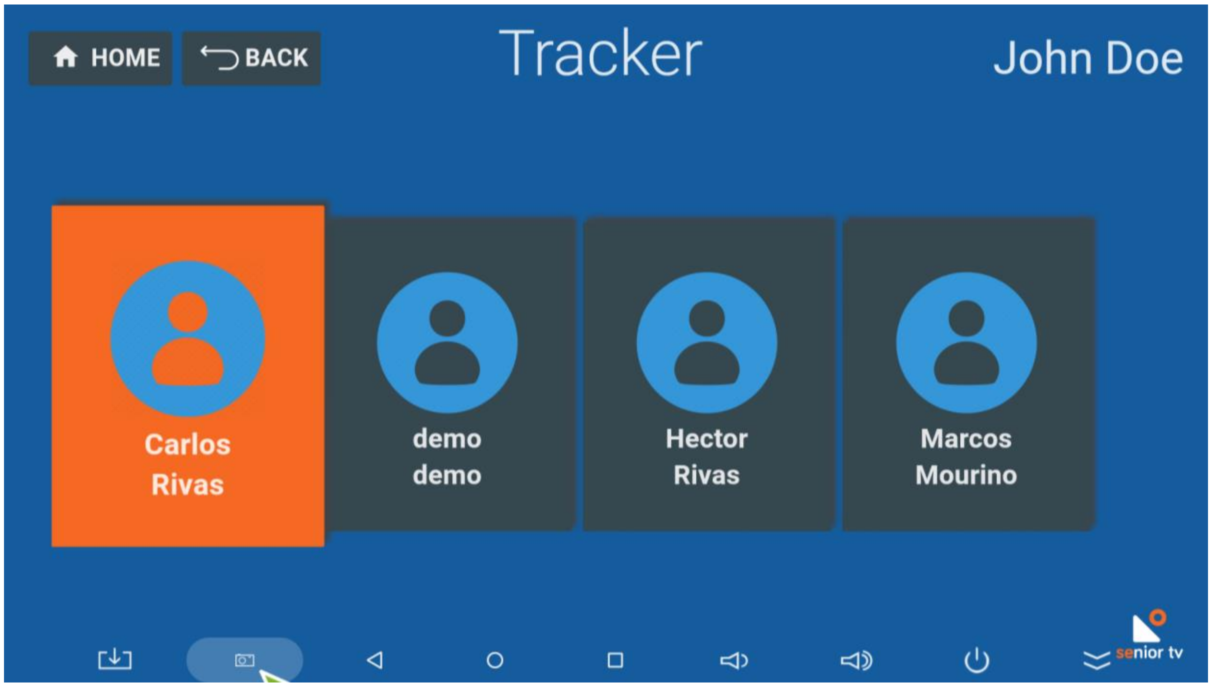


FIGURE 47. TRACKER TV FRIENDS LIST SCREEN

4.2.4.2 Tracker Mobile

The mobile app includes all TV functionalities and adds some more regarding walks recording and "my home" route. These are the specific Tracker Mobile functionalities:

1. Record walk: collects users' walk information, such as: starting time of a walk, GPS positions, walked distance, average speed, ending time and, total walked time.
   - Start walk: Once the walk starts, the application gets the current GPS position, and sends to the Tracker Cloud the starting time and the initial position of the walk. Then, every 15 seconds, the application gets the current GPS position and sends it to the Cloud. Besides, the application calculates the current distance walked and average speed of the walk.
   - Pause walk: allows users to stop the data collection. While the walk is paused, the application does not send GPS information to the Cloud.
   - Resume walk: allows restore the data collection.
   - Stop walk: finalises the data collection. The application sends to Tracker Cloud the ending time, the total walked distance, the average speed and, the total walked time.
2. Set home location: allows users to establish the location of their homes. Every user can only have one *Home position*. Therefore, when the user sets a home location the previous information will be deleted.

3. Help user to go home: shows the route and indications from senior's location to their home. This functionality is only available after seniors' *Home location* set.

The following charts show the Tracker TV app architecture and some interface screens:

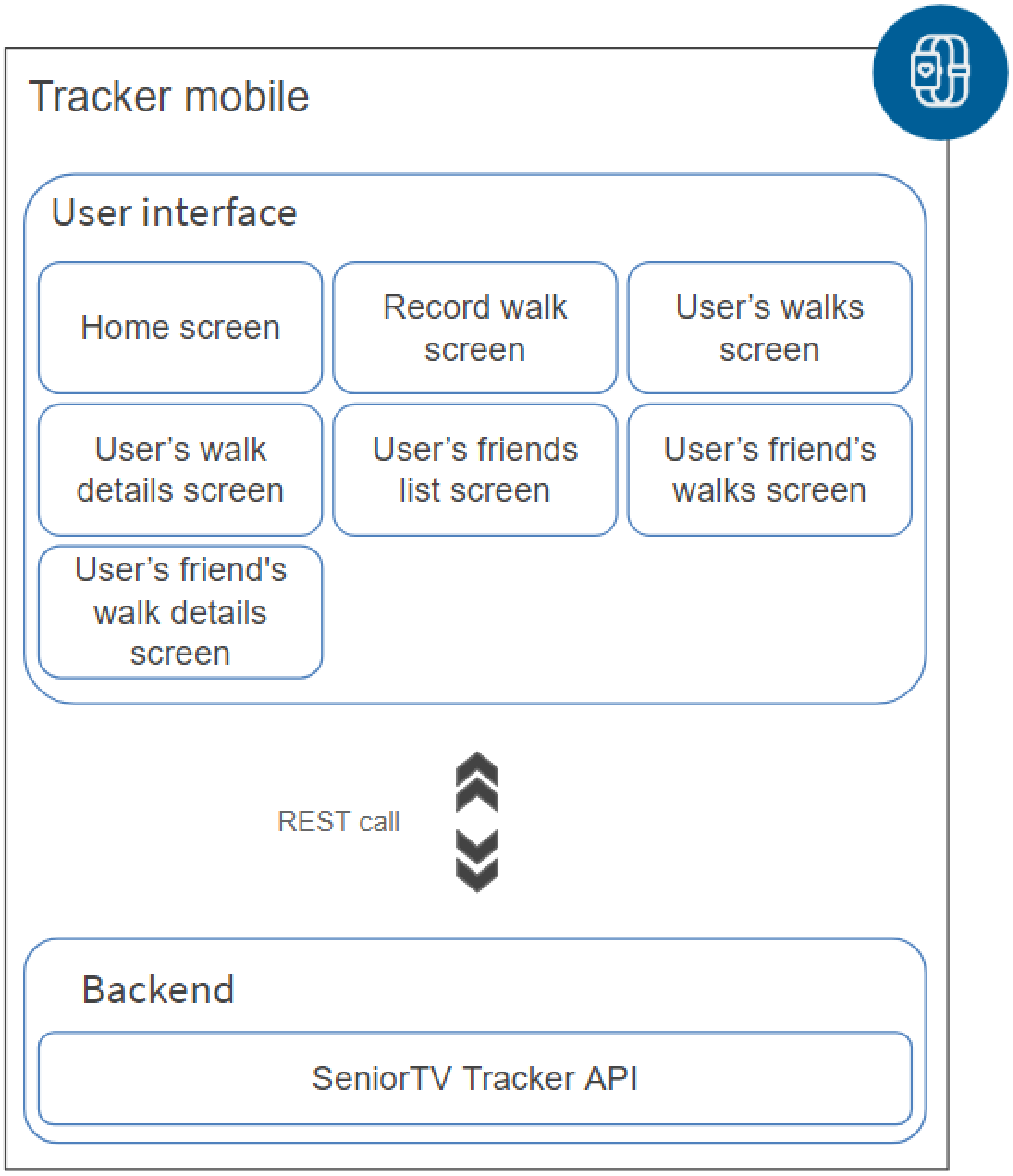

Tracker mobile
User interface
Home screen
Record walk screen
User's walks screen
User's walk details screen
User's friends list screen
User's friend's walks screen
User's friend's walk details screen
REST call
Backend
SeniorTV Tracker API

FIGURE 48. TRACKER TV ARCHITECTURE

1. Home screen

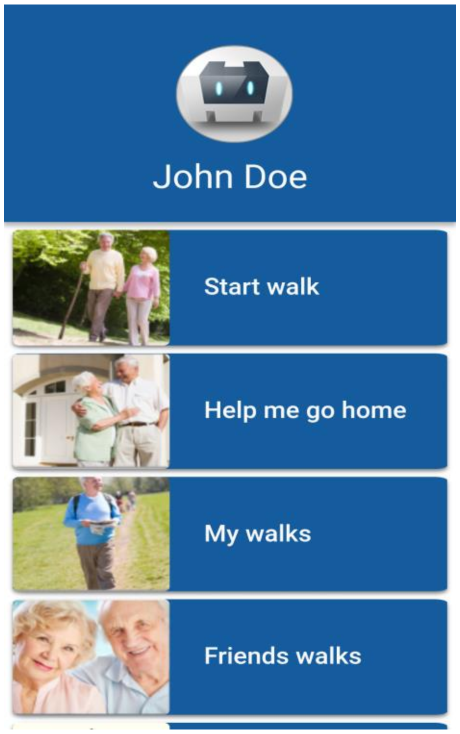


FIGURE 49. TRACKER MOBILE HOME SCREEN

2. Record walk screen

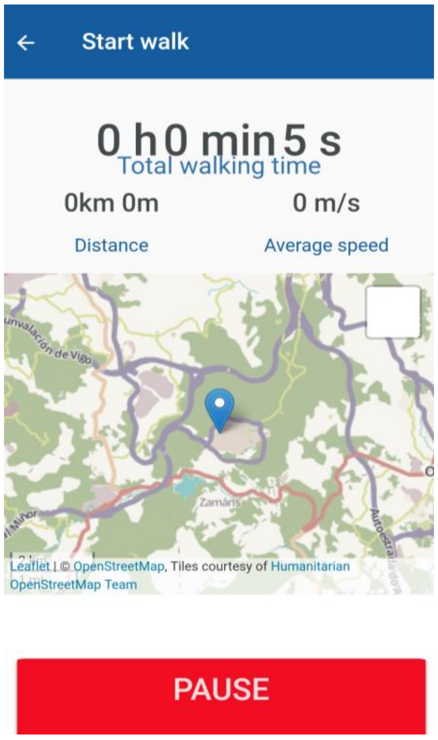
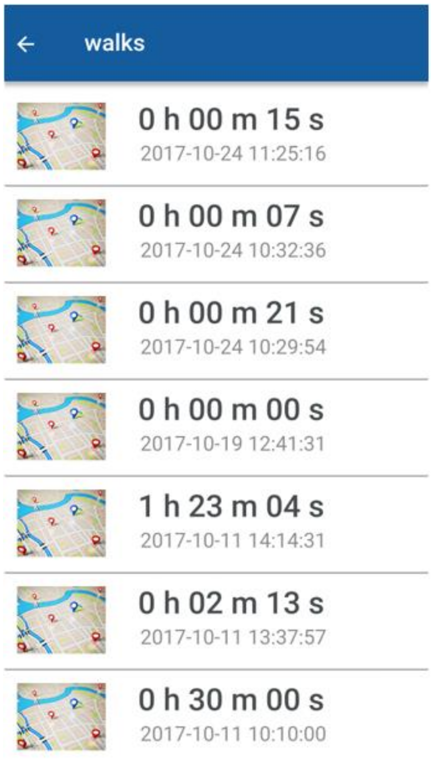


FIGURE 50. TRACKER MOBILE RECORD WALK SCREEN

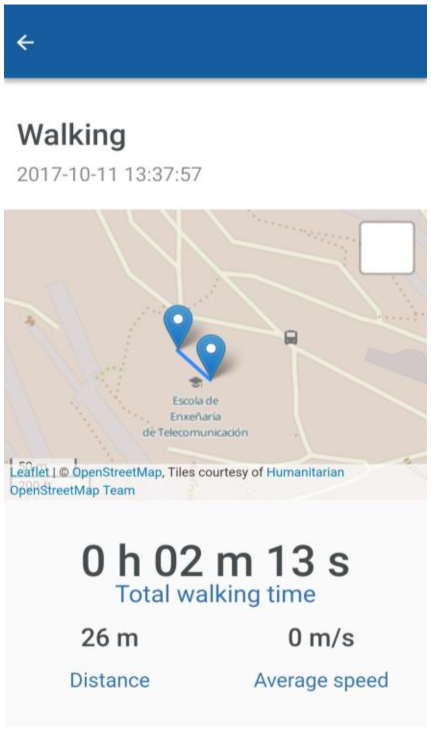
3. Walk list screen


walks

0 h 00 m 15 s	2017-10-24 11:25:16
0 h 00 m 07 s	2017-10-24 10:32:36
0 h 00 m 21 s	2017-10-24 10:29:54
0 h 00 m 00 s	2017-10-19 12:41:31
1 h 23 m 04 s	2017-10-11 14:14:31
0 h 02 m 13 s	2017-10-11 13:37:57
0 h 30 m 00 s	2017-10-11 10:10:00

FIGURE 51. TRACKER MOBILE WALK LIST SCREEN

4. Walk details screen


Walking

2017-10-11 13:37:57

Escola de Enxeñaría de Telecomunicación
Leaflet | © OpenStreetMap, Tiles courtesy of Humanitarian OpenStreetMap Team

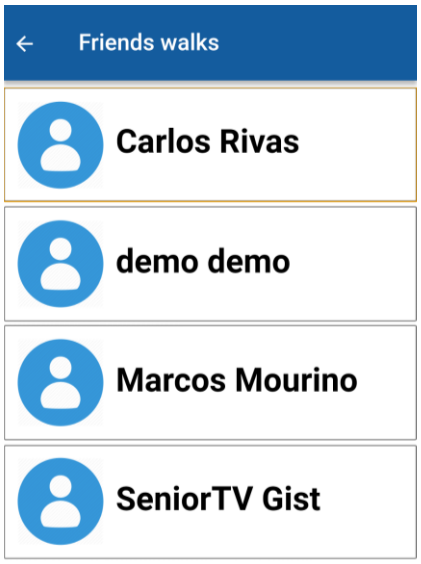
0 h 02 m 13 s
Total walking time

26 m
Distance

0 m/s
Average speed

FIGURE 52. TRACKER MOBILE WALK DETAILS SCREEN

5. Friend list screen


Friends walks

Carlos Rivas

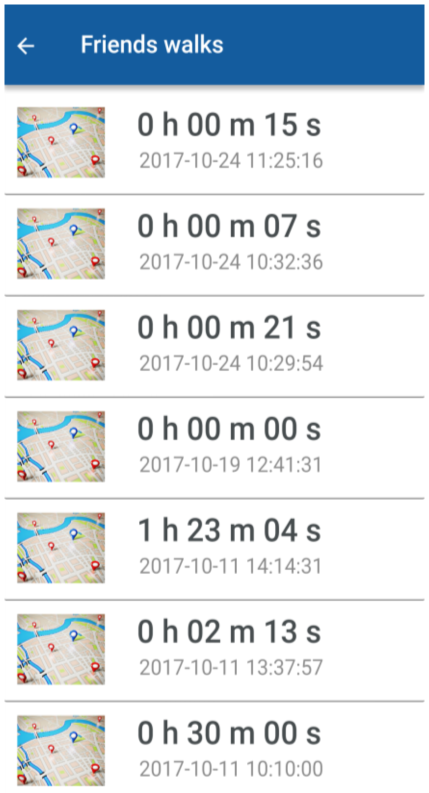
demo demo

Marcos Mourino

SeniorTV Gist

FIGURE 53. TRACKER MOBILE FRIEND LIST SCREEN

6. Friend walks list screen


Friends walks

0 h 00 m 15 s
2017-10-24 11:25:16

0 h 00 m 07 s
2017-10-24 10:32:36

0 h 00 m 21 s
2017-10-24 10:29:54

0 h 00 m 00 s
2017-10-19 12:41:31

1 h 23 m 04 s
2017-10-11 14:14:31

0 h 02 m 13 s
2017-10-11 13:37:57

0 h 30 m 00 s
2017-10-11 10:10:00

FIGURE 54. TRACKER MOBILE FRIEND WALKS LIST SCREEN

4.2.5 Virtual Center

There are many reasons why an older person should stay at home reducing their contact with the outside and disrupting their daily routine. The Virtual Center service allows seniors to keep contact with their daily care center from their home through the TV. Elderly associations can stream the activities which take place in their facilities (courses, gym maintenance, etc.) and seniors will be able to follow these activities without need to move until their center.

This service also may be used as a complement of daily care center activities. For example, doing morning exercises at the center and evening exercises at home.

The Virtual Center service is composed by two elements: the Streamer and the TV app. The TV app is focussed on seniors and the Streamer is oriented to daily care centres. The following chart shows the architecture of the service:

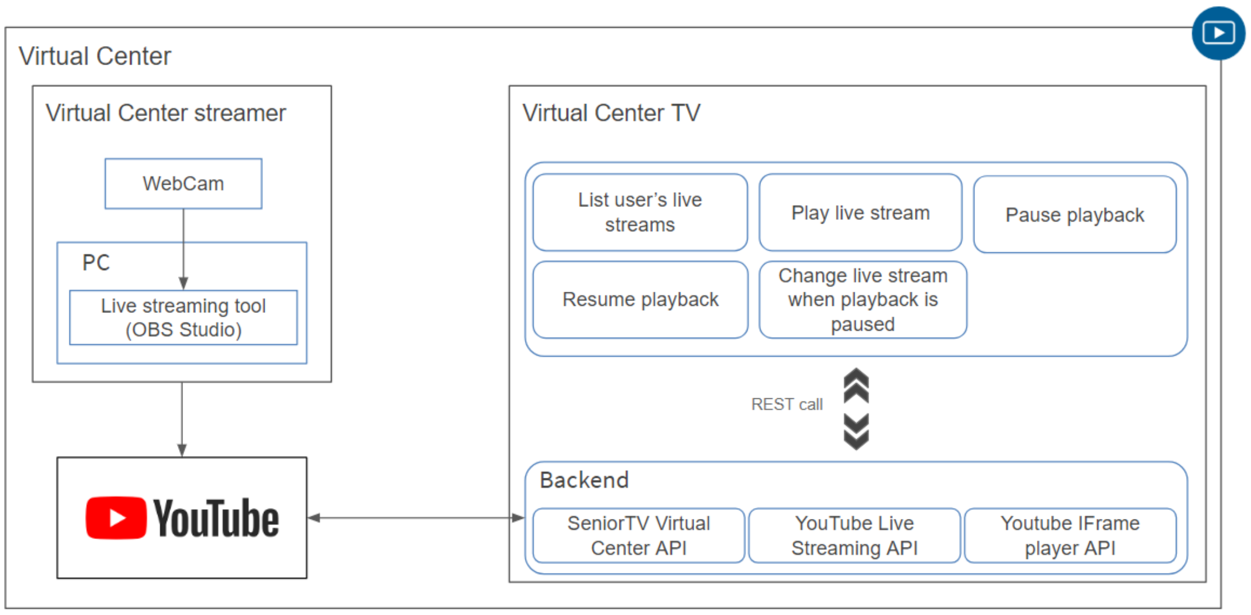


FIGURE 55. VIRTUAL CENTER GENERAL ARCHITECTURE

4.2.5.1 Virtual Center Streamer

The Streamer is responsible to stream the elderly centre activities. It is composed by a webcam, a microphone and a computer with a live streaming tool installed. In particularly, in this project the *Open Broadcaster Software Studio*[19] (OBSS) is used because it is an open source tool for Windows, macOS and Linux.

[19] Open Broadcaster Software Studio official site: https://obsproject.com/

The OBSS tool allows to stream audio and video using through a channel YouTube[20]. YouTube is a video-sharing service that allows users to upload, view, rate, share, add to favourites, report, comment on videos, and subscribe to other users. Among the features provided by YouTube, it allows to create live streams in order users can real-time broadcast events. Also, YouTube has an open Live Streaming API[21] which permits update and manage live events easily.

4.2.5.2 Virtual Center TV

This application is responsible for playing on the TV the Virtual Center streams. Seniors can navigate through the channels and selected one of them to watch the current online streaming.

The next chart shows the Weather TV architecture. The cloud backend is composed by three services:

- SENIOR-TV Virtual Center API: allows manage the seniors channels.
- YouTube Live Streaming API: lets create, update, and manage live events on YouTube. It allows schedule events and relate them to video streams, which represent the broadcast content.
- Youtube IFrame player API: permits to embed a YouTube video player on a website and control the player using JavaScript. Queue videos for playback (play, pause, or stop videos), adjust the player volume or retrieve information about the video being played.

[20] YouTube official site: https://www.youtube.com/
[21] YouTube Live Streaming API site: https://developers.google.com/youtube/v3/live/getting-started

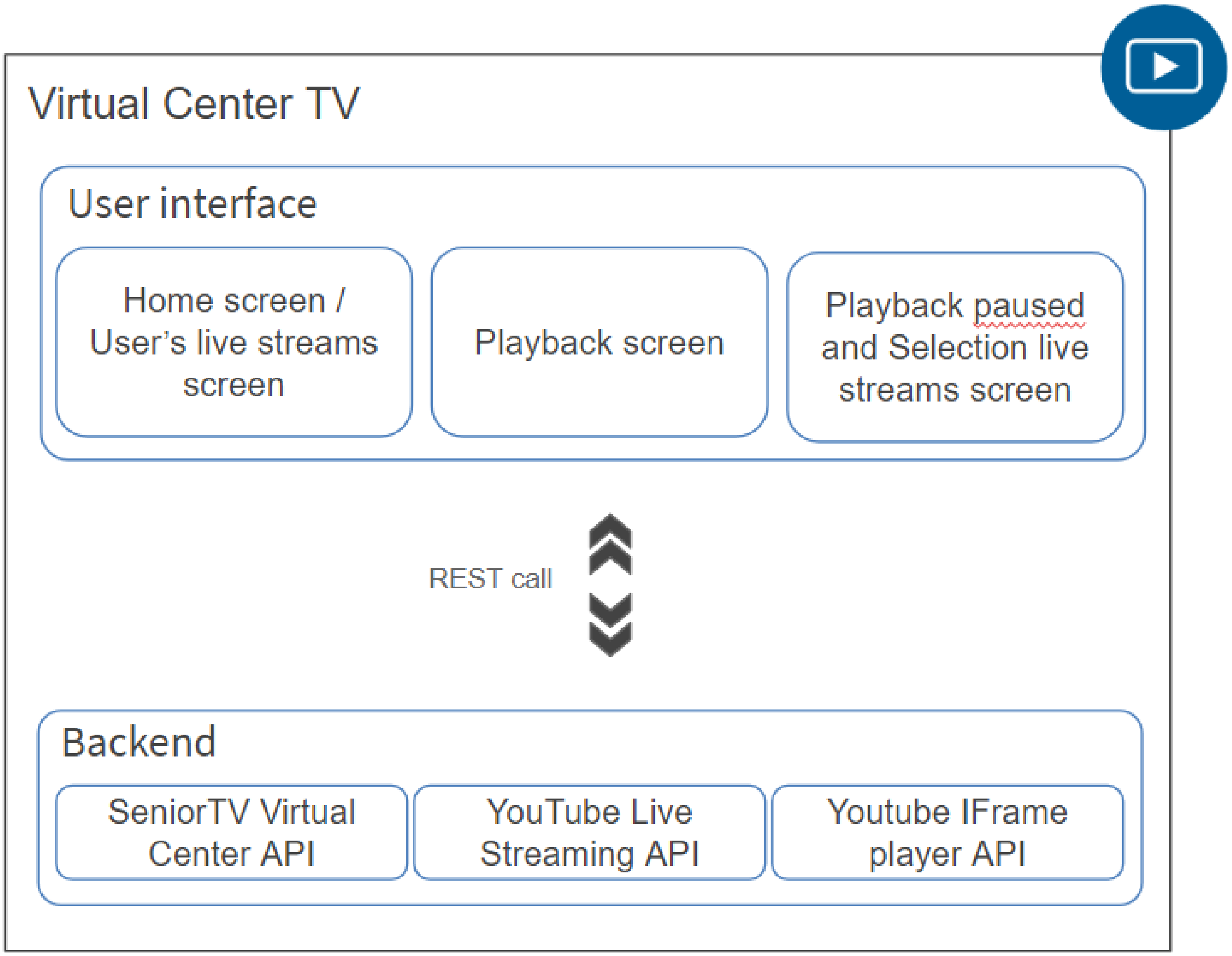

Virtual Center TV
User interface
Home screen / User's live streams screen
Playback screen
Playback paused and Selection live streams screen
REST call
Backend
SeniorTV Virtual Center API
YouTube Live Streaming API
Youtube IFrame player API

FIGURE 56. VIRTUAL CENTER TV ARCHITECTURE

The Virtual Center TV app has been implemented using Angular and Apache Cordova frameworks. These are the app main functionalities:

1. Play live stream: allows the user to play a specific live stream.
2. Pause playback: allows the user to pause the playback of the current live stream using the Youtube IFrame Player API.
3. Resume playback: allows the user to resume the playback of the current live stream using the Youtube IFrame Player API.
4. Change to another live stream: when the playback is paused, seniors can change to another live stream channel from the video streaming screen.

The following charts show the main interface screens of the TV app:

1. Home screen

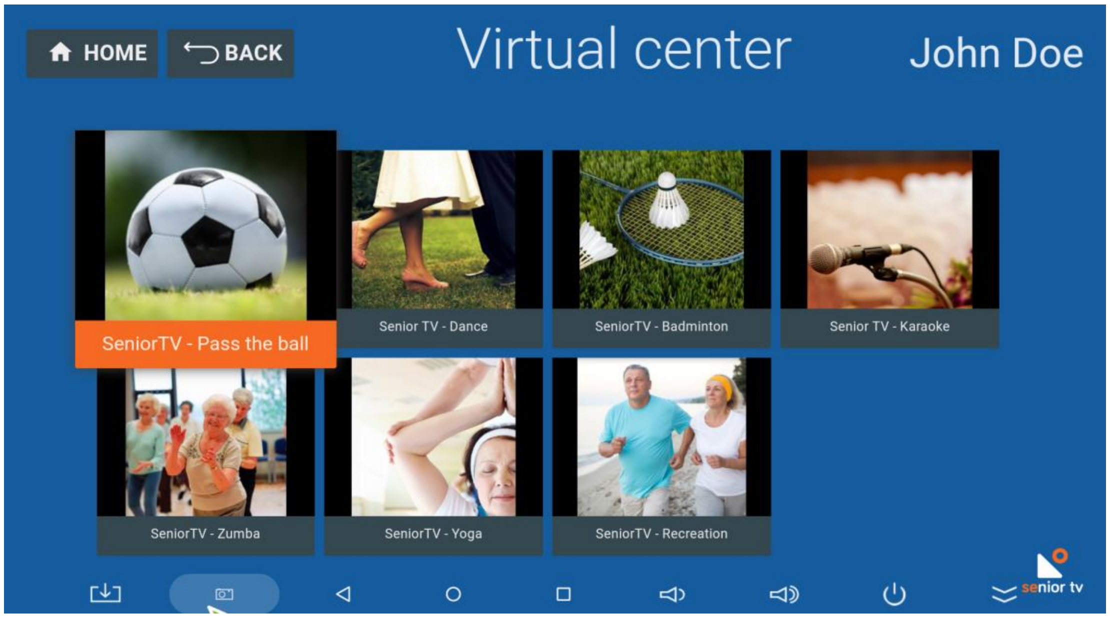


FIGURE 57. VIRTUAL CENTER HOME SCREEN

2. Playback screen

FIGURE 58. VIRTUAL CENTER PLAY BACK SCREEN

3. Pause playback and list channels screen

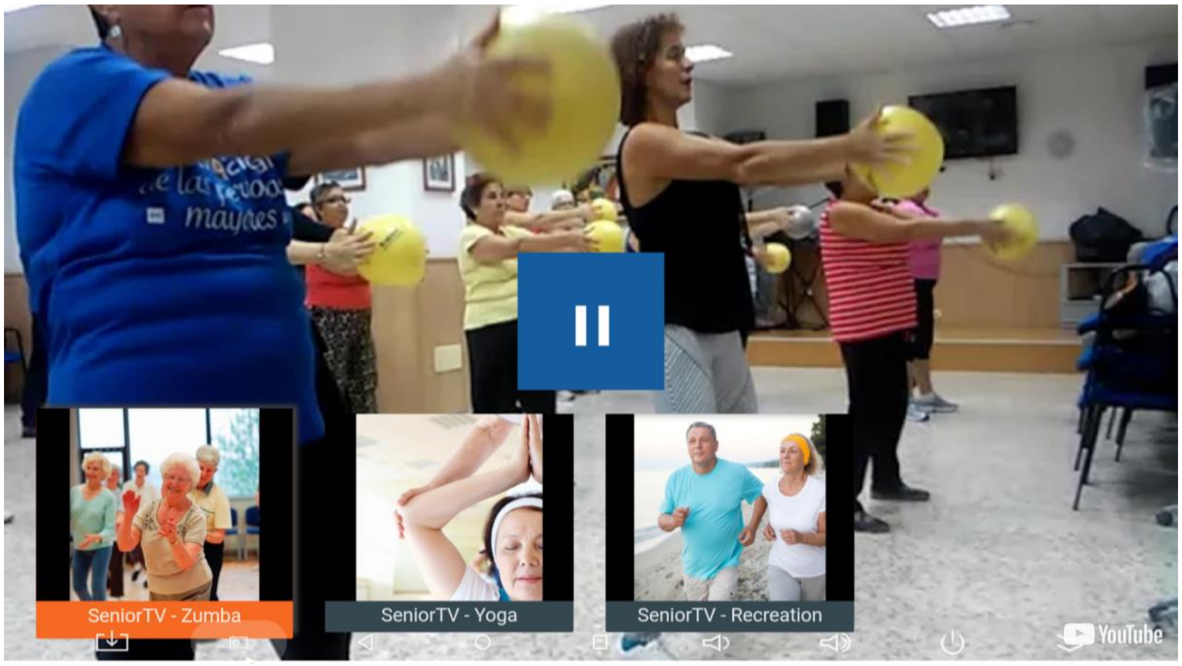


FIGURE 59. VIRTUAL CENTER PAUSE PLAYBACK AND LIST CHANNELS SCREEN

4.2.6 Wikipedia

Wikipedia is a global, free and multilingual online encyclopaedia, with the mission of allowing anyone to read, create or edit articles. It currently houses more than 46 million articles distributed in 277 editions, being the largest and most popular general reference work on the Internet.

Access to Wikipedia allows seniors to access a large amount of information increasing their contact with external world. The Wikipedia service is a simplified senior-oriented Wikipedia client for TV. The service is designed to interact with the well-known web service in an easy and simple way using the TV. Users can save their usual searches or mark the articles as favourite to read later. Besides, they can use different Wikipedia versions depending upon the language (e.g. the English version is much larger than other minority languages).

The Wikipedia service consists only of the Wikipedia TV app which has been developed using Javascript programming language and the Apache Cordova framework. The application fetches the data via HTTP requests to the public MediaWiki action API[22]. This API provides convenient access to wiki features, data, and metadata over HTTP requests.

The following chart shows the service architecture which comprises the functionalities described below.

[22] MediaWiki action API official side: https://www.mediawiki.org/wiki/API:Main_page/en

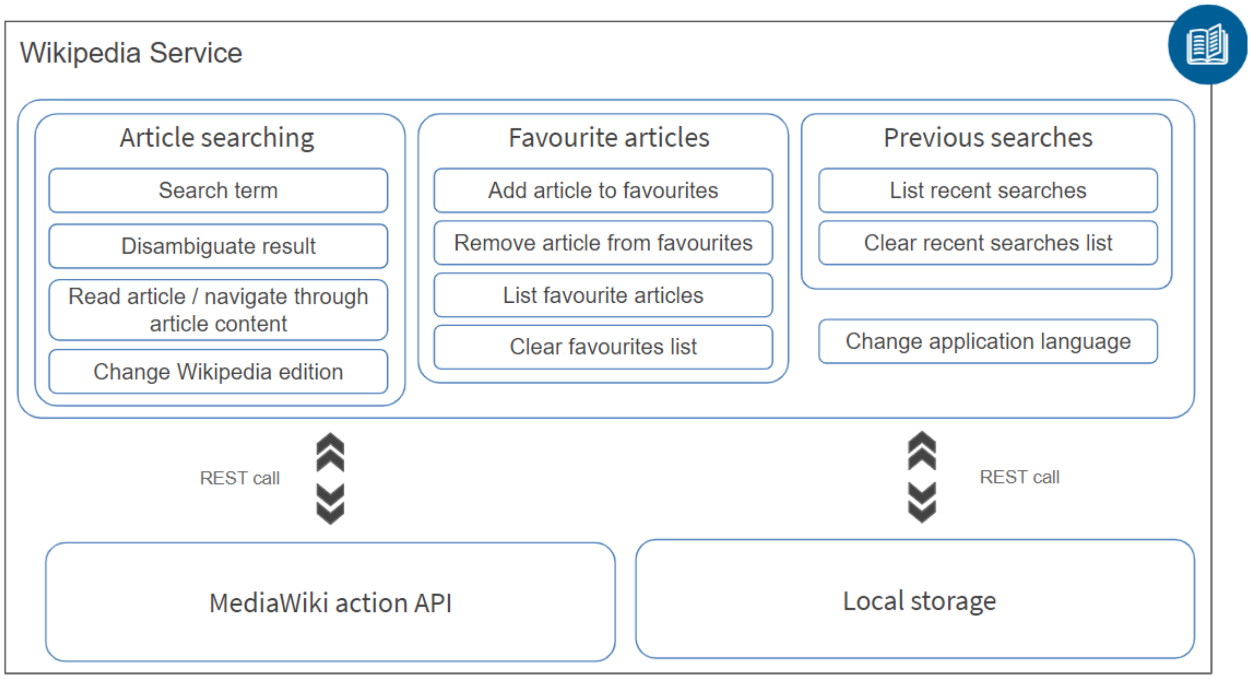


FIGURE 60. WIKIPEDIA GENERAL ARCHITECTURE

The service functionalities are organized in three groups: article searching, favorite articles, previous searches.

1. Search an article: allows to look for a specific article in a particular Wikipedia version.
2. Read article or Navigate through article content: permits to read the selected Wikipedia article content and to navigate through its different sections.
3. Manage favorite articles: seniors can bookmark articles in order to access them later.
4. Manage recent articles: the TV app stores the latest read articles references automatically. Users can access to the recently viewed articles list in order to repeat the searches without typing search terms specifically.
5. Change Wikipedia edition: allows users to perform requests on different Wikipedia editions. The current available editions are English, Spanish, Greek, Romanian, and Slovenian.
6. Change application language: permits to translate the display texts to several languages: English, Spanish, Greek, Romanian, and Slovenian.

Following, the main TV app screenshots are shown:

1. Home screen

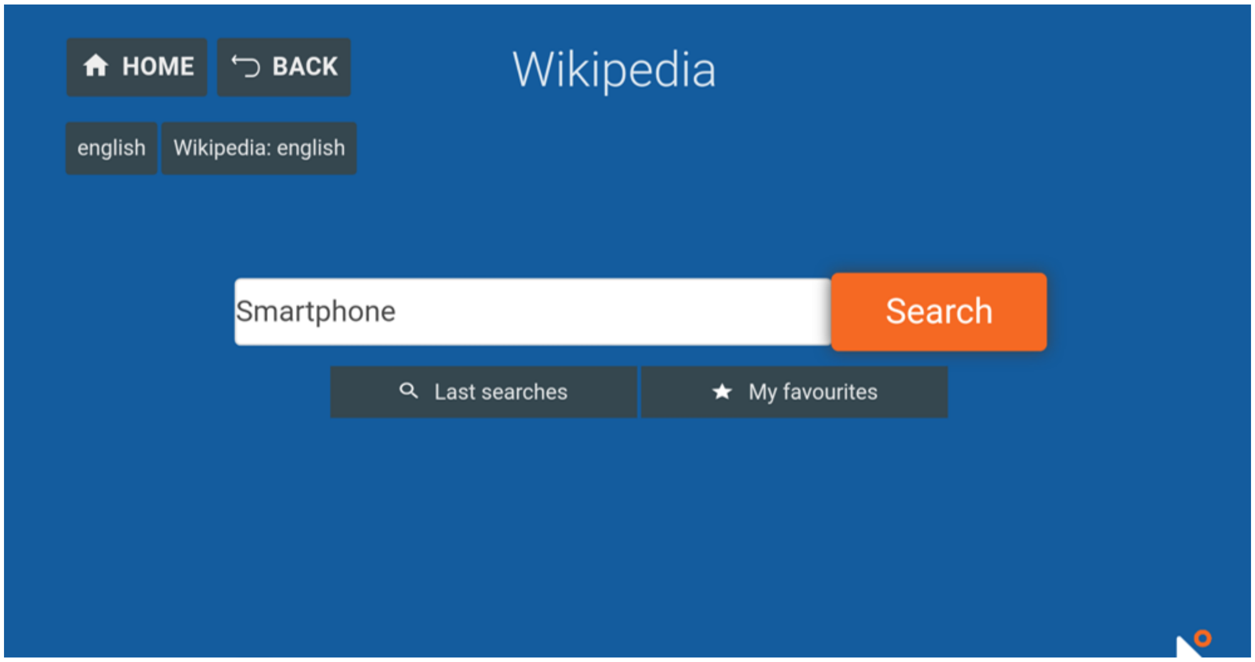
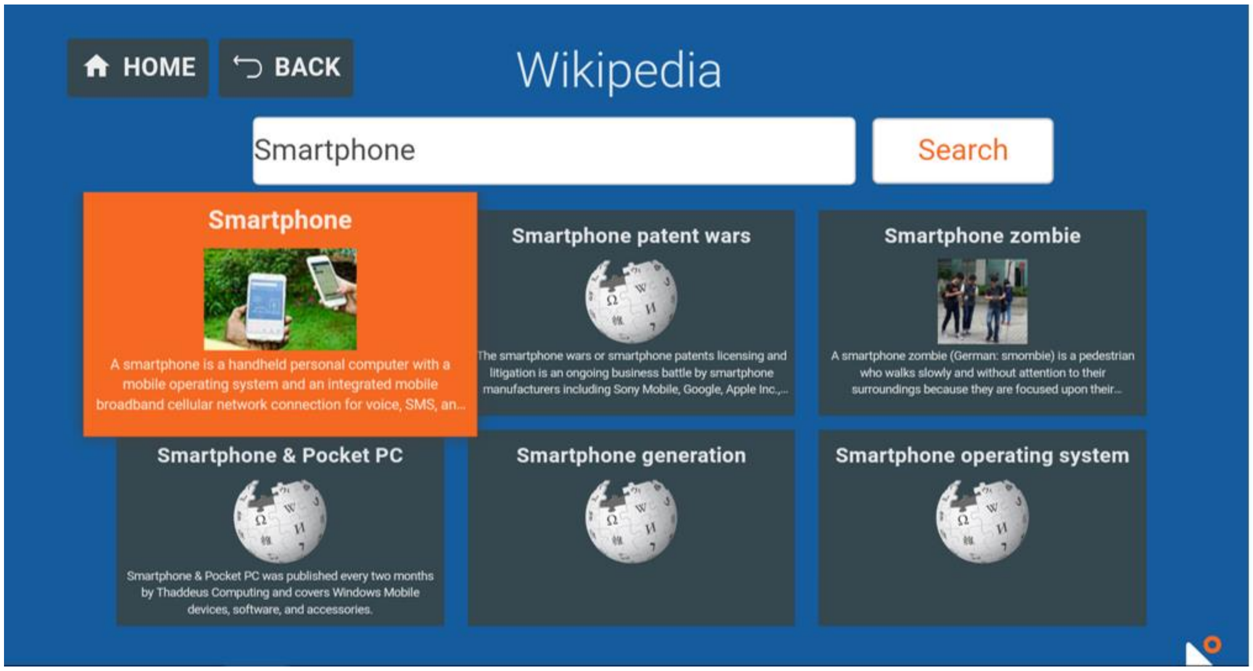


FIGURE 61. WIKIPEDIA HOME SCREEN

2. Search results screen: implements the “Disambiguate result” functionality



A smartphone is a handheld personal computer with a mobile operating system and an integrated mobile broadband cellular network connection for voice, SMS, an...

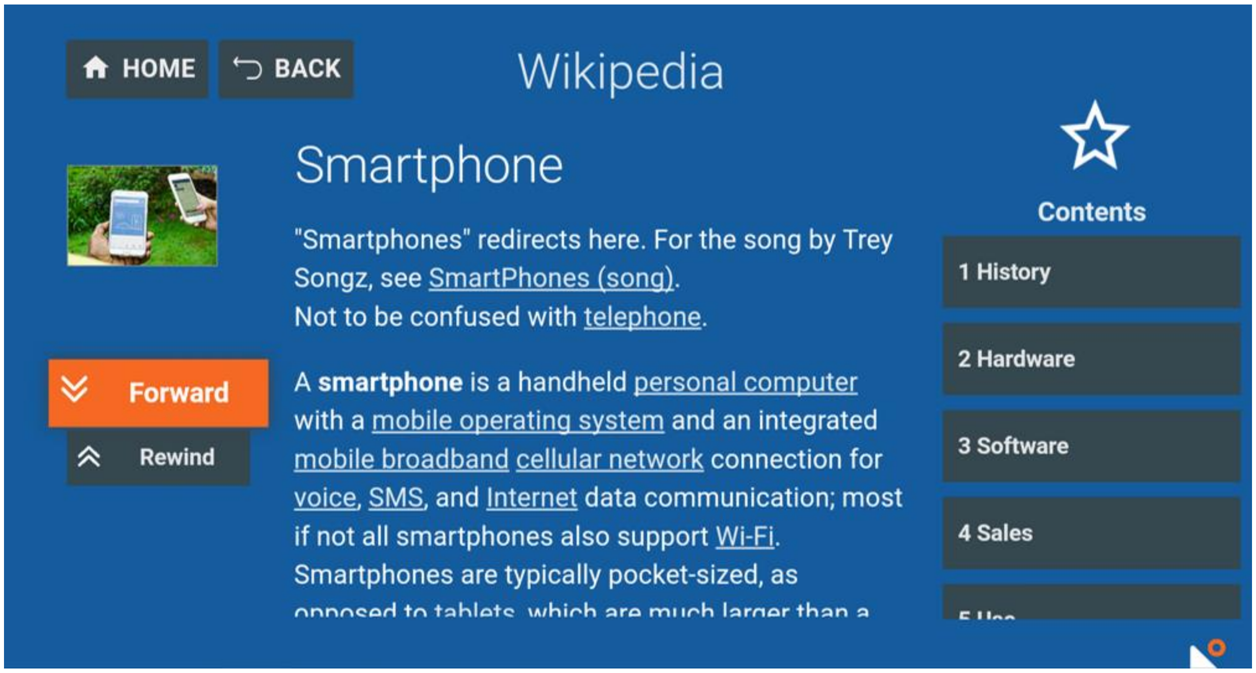
The smartphone wars or smartphone patents licensing and litigation is an ongoing business battle by smartphone manufacturers including Sony Mobile, Google, Apple Inc.,...

A smartphone zombie (German: smombie) is a pedestrian who walks slowly and without attention to their surroundings because they are focused upon their...

Smartphone & Pocket PC was published every two months by Thaddeus Computing and covers Windows Mobile devices, software, and accessories.

FIGURE 62. WIKIPEDIA SEARCH RESULTS SCREEN

3. Article Screen: shows the selected article content and some buttons to navigate through it.



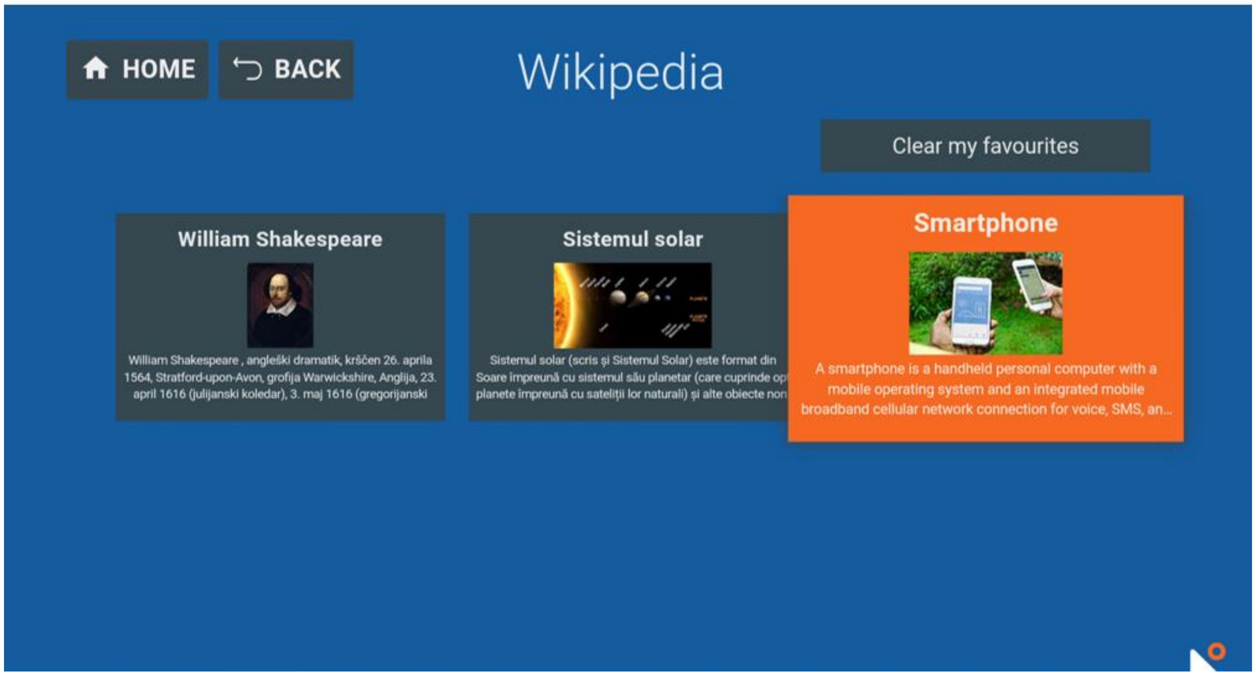
Smartphone

"Smartphones" redirects here. For the song by Trey Songz, see SmartPhones (song).
Not to be confused with telephone.

A **smartphone** is a handheld personal computer with a mobile operating system and an integrated mobile broadband cellular network connection for voice, SMS, and Internet data communication; most if not all smartphones also support Wi-Fi. Smartphones are typically pocket-sized, as

FIGURE 63. WIKIPEDIA ARTICLE SCREEN

4. Favourites screen



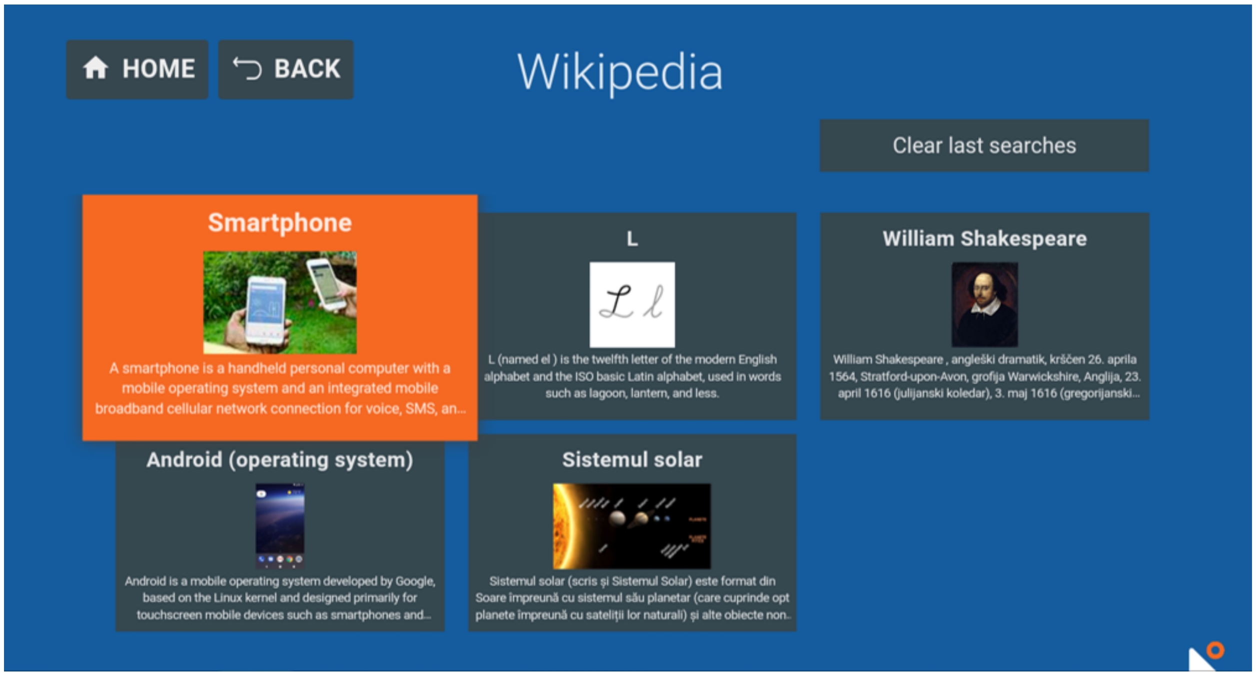
William Shakespeare , angleški dramatik, krščen 26. aprila 1564, Stratford-upon-Avon, grofija Warwickshire, Anglija, 23. april 1616 (julijanski koledar), 3. maj 1616 (gregorijanski

Sistemul solar (scris și Sistemul Solar) este format din Soare împreună cu sistemul său planetar (care cuprinde opt planete împreună cu sateliții lor naturali) și alte obiecte non

A smartphone is a handheld personal computer with a mobile operating system and an integrated mobile broadband cellular network connection for voice, SMS, an...

FIGURE 64. WIKIPEDIA FAVOURITES SCREEN

5. Last searches screen



Smartphone

A smartphone is a handheld personal computer with a mobile operating system and an integrated mobile broadband cellular network connection for voice, SMS, an...

L

L (named el) is the twelfth letter of the modern English alphabet and the ISO basic Latin alphabet, used in words such as lagoon, lantern, and less.

William Shakespeare

William Shakespeare , angleški dramatik, krščen 26. aprila 1564, Stratford-upon-Avon, grofija Warwickshire, Anglija, 23. april 1616 (julijanski koledar), 3. maj 1616 (gregorijanski...

Android (operating system)

Android is a mobile operating system developed by Google, based on the Linux kernel and designed primarily for touchscreen mobile devices such as smartphones and...

Sistemul solar

Sistemul solar (scris şi Sistemul Solar) este format din Soare împreună cu sistemul său planetar (care cuprinde opt planete împreună cu sateliții lor naturali) şi alte obiecte non...

FIGURE 65. WIKIPEDIA LAST SEARCHES SCREEN

6. Change language or Change Edition screen

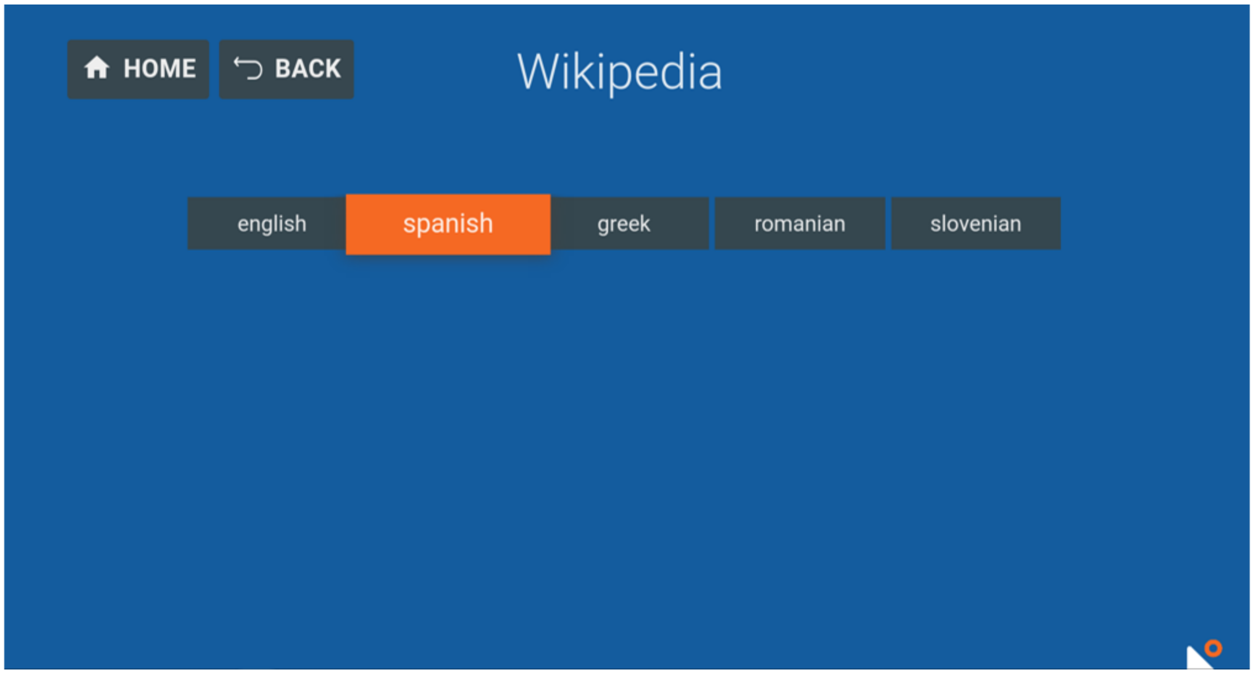


FIGURE 66. WIKIPEDIA CHANGE LANGUAGE OR CHANGE EDITION SCREEN

4.2.7 Weather 2.0

This service is an upgrade of Weather service developed to first pilot. The initial version was only used to get the seniors feedback about the interface usability the remote-control interaction and also the service itself.

The new version of this service is composed by two different modules: a TV and a Web app. The TV app is focussed on seniors and the Web app is oriented to secondary users. The following chart shows the service general architecture:

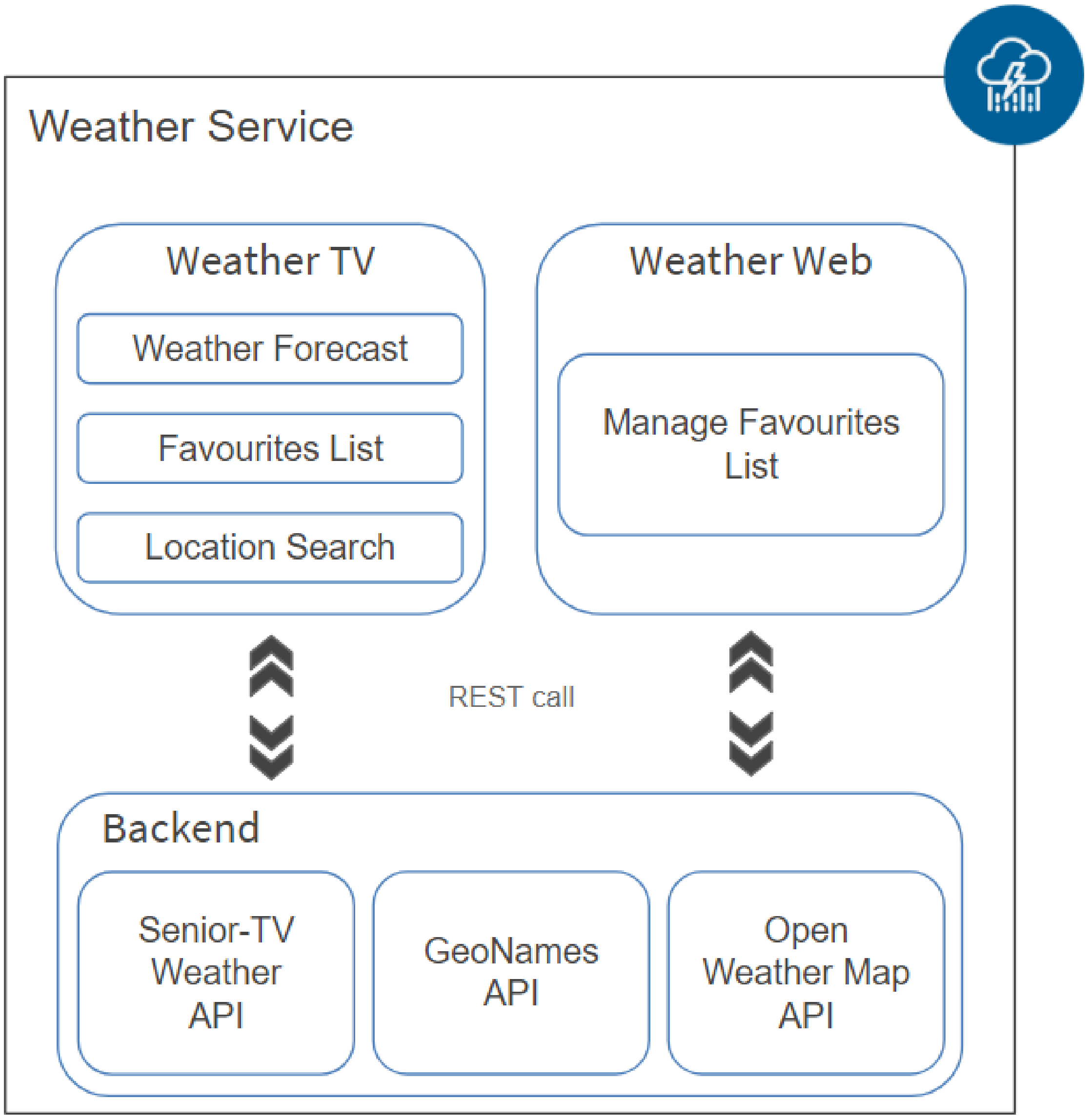


FIGURE 67. WEATHER GENERAL ARCHITECTURE

Both, Weather TV and Web, have been developed using Angular and Apache Cordova frameworks. The cloud backend has been developed using Java EE[23] technology. The two apps use HTTP requests to communicate with the backend to obtain the seniors' favorite locations. The cloud backend is composed by three services:

- SENIOR-TV Weather API: allows manage the favorite locations (add, remove or update the database information).
- GeoNames[24] API: is used to translate the location names to their geographical latitude and longitude coordinates to avoid potential spelling errors. A location name could be different spellings depending on language. This issue can cause errors in the forecasts given by OpenWeatherMap API because it cannot identify the selected location.
- OpenWeatherMap[25] API: obtains the selected location weather forecast using the geographic coordinates obtained previously from GeoNames API.

The next paragraphs include the TV and Web applications architecture and the main interface screens description.

4.2.7.1 Weather TV

The scope of the TV app has not change from the initial version: use the TV screen to obtain the weather forecast of senior's location or other locations. Although the version 2.0 incorporates some new functionalities:

1. Search other locations forecast: users have to type the location name using the virtual keyboard.
2. Manage favourite locations: users can mark a location as favourite to not retype the location name any more.

The next chart shows the Weather TV architecture:

[23] Java EE documentation: https://www.oracle.com/technetwork/java/javaee/documentation/index.html
[24] GeoNames official site: http://www.geonames.org/
[25] OpenWeatherMap official site: https://openweathermap.org/

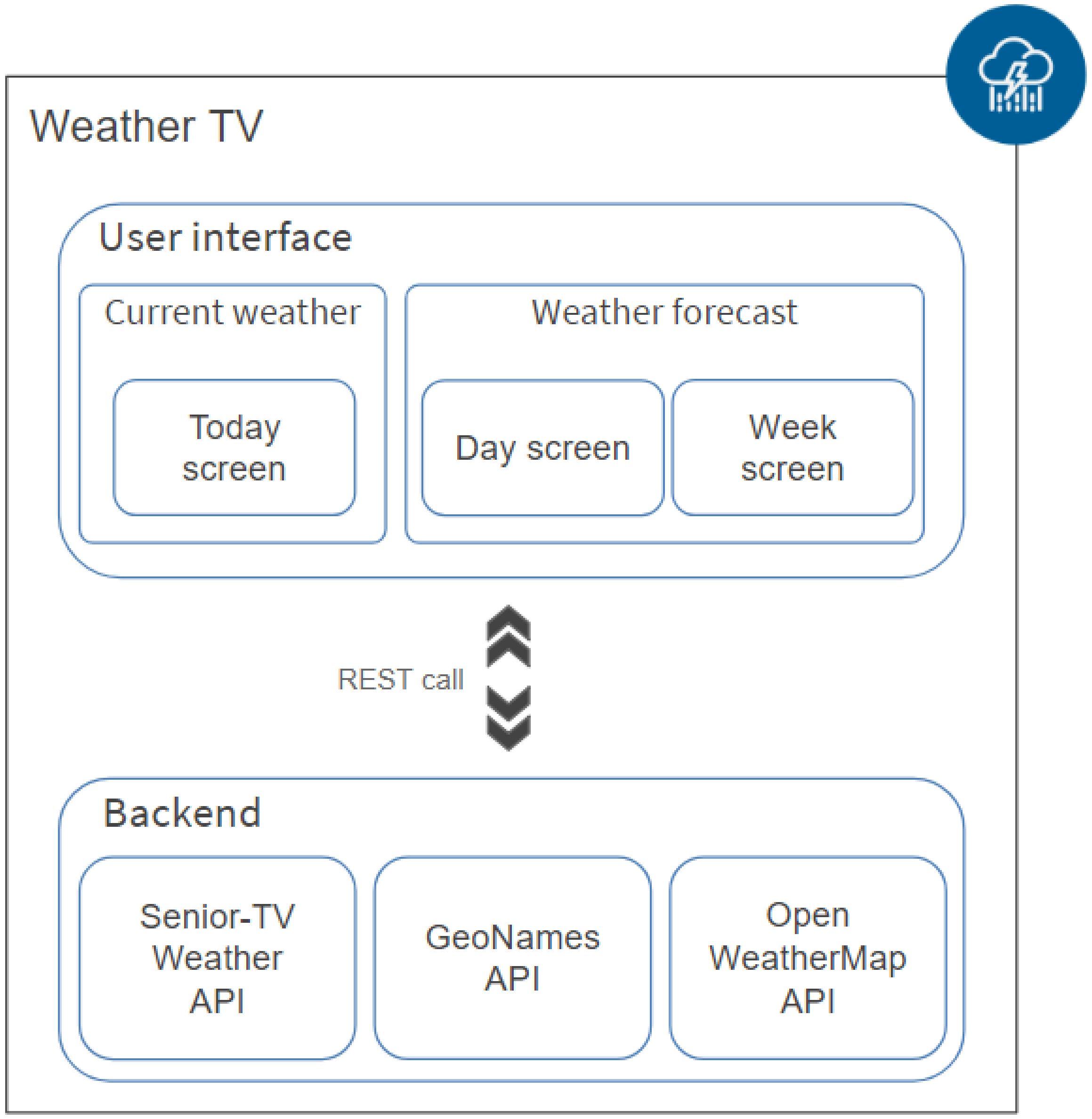


FIGURE 68. WEATHER TV GENERAL ARCHITECTURE

A new interface design was created for the Weather 2.0 to adapt the color palette and font sizes to the suggestions made by elderly during first pilot. The main screens are included following:

1. Daily forecast screen

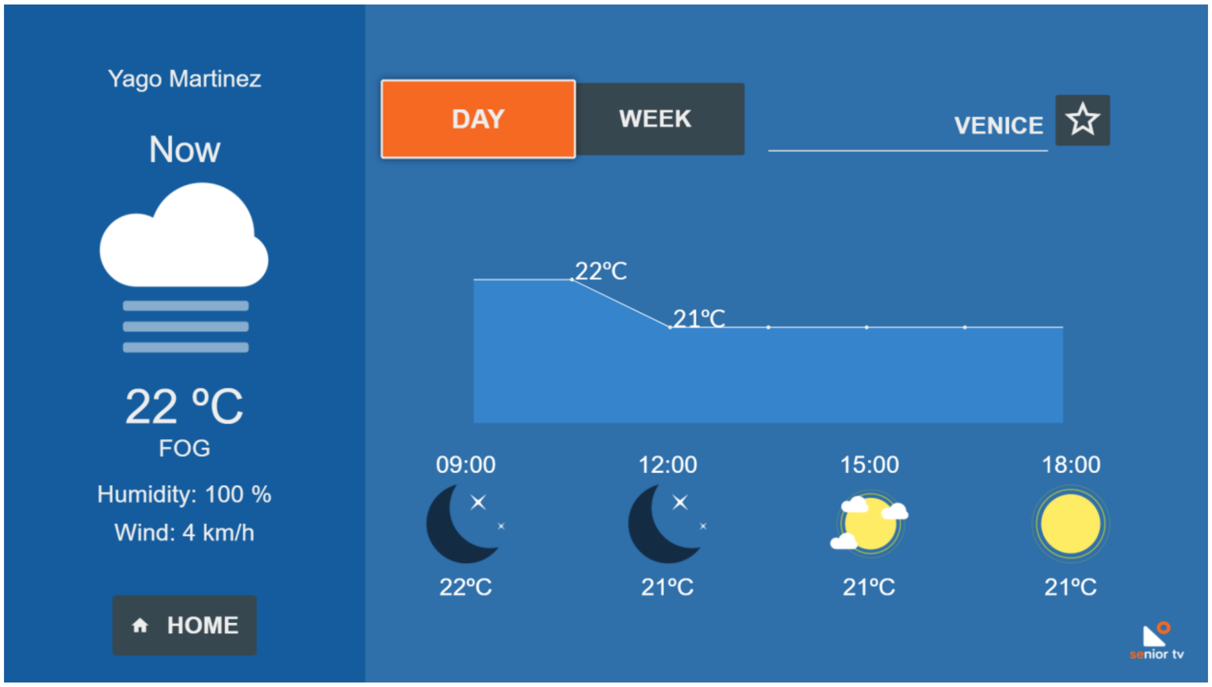
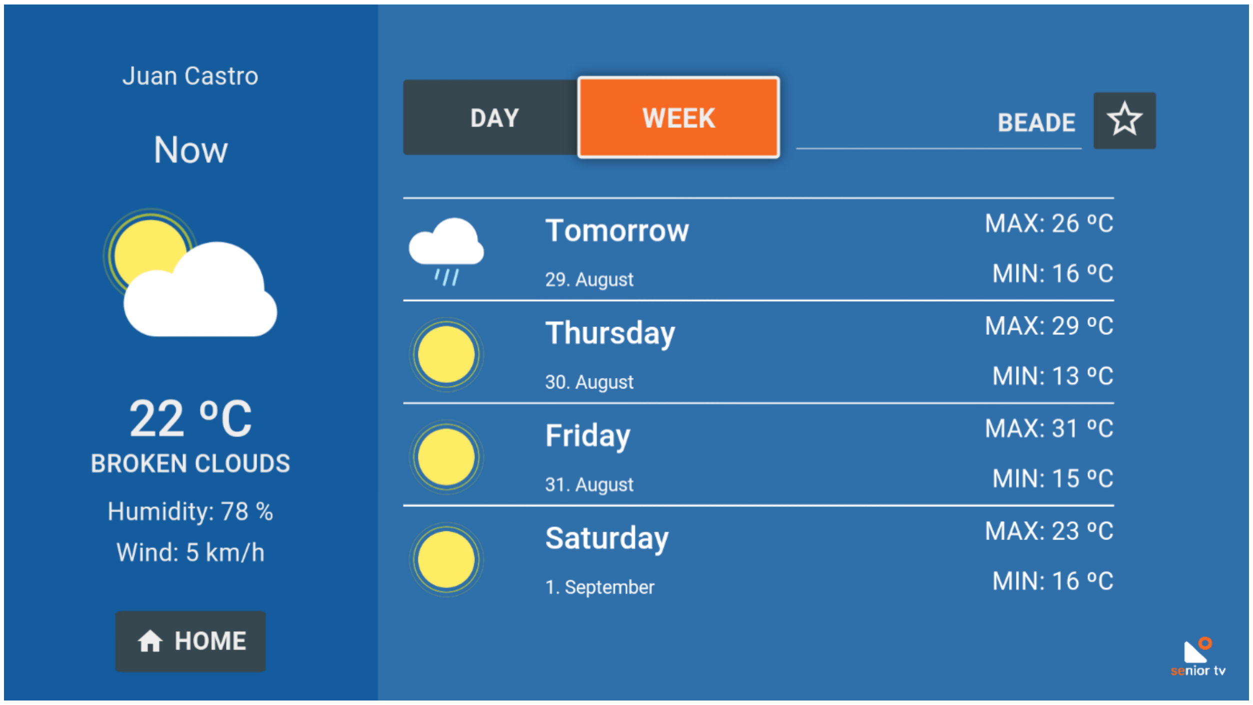


FIGURE 69. WEATHER TV DAILY FORECAST

2. Weekly forecast screen



Tomorrow	29. August	MAX: 26 °C / MIN: 16 °C
Thursday	30. August	MAX: 29 °C / MIN: 13 °C
Friday	31. August	MAX: 31 °C / MIN: 15 °C
Saturday	1. September	MAX: 23 °C / MIN: 16 °C

FIGURE 70. WEATHER TV WEEKLY FORECAST

4.2.7.2 Weather Web

The Weather Web app allows managing the seniors' favourite locations. This application is focussed to secondary users (caregivers or relatives) but primary users can also access to the app. These are the app main functionalities:

1. Add a favourite location to a particular user.
2. Remove elements from to a user favourite list.

The next charts show the Weather Web architecture and some interface screens:

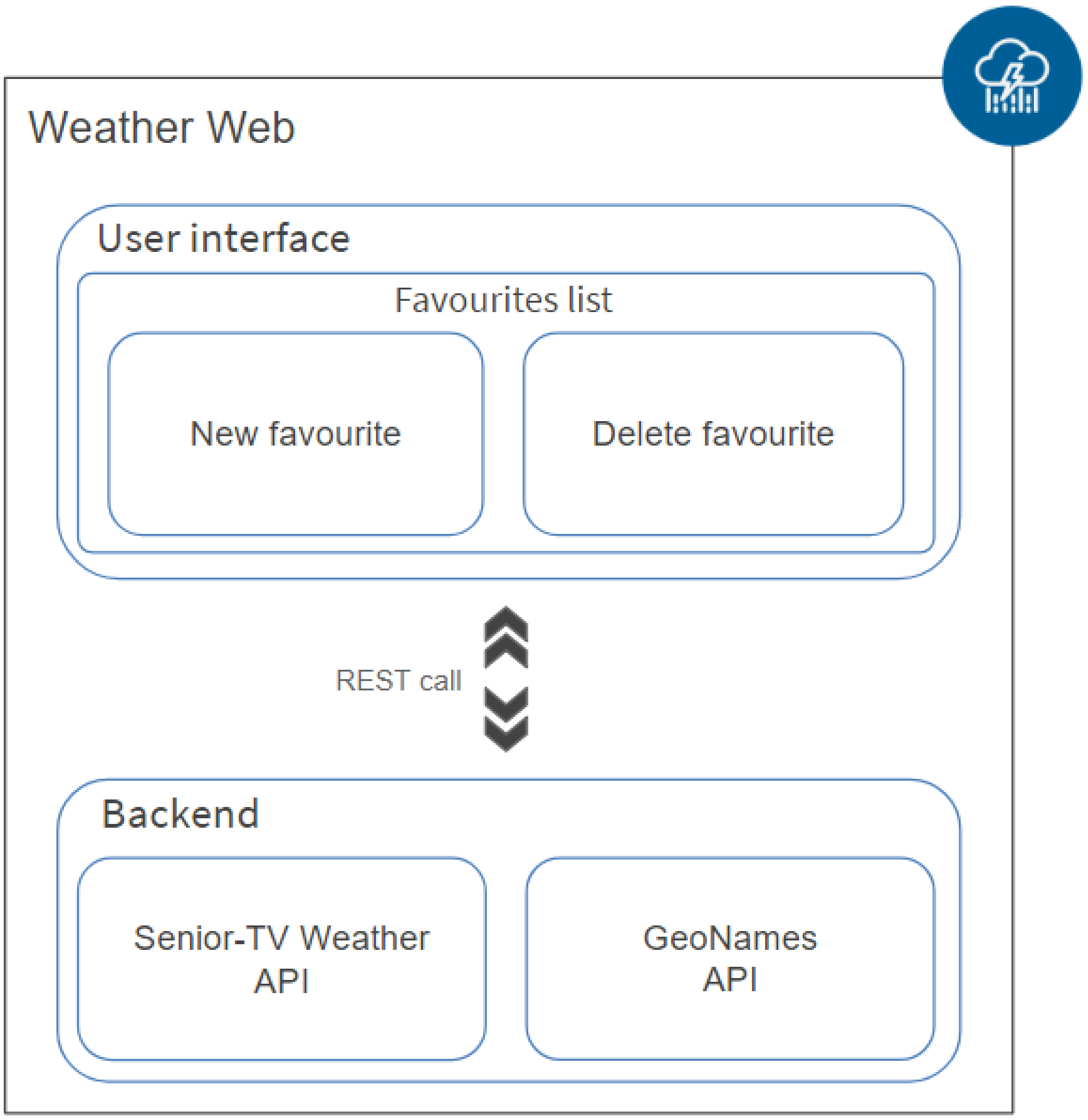
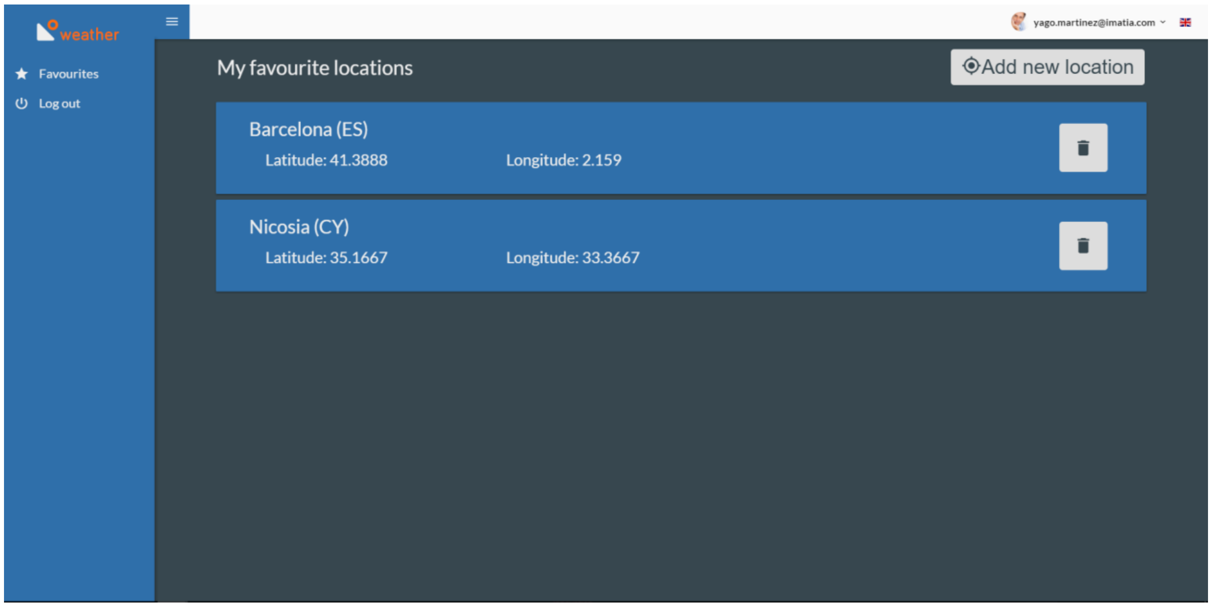


FIGURE 71. WEATHER WEB GENERAL ARCHITECTURE



My favourite locations

Barcelona (ES)
Latitude: 41.3888 Longitude: 2.159

Nicosia (CY)
Latitude: 35.1667 Longitude: 33.3667

FIGURE 72. WEATHER WEB INTERFACE

5. Formal services

The following subsections show the main characteristics and the formal services state after the development cycles (months 1 to 25).

5.1. First pilot

The platform obtained after the first cycle only includes services which run in the smart-TV devices, as the following chart is showing:
During the first cycle of the development of the formal services we analysed the potential possibilities for developing formal apps for the SENIOR-TV platform.

This procedure took place taking in mind many aspects, such as:

- Market status and future trends.
- End users' outcome from the first iteration.
- Extensibility of the app with existing technologies and future trends.
- Integration of sensors so as to be used as input device in the app.
- Data protection, confidentiality and ethics considerations.

During this period the following technical objectives have been identified:

Technical objective	**Details**
Definition of apps, data collection analysis and predictions	One of the formal bundles that we are interested to develop and consists of subcomponents is the health bundle. These consist of: a) Body Weight App b) Blood pressure app c) Pulse rate app d) Blood glucose level The initial idea is that this bundle will be implemented by using a set of sensors which are connected remotely to the STB (if finally, SENIOR-TV will use an STB). The users will have the ability to view the parameters in their screen and if they wish then by just pushing a button they could transmit this info to e.g. their doctor. Then, the doctor through a web API could give feedback, monitor the status of the elderly, etc.

Selection of commercial of the shelf sensors for the selected applications	Selection criteria will include: • Depending on the applications sensor types Power consumption evaluation of sensors and actuators for autonomy evaluation. • Sampling frequency of sensors, accuracy and sample size. • Integration possibilities
Selection of wireless technology for connectivity of the remote devices	Selection criteria will include: • Low-power connectivity solutions. • Depending on the use case corresponding design parameters such as radio transceivers density per square meter, and wireless link distance. • Low packet error rates, and wireless capacity to fulfil sensor sampling/data rates. Mesh and star topologies with the corresponding routing protocols.
Integration of lightweight communication protocol COAP of the remote devices	Traditional Web protocols (HTTP/TCP) create heavy traffic for constrained devices, therefore lightweight Web protocols needed. Constrained Application Protocol (CoAP) is a software protocol intended to be used in very simple electronics devices that allows them to communicate interactively over the Internet.
Integration of 6LoWPAN in order the Internet Protocol to be applied even to the smallest devices.	Low-power devices with limited processing capabilities should be able to participate in the Internet of Things. The 6LoWPAN group has defined encapsulation and header compression mechanisms that allow IPv6 packets to be sent to and received from over IoT based networks. Other functionalities include: address resolution, routing considerations and protocols for mesh topologies, device, service discovery and security.
Cloud integration	System architecture and development for cloud data acquisition, storage and computation. Data stored in the cloud can be processed for developing business logics. Moreover, the data can be analyzed for anomaly detection, predictions, etc. depending on the use case.

Web API integration	Web API to retrieve the data and other information from the cloud and the network provides ease of integration by other nodes or free application development on the provided data.
Analytics and prediction dashboard	Optional feature – to be implemented in case there is still remaining Budget.

5.2. Second pilot

In order to get more seniors' feedback regarding possible formal apps to include in SENIOR-TV platforms, a Health service has been developed for second pilot. The main objective of this service is collected seniors' health data using the TV. Besides, the Health service can be used by relatives or doctors to access to seniors' data because all the information is stored in the cloud.

5.2.1. Health service

The Health App is the interaction of the elderly person with his/her doctor, medical organization, etc. using his/her SMART-TV. All the data collected by users through the app are transmitted and stored in the cloud. Then, the secondary user side (nurse, doctor, family, etc.) through a web interface they can see the values transmitted. The next image presents the data process:

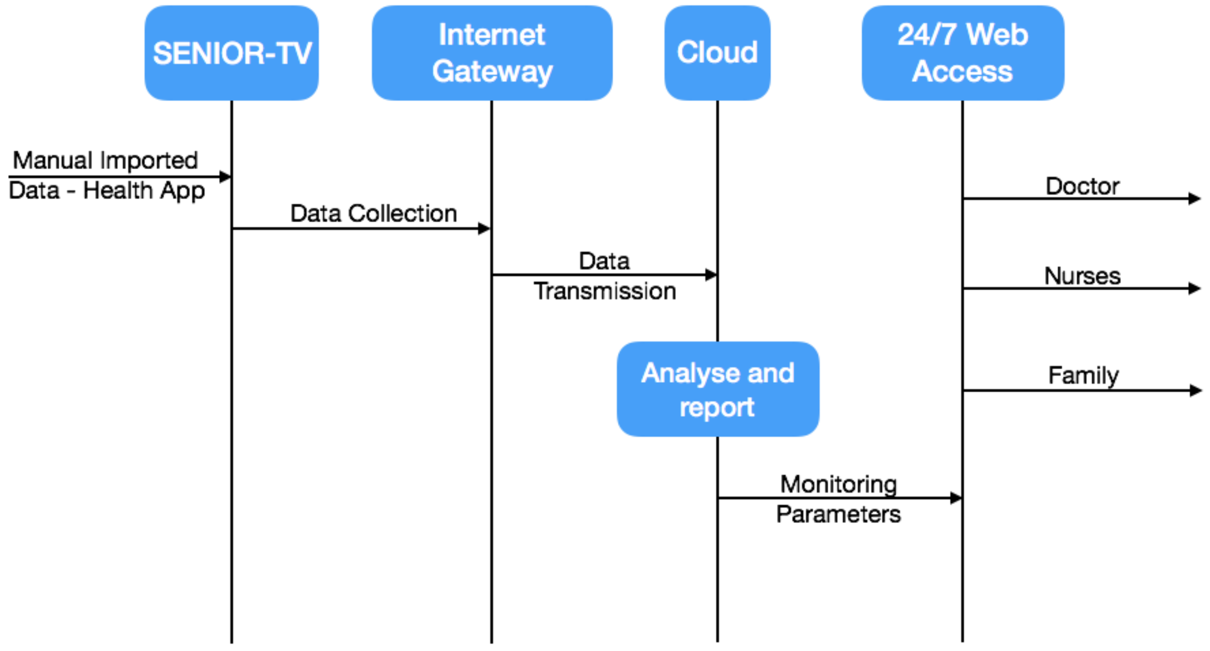


FIGURE 73. HEALTH DATA FLOW SCHEME

The app offers the following functionalities:

1. Login: via username/password screen.
2. Body weight measuring: user provides his/her blood pressure using the D-pad of the remote-control.
3. Blood pressure measuring: user provides his/her blood pressure using the D-pad of the remote-control.
4. Glucose level measuring: user provides his/her blood pressure using the D-pad of the remote-control.
5. Interface personalisation: user has the possibility to change the colours and the text size of the application.

The TV app has been developed using the Android TV SDK and is according the guidelines set to be compatible with TV devices. The initial idea during the application design was to be as simple as it could be in order to be used by the elderly. In this application the full features control is taking place using only the D-pad of the remote-control. The only necessity to use keyboard letters is during the authentication procedure (username and password).

The deployment of the web API is taking place using Microsoft Technologies, IIS 8.5 and we use Azure cloud services. The app uses Microsoft SQL Server 2012 as for the database utilization.

6. Ergonomic design

In this section, the TV interface design guidelines are presented. Only the Attentix and Episodix games are out of these guidelines.

While planning the design and outlook of SENIOR-TV applications, we conducted a study with University of Maribor focused on what are the major points that need to be considered when building a senior friendly interface.

Mostly we concentrated on following five aspects to make the solution as senior friendly as we could:

1. **Navigation**
   a. Avoid scrolling when possible, because this is something that seniors are not accustomed to and have difficulty comprehending it.
   b. Headings of different pages must be unique but they should also have a related look, so users know they are a part of the same solution.
   c. All the pages must have titles on top, so users know where they are currently positioned.
   d. Entire navigation must be simple and consistent throughout the applications.
2. **Language**
   a. Expert terms should be avoided and language used should be simple.
   b. Common words should be used and if possible, avoid newer or slang words.
3. **Text**
   a. Font used should be easy to read (simple font).
   b. It is advisable to use only one font throughout all the applications.
   c. Font must be large enough as seniors have difficulties reading smaller text.
4. **Buttons**
   a. Selected items should be clearly highlighted so it is easy to distinguish between selected and non-selected items.
   b. There should be high contrast between buttons, background color, and text color.
   c. Buttons should be big and there must be enough space between them.
5. **Colors**
   a. Bright colors must be avoided.
   b. Negative polarity is preferred over positive (light color text on dark color background is better than vice versa).

6.1. First pilot

Based on the University of Maribor knowledge and the SENIOR-TV logo colors the first front end guidelines were designed. Next paragraphs show that design details.

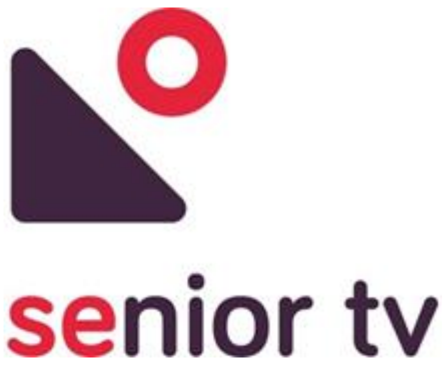
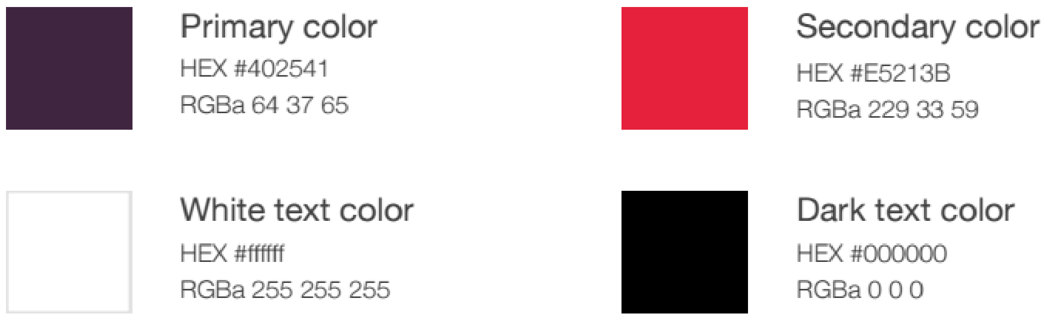


FIGURE 74. SENIOR-TV LOGO

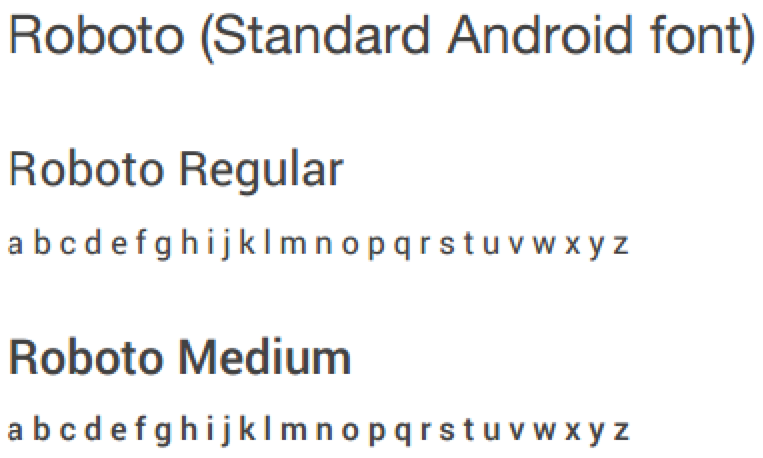
6.1.1. Color swatches



🟪	Primary color HEX #402541 RGBa 64 37 65	🟥	Secondary color HEX #E5213B RGBa 229 33 59
⬜	White text color HEX #ffffff RGBa 255 255 255	⬛	Dark text color HEX #000000 RGBa 0 0 0

FIGURE 75. FIRST PILOT COLOR SWATCHES

6.1.2. Typography



Roboto (Standard Android font)

Roboto Regular

a b c d e f g h i j k l m n o p q r s t u v w x y z

Roboto Medium

a b c d e f g h i j k l m n o p q r s t u v w x y z

FIGURE 76. FIRST PILOT TYPOGRAPHY

6.1.3. Button states

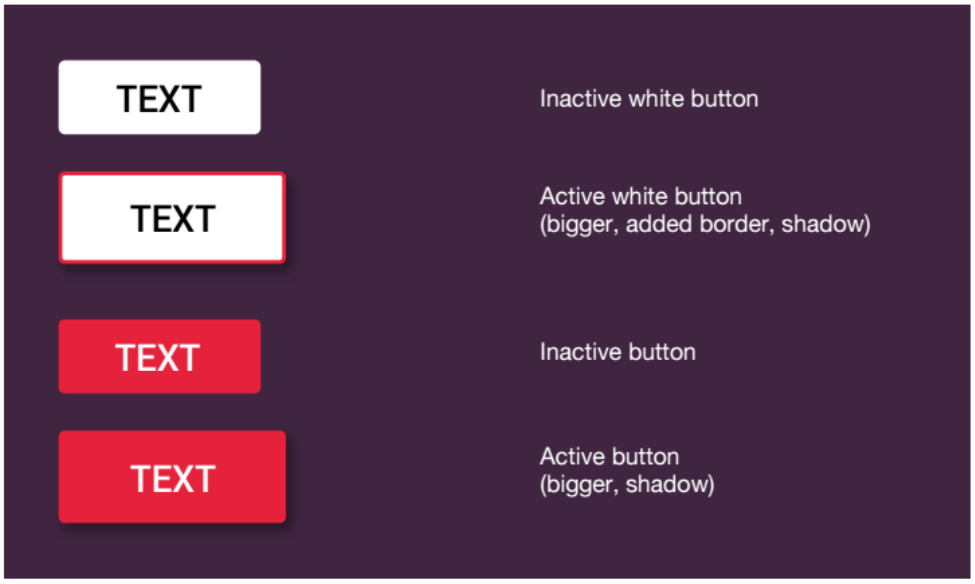


FIGURE 77. FIRST PILOT BUTTONS

6.1.4. TV interact and remote-control

Users navigate apps using a directional pad (D-Pad) and using the air mouse pointer. This type of controller limits movement to up, down, left, and right.

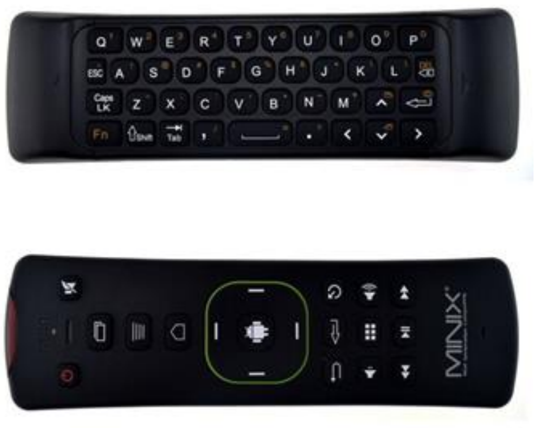


FIGURE 78. AIR MOUSE REMOTE-CONTROLLER USED IN FIRST PILOT TESTING

6.1.5. Navigation, focus and selection

A key aspect of making application work well with a D-pad controller is to make sure that there is always an object that is obviously in focus. App must clearly indicate what object is focused, so users can easily see what action they can take. Use scale, shadow brightness, opacity, animation or a combination of these attributes to help users see a focused object.

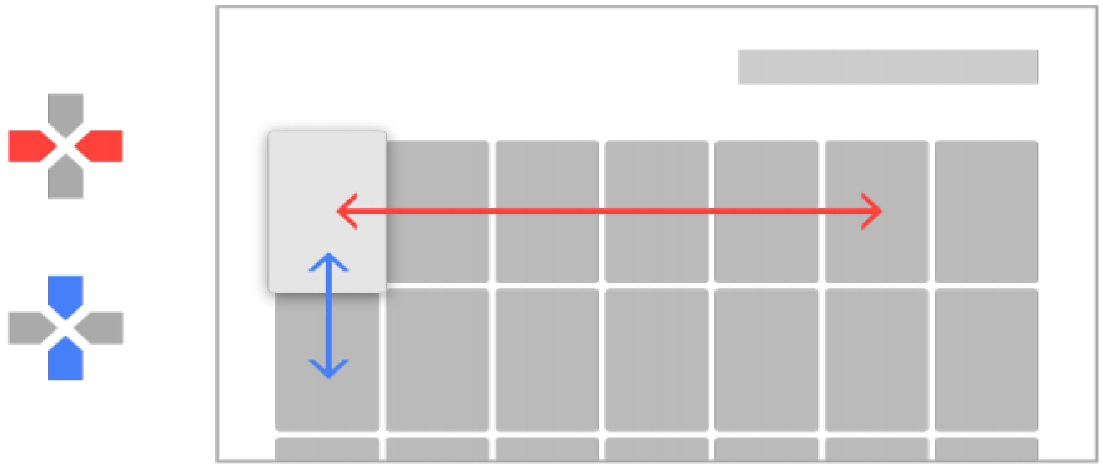


FIGURE 79. NAVIGATION

6.1.6. Senior feedback

After presenting our applications to end users during first pilot testing, the vast majority commented negatively on the colors we used, especially the background color, describing it dull and depressive. We took this feedback into account and prepared several different color schemes.

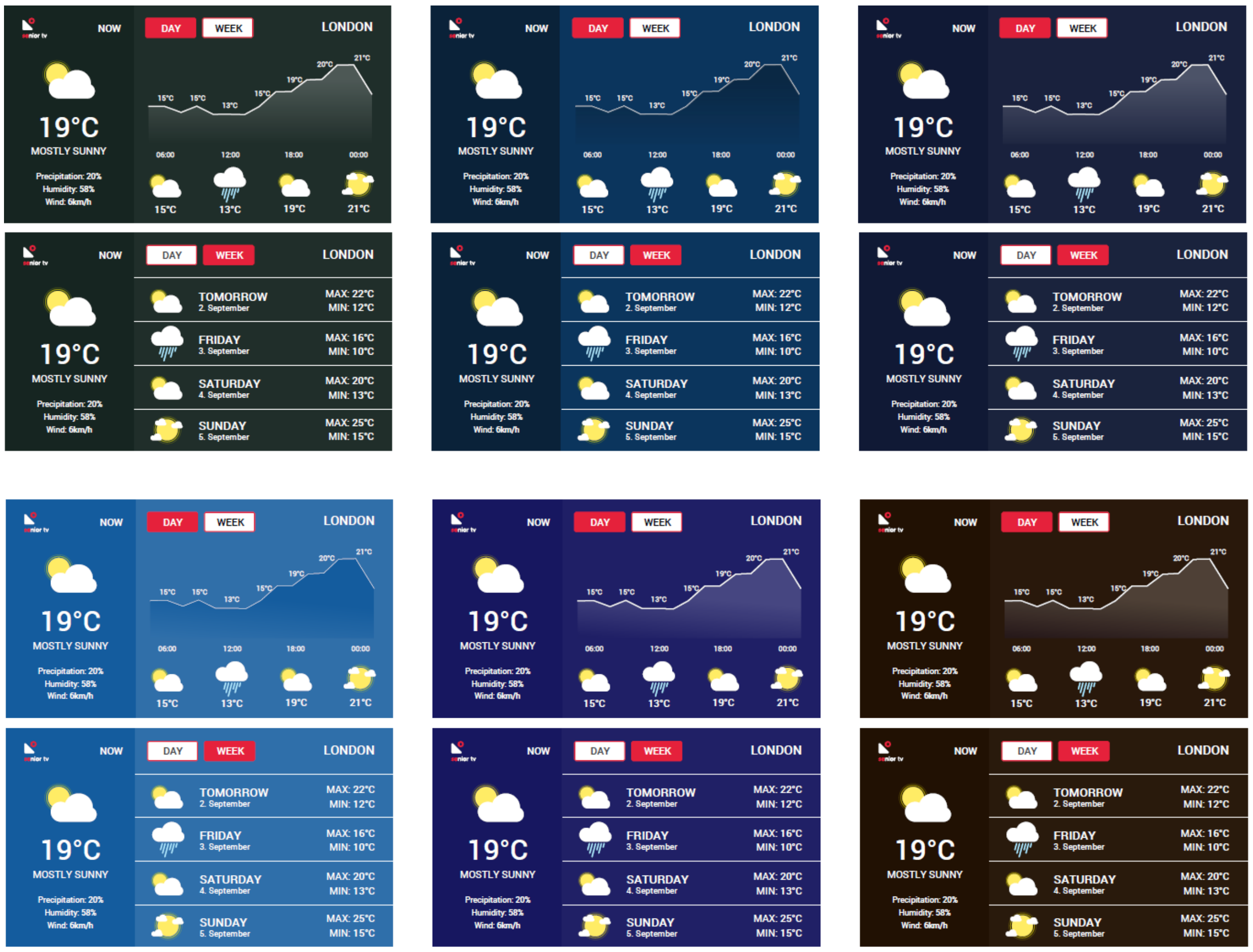

NOW
DAY
WEEK
LONDON
19°C
MOSTLY SUNNY
Precipitation: 20%
Humidity: 58%
Wind: 6km/h
TOMORROW 2. September MAX: 22°C MIN: 12°C
FRIDAY 3. September MAX: 16°C MIN: 10°C
SATURDAY 4. September MAX: 20°C MIN: 13°C
SUNDAY 5. September MAX: 25°C MIN: 15°C

FIGURE 80. ALTERNATIVE COLOR SCHEMES

After presenting the color schemes to several groups of end users, they were all in favor of the light blue background scheme. This is why we chose the blue color scheme and implemented it into the second pilot's applications.

While conducting our first pilot testing, we noticed that it was hard for end users to control SENIOR-TV applications by using the air mouse remote-controller. They had problems keeping steady hands while pointing the device and it was hard for them to keep the remote still while pressing buttons.

6.2. Second pilot

6.2.1. Color swatches

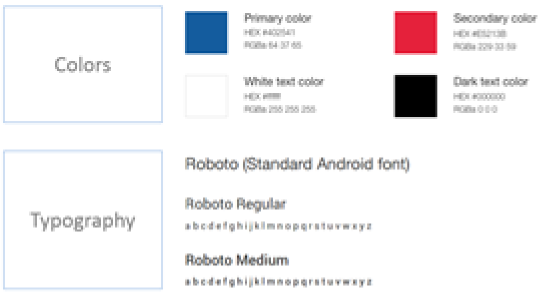


FIGURE 81. SECOND PILOT COLOR SWATCHES

6.2.2. Button states

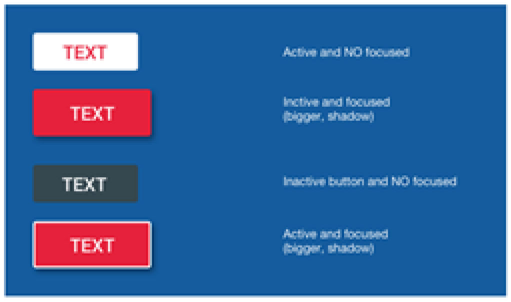

TEXT
Active and NO focused
TEXT
Inctive and focused (bigger, shadow)
TEXT
Inactive button and NO focused
TEXT
Active and focused (bigger, shadow)

FIGURE 82. SECOND PILOT BUTTONS

6.2.3. TV interact and remote-control

Due to seniors' problems with the air mouse, some alternative D-pad remote-controls have been studied. We used three groups of end users to test which controller was most suitable for them and the results were unanimous – they all preferred the Minix standard remote-controller with very few buttons and easy to use navigation arrows. They were accustomed to the use of such remote-controllers and their unsteady hands were no longer a problem.

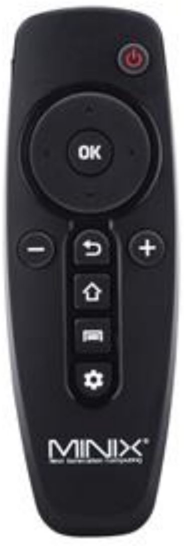


FIGURE 83. D-PAD REMOTE-CONTROLLER USED IN SECOND PILOT TESTING

6.2.4. Other interface considerations

End users often had trouble using physical BACK button on our remote-control and we also got feedback while holding our midterm review meeting that both HOME and BACK buttons should be always present on every screen. This is psychologically positive because it is reassuring to seniors that they always have a way out of the application and back to home screen. This is something that makes them feel safer. We thus added HOME and BACK buttons to every screen to help end users with their navigation and to help them feel more at ease while using our applications.

The following image shows the main elements in the SENIOR-TV apps:

1. Service name.
2. Logged-in user name.
3. Back and Home buttons.
4. Platform logo.

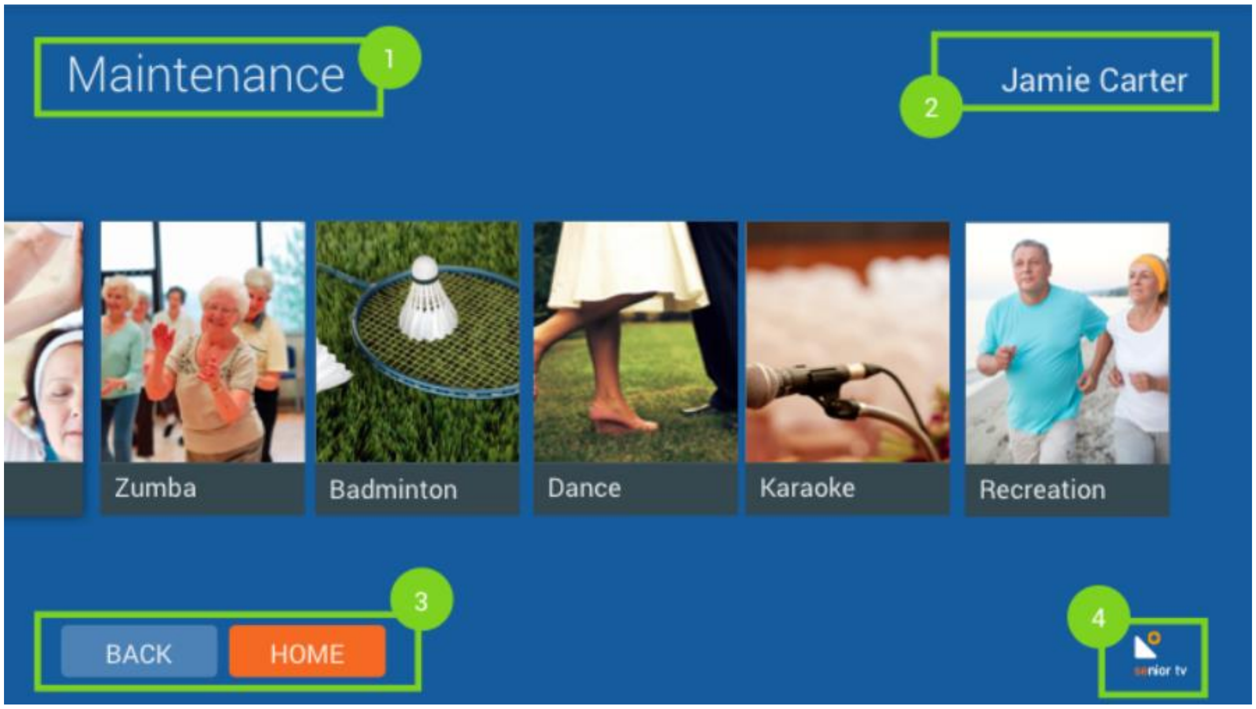


FIGURE 84. GENERAL INTERFACE LAYOUT

7. System integration

All the SENIOR-TV platform services are supported by a "System integration" module. This module includes some common functionalities regarding authentication, notification and device manager. The next subsections describe the integration module architecture and its functionalities.

7.1. General architecture overview

The integration module is composed by two sections: a TV app and the cloud services - developed around the Firebase Cloud platform. The formal and informal services use this module to identify the platform users and to manage the devices and notifications.

The following diagram shows the most important components of integration module:

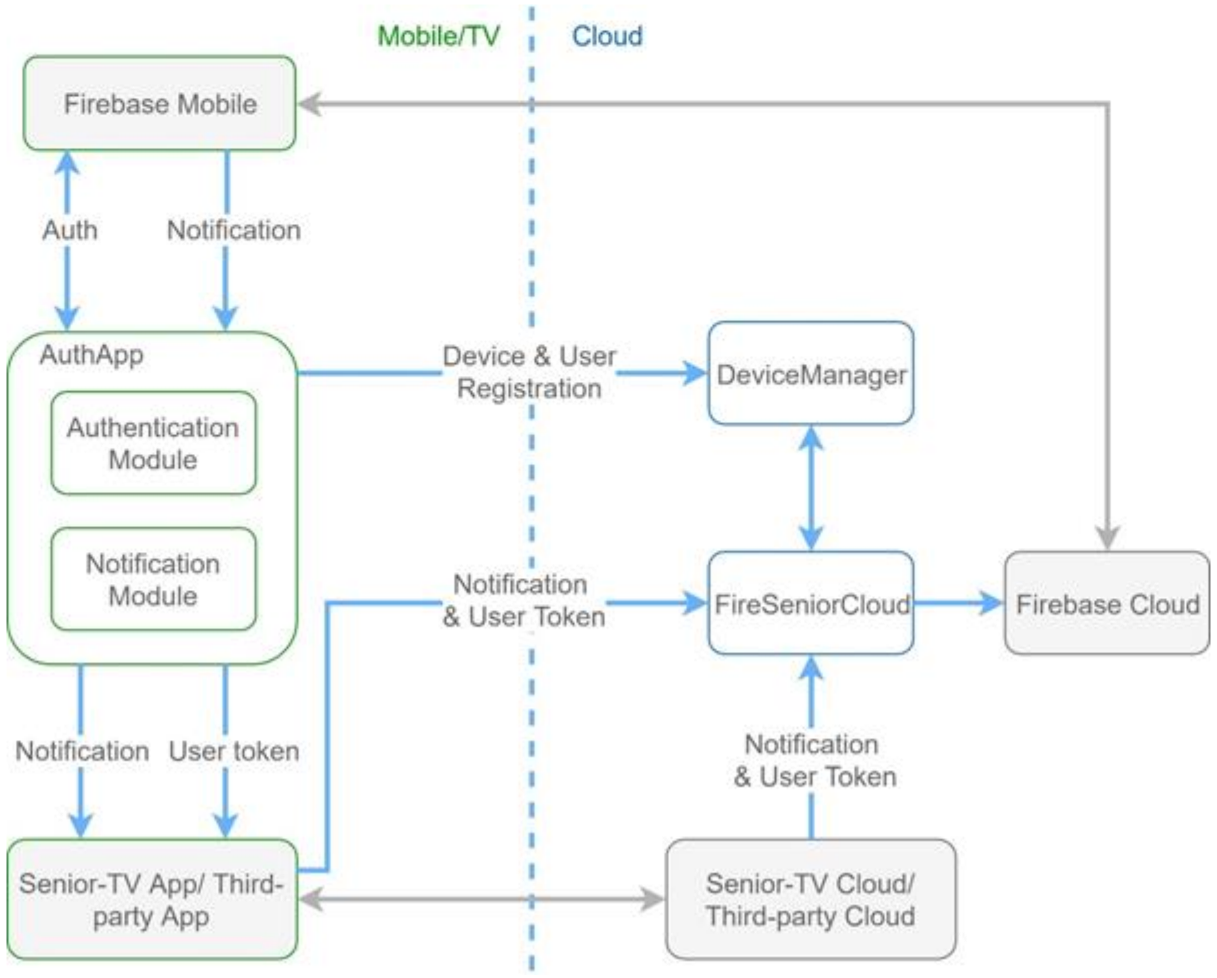


FIGURE 85. GENERAL ARCHITECTURE OVERVIEW

1. **AuthApp**: The Android application responsible for user authentication and dispatching notifications to corresponding Android ThirdPartyClients.

2. **DeviceManager**: The Cloud Service responsible for managing the device list of users.
3. **FireSeniorCloud**: The Cloud Service responsible for receiving Notification Messages from ThirdPartyClients and sending the notifications to the AuthApp.
4. **UserDevice**: The Android Device where SENIOR-TV apps reside
5. **UserToken**: The token used to identify the user across services (mobile or cloud).
6. **DeviceToken**: The token used to identify a specific device.
7. **ThirdPartyClients**: Android applications or Cloud services that send or receive information in SENIOR-TV.
8. **Firebase Mobile**: The client service for firebase (part of Google Play Services – In order for firebase to work PLAY SERVICES version 10.2.0 must be installed on the Android Device).
9. **Firebase Cloud**: The Cloud service for firebase.
10. **SENIOR-TV Cloud/Third-party Cloud**: Cloud services belonging to SENIOR-TV apps or third-party apps.
11. **SENIOR-TV App/Third-party App**: Android application running on the users' device.

7.2. Cloud components

7.2.1. DeviceManager Service

This is the cloud service responsible for managing the device list of users. The scope of the DeviceManager is to store and manage the DeviceTokens belonging to certain UserTokens.

AuthApp will register a new DeviceToken belonging to a UserToken directly with the DeviceManager. When FireSeniorCloud needs to send a notification to a certain UserToken, it will first query the DeviceManager Service to retrieve the list of DeviceTokens belonging to a specific UserToken.

- **Inputs**: The Device and User Token from AuthApp at JSON[26] format.
- **Outputs**: The Device list and User Token Response to FireSeniorCloud at JSON format.

7.2.2. FireSeniorCloud service

FireSeniorCloud service is responsible for receiving Notification Messages from ThirdPartyClients and sending the notifications to the AuthApp. The scope of FireSeniorCloud is to receive Notification messages from ThirdPartyClients and forward these messages to the corresponding UserDevices belonging to specific users.

- **Inputs**: A Notification Message from ThirdPartyClients and the DeviceToken list from DeviceManager at JSON format.

[26]JSON Wikipedia definition: https://en.wikipedia.org/wiki/JSON

- **Outputs**: The Notification message to FirebaseCloud.

7.3. TV/Mobile components

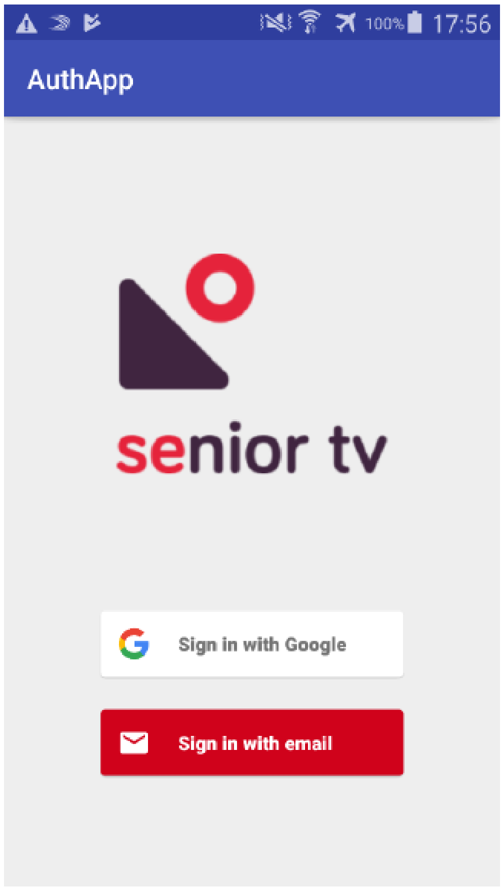
7.2.1. AuthApp

The Android app is responsible for authenticating the user on the SENIOR-TV platform and also managing the notifications. The users are registered or authenticated using an email and password.

Once a user is authenticated, the Device Manager will receive the device identifier and the UserToken. As long as a user keeps his session started in the AuthApp, the user token is broadcasted every minute to the rest of Android ThirdPartyClients using Android intents. The ThirdPartyClients apps can also ask to the UserToken through an Android OrderedBroadcast.

The UserToken can also be used by the ThirdPartyClients to create notifications for users from the Android apps. Besides, the notifications can be created from third-party clouds using the UserToken.

- **Inputs**:
  - Notifications from Firebase.
  - Authentication response from Firebase (with the UserToken).
  - Broadcast intent from ThirdPartyClients requesting UserToken.
- **Outputs**:
  - Notifications to corresponding Android ThirdPartyClients.
  - UserToken to registered Android ThirdPartyClients.



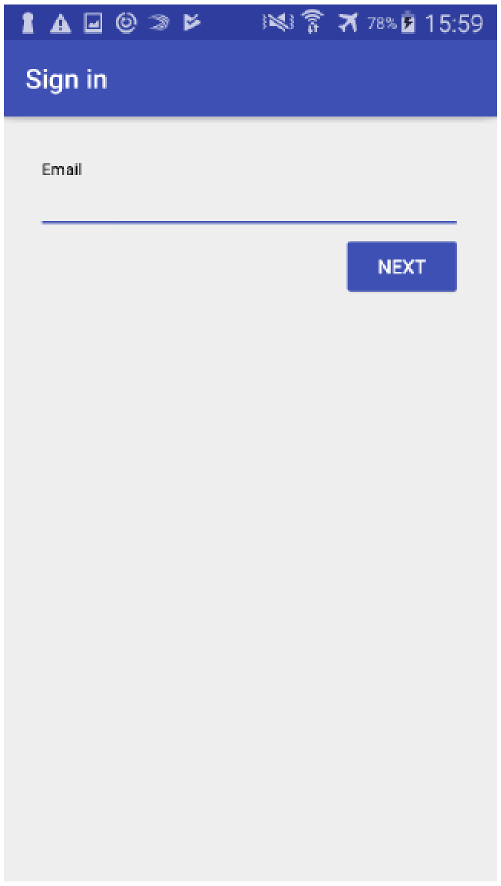
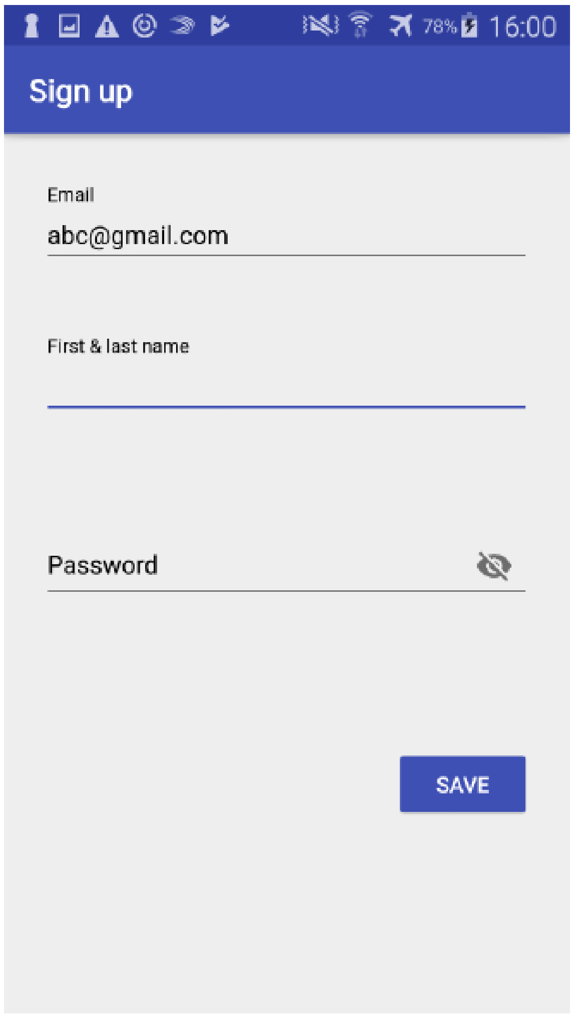
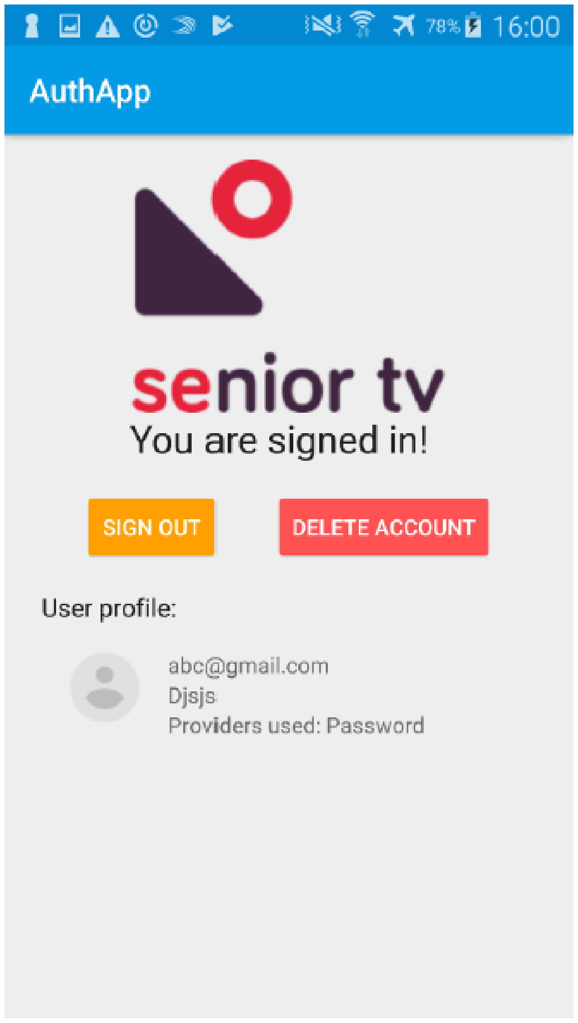


FIGURE 86. AUTHAPP SCREENS 1



abc@gmail.com



You are signed in!

User profile:

abc@gmail.com
Djsjs
Providers used: Password

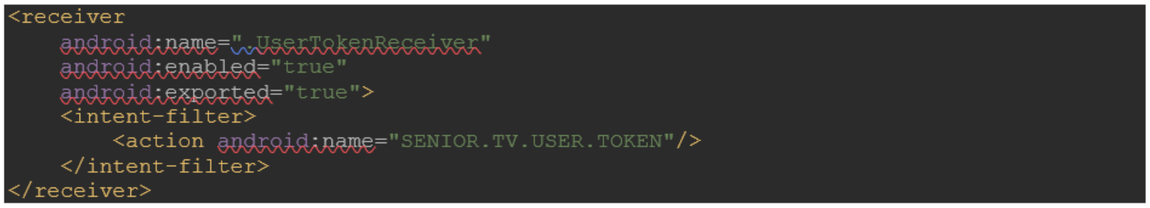
FIGURE 87. AUTHAPP SCREENS 2

7.2.1. Third-Party clients &AuthApp interaction

Android supports applications developed by hybrid and native technologies (see section 2). This section describes how to integrate each kind of apps with the AuthApp.

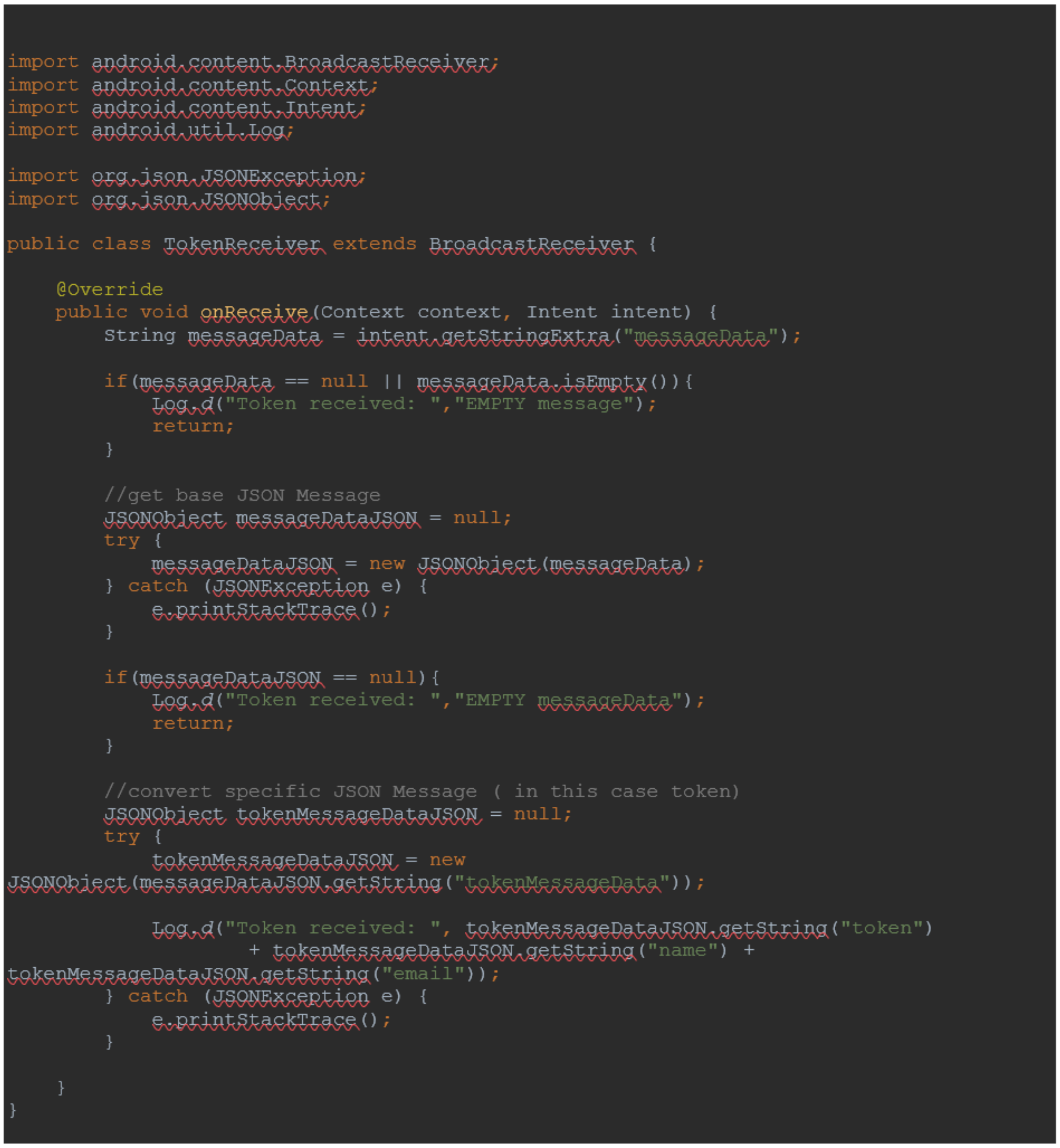
1. Native Java apps

ThirdPartyClients must implement BroadcastReceivers in order to be able to receive the notifications or the user token on the Android device. Below the UserToken Broadcast and the Notification Broadcast receivers are included:



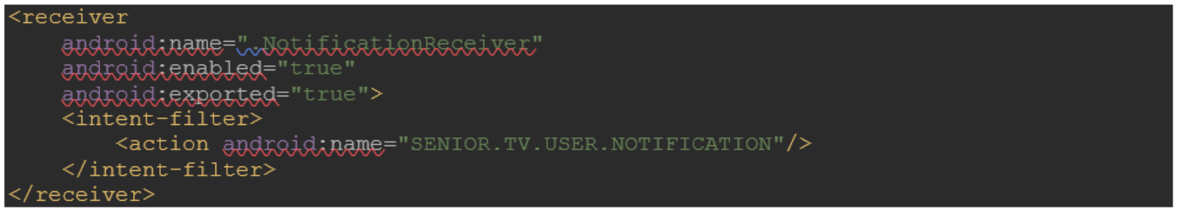
```
<receiver
    android:name=".UserTokenReceiver"
    android:enabled="true"
    android:exported="true">
    <intent-filter>
        <action android:name="SENIOR.TV.USER.TOKEN"/>
    </intent-filter>
</receiver>
```

FIGURE 88. USERTOKEN BROADCAST RECEIVER



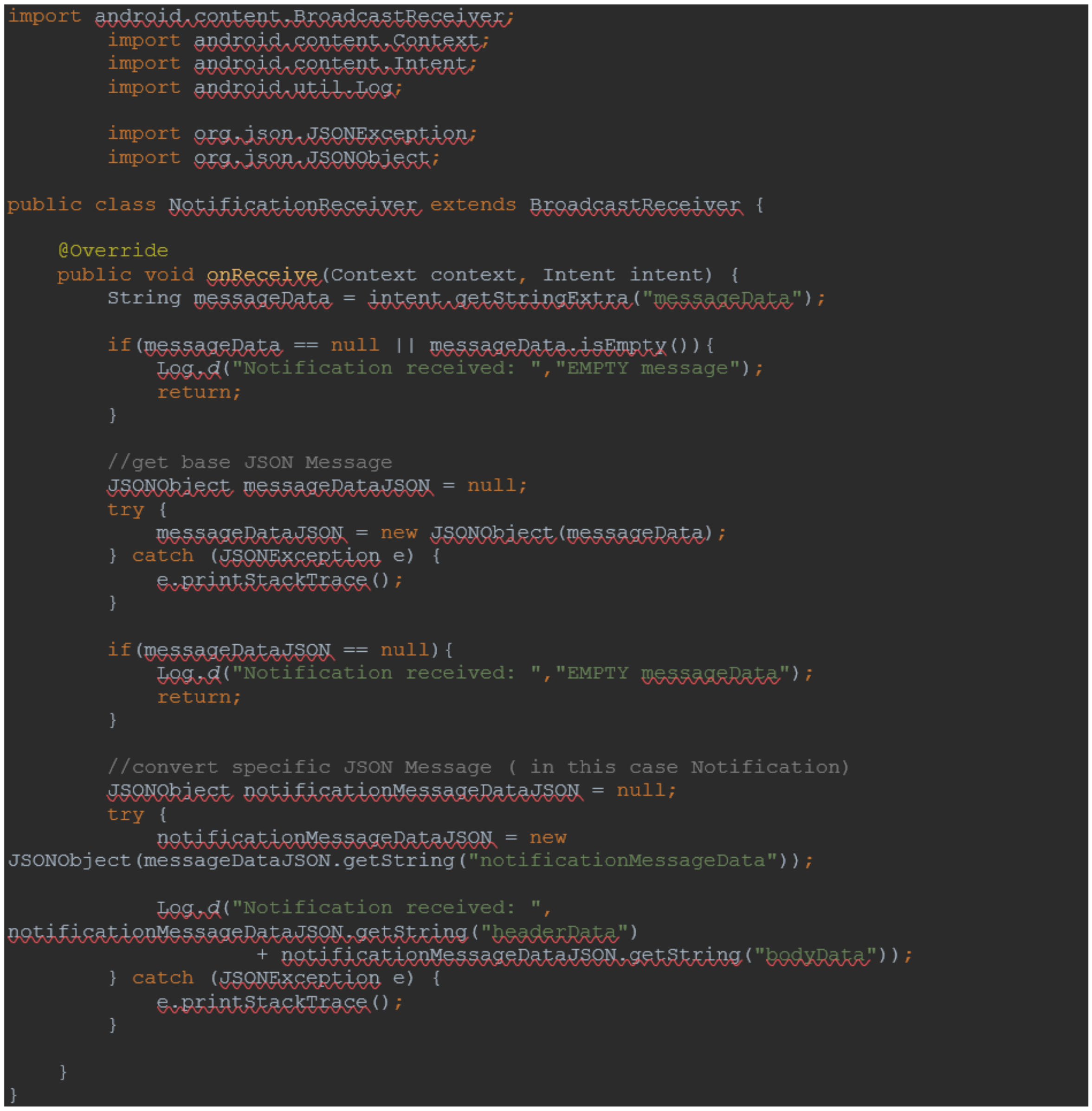
```
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.util.Log;

import org.json.JSONException;
import org.json.JSONObject;

public class TokenReceiver extends BroadcastReceiver {

    @Override
    public void onReceive(Context context, Intent intent) {
        String messageData = intent.getStringExtra("messageData");

        if(messageData == null || messageData.isEmpty()){
            Log.d("Token received: ","EMPTY message");
            return;
        }

        //get base JSON Message
        JSONObject messageDataJSON = null;
        try {
            messageDataJSON = new JSONObject(messageData);
        } catch (JSONException e) {
            e.printStackTrace();
        }

        if(messageDataJSON == null){
            Log.d("Token received: ","EMPTY messageData");
            return;
        }

        //convert specific JSON Message ( in this case token)
        JSONObject tokenMessageDataJSON = null;
        try {
            tokenMessageDataJSON = new
JSONObject(messageDataJSON.getString("tokenMessageData"));

            Log.d("Token received: ", tokenMessageDataJSON.getString("token")
                    + tokenMessageDataJSON.getString("name") +
tokenMessageDataJSON.getString("email"));
        } catch (JSONException e) {
            e.printStackTrace();
        }

    }
}
```

FIGURE 89. USERTOKEN RECEIVER USING



```
<receiver
    android:name=".NotificationReceiver"
    android:enabled="true"
    android:exported="true">
    <intent-filter>
        <action android:name="SENIOR.TV.USER.NOTIFICATION"/>
    </intent-filter>
</receiver>
```

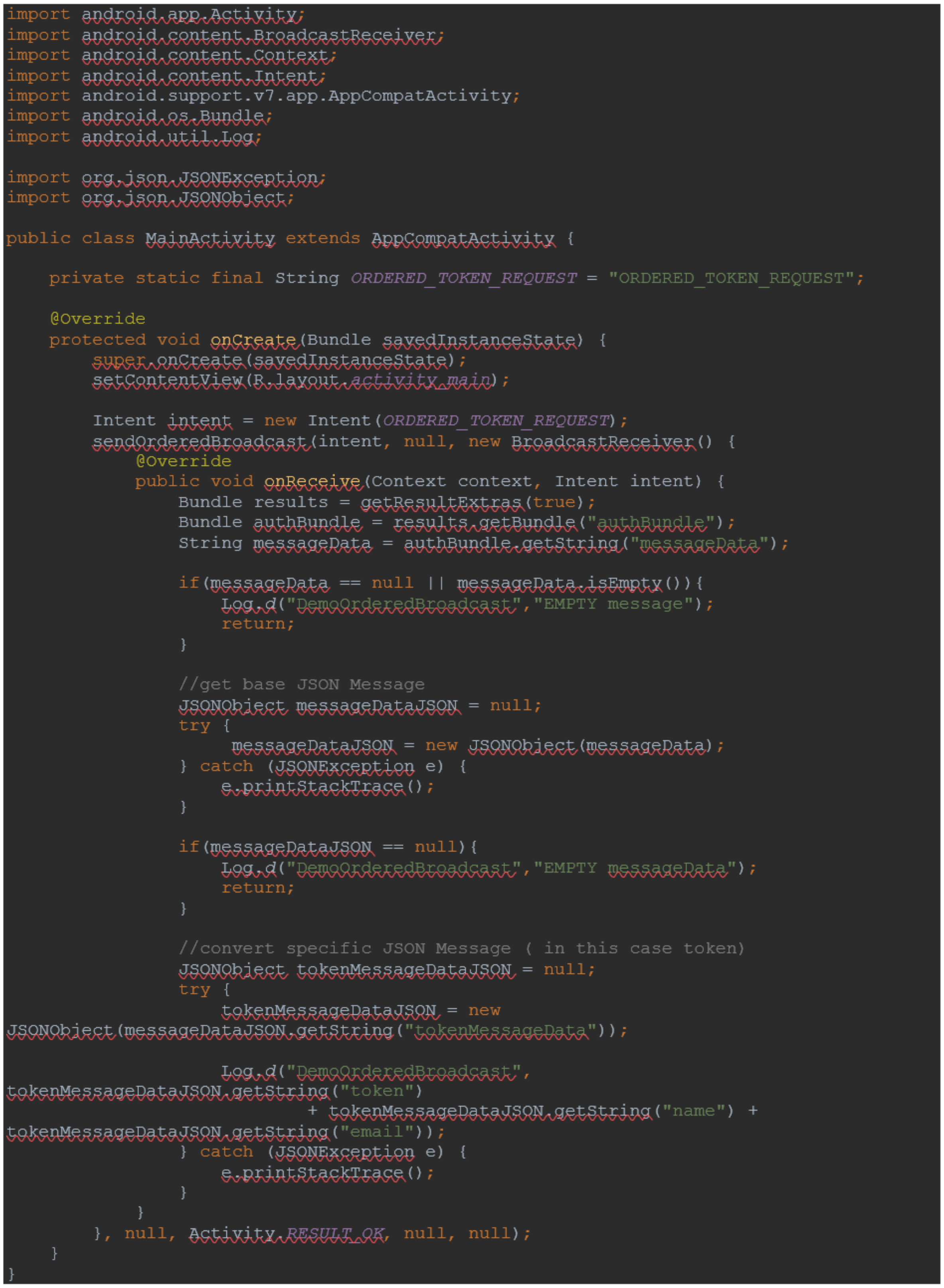
FIGURE 90. NOTIFICATION BROADCAST RECEIVER



```
import android.content.BroadcastReceiver;
        import android.content.Context;
        import android.content.Intent;
        import android.util.Log;

        import org.json.JSONException;
        import org.json.JSONObject;

public class NotificationReceiver extends BroadcastReceiver {

    @Override
    public void onReceive(Context context, Intent intent) {
        String messageData = intent.getStringExtra("messageData");

        if(messageData == null || messageData.isEmpty()){
            Log.d("Notification received: ","EMPTY message");
            return;
        }

        //get base JSON Message
        JSONObject messageDataJSON = null;
        try {
            messageDataJSON = new JSONObject(messageData);
        } catch (JSONException e) {
            e.printStackTrace();
        }

        if(messageDataJSON == null){
            Log.d("Notification received: ","EMPTY messageData");
            return;
        }

        //convert specific JSON Message ( in this case Notification)
        JSONObject notificationMessageDataJSON = null;
        try {
            notificationMessageDataJSON = new
JSONObject(messageDataJSON.getString("notificationMessageData"));

            Log.d("Notification received: ",
notificationMessageDataJSON.getString("headerData")
                    + notificationMessageDataJSON.getString("bodyData"));
        } catch (JSONException e) {
            e.printStackTrace();
        }

    }
}
```

FIGURE 91. NOTIFICATION BROADCAST RECEIVER USING

The ThirdPartyClient can also request the token and metadata without having to wait for the usual broadcast. The Ordered Broadcast Receiver is used in these situations:

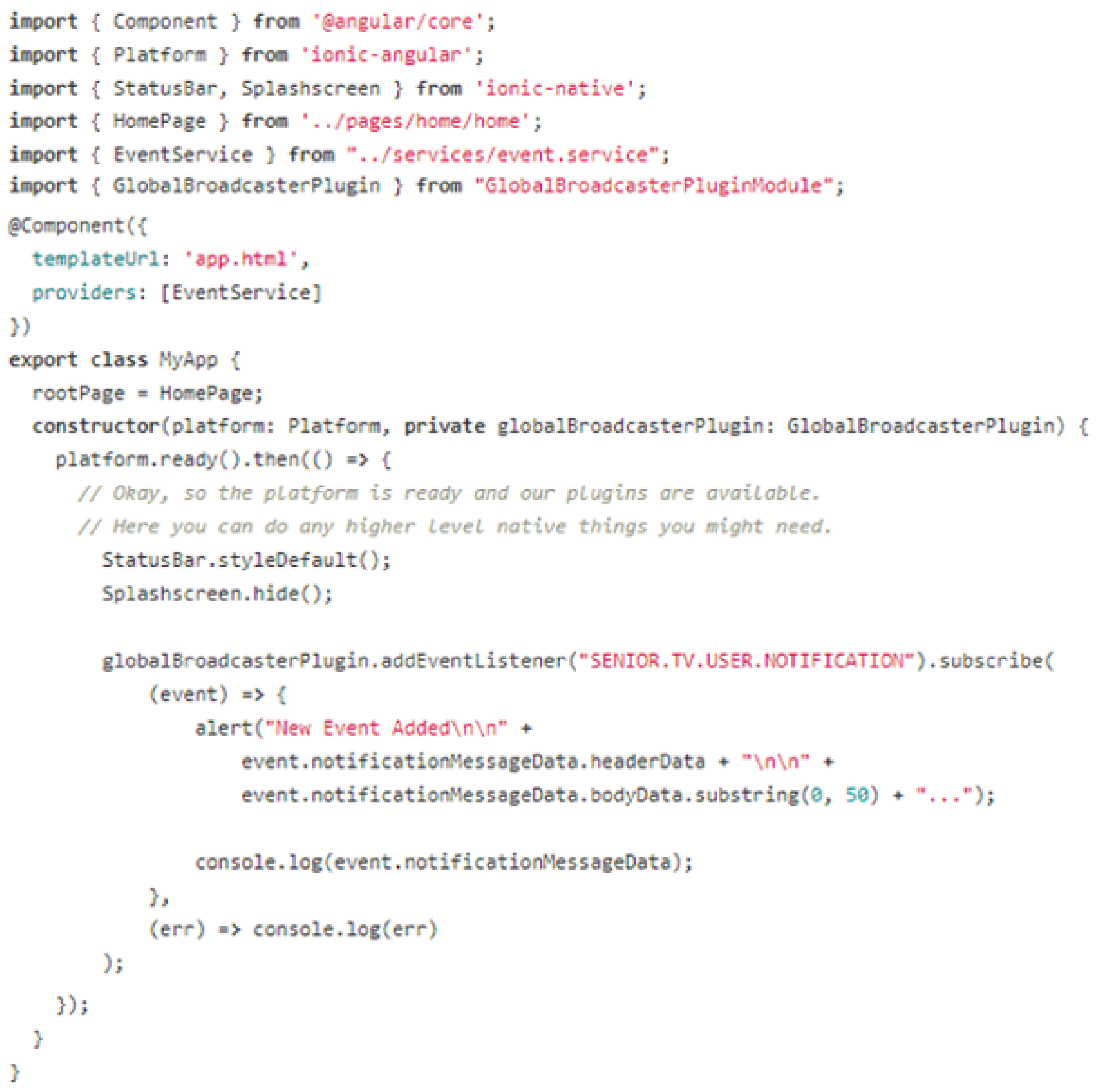


```
import android.app.Activity;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;

import org.json.JSONException;
import org.json.JSONObject;

public class MainActivity extends AppCompatActivity {

    private static final String ORDERED_TOKEN_REQUEST = "ORDERED_TOKEN_REQUEST";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        Intent intent = new Intent(ORDERED_TOKEN_REQUEST);
        sendOrderedBroadcast(intent, null, new BroadcastReceiver() {
            @Override
            public void onReceive(Context context, Intent intent) {
                Bundle results = getResultExtras(true);
                Bundle authBundle = results.getBundle("authBundle");
                String messageData = authBundle.getString("messageData");

                if(messageData == null || messageData.isEmpty()){
                    Log.d("DemoOrderedBroadcast","EMPTY message");
                    return;
                }

                //get base JSON Message
                JSONObject messageDataJSON = null;
                try {
                     messageDataJSON = new JSONObject(messageData);
                } catch (JSONException e) {
                    e.printStackTrace();
                }

                if(messageDataJSON == null){
                    Log.d("DemoOrderedBroadcast","EMPTY messageData");
                    return;
                }

                //convert specific JSON Message ( in this case token)
                JSONObject tokenMessageDataJSON = null;
                try {
                    tokenMessageDataJSON = new
JSONObject(messageDataJSON.getString("tokenMessageData"));

                    Log.d("DemoOrderedBroadcast",
tokenMessageDataJSON.getString("token")
                            + tokenMessageDataJSON.getString("name") +
tokenMessageDataJSON.getString("email"));
                } catch (JSONException e) {
                    e.printStackTrace();
                }
            }
        }, null, Activity.RESULT_OK, null, null);
    }
}
```

FIGURE 92. ORDERED BROADCAST RECEIVER USING

2. Hybrid apps

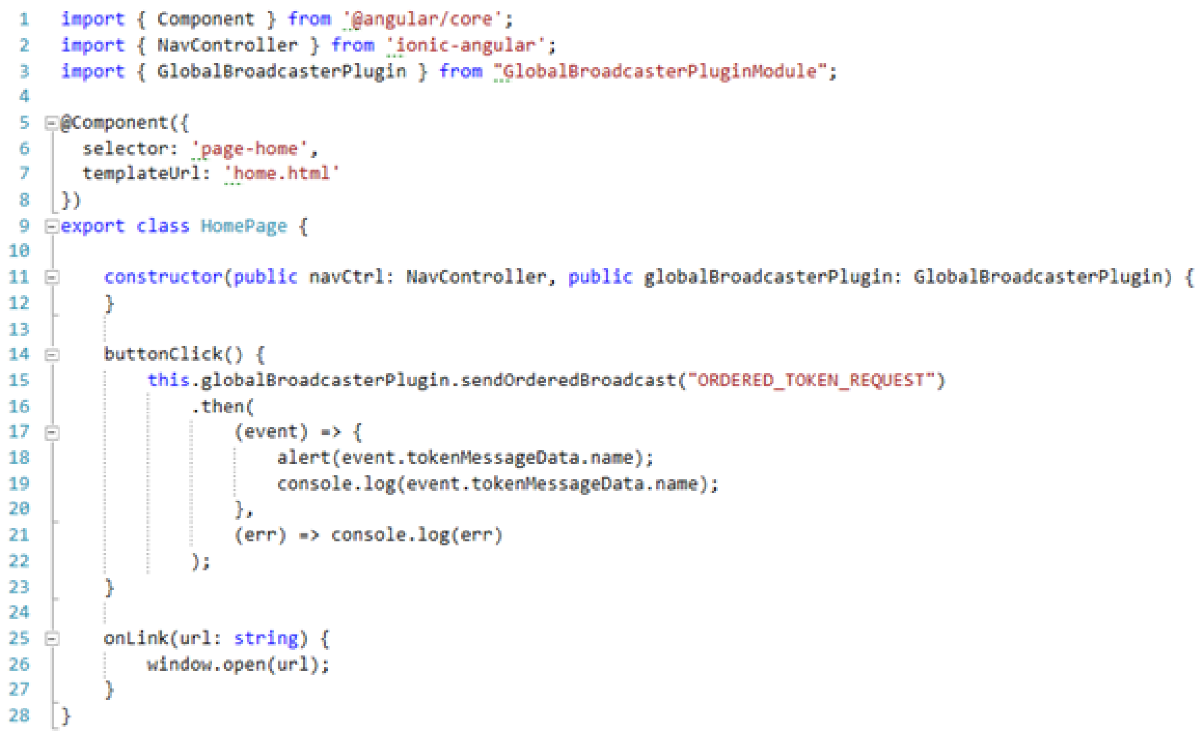
Hybrid ThirdPartyClients must import the custom GlobalBroadcasterPlugin in order to be able to receive the notifications or the user token on the Android device inside the Ionic Application. A GlobalBroadcasterPlugin using example is included below:



```
import { Component } from '@angular/core';
import { Platform } from 'ionic-angular';
import { StatusBar, Splashscreen } from 'ionic-native';
import { HomePage } from '../pages/home/home';
import { EventService } from "../services/event.service";
import { GlobalBroadcasterPlugin } from "GlobalBroadcasterPluginModule";
@Component({
  templateUrl: 'app.html',
  providers: [EventService]
})
export class MyApp {
  rootPage = HomePage;
  constructor(platform: Platform, private globalBroadcasterPlugin: GlobalBroadcasterPlugin) {
    platform.ready().then(() => {
      // Okay, so the platform is ready and our plugins are available.
      // Here you can do any higher level native things you might need.
        StatusBar.styleDefault();
        Splashscreen.hide();

        globalBroadcasterPlugin.addEventListener("SENIOR.TV.USER.NOTIFICATION").subscribe(
            (event) => {
                alert("New Event Added\n\n" +
                    event.notificationMessageData.headerData + "\n\n" +
                    event.notificationMessageData.bodyData.substring(0, 50) + "...");

                console.log(event.notificationMessageData);
            },
            (err) => console.log(err)
        );
    });
  }
}
```

FIGURE 93. GLOBALBROADCASTERPLUGIN IN EVENTS SERVICE

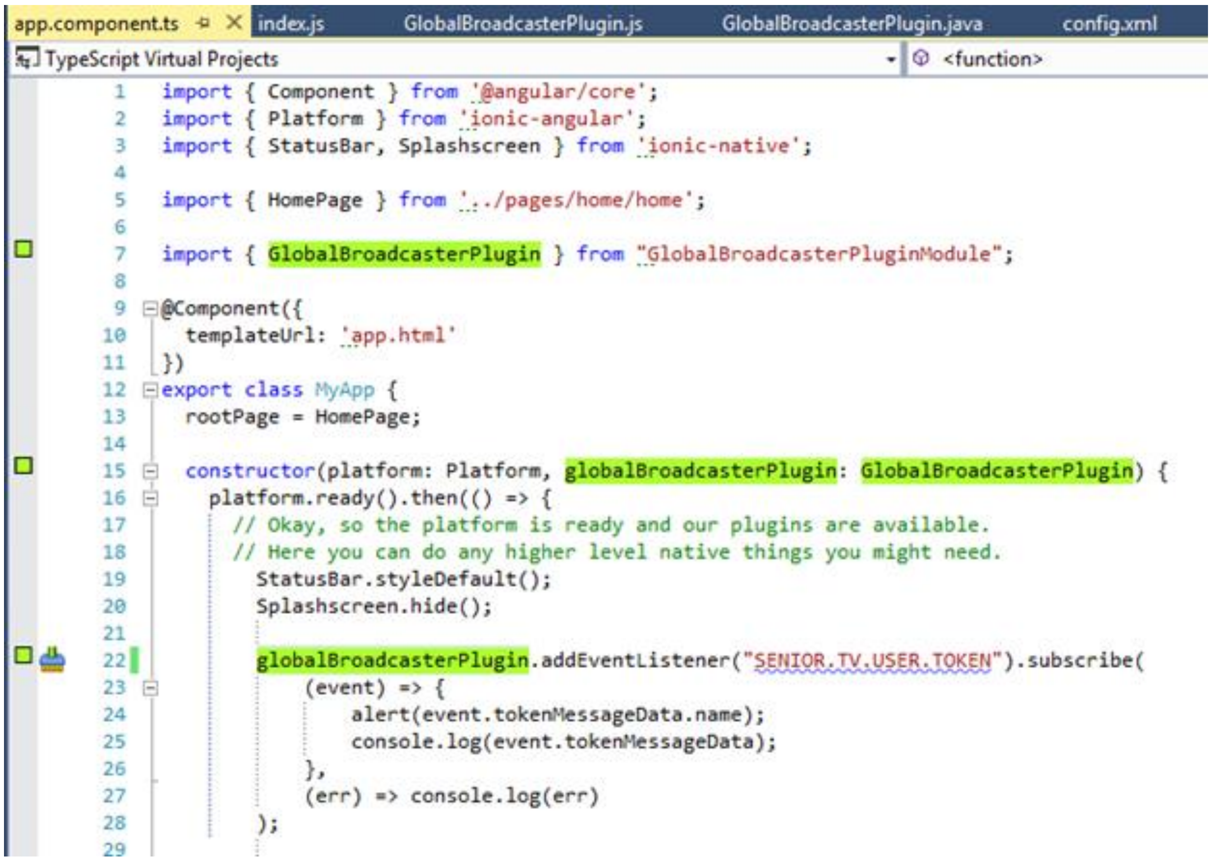
Following an example of “sendOrderedBroadcast” function to retrieve on demand the UserToken:



```
import { Component } from '@angular/core';
import { NavController } from 'ionic-angular';
import { GlobalBroadcasterPlugin } from "GlobalBroadcasterPluginModule";

@Component({
  selector: 'page-home',
  templateUrl: 'home.html'
})
export class HomePage {

    constructor(public navCtrl: NavController, public globalBroadcasterPlugin: GlobalBroadcasterPlugin) {
    }

    buttonClick() {
        this.globalBroadcasterPlugin.sendOrderedBroadcast("ORDERED_TOKEN_REQUEST")
            .then(
                (event) => {
                    alert(event.tokenMessageData.name);
                    console.log(event.tokenMessageData.name);
                },
                (err) => console.log(err)
            );
    }

    onLink(url: string) {
        window.open(url);
    }
}
```

FIGURE 94. RETRIEVE ON DEMAND THE USERTOKEN

After installing the plugin in a hybrid project, the GlobalBroadcasterPlugin module can be imported as a component and the plugin methods are accessible:


app.component.ts
index.js
GlobalBroadcasterPlugin.js
GlobalBroadcasterPlugin.java
config.xml
TypeScript Virtual Projects
<function>

```
import { Component } from '@angular/core';
import { Platform } from 'ionic-angular';
import { StatusBar, Splashscreen } from 'ionic-native';

import { HomePage } from '../pages/home/home';

import { GlobalBroadcasterPlugin } from "GlobalBroadcasterPluginModule";

@Component({
  templateUrl: 'app.html'
})
export class MyApp {
  rootPage = HomePage;

  constructor(platform: Platform, globalBroadcasterPlugin: GlobalBroadcasterPlugin) {
    platform.ready().then(() => {
      // Okay, so the platform is ready and our plugins are available.
      // Here you can do any higher level native things you might need.
        StatusBar.styleDefault();
        Splashscreen.hide();

        globalBroadcasterPlugin.addEventListener("SENIOR.TV.USER.TOKEN").subscribe(
            (event) => {
                alert(event.tokenMessageData.name);
                console.log(event.tokenMessageData);
            },
            (err) => console.log(err)
        );

```

FIGURE 95. HYBRID PROJECT USING THE GLOBALBROADCASTERPLUGIN

8. Next steps

This section includes an overview of the main technical issues that will analyze in the following versions of SENIOR-TV platform.

8.1. System integration

In next steps of the system integration work, we will further investigate standards in order to enable easy interaction between the various parts of the system. We will plan to incorporate different user roles and adapt the current services to distinguish the functionalities for each of them.

Besides, we are going to study how to adapt the TV launcher to create a compact and unified ecosystem to made easy the seniors' interaction with the Android STB.

8.2. More services

For the third version we will complete the platform informal services including new applications regarding social and entertainment topics in order to improve the relationship of the elderly with their environment and with society. The new services are related to the following services mentioned in the DOW:

1. Social (Facebook, Twitter).
2. Tele-rehab (games, audiovisual repositories).
3. Smart leisure. Personal recommendations for leisure activities.

The next version of the Health App will try to integrate sensors for the input devices. So accordingly, the app will support dual method for the data entry. Sensor based and manual based using the remote-control. In the sensor-based method the user should use a bluetooth based wearable to measure his/her parameters and then automatically the measurement is transmitted via bluetooth to the app without any manual intervention from the user. Then using third-party API and server we collect the data and then are permanently stored in SENIOR-TV server for further processing and publication into the SENIOR-TV web API.

8.5. Control & input devices

During SENIOR-TV V2 develop, all the TV apps were adapted to interact with D-pad remote-controls in order to reduce the pointers problems reported by the seniors during first pilot. Looking to the third platform version, touch devices (as tablets or mobiles) will be studied to interact with TV. Besides, bracelets and similar devices are being investigated to monitoring the physiological signals in order to integrate them into Health service.

9. Conclusions

Once we have selected the development technologies and the operating system to development the platform, our efforts have been focused on adapting the interfaces and interaction devices to the seniors' needs.

Although the presented applications during first pilot were not complete, a very useful feedback has been obtained from the elderly. In general, the application objective and its functionalities were easy understood and, the seniors' expectations were fulfilled. However, several deficiencies were reported both in the interface and in the controls. The current interface design has solved most of detected problems. The main modifications are regarding reduce screens complexity and using only three type of buttons to interact with TV apps: the 4 cursors and the Home and Back buttons.

On the other hand, great progress has been made in the integration of the services and the platform in order to obtaining a compact and integrated final product: the user registration system has been unified, and authentication, notification and device manager services have been added. These common services are used by all platform services.

Finally, progress has been made in the integration of other platforms (web and mobile / tablet) to facilitate secondary users to access to the SENIOR-TV platform services to check their elders data from other devices besides the TV.

In the next platform version, we will advance in entertainment services to promote an active and informed mind. In addition, it is essential to investigate deeper about sensors integration with the formal service in order to make easier and attractive its management.

aal-2014-171

SENIOR-TV

PROVIDING ICT-BASED FORMAL AND INFORMAL CARE AT HOME

Quality Checklist
Quality Control of D2.2

Peer Reviewer	
Reviewer	Partner
Luis Anido-Rifón	Advisory Board

CRITERIA	VERIFIED
1) Conformity to Standards and Project templates	
Logos (AAL, SENIOR-TV)	X
Project title, reference, author, version, revision, data	X
Mandatory statements (disclaimer)	X

Conformance to the standard structure required by EACEA (ex. Disclaimer, Executive summary, Acknowledgement, Introduction, page numbers, etc.)	**Executive summary missing**
2) Language check (typing mistakes, grammar, etc.)	**X**
3) Coherence with objectives declared in the Technical Annex	
Obj. 1: To elaborate the project's Quality Plan following well-accepted methodologies tailored to the learning domain and based on a detailed description of projects objectives, success indicators and work plan.	**This document needs to be updated with the actual objectives related to this D2.2**
Obj. 2: To monitor all project activities and provide quality control of all project results as well as recommendations for improvements and identification of best practices.	**This document needs to be updated with the actual objectives related to this D2.2**
4) Reliability of data	
Information and sources well identified	**X**
Data and information are free from factual or logic errors	**X**
The analysis (if applicable) is reliable, i.e. previous studies have been sufficiently reviewed; qualitative information and quantitative data are balanced and appropriate	**X**
5) Credibility of findings	

Findings supported by evidence based on data analysis	**Not applicable**
Replicability of findings	**Not applicable**
6) Validity of conclusions	
Conclusions meet evaluation questions and information needs	**No**
Conclusions supported by proper evaluation findings	**No**
No conclusions missing according to the evidences presented	**No**

7) Please indicate any deviations from contractual conditions (WP objectives declared in the technical annex)

None

8) Comments/Suggestions for revision

Executive summary is missing.

The technical quality of the deliverable is good from the perspective of the software developments carried out during the second year of the project. Nevertheless, the strategy behind the methodology described in the Description of Work is not sufficiently described in the contents of this deliverable.

There is no analysis of data from the first cycle of pilots, which is assumed to feed the developments under WP2 for the second pilot. This information is completely missing. The lack of information from WP3 impacts on the quality of the deliverable.

It is also not clear whether the objectives from T2.3 were properly addressed as insufficient information is provided.

Conclusions are not well elaborated and do not provide an overall perspective on how the works under this WP fit into the global project objectives.

9) Implementation of revisions/modifications suggested and explanation for eventual rejections (performed by the Responsible of the Deliverable)

Feedback from WP3 (from first cycle) and WP1 (from second cycle) needs to be included.

More elaboration on the description of individual tasks is required.

Conclusions requires updating considering the above and proposing guidelines for the development of the next pilot with end-users.

10) Deliverable accepted

☐ YES

☐ NO

If NO, please state reasons:

The deliverable describes the technical developments carried out under this WP, Task 2.2. However, this document lacks a clear description on how the outputs from other WPs (e.g. WP1 and WP3) have been taken into account. It also does not sufficiently describe how WP3 should use the outcomes form this WP to develop the next cycle of pilots.

aal-2014-171

SENIOR-TV

PROVIDING ICT-BASED FORMAL AND INFORMAL CARE AT HOME

Quality Checklist
Quality Control of D2.2_V2

Peer Reviewer	
Reviewer	**Partner**
Catalina Anghelache -Tutulan	COMPEXIN SA

CRITERIA	VERIFIED
1) Conformity to Standards and Project templates	
Logos (AAL, SENIOR-TV)	**Y**
Project title, reference, author, version, revision, data	**Y**
Mandatory statements (disclaimer)	**Y**

Conformance to the standard structure required by EACEA (ex. Disclaimer, Executive summary, Acknowledgement, Introduction, page numbers, etc.)	**Y**
2) Language check (typing mistakes, grammar, etc.)	**N**
3) Coherence with objectives declared in the Technical Annex	**Y**
Obj. 1: To elaborate the project's Quality Plan following well-accepted methodologies tailored to the learning domain and based on a detailed description of projects objectives, success indicators and work plan.	**Y**
Obj. 2: To monitor all project activities and provide quality control of all project results as well as recommendations for improvements and identification of best practices.	**Y**
4) Reliability of data	
Information and sources well identified	**Y**
Data and information are free from factual or logic errors	**N/A**
The analysis (if applicable) is reliable, i.e. previous studies have been sufficiently reviewed; qualitative information and quantitative data are balanced and appropriate	**Y**
5) Credibility of findings	
Findings supported by evidence based on data analysis	**Y**
Replicability of findings	**N/A**
6) Validity of conclusions	
Conclusions meet evaluation questions and information needs	**Y**
Conclusions supported by proper evaluation findings	**Y**
No conclusions missing according to the evidences presented	**Y**
7) Please indicate any deviations from contractual conditions (WP objectives declared in the technical annex)	
8) Comments/Suggestions for revision	

- There are quite a few grammar mistakes

9) Implementation of revisions/modifications suggested and explanation for eventual rejections (performed by the Responsible of the Deliverable)

10) Deliverable accepted

☐ **YES**

☐ NO

If NO, please state reasons: Implement changes and grammar corrections before.