**eWALL installation-home**



**This document presents an overview of eWALL that can be used by the eWALL Open Source Community. The document belongs to eWALL consortium and any use of the document requires the reference to eWALL Open Source (http://gitghub.com/ewallprojecteu)**

Table of Contents

[1 eWALL Devices and Software 4](#_Toc510695111)

[1.1 Software repository 5](#_Toc510695112)

[2 *Virtual Testbed Environment* 7](#_Toc510695113)

[3 eWALL prototype for the intra-pilots and the user pilots 9](#_Toc510695114)

[3.1 User Pilots prototypes 9](#_Toc510695115)

[4 Building the eWALL prototype 12](#_Toc510695116)

[4.1 Preparation 13](#_Toc510695117)

[4.1.1 Hardware Setup per eWALL 13](#_Toc510695118)

[4.1.2 Connect all the devices 14](#_Toc510695119)

[4.1.3 Prepare Ubuntu Installation Media 14](#_Toc510695120)

[4.2 Ubuntu Installation 15](#_Toc510695121)

[4.2.1 Changing the PC Boot Order 15](#_Toc510695122)

[4.2.2 Boot Ubuntu from USB (GigaByte Brix) 15](#_Toc510695123)

[4.2.3 Boot Ubuntu from USB (Zotac) 15](#_Toc510695124)

[4.2.4 Installing Ubuntu 14.04.3 15](#_Toc510695125)

[*TABLE: Ubuntu PC Name and Username Scheme* 16](#_Toc510695126)

[4.3 Configuring Fresh Ubuntu Install 17](#_Toc510695127)

[4.3.1 Connect to the Internet 17](#_Toc510695128)

[4.3.2 Disable Automatic Updates 17](#_Toc510695129)

[4.3.3 Install TeamViewer 17](#_Toc510695130)

[4.3.4 Configuring TeamViewer 18](#_Toc510695131)

[4.3.5 Disable Screen Lock (Verified) 18](#_Toc510695132)

[4.4 Third-Party Software Installations 19](#_Toc510695133)

[4.4.1 Install Google Chrome (Verified) 19](#_Toc510695134)

[4.4.2 Install VNStat Network Monitoring tool (Verified) 19](#_Toc510695135)

[4.4.3 Install Galileo FitBit Sync Tool (In Progress) 20](#_Toc510695136)

[4.5 Install eWALL Home Environment 20](#_Toc510695137)

[4.5.1 Downloading latest eWALL Home Environment Software 20](#_Toc510695138)

[4.5.2 Configuring Environmental Sensing Nodes 21](#_Toc510695139)

[4.5.3 Installing eWALL Home Environment 22](#_Toc510695140)

[4.6 Configuring Philips Hue 22](#_Toc510695141)

[4.7 Installing eWALL Local Platform 23](#_Toc510695142)

[4.7.1 Configuring Local Platform software 23](#_Toc510695143)

[4.7.2 Running and Using the Local Platform software 24](#_Toc510695144)

[4.7.3 Updating Local Platform software 25](#_Toc510695145)

[4.8 Pairing Nonin & Omron 25](#_Toc510695146)

[4.9 Configuring Fitbit 26](#_Toc510695147)

[4.10 Configuring Chrome Kiosk Mode 29](#_Toc510695148)

[5 eWALL Integrated applications and services 31](#_Toc510695149)

[5.1 eWALL main screen 31](#_Toc510695150)

[5.2 New eWALL applications 31](#_Toc510695151)

[5.3 eWALL service bricks 35](#_Toc510695152)

[5.4 eWALL reasoners 36](#_Toc510695153)

[5.5 Benefits to COPD and MCI target groups 36](#_Toc510695154)

[Bibliography 38](#_Toc510695155)

# eWALL Devices and Software

Part of eWALL is installed in the client’s home. This comprises devices and software. The devices are fitted and configured, while the software (eWALL-specific or 3rd party) is installed in the Home PC or the smartphone.

The full list of devices that are could be located at the care recipient’s home are:

* Reference main screen (touchscreen): ELO 4201L
* Home PC: Gigabyte Brix
* Android smartphone: Samsung Galaxy S4
* Kinect for Xbox One & PC adapter
* Trivia: USB hub, cables (Ethernet, USB, HDMI), keyboard & mouse
* Socket sensors: Plugwise Stick and Circles
* Home environment sensors: Arduino explorer and custom packaged Arduino sensing solution
* Controlled lighting: Philips Hue gateway and lamps
* Wearable: Fitbit ChargeHR
* Omron 708-BT Blood Pressure Monitor
* Nonin, Onyx II Model 9560 Bluetooth Pulse Oximeter

The connections of these devices at home are summarized in Figure 1.

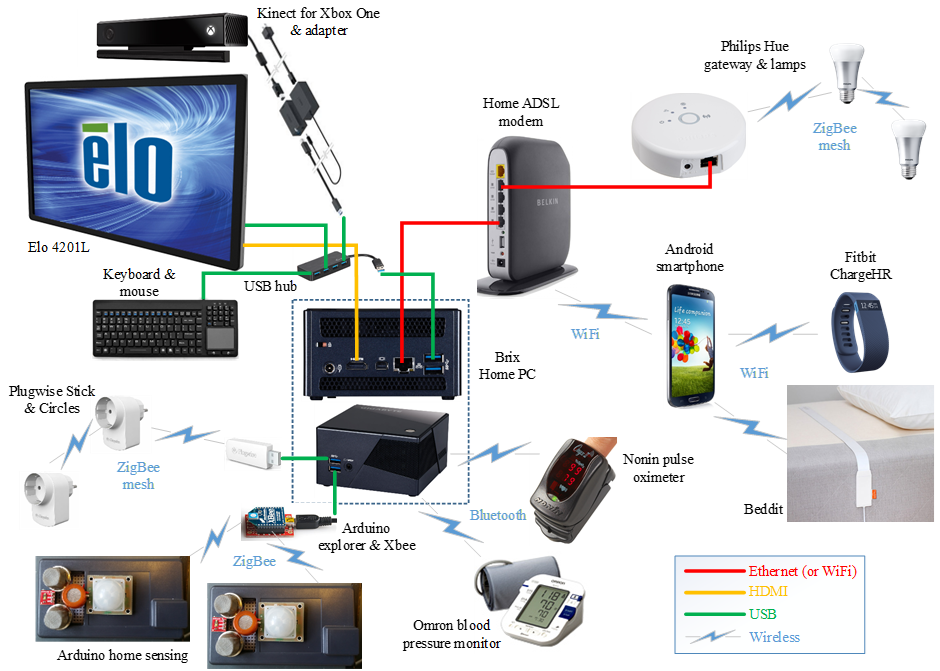
****

Figure 1. Wireless and wired connections of all home eWALL devices

## Software repository

The eWALL software for released for the user pilots is is hosted at the UPB GitLab server. The home environment can be found at:

<http://serv2.radio.pub.ro/gitlab/pilots/ewallhome/raw/master/ewallhome.tar.gz>

Access credentials are available for the eWALL consortium.

The developers’ repository resides at the same servers on address:

<http://serv2.radio.pub.ro/gitlab>

The home environment is split into:

* Arduino\_uC\_code: interface with the sensors, running on the sensor boards
* SocketSense: Power socket sensing code
* UserSim: Simulator for care recipient behavior
* DeviceGateway: The various device gateway components
  + Accelerometer: Class definition and implementation for the Accelerometer DGw client (processing component). Connects to the DGw via sockets, receives accelerometer data processes the data into IMA, ISA, steps and stores the results in CouchDB.
  + DGwClientConsole: Example console to run DGw clients.
  + DeviceGatewayConsole: Contains console applications to run DGw with Arduino (DeviceGatewayConsoleArduino.cpp), with WaspMotes (DeviceGatewayConsoleWM.cpp), with Kinect (DGwKinectTest.cpp) and with prerecorded data (DGwConsolePrerecorded.cpp). Configuration files are filled with json data, motesArduino.cfg for Arduino, motes.cfg for WaspMotes.
  + curl\_helper: curl\_helper library.
  + includeDir: Contains the interface classes, functions and structures definitions needed in all the DGw libraries.
  + json\_spirit: Library for JSON message parsing, included from the json\_spirit project (https://github.com/png85/json\_spirit).
  + json\_spirit\_dgw: Library for parsing dgw structures (defined in includeDir/DGWDefs.h into json strings). Needed for the socket interface and for reading of the configuration files.
  + libAVDevices: Contains the class definitions and implementations of AV Devices for the DGw: example AV device (libAVDeviceExample.h and libAVDeviceExample.cpp) and kinect AV device (libAVKinect.h and libAVKinect.cpp). freenectwrp.hpp is a wrapper for the libfreenect driver.
  + libDGwClient: Contains the class definitions and implementations of DGw clients utilizing the measurement and streaming data via sockets. libDGwClient.h and libDGwClient.cpp represent example implementations of the DGw client. libDGwClientSensors.h and libDGwClientSensors.cpp accept sensor measurements (temperature, humidity, gasses, liminocity...) and store them into CouchDB. sensorsToDB.cpp is a console application to run the DGwClientSensors.
  + libDGwSockets: Contains the class definitions and implementations socket interface between the Device Gateway (socket server) and the DGw clients (socket clients). The data is transferred as json strings using the json\_spirit\_dgw library.
  + libDeviceGateway: Contains the class definitions and implementations of the device gateway. The DGW can connect multiple sensor devices and multiple AV devices on one end and multiple clients (processing algorithms) via sockets on the other end.
  + libSensorDevices: Contains the class definitions and implementations of the sensor devices. The \libSensorDeviceArduino subdirectory contains the implementation of the Arduino sensor devices. The \libSensorDeviceWaspMote subdirectory contains the implementation of the WaspMote sensor devices. The \libSensorDeviceExample subdirectory contains an example sensor devices implementation. The \libSensorDevicePrerecorded subdirectory contains the implementation of prerecorded data reading and streaming
  + time\_funcs: Contains the c code time\_funcs.c with time-related function implementations.
  + libDeviceGatewayJavaWrapper: Contains the DeviceGateway library wrapper for JAVA developed using SWIG.
  + Antidote-master: Continua Alliance porting for USB and Bluetooth connectivity.
  + FaceProcessing: Face tracking application: tracks faces of persons, estimates expression and posts results to CouchDB
  + VisualProcessing: Simple visual processing application: calculates luminance and activity (boolean) in a frame and posts results to CouchDB. Also includes two sample programs, expRecKinect for plain face detection and expression recognition, and scrActKinect for activity recognition from Kinect dept images.
  + libVisual: Tracking and video processing libraries used by FaceProcessing
  + libTUSAlgos: Expression recognition and other libraries. Expression recognition is used by FaceProcessing and expRecKinect.

# Virtual Testbed Environment

The Virtual Test Bed Environment (VTBE) has been upgraded within WP6 for validating the M24 and M36 eWALL prototype. VTBE is important because, enabling thorough testing at different levels (unit test at development level, inter-components and end-to-end test at integration level), allows for efficiently testing, fixing and optimizing the eWALL prototype via an iterative process, towards the validation of the M24 prototype.

System Integration Testing (SIT) represents a set of tests that interrogate functionality of all the connected parts of the M24 and M36 eWALL prototype. The tests are performed manually and/or automatically.

**Error! Reference source not found.** represents the eWALL VTBE, which is used as testing and validation environment.

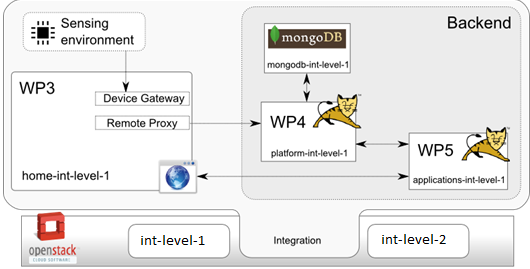


Figure 2. VTBE for validation of eWALL prototypes

The eWALL VTBE comprises the following environments:

1. **Development Environment:** consists of a set of virtual machines for the deployment and execution of the software artefacts created by the developers. This environment acts as the first integration and unit testing point, where developers coming from different companies and working on different aspects of the platform try to connect and make the software modules interact. A Continuous Integration and Delivery system automatically builds and unit-tests the developed components whenever new code is released on the code versioning system and, in case these phases are successful, automatically deploys them into the proper hosting servers in this environment.
2. **Integration Level 1** **Environment:** consists of a replica of the development environment, but is dedicated to the integration testing of the software modules. This environment is not accessed directly by developers or testers because it is dedicated to continuous integration and automated testing of specific parts of the eWALL platform.
3. **Integration Level 2 Environment:** consists of a replica of the development Environment, and is dedicated to the end-to-end testing of the whole platform. Int-level-2 environment is used for the validation of every release of the eWALL prototype. The eWALL system components from the development environment are deployed on int-level-2 environment to represent the currently released components of the eWALL platform and applications to be validated. On int-level-2 environment there are currently 6 active virtual machines, currently used for: data storage (mongodb-int-level-2); simulating home environments (home-int-level-2 and home-int-level-2- b); hosting WP4 platform services (platform-int-level-2); hosting services and applications (applications-int-level-2); caching data for efficient resource management (memcached-int-level-2). More in detail: the home-int-level-2 and home-int-level-2-b virtual machines are used for simulating data coming from the wearables and domotics sensors, and for their management via the device gateway, CouchDB and remote proxy components; the platform-int-level-2 virtual machine is used for the deployment of cloud gateway, eWALL portal, IDSS components, notification manager, profiling server and lifestyle reasoners; the application-int-level-2 virtual machine is used for deployment of all the applications and service bricks which enable the applications. Just to mention some of the applications: the DFM, DPAM and Healthcare Monitoring, the eWALL main screen, the cognitive games, the Caregiver Web Application. As mentioned, due to the difficulty of connecting real sensors to a testing environment, we set up simulated environments representing so-called “Shadow Users”. The generated data are used further in the end-to-end testing system for verification of data flow from CouchDB to eWALL applications.
4. **Production Environment** consists of the final environment used for the real pilot execution. This is not part of the VTBE, but we mention it as it is hosted on the same eWALL Cloud infrastructure and resembles the other environments (with the exception of the “home” machines, which will be obviously replaced by real remote installations in the primary users’ homes).
5. **Application Lifecycle Management Environment** consists of a set of servers, which host the tools enabling the whole application lifecycle management: versioning, continuous integration and delivery, web proxies, software artefacts repository and the monitoring of all the resources involved (Zabbix).

# eWALL prototype for the intra-pilots and the user pilots

In preparation for the eWALL end-user pilots we built and deployed several eWALL installation at the premises of the pilot partners. This process is assumed to be very important and critical for the respective medical parties since it represents and simulate the whole process of primary eWALL user home installation deployment and usability. Every of the medical partners deployed at least one full eWALL system configuration in own real home environment to assess the usability and mitigate the risks during deployment in real users’ home scenario.

In order to pass by the limitation of having one single physical cloud servers infrastructure and to ensure redundancy and reliability to the eWALL production (prod) environment, the decision was taken to migrate the prod environment to a cloud hosting provider. Assessing several options over an internal survey, Amazon Web Services (AWS) was selected as the cloud provider for the eWALL prod environment. AWS has been found by the EC to be fully compliant with EU Data Protection Directives therefore satisfying all the requirements for sensitive medical data. As result we successfully migrated our own OpenStack servers based eWALL cloud environment to the Amazon servers. This way, eWALL ensures 24/7 operational availability and significantly reduced risk of cloud infrastructure failures.

## User Pilots prototypes

The eWALL user pilots are planned to be executed in 4 waves; each of them for the period of 6 weeks. While the intra-pilots are already running since beginning of February, the demonstrations with actual subjects (COPD, MCI, ARI) started beginning of April using the current version of the software, namely v1.0. In the meanwhile, technical partners are developing the new functionalities, and a “code-freeze” is planned by the beginning of May. The feedback from the intra-pilots leverages a faster bug fixing process.

For the needs of the intra-pilots we have created several users on the aws-prod environment, which resides at the Amazon cloud and they are used for testing in the Amazon environment. We currently have the same user names in int-level-2, and the systems are up-and-running and generating sensing data from real users and devices.

At RRD, Netherlands, we have several reference home installations up and running. An example of operational eWALL metaphor is given on Figure 3.



Figure 3. eWALL home installation in Netherlands

As expected, the technology is hidden form the end users and they have access only to the eWALL main screen and applications. This eWALL metaphor was assess at RRD by several experts and users from the eWALL actual target groups (COPD, MCI, ARI), it was well accepted and was reported to be well presumed and non-intrusive.

At AAU, Denmark, a different metaphor will be used, as given in Figure 4. The main screen will be mounted on a screen stand and the HomePC will be on a small shelf below the screen. This change approach was adopted to avoid mounting the screen on the users’ walls at their homes, since that process would have required drilling.



Figure 4. eWALL lab installation at AAU, Denmark

# Building the eWALL prototype

In order to support the prototype building and deployment at the user partners’ sites, the eWALL technical partners jointly described in details the whole process for the HomePC installation and configuration in a single step-by-stem installation manual. This instructions are given in a way that non-technical expert person will be able to follow and complete. The step-by-step installation manual document is a part of the current deliverable and is described in this section.

On Figure 5 and Figure 6 some of the physical assembly components are given. The actual process of assembling and configuring a single whole home installation for a fist time was reported to take and average of one single person day by a technical partner who is familiar with the components. Thus the medical partners are planning to pre-assemble the home installations at lab environment and deliver them at the users’ homes - configured and ready to be plugged in. This will aim to reduce the time spend at the users’ homes for technical work.

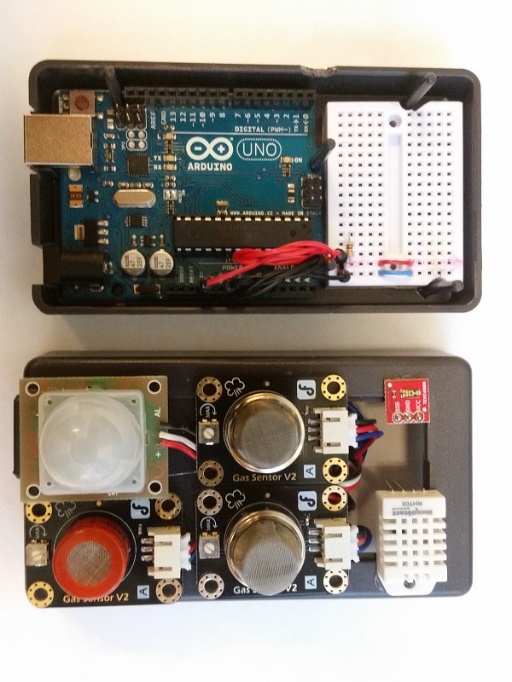


Figure 5. Arduino environmental module



Figure 6. Assembling the eWALL prototype at RRD

## Preparation

### Hardware Setup per eWALL

|  |  |  |
| --- | --- | --- |
| **#** | **Hardware Item** | **Link** |
| **Home PC** | | |
| 1 | GigaByte Brix i7 PC  - Additional 256GB+ SSD Drive  - Additional 8GB+ RAM |  |
| 1 | Ewent EW1131 4-port USB 3.0 Hub | [Link](http://www.ewent-online.com/ew1131-4-port-usb-3-1-hub.html) |
| 1 | Logitech Wireless Keyboard/Mouse | [Link](http://www.logitech.com/en-roeu/product/wireless-touch-keyboard-k400-plus) |
| 1 | Logitech Z150 Multimedia Speaker | [Link](http://www.logitech.com/en-roeu/product/multimedia-speakers-z150) |
| 1 | IIYAMA ProLite 42” Touch Screen | [Link](http://www.iiyama.com/gb_en/products/prolite-tf4237msc-b1ag/) |
| 1 | CR 2025 Battery for Screen Remote |  |
| 1 | HDMI Cable |  |
| 1 | 9-Socket Power Extension Cord |  |
| 1 | TP-LINK 4020 2-Port Powerline Adapter | [Link](http://www.tp-link.com/en/products/details/cat-18_TL-PA4020PKIT.html) |
| **Cabinet** | | |
| 1 | IKEA Nornas Cabinet | [Link](http://www.ikea.com/gb/en/catalog/products/00282231/) |
| 1 | IKEA Fado Table Lamp | [Link](http://www.ikea.com/gb/en/catalog/products/10096375/?query=FADO+Table+lamp) |
| 1 | Screen mounting DIY equipment |  |
| **Commercial Sensors** | | |
| 1 | Microsoft Kinect for XBox One | [Link](http://www.ikea.com/gb/en/catalog/products/10096375/?query=FADO+Table+lamp) |
| 1 | Microsoft Kinect Adapter for Windows | [Link](http://www.microsoftstore.com/store/msusa/en_US/pdp/Kinect-Adapter-for-Windows/productID.308803600) |
| 1 | Kinect Camera Holder/Stand |  |
| 1 | FitBit Charge HR | [Link](https://www.fitbit.com/uk/chargehr) |
| 1 | Philips Hue Starter Pack E27 |  |
| 1 | Nonin Bluetooth PulseOximeter WristOx | [Link](http://www.nonin.com/OEMSolutions/WristOx23150-OEM) |
| 1 | Omron 708-BT Blood Pressure Monitor | [Link](http://www.healthcare.omron.co.jp/bt/english/) |
| **Environmental Sensors** | | |
| 1 | Environmental Sensing Receiver |  |
| 4 | Environmental Sensor |  |

### **Connect all the devices**

* **Attention:** the Home-PC and the Philips Hue Hub must be in the same local network in order for these devices to be able to communicate with each other.
* **Attention:** the IIYAMA TF4237MSC 42” Touch Screen requires a lot of power. If the power supply is connected to a “power hub” together with all the other devices, it may not get enough power to operate! In this case, plug the screen directly into a wall socket.

### Prepare Ubuntu Installation Media

**Attention:** Make sure to download **14.04.3** and not the latest Ubuntu version!

1. Download Ubuntu 14.04.3 LTS from the Ubuntu website: <http://releases.ubuntu.com/14.04.3/ubuntu-14.04.3-desktop-amd64.iso>
2. Create a Bootable USB stick by following the instructions below
   1. Creating bootable USB from Windows: <http://www.ubuntu.com/download/desktop/create-a-usb-stick-on-windows>
   2. Creating bootable USB from Ubuntu: <http://www.ubuntu.com/download/desktop/create-a-usb-stick-on-ubuntu>

Creating bootable USB from Mac OSX: <http://www.ubuntu.com/download/desktop/create-a-usb-stick-on-mac-osx>

## Ubuntu Installation

### Changing the PC Boot Order

In order to install Ubuntu from a USB stick, first set the correct boot priority on the machine.

**ATTENTION:** Additional help can be found at: <http://www.boot-disk.com/boot_priority.htm>

### Boot Ubuntu from USB (GigaByte Brix)

1. Insert the USB bootable Ubuntu Install Media in a free USB slot on the Brix PC.
2. While the Bric PC is starting up, press [Del] to enter the BIOS mode.
3. Use the arrow keys to navigate to the “Boot” tab.
4. Select “Boot Option #1” and press [Enter].
5. Select the USB key option (e.g. “[UEFI: USB2.0 Flash…]”) and press [Enter].
6. Go to the ‘Save & Exit’ tab.
7. Highlight “Save Changes and Reset” and press [Enter].
8. Select “Yes” on the confirmation screen and press [Enter].

### Boot Ubuntu from USB (Zotac)

1. Insert the USB bootable Ubuntu Install Media in a free USB slot on the Zotac box.
2. While the Zotac PC is starting up, press [Del] to enter the BIOS mode.
3. Use the arrow keys to navigate to the “Boot” tab.
4. Use the arrow keys to go down to the option “Hard Drive BSS Priorities” and press [Enter].
5. Highlight “Boot Option #1” and press [Enter].
6. Select the “USB Flash Disk” option (or similar) and press [Enter].
7. Press [Esc] to go back to the main Boot menu.
8. Highlight “Boot Option #1” make sure it shows the USB Flash Disk option.
9. Navigate to the right “Save&Exit” and select “Save Changes and Exit”.
10. In the popup menu select “yes”.

### Installing Ubuntu 14.04.3

When the boot option configuration was successful, the machine should restart and boot from the USB installation media and show the Ubuntu installation menu:

**ATTENTION:** Additional help can be found at: <https://help.ubuntu.com/community/Installation/FromUSBStick>

1. Select “Install Ubuntu”.
2. Select installation language “English” and press [Continue].
3. Select [I don’t want to connect to a wi-fi network right now] and press [Continue].
4. If there was already an Ubuntu version installed on the machine, select [Erase disk and install Ubuntu] and press [Install Now].
5. In the “Write the changes to disk?” pop-up select [Continue].
6. Select the Time zone in which you’re in by clicking on your location at the map and press [Continue].
7. For the keyboard layout select the default option of “English (US)” with “English (US) - English (US, International with dead keys) and select [Continue].
8. Set the following information as indicated by the naming scheme Table below:
   1. Your name: [PC Name]
   2. Your computer’s name: [PC Name]
   3. Pick a username: [Username]
   4. Choose a password: [Ubuntu Password]
   5. Confirm your password: [Ubuntu Password]
   6. ATTENTION: Select [Log in automatically] for convenience.
9. Press [Continue] and wait for the installation to Finish.
10. Press [Restart now] after the installation has finished.

## TABLE: Ubuntu PC Name and Username Scheme

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Country** | **Notes** | **PC Name** | **Username** | **Ubuntu Password** |
| Netherlands |  | ewall-nl1 | ewall-nl | XXXXX |
|  |  | ewall-nl2 | ewall-nl | XXXXX |
|  |  | ewall-nl3 | ewall-nl | XXXXX |
|  |  | ewall-nl4 | ewall-nl | XXXXX |
|  |  | ewall-nl5 | ewall-nl | XXXXX |
| Italy |  | ewall-it1 | ewall-it | XXXXX |
|  |  | ewall-it2 | ewall-it | XXXXX |
|  |  | ewall-it3 | ewall-it | XXXXX |
|  |  | ewall-it4 | ewall-it | XXXXX |
|  |  | ewall-it5 | ewall-it | XXXXX |
| Denmark |  | ewall-dk1 | ewall-dk | XXXXX |
|  |  | ewall-dk2 | ewall-dk | XXXXX |
|  |  | ewall-dk3 | ewall-dk | XXXXX |
|  |  | ewall-dk4 | ewall-dk | XXXXX |
|  |  | ewall-dk5 | ewall-dk | XXXXX |
| Austria |  | ewall-au1 | ewall-au | XXXXX |
|  |  | ewall-au2 | ewall-au | XXXXX |

## Configuring Fresh Ubuntu Install

### Connect to the Internet

First, connect the Brix PC to a wired or wireless network.

* **Attention:** The Brix-PC and Philips Hue Bridge must be connected to the same local network in order for communication between these devices to work!

### Disable Automatic Updates

1. Open the Ubuntu Search (upper left corner of the Ubuntu desktop)
2. Type Software & Updates and open the application
3. Select the tab “Updates”
4. Uncheck all updates checkboxes
5. Select “Never” from the “Automatically check for updates:” dropdown menu

**ATTENTION:** If an “Automatic updates” window ever pops up, press cancel. Do not install Ubuntu updates automatically.

### Install TeamViewer

To make sure you can do the rest of the installation and configuration from a remote location, first install TeamViewer 11.

1. On the Brix PC, open FireFox and go to: [https://www.teamviewer.com/en/download/linux.aspx](https://www.teamviewer.com/nl/download/linux.aspx)
2. Under “Ubuntu, Debian”, select “Download deb v11.x.xxxxx”. (32-Bit / 64-Bit Multiarch)
3. Select [Save file] in the FireFox popup menu.
4. Open the downloaded file (this will open the Ubuntu Software Centre).
5. At this point Ubuntu Software Center will give a dependency error (libc6).
6. Close Ubuntu Software Center.
7. Go to System Settings -> Software & Updates -> Updates
8. Deselect “Important security updates (trusty-security)”
9. Select again “Important security updates (trusty security)”
10. Click the “Close” button
11. In the popup-menu select “Reload”.
12. Open a File Explorer and double click the downloaded TeamViewer install file again.
13. At this point the dependency error should no longer show and you can click “Install”.

### Configuring TeamViewer

Before running and configuring TeamViewer on the Brix PC, make sure there is a local (country-wide) TeamViewer account setup.

**Create Teamviewer Account:**

1. Open TeamViewer 11 and Accept the License Agreement.
2. In the “Computers & Contacts” window click “Sign Up”.
3. Fill in the form with the following info:
   1. Select “Create a free TeamViewer account”
   2. Your Name: “eWALL [COUNTRY]” (e.g. “eWALL NL”)
   3. Email / username: Enter an existing valid email address of the country’s main local administrator.
   4. Password: Choose a secure password
   5. Click Next.

**ATTENTION:** For security and privacy reasons the password selected for the TeamViewer account should be secure and kept only for the local administrators (unique for every user and/or country). This password should be protected under the approved form eWALL security and privacy terms and procedures.

**Configuring Teamviewer:**

1. In the “Computers & Contacts” window, sign in to your TeamViewer account (**ATTENTION:** Don’t forget to check the box “Keep me signed in”).
2. In the “TeamViewer” main window, select the checkbox “Start TeamViewer with system”.
3. Select the checkbox “Assign device to account” and click “Assign”.
4. Select the checkbox “Grant easy access”.
5. Restart the Ubuntu machine and make sure TeamViewer automatically starts up and connects to your TeamViewer account.

### Disable Screen Lock (Verified)

1. Open the “System Settings” menu from the Launcher.
2. Select “Brightness & Lock”.
3. Set the value of “Turn screen off when inactive for:” to “Never”.
4. Set the “Lock” option to “Off”.
5. Uncheck “Require my password when waking from suspend”.
6. Close the Setting window.

## Third-Party Software Installations

### Install Google Chrome (Verified)

1. On the Brix PC, use FireFox to go to: <https://www.google.com/intl/en-US/chrome/browser>
2. Click “Download Chrome”
3. Select “64 bit .deb (For Debian/Ubuntu) and click “Accept and Install”.
4. In the pop-up window select “Open with: Ubuntu Software Center (default)” and click “OK”.
5. In the Ubuntu Software Center click “Install”.
6. Close Ubuntu Software Center and go to “Search your computer and online sources”.
7. Type “Chrome” and launch Google Chrome.
8. Check “Make Google Chrome the default browser” and click “OK”.
9. In the Launcher, right-click the Chrome icon and select “Lock to Launcher”.
10. Close Google Chrome.

### Install VNStat Network Monitoring tool (Verified)

**ATTENTION:** Additional help can be found here: <http://www.thegeekstuff.com/2011/11/vnstat-network-traffic-monitor/>

1. Open a Terminal window.
2. Type **sudo apt-get install vnstat** [Enter].
3. Configure vnstat to monitoring the correct network interface (e.g. eth0 or wlan0):
   1. To check all available network interfaces go to your Terminal window and type **vnstat --iflist** [Enter].
   2. If the machine is configured to connect wirelessly to the internet, choose the **wlan0** or similar interface in the following step.
   3. If the machine is connected to a wired network, choose **eth0** or similar in the following step.
4. In the Terminal, type **sudo vnstat -u -i wlan0** [Enter] (replacing the “wlan0” with your network interface).
5. Start the vnstat service by typing **sudo service vnstat start** [Enter].
6. Make sure all files in the folder /var/lib/vnstat are owned by “vnstat:vnstat”:
   1. In a Terminal window type **cd /var/lib/vnstat** [Enter].
   2. Type **sudo chown vnstat:vnstat \*** [Enter].
   3. Type **ls -al** [Enter] to check if all files (e.g. wlan0, .wlan0,) are owned by vnstat user and vnstat group. If this is not the case, use the chown vnstat:vnstat command on the individual files in this directory until they all have the correct user:group permissions.
7. Use some internet data (e.g. open a YouTube movie).
8. Type **sudo vnstat -u** [Enter] to force a database update.
9. Type **vnstat** [Enter] to check the functionality of the tool (this should show a table with network usage up till now).

### Install Galileo FitBit Sync Tool (In Progress)

1. Make sure the FitBit dongle is plugged directly into the eWALL machine (not through a USB hub)
2. Open a Terminal window and perform the following steps:
   1. sudo add-apt-repository ppa:cwayne18/fitbit
   2. sudo apt-get update
   3. sudo apt-get install galileo
   4. sudo start galileo
3. Type cd /etc/udev/rules.d/ [Enter]
4. Type sudo touch 99-fitbit.rules
5. Type sudoedit 99-fitbit.rules
6. Add the following to the 99-fitbit.rules file (in one line):

SUBSYSTEM=”usb”, ATTR{idVendor}==”2687”, ATTR{idProduct}==”fb01”, SYMLINK+=”fitbit”, MODE=”0666”

1. Save the file (CRTL+O) [Enter] and exit the editor (CRTL+X)
2. Type sudo service udev restart
3. Done

## Install eWALL Home Environment

**ATTENTION:** At this point, before installing the Home Environment software, make sure that the Microsoft Kinect is physically connected to the machine (on a USB3.0 port on the machine itself, not on the USB Hub). If the Kinect is not attached the software will fail to launch.

### Downloading latest eWALL Home Environment Software

1. Open Google Chrome and go to: <http://serv2.radio.pub.ro/gitlab/pilots/ewallhome/raw/master/ewallhome.tar.gz>
2. Login to Gitlab using the following info:
   1. Username: “ewall”
   2. Password: “\*\*\*\*\*\*\*\*\*\*”
   3. Click “Sign In”
3. Open a File Explorer and go to “Downloads”
4. Right-click the “ewallhome.tar.gz” file and select “Extract here”.
5. Right-click the “ewallhome” folder and select “Cut”.
6. Go to the “Home” folder in the File Explorer, then Right-click -> Paste.
7. **ATTENTION:** When updating from a previous version, make sure the old folder “ewallhome” is removed first (see Section 5.4).

### Configuring Environmental Sensing Nodes

It is very important to note that you need to change or set content the motesArduino.cfg file. For every HomePC you will have a different set of sensors with different cfg. files respectively.

If you already have the preconfigured file, you can update the motesArduino.cfg file before you build the project,following this path:

*…\DeviceGateway\DeviceGatewayConsole*

Inside this folder, open the motesArduino.cfg file, using a text editor and replace with the the .cfg file that has been preliminarily prepared for the installation.

or directly change the file in the build folder:

…*.\build\DeviceGateway\DeviceGatewayConsole*

If you start from a scratch with building or setting the Arduino environment modules, you will need to follow the building and configuration instructions that are part of WP3 and could be found at:

<https://docs.google.com/document/d/14OH8SPcFdRKb2QXY-bsJqOs3MbMADl0BF9o_eONpx-A/edit>

### Installing eWALL Home Environment

1. Open a Terminal window.
2. Type **cd /home/{ewall-xx}/ewallhome** [Enter] (replace “{ewall-xx}” with your country’s username, see Section 2.3).
3. **ATTENTION:** Before running the following install script, make sure the Kinect is physically connected to the machine on a USB3.0 port (not through a USB Hub). The DeviceGateWay will fail to run if no Kinect is detected.
4. Type **./install\_dgw.sh** [Enter]
5. When the installation script is done, you will have to login to Ubuntu.

The update of all ewallhome components is done with the update\_ewallhome.shscript located in your ewallhome installation directory **/home/{ewall-xx}/ewallhome**.

1. Open a Terminal window.
2. Type **cd /home/{ewall-xx}/ewallhome** [Enter] (replace “{ewall-xx}” with your country’s username, see Section 2.3).
3. Type **./update\_ewallhome.sh** [Enter](you will be asked for sudo password)
4. When the update is done, you will have to login to Ubuntu.

ATTENTION! If update\_ewallhome.sh is missing in your ewallhome installation directory, you will have to download it from the latest ewallhome.tar.gz package. **Copy only** the **update\_ewallhome.sh** script in your current ewallhome installation directory /home/{ewall-xx}/ewallhome. Do not overwrite the whole directory, you will lose configurations.

## Configuring Philips Hue

1. Make sure that that the Philips Hue Hub and the Home PC are connected to the same network.
2. in terminal type **cd /home/{ewall-xx}/ewallhome/local-platform/** (where ***ewall-XX*** is the PC username) to move to local-platform folder

*Note: Make sure file*  ***configureLight-controller.jar*** *is present in this folder (use command* ***ls -a*** *to see list of files), if not, you must update eWALL Home Environment software (see chapter 5.4)*

1. Type **java -jar configureLight-controller.jar** and follow the instructions in terminal
2. When configuration is complete, exit (press **Ctrl+C**)

## Installing eWALL Local Platform

*ATTENTION!*  Before continuing further, for security reasons, the user, sensing environment and devices details must be added to eWALL system. Each home sensing environment deployment must have its own unique **sensing environment ID** and related primary user's credentials (**username** and **password**). These parameters are obtained from the region administrator who provisions related user and data, designated home sensing environment and devices data in eWALL Cloud using eWALL Portal before installation and configuration of software in home sensing environment. Contact your region administrator for more information on that!

*(Instructions for provisioning of data at eWALL Cloud:*

[*https://docs.google.com/document/d/124zFOlgowhbycxljzngKndGYw3-vFW3vlQgNwuK4Osc/edit#heading=h.6dttx0ux1c7h*](https://docs.google.com/document/d/124zFOlgowhbycxljzngKndGYw3-vFW3vlQgNwuK4Osc/edit#heading=h.6dttx0ux1c7h)*)*

### Configuring Local Platform software

**Sensing Environment Configuration:**

1. Open a File Explorer and go to **/home/{ewall-xx}/ewallhome/local-platform/** (replace “{ewall-xx}” with your country’s username).
2. Right-click the file “config.properties” and select “open with gedit”
3. Set the following parameters:
   1. eu.ewall.platform.remoteproxy.sensingenvironment.id = xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
   2. eu.ewall.platform.remoteproxy.system.username = primary-user-name
   3. eu.ewall.platform.remoteproxy.system.password = primary-user-password
4. You can get the parameter values by logging in the eWALL Portal. Here is a guide to doing that: [https://docs.google.com/document/d/124zFOlgowhbycxljzngKndGYw3-vFW3vlQgNwuK4Osc/edit#](https://docs.google.com/document/d/124zFOlgowhbycxljzngKndGYw3-vFW3vlQgNwuK4Osc/edit)
5. Save the change

**Configuring Start/Stop Script**

1. Open a File Explorer and go to **/home/{ewall-xx}/ewallhome/local-platform/**
2. Right-click the file “local-platform.conf” and select “Open with gedit”.
3. Find line numbered 11 and 14 and change the path to match the Local Platform install path:
   1. [line 11] “chdir **/home/{ewall-xx}/ewallhome**/local-platform/”
   2. [line 14] “java -jar **/home/{ewall-xx}/ewallhome**/local-platform/local-platform-all.jar -resume”
4. Save the file and close the gedit program.
5. Launch a terminal
6. Type **sudo mv /home/{ewall-xx}/ewallhome/local-platform/local-platform.conf /etc/init/** and press Enter.

**Reconfiguring Sensing Environment to connect to different eWALL Cloud system:**

1. Open a File Explorer and go to **/home/{ewall-xx}/ewallhome/local-platform/** (replace “{ewall-xx}” with your country’s username).
2. Right-click the file “config.properties” and select “open with gedit”
3. Set the parameter *eu.ewall.platform.remoteproxy.root* to proper value:
   1. if using production environment (Amazon Web Services)

*eu.ewall.platform.remoteproxy.root=***https\://ewall.upb.ro/platform-prod-aws/**

* 1. if using int-level-2 environment (for testing)

*eu.ewall.platform.remoteproxy.root=***http\://ewall.radio.pub.ro/platform-int-level-2/**

1. Save the changes to the file
2. Start or restart the local-platform

**(Optional) HTTP Proxy Configuration:**

If an http proxy is used to access the public internet, follow the steps below to configure the local platform accordingly:

1. Open a File Explorer and go to “/home/ewall-XX/local-platform/
2. Right-click the file “config.properties” and select “open with gedit”
3. Find the “# proxy settings” line. Uncomment and fill in the host and proxy values for the two lines below:
   1. http.proxyHost={xxx.xxx.xxx.xxx}
   2. http.proxyPort={xxxx}

### Running and Using the Local Platform software

1. To start local-platform type **sudo service local-platform start** [Enter] in terminal (wait 30 seconds)

**ATTENTION:** The local-platform software will start automatically when the system boots (you do not need to start it manually if you reboot your home PC).

1. To see the current status of the local-platform software, open a terminal and type **sudo service local-platform status** [Enter]. If the software is running, the output given should be “local-platform start/running”.

(Optional) To control the local-platform software, use any of the following commands in a terminal window:

* Start: **sudo service local-platform start** [Enter]
* Stop: **sudo service local-platform stop** [Enter] (wait a few seconds for graceful shut-down)
* Restart: **sudo service local-platform restart** [Enter]

### Updating Local Platform software

1. Local platform is updated through update of eWALL Home Environment software, see chapter 5.4

## Pairing Nonin & Omron

**Preparation:**

1. Make sure Bluetooth is enabled on the Brix P
   1. Go to System Settings -> Bluetooth and make sure Bluetooth is set to “On”.
2. Open a Terminal window and stop the eWall Home Service by typing **sudo service dgw-daemon stop** [Enter]
3. In the same Terminal window type **cd /home/ewall-XX/ewallhome** [Enter]
4. Leave the Terminal window open.

**Pairing the Nonin:**

1. Make sure the Nonin device has proper batteries inside and turn it on:
   1. For the WristOx Sensor: plug the finger connector into the wrist-worn part, and insert your finger into the finger clip.
   2. Check if the BlueTooth icon is shown on the Nonin screen. Press the gray button on the nonin device until the bluetooth icon appears.
2. Run the pairing script with **./pairNonin.sh** [Enter]
   1. The pairing script should finish with the message: “*Pairing succesfull. :)*”
   2. If pairing script outputs “Nonin device not discovered, try again!” run the pairing script again until successful.

**Pairing the Omron:**

1. Get the Omron Device and put the device in “Pairing Mode”:
   1. Press the “<” button to go to a previous measurement screen (or an empty previous measurement screen if no measurements were taken yet).
   2. Press and hold the “Upload” button for three seconds (a ‘P’ should appear, combined with an animation on the screen.
2. In the Terminal window type **./pairOmron.sh** [Enter] to start the pairing procedure
3. If this fails, run the command again until you see the message “Pairing successfull. :)”

**Testing:**

1. Start up the device gateway service again by typing **sudo ./xdgw-restart** [Enter]
2. Wear the Nonin PulseOximeter for a few minutes.
3. Take a Blood Pressure measurement with the Omron, finish by pressing “Upload” on the device, a message of “Err”, or “OK” should appear. Retry until the message “OK” appears.
4. Check the My Health application to see if the values appear.

## Configuring Fitbit

**ATTENTION:** A FitBit.com account can only be “paired” to a single eWALL user across environments. If you have your Fitbit.com account “[dave@gmail.com](mailto:dave@gmail.com)” connected to a user in the dev-environment, the same account cannot be paired to a user in the int-level-2 environment.

To complete the pairing of a Fitbit.com account to an eWALL user you need the following info:

* FitBit.com account;
* FitBit.com password;
* eWALL regional admin account;

The procedure of configuring Fitbit is done using the FitbitAuthenticator app. The link to the app is:

<http://ewall.radio.pub.ro/applications-int-level-2/FitbitAuthenticator/>

Execute the link in Google Chrome browser which should open the login page presented below- Figure 7:

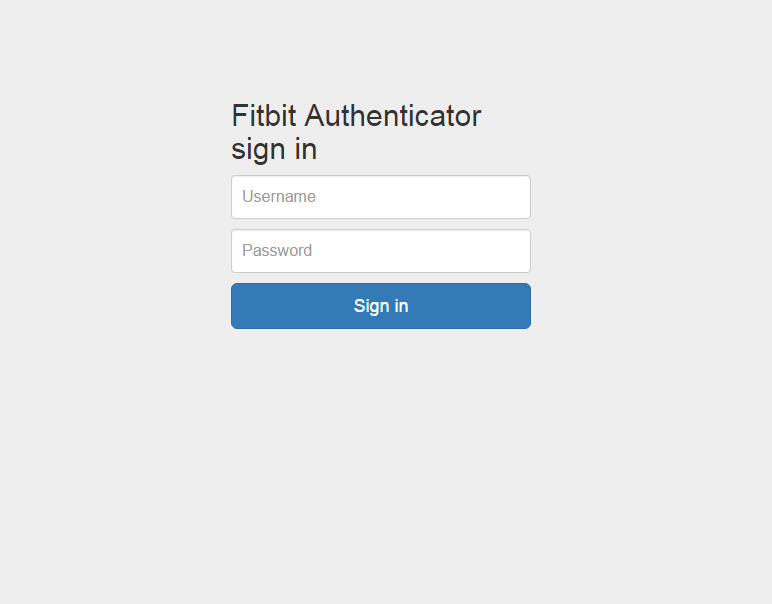


Figure 7 Fitbit Authenticator-sign in window

In the *Username/Password* fields enter the corresponding eWALL regional admin’s username and password and press *Sign In.* If they are valid you will see the main page of the application. Note that the application is only in English.

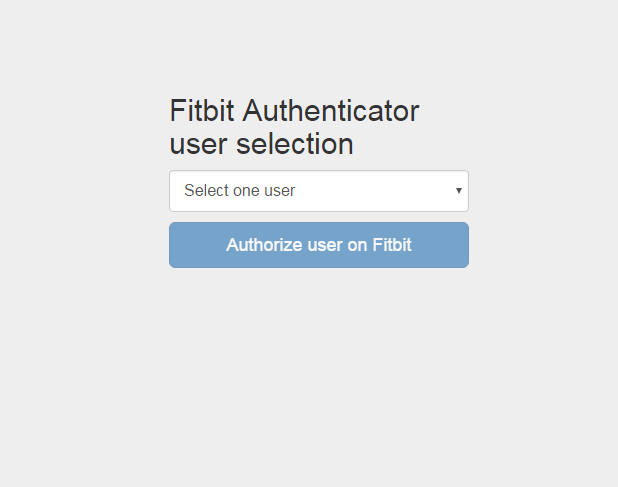


Figure 8 Fitbit Authenticator - user selection window

In the main page there is a drop down menu with text *Select one user (*Figure 8*)* Press the menu and a list with the users assigned to the regional admin will popup. Select the user you want to configure and press the button *Authorize user on Fitbit*. This will open the Fitbit page in new tab of the browser where you need to enter the username and password of the Fitbit account. Below is screenshot of the Fitbit’s website login page-Figure 9:

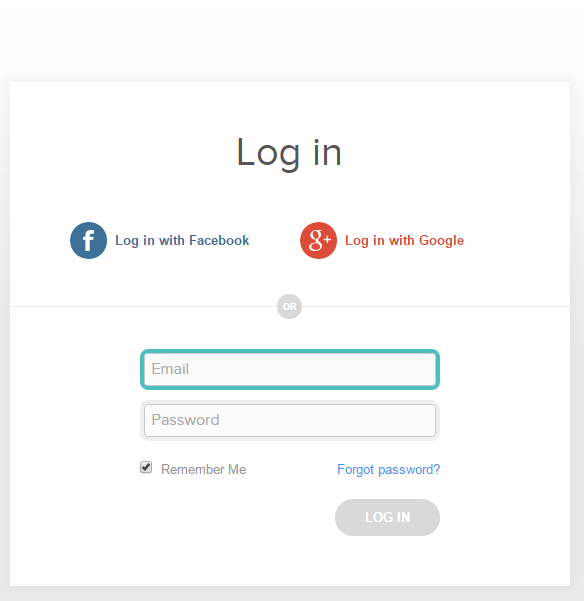
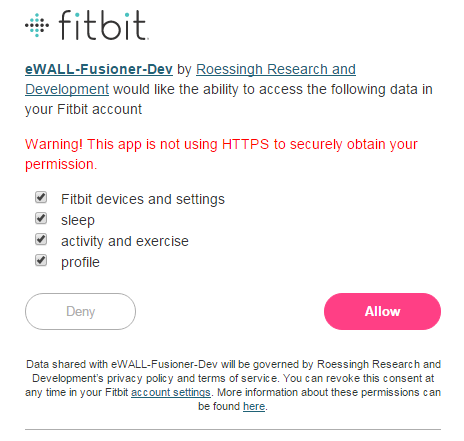


Figure 9 Login to Fitbit account

When ready with the credentials press the *Log In* button. This will lead you to the user’s personal Fitbit page. It consists of multiple checkboxes each one selecting the type of data to be delivered by Fitbit to eWALL. Select all of these and press the button *Allow*. Example screenshot is presented below.



Find the button *Allow* and press it to grant FitBit.com permission to the eWALL Fusioner to read data from this account. This will automatically close the Fitbit page (i.e. the tab in which the Fitbit page was opened) and redirect you to the Fitbit authenticator page. On that page you will see message with the status of the procedure. If it is successful you will get a message with title Success. Press the OK button of the message and the message will close and the main page of the application will remain open. Press the button located in the upper right corner to log out.

**ATTENTION!** It could happen that you want add authenticate other users, you must close the browser, then reopen it and repeat the complete procedure since the beginning. This is due to the temporary cache used by the browser.

## Configuring Chrome Kiosk Mode

1. **Using Google Chrome**, access the Kiosk extension URL <https://goo.gl/AestrL>
2. Click on Add to Chrome button (top right)
3. Follow the installation instructions -> you will see a new tab saying it was installed
4. Go back to the Chrome Web Store and click on the Launch App-Figure 10
5. Choose a username and password for the administrator account
6. Configure Kiosk for eWALL
   1. For WP6 partners use the following URL: [**http://ewall.radio.pub.ro/applications-int-level-2/mainscreen-new/**](http://ewall.radio.pub.ro/applications-int-level-2/mainscreen-new/)
   2. For Medical partners use the following URL: [**https://ewall.upb.ro/applications-prod-aws/mainscreen-new/**](https://ewall.upb.rp/applications-prod-aws/mainscreen-new/)

6. Click Save -> Kiosk will close

7. Open a terminal and navigate to the installation directory: **/home/{ewall-xx}/ewallhome/**

8. Type the command **sudo cp kiosk-start.desktop /home/{ewall-xx}/.config/autostart/**

9. Log out and log into Ubuntu again, the Kiosk will start after 30 seconds -> It will display the eWALL main screen in kiosk mode.

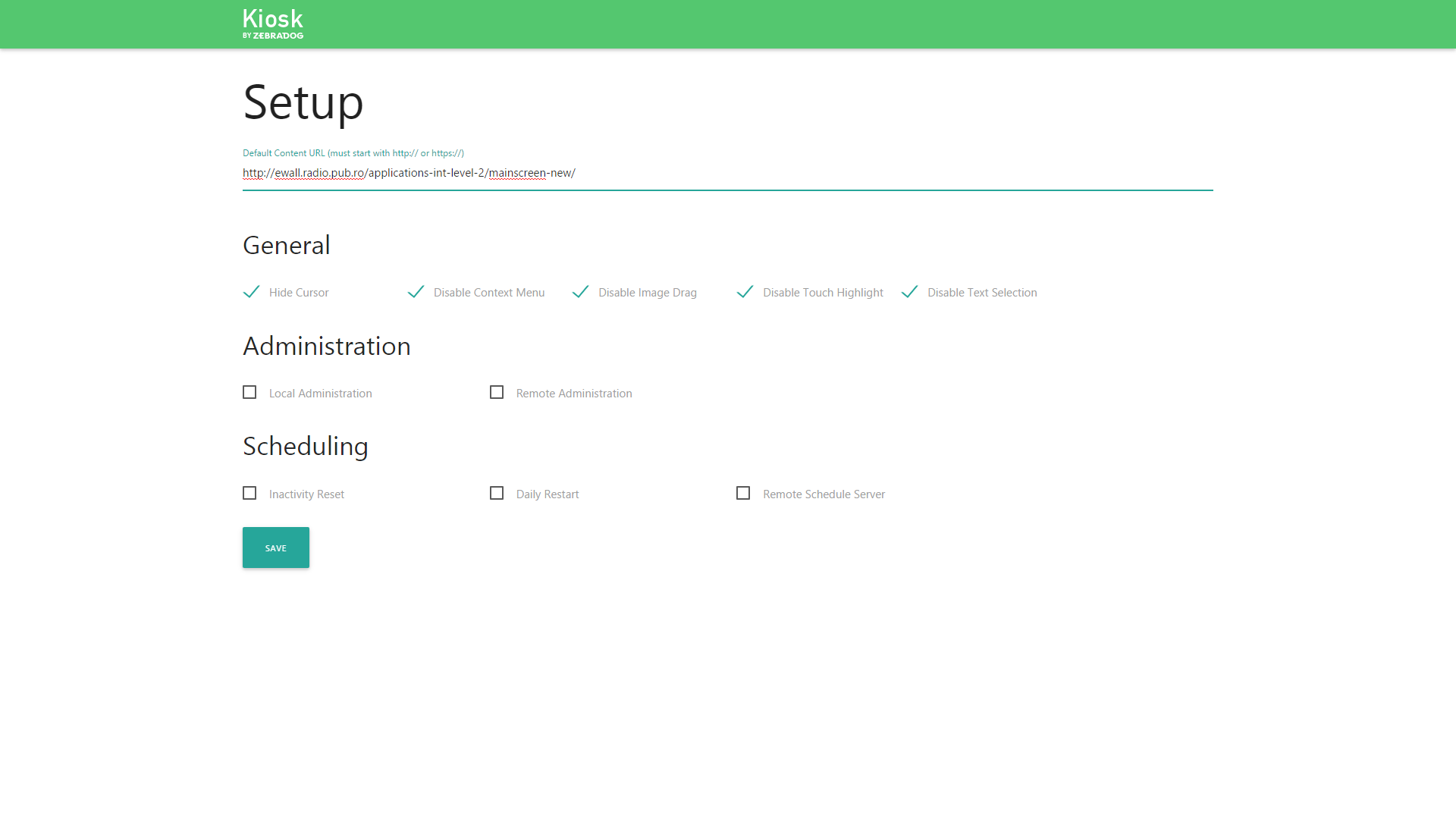


Figure 10 Kiosk setup configurations window

# eWALL Integrated applications and services

In M36 eWALL Prototype focus was to develop additional software components as well as to improve the existing ones (from M24). Furthermore in addition to the existing applications and services eWALL consortium defined and described a list of new features and applications that will be an objective of future development.

## eWALL main screen

The redesigned eWALL main screen has the purpose of supporting the eWALL widgets and applications. It offers a space and context of interaction, through which the end-user can view and interact with the eWALL applications. It can be defined as an operating system user interface, specialized and personalized by the user’s profile data. The main screen acts as a “container” for all the applications and GUI elements exposed to the end user. Indeed, in addition to the visual aspects, the main screen implements behind the scenes a set of horizontal functionalities which allow applications to interact in a secure way with the remote side of the eWALL platform, and that provide personalization of the GUI based on the user profile and remote service configuration.



Figure 11. eWALL main screen

## New eWALL applications

A wide list of potential applications was discussed and analysed in series of cross work package meetings and collaboration. After extending the list of applications that will bring more of the eWALL intelligence to primary end users, the consortium agreed on a short list which is feasible for development before the start of the user trails. The full list of functionalities contains, Fall Preventions Program, My Cookbook Application, My Shopping List, My Settings, Personal Daily Support Service, Sleep Diary, Notification System for Environmental Monitoring, Reward System for Physical Activity and a set of new Cognitive Games as well as improvements to existing functionalities, such as My Health, Calendar, along with other additional improvements. These applications are extensively described in D5.3.2.

The applications are one of the core components of eWALL, as they provide the interface between the user and the system. The eWALL main screen has the purpose of supporting the eWALL widgets and applications. It offers a space and context of interaction, through which the end-user can view and interact with the eWALL applications.

The eWALL prototype (Ph3/M36) has been developed by work package WP4. It builds on M18 and M24 versions which have been reported in deliverables D4.1.2, D4.2.2, D4.3.2, D.4.4.2, and D.4.4.3.

WP4 components are stated in the following table (Table 1)

Table 1. WP4 components in M24

|  |  |
| --- | --- |
| **WP4 component** | **Responsible partner(s)** |
| Remote Proxy | ENT UPB |
| Cloud Gateway | ENT, UPB |
| Profiling Server | ENT |
| Commons data model | ENT |
| IDSS reasoner automatic goal-setting | RRD |
| IDSS reasoner sleep anomaly classifier | TUS, UOZ |
| Notification Manager | UKIM |
| eWALL Portal | HP, ENT |
| eWALL Platform login | HP |
| Data manager | ENT |
| eWALL Eureka server | ENT |
| IDSS exercise reasoner | ENT |
| IDSS fall detection reasoner | UKIM |
| IDSS activity coach reasoner | RRD |
| Local data manager | ENT, AAU |

All WP4 components can be found in WP4 project’s GitLab[[1]](#footnote-2) server (<http://serv2.radio.pub.ro/gitlab>), which is connected to project’s Jenkins[[2]](#footnote-3) Continuous Integration server (http://ewall.radio.pub.ro/jenkins/) configured to automatically deploy components on the Platform-dev cloud environment[[3]](#footnote-4).

In comparison with M18 delivery, following new features and components have been added:

* **Remote Proxy component** – update of M18 version (most relevant: improvement in robustness, new data support added, notification bus interface added )
* **Cloud Gateway component** – update of M18 version (most relevant: new data support added – e.g. vitals sensing, sensing environment, appliance power sensing, domotics and actuator support).
* **Profiling Server component** – update of M18 version (most relevant: new data support added
* **Commons data model** - update of M18 version with new ontologies and refinement of existing ones with respect to new requirements.
* **IDSS reasoner automatic goal-setting component** - minor updates with respect to M18 version.
* **IDSS reasoner sleep anomaly classifier** **component** – small updates in respect to M18 version
* **Notification manager component-** update of M18 version with support for configuration of user and notification selection
* **eWALL Portal component** – update of M18 version (most relevant: support for time zone handling for users, different data model related update, localizations updates, devices configuration added)
* **eWALL Platform login component** – update of M18 component
* **Data manager** **component** - initial version provided for M24 (did not exist in previous M18 delivery).
* **eWall Eureka server component** - initial version provided for M24 (did not exist in previous M18 delivery). It acts as a central point where eWALL services are registered and discovered.
* **IDSS exercise reasoner component** - initial version provided for M24 (did not exist in previous M18 delivery).
* **IDSS fall detection reasoner component** - initial version provided for M24 (did not exist in previous M18 delivery).
* **IDSS activity coach reasoner component** - initial version provided for M24 (did not exist in previous M18 delivery), but it also includes functionality of the M18 IDSS reasoner motivation messages.
* **Local data manager component** - initial version provided for M24 (did not exist in previous M18 delivery).

As a help to the developers using WP4 components following documentation has been provided:

* **Swagger interactive documentation** (described in D4.4.2 [10]) has been added on all components exposing RESTful API (Cloud Gateway, Profiling Server, Notification Manager...).
* **Javadoc** (described in D4.4.2 [10]) has been added for new functionalities as well as improved for old ones so that every class and method is documented properly.

WP4 components wiki pages describing different functionalities and offering guidelines and usage instructions: <http://serv2.radio.pub.ro/gitlab/wp4/wp4project/wikis/home>

eWALL Platform (WP4) components are illustrated below on Figure 12**Error! Reference source not found.**:



Figure 12 eWALL Platform components, from D4.4.2

## eWALL service bricks

In WP5, a refactoring process on the service bricks and the main screen has been carried out.

The new service bricks are as follows:

*DOMOTICS Service Brick*

Block diagram of this service brick is depicted on Figure 13.

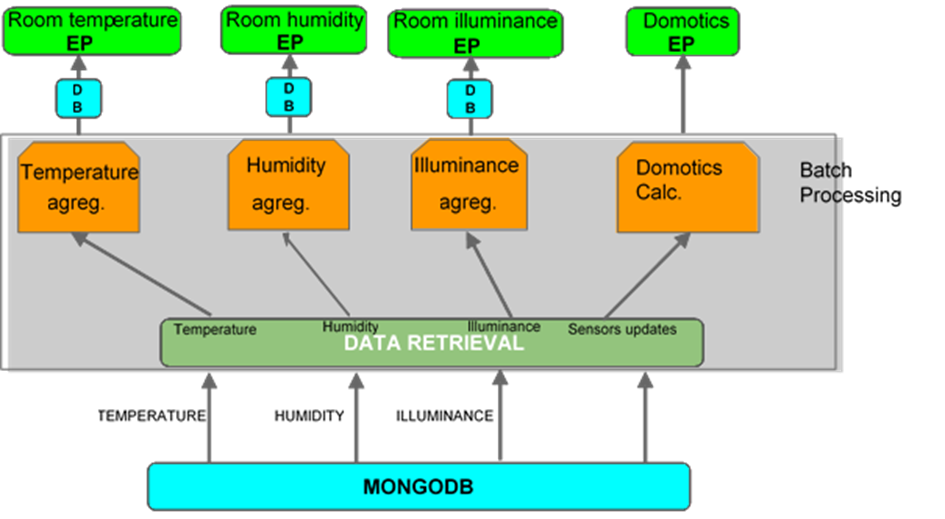


Figure 13. DOMOTICS Service Brick

The batch processing module contains all computational sub-modules that are running offline and calculate and aggregate data in a format requested by the DOMOTICS Application and this information is exposed by endpoints. The following information is provided:

* Room temperature
* Room humidity
* Room illunminance
* Domotics-current values of sensors

The Data Retrieval sub-module is used to retrieve new data. This is achieved by two approaches: polling and subscription.

The following additional new components have been developed in WP5:

* **Environment SB** – delivers information about the sensors in user environment.
* **Vitals SB**
* **Memory Questions Management**
* **Caregiver SB**

## eWALL reasoners

Lifestyle Reasoners within eWALL are components that process and store long term data that follows certain patterns or routines that define the lifestyle of the user. The aim of these components is to predict behaviour and to detect (slow or fast) changes that might indicate a change in the user’s health status. To do so, the lifestyle reasoners consume medium level data (e.g. processed data from service bricks) from multiple sources and derive semantically meaningful patterns. The data is processed, stored, and compared with medium- and long-term data stored in the cloud and the reasoner determines e.g. whether a variation falls within the expected thresholds, or employs more complex methods to determine deviation. These results can then be exposed through an API for use in other applications or other processing components. The output of a reasoner can also be an input for a combined lifestyle reasoner. The main distinction between lifestyle reasoners and the Intelligent Decision Support System (IDSS) is that the lifestyle reasoners make decisions about the short-, medium-, or long-term past, whether the IDSS reasoners reason about the now.

## Benefits to COPD and MCI target groups

Following the reviewer concerns that the services and applications will not address appropriately the specific health challenges faced by primary end-users, the corresponding actions were taken:

* Re-analysis on user requirements for COPD and MCI patients;
* Introduced new component named Functionality Decline Lifestyle Reasoner dedicated to address MCI patients’ needs for assessment on self-care, and help planning caring needs. This module implements intelligent decision on specific user functioning to assess the self-care of the user. It utilizes low-level intelligence in terms of estimated daily functioning provided by Daily Functioning Service Brick;
* Extended the research on cognitive games and performed deeper analysis for games selection. Prioritized their integration;
* Introduced new gentle reminder to MCI patients for playing cognitive games frequently enough;
* Strengthened the requirements on IDSS Sleep Anomaly classifier to address sleep quality and abnormalities which is important for MCI patients. The module implements intelligence in the form of statistical estimates and low level context extraction using machine learning;
* Introduced new component Vital Signs Lifestyle Reasoner to monitor the vital signs measurements and reasoning upon them. This component is dedicated to COPD patients since the frequently measurements and tracking is of importance for them. Using sensory data and low level intelligence it provides reasoning upon extremes in measurements or trends by utilizing statistical estimation;
* Adapted the Healthcare monitor web app to display important for COPD patients information;
* Widened the range of available training programs by introducing on-line training programs;
* Additional to these, the following benefits delivered by developed (or in development) eWALL components are available:
* Physical activities monitoring with personalized goal setting and statistics on different activities. Providing this functionality involves intelligence in goal setting and physical activity estimation;
* Daily functioning monitor with reasoning upon different activities in order to improve self-care. In the heart of this functionality are intelligent algorithms for functioning estimation and deviation detection;
* Sleep monitor with sleep quality assessment done by decision making algorithms;
* Mood estimation and tracking with mood improvement recommendations involving diverse mood features and statistical estimation;
* Socializing estimation and tracking with recommendations build on diverse socializing features and decision making;
* Personalized training programs;
* Multiple alarms on critical events;
* In-house appliances and sensors monitoring with partial control;
* Personal organizer with integrated eWALL functionality;

# Bibliography

1. The eWALL Consortium, “D3.1.2, eWALL devices and networked infrastructure,” eWALL for Active Long Living FP7 project, April 2015.
2. The eWALL Consortium, “D3.2.2., eWALL gateway,” eWALL for Active Long Living FP7 project, April 2015.
3. The eWALL Consortium, “D3.3.2., Perception from sensors (non-A/V),” eWALL for Active Long Living FP7 project, April 2015.
4. The eWALL Consortium, “D3.4.2., Audio/Visual perception,” eWALL for Active Long Living FP7 project, April 2015.
5. The eWALL Consortium, “D3.5.2., eWALL configurable metadata streams,” eWALL for Active Long Living FP7 project, April 2015.
6. The eWALL Consortium, “D4.1.2., Semantic modelling of eWALL middleware services,” eWALL for Active Long Living FP7 project, April 2015.
7. The eWALL Consortium, “D4.2.2., Intelligent support system for eWALL,” eWALL for Active Long Living FP7 project, April 2015.
8. The eWALL Consortium, “D4.3.2., Cloud middleware services for eWALL,” eWALL for Active Long Living FP7 project, April 2015.
9. The eWALL Consortium, “D3.3.2., Perception from sensors (non-A/V),” eWALL for Active Long Living FP7 project, April 2015.
10. The eWALL Consortium, “D4.4.2., eWall Platform,” eWALL for Active Long Living FP7 project, April 2015.
11. The eWALL Consortium, “D5.1.2., eWALL metadata from the outside: filtered and retrieved,” eWALL for Active Long Living FP7 project, April 2015.
12. The eWALL Consortium, “D5.2.2., Lifestyle reasoning: fusion for activities, situation and their patterns,” eWALL for Active Long Living FP7 project, April 2015.
13. The eWALL Consortium, “D5.3.2., Intelligent context-aware services & applications,” eWALL for Active Long Living FP7 project, April 2015.

1. <https://about.gitlab.com/gitlab-ce/> [↑](#footnote-ref-2)
2. <http://jenkins-ci.org/> [↑](#footnote-ref-3)
3. DEV environment is a set of virtual machines for development. *Applications-dev*: vm for service bricks and applications, *Platform-dev*: vm for central system, *Home-dev*: vm for hosting what is installed in a home environment, *Mysql-dev* and *mongodb-dev*: vms for databases. [↑](#footnote-ref-4)