UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Título del Proyecto:

Proyecto de monitorización domótica en el domicilio

Autor:

Ruz Castro, Javier

Director:

Tomás Gironés, Jesús

TESINA PARA LA OBTENCIÓN DEL TÍTULO DE:

Diploma de Especialización en Desarrollo de Aplicaciones para Android

Septiembre del 2015

Contenido

Introducción

Descripción del problema

Prodomos nace como un proyecto de ingeniería cuyo objetivo consiste en diseñar y fabricar una red de sensores inalámbrica de bajo coste para uso doméstico. Estos sensores realizan una serie de lecturas periódicas y envían los datos inalámbricamente a el coordinador de la red, que se encarga de almacenar dichas lecturas en un servidor web. La forma que tendrá el usuario final de acceder a estos datos será mediante una aplicación en Android.

Objetivos

EL objetivo de la aplicación consiste en facilitar al usuario final las medidas realizadas por la red en su domicilio a tiempo real, atendiendo a las especificaciones del producto global:

- Si bien cada usuario tendrá instalada su propia red sensorial en su domicilio, se debe facilitar la configuración de cada elemento de consulta, es decir, que cada usuario pueda añadir sus habitaciones y cada habitación se corresponda con un nodo sensor.
- Se pretende habilitar un sistema de registro de usuario y login, para que cada usuario almacene en un servidor web su configuración. Esto le permitirá recuperarla en cualquier dispostivo Android o en caso que desinstale la aplicación.
- La interfaz de usuario ha de ser sencilla e intuitiva, pero vistosa, de forma que cualquiera pueda utilizarla cómodamente sin ningún tipo de conocimiento previo.

Motivación

Este proyecto fue iniciado como trabajo de fin de grado en el grado de ingeniería en tecnologías y servicios de telecomunicaciones, tratándose de un diseño complejo en hardware y software, trabajando con sistemas electrónicos reales entre sí y viable dentro del mercado domótico actual. La aplicación móvil forma una parte fundamental dentro del proyecto, y entra en juego una vez finalizado la parte del desarrollo hardware .

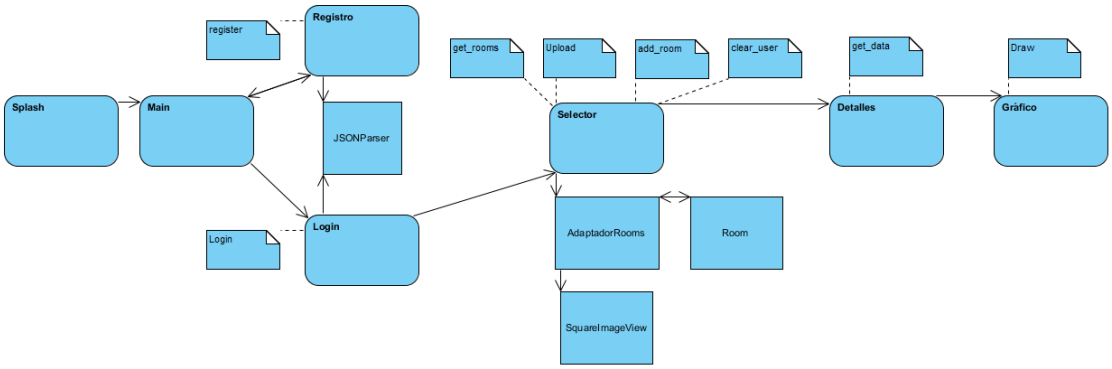
Situación de… / Tecnologías utilizadas

Para la fabricación de los prototipos de la red de sensores se han utilizado placas de microcontroladores de la marca Arduino. El diseño de las placas de circuito impreso se ha realizado utilizando Eagle. La programación del software de cada dispositivo se ha realizado mediante C++. La aplicación y sus servicios WEB se han realizado acorde a los conocimientos adquiridos a lo largo del diploma, utilizando elementos de php, JSON y mysql.

Arquitectura de la aplicación

A continuación se explica punto por punto la arquitectura de la aplicación y se justifican las principales decisiones que han sido tomadas a lo largo del proceso de diseño y verificación:

Esquema del diseño



En la figura anterior se puede observar un diagrama con todas las clases de las que se compone la aplicación, así como todos los servicios WEB a los que ha de acceder. A continuación se explica qué función desempeña cada una:

SplashActivity:

Esta clase es una simple actividad que da la bienvenida a la aplicación. Se mostrará durante tres segundos y será destruída, de modo que no se podrá volver a ella sin reiniciar la aplicación.

Main, Registro y Login:

La clase Main es un sencillo menú que permite al usuario elegir entre darse de alta con su Nick y su contraseña o si registrarse en la base de datos para poder acceder a la aplicación. Las vistas de estas tres actividades son bastante sencillas y serán utilizadas de forma intuitiva. Para darle un aspecto más visible se han definido vario “shapes” en el directorio drawable así como botones personalizados. Estos elementos serán vistos más adelante.

JSONParser

La clase JSONParser contiene un inputStream y un objeto JSON estáticos y un método genérico que será utilizado en las dos clases mencionadas anteriormente para realizar una petición http y obtener la respuesta correspondiente en forma de objeto JSON. Con el mismo método podremos gestionar el acceso a los servicios WEB tanto para el registro como para el Login.

Room

Debido a que cada nodo sensor de la red está destinado a ocupar una habitación diferente en el domicilio, cada usuario deberá configurar la aplicación añadiendo habitaciones. Para ello se ha creado la clase Room.

La clase Room contiene los campos de información básicos para ser descritas: Nombre y foto, así como lo campos asociados a la información que será mostrada al usuario una vez recuperada del servidor: Temperatura, Humedad, Luz y Fecha de captura. Además, cada objeto Room tendrá un indicativo alfanumérico (source) que se emparejará con el mismo indicativo único para cada sensor. De este modo el usuario sólo tendrá que mirar qué indicativo tiene el sensor ubicado en su salón y atribuírselo a el objeto Room "Salón" creado por él. Como campo adicional se añade un String que será utilizado para almacenar el url correspondiente a la foto de la habitación una vez ésta sea subida a internet. De este modo podrá ser recuperada cuando se cargue la información del usuario.

AdaptadorRoom

Para que todas las habitaciones definidas por el usuario puedan ser mostradas al usuario en forma de cuadrícula utilizando una vista personalizada y gestionar la interacción con la misma, es necesario definir un adaptador. La clase adaptador Room contiene los métodos necesarios para reciclar las vistas que desaparezcan de la pantalla e inflar las nuevas, obtener el id de un objeto Room dentro de un vector e incluso obtener el propio objeto Room pulsando sobre su vista.

SquareImageView

Uno de los problemas que aparecen al trabajar con imágenes en un gridview es que no todas las imágenes elegidas tienen el mismo tamaño. Algunas son más anchas, otras son más altas, y unas son mas grandes que otras. Como resultado, al adaptarse a la cuadrícula, se generan espacios vacíos o se deforma la tabla. Para evitar esto y conseguir que todas las imágenes se muestren cuadradas y por lo tanto ordenadas, la clase SquareImageView se extiende de ImageView y fuerza a la imagen a ser cuadrada. Cuando se definan los elementos de la vista personalizada para cada elemento de la cuadrícula, se utilizará SquareImageView en lugar de ImageView en el layout.

Selector

La actividad Selector es la actividad que se encarga de gestionar la configuración de habitacones de cada usuario y mostrarla en pantalla: Para ello hace uso del adaptador descrito con anterioridad y de un vector de objetos de la clase Room. Se inflará por lo tanto un GridView formado por las imágenes de cada una de las habitaciones definidas por el usuario.

Mediante una pulsación simple en cualquier elemento de la cuadrícula se lanzará la actividad Detalles a través de una sencilla transición definida utilizando el método

makeSceneTransitionAnimation de la clase ActivityOptionsCompat. A esta actividad se le mandarán los parámetros correspondientes al objeto Room seleccionado.

Mediante una pulsación larga se mostrará un menú que permitirá al usuario editar o borrar el objeto seleccionado, así como añadir uno nuevo. A la hora de crear habitaciones, el usuario deberá introducir el nombre, el código src correspondiente al indicador único de cada nodo sensor, y definir una imagen. Para ello podrá tomar una foto directamente con el dispositivo o bien buscar en la galería. Estas fotos son almacenadas en un servidor web una vez definidas, y se almacena en el campo correspondiente del objeto room en cuestión un url para poder recuperarlas posteriormente.

Una vez definidas todas las habitaciones, el usuario puede guardar la configuración mediante un botón ubicado en el toolbar de la actividad. Al guardar la configuración se accede a un servicio web al cual se le envían los parámetros nombre, src y url de cada objeto Room dentro del vector, los cuales serán almacenados en una tabla de la base de datos. Cada vez que se guarda la configuración se eliminan de la base de datos los elementos almacenados previamente por el usuario.

De forma automática, cada vez que un usuario inicie sesión, se utilizará un servicio web para acceder a la tabla de la base de datos y cargar la configuración almacenada por el usuario. Esta configuración se recupera utilizando un objeto JTAG. Si el objeto JTAG está vacío significa que el usuario es nuevo o nunca ha guardado su configuración, por lo que se mostraría la cuadricula vacía. En caso contrario, el vector de habitaciones se llena accediendo a los datos de este objeto. Entonces se procede a la descarga de cada una de las imágenes utilizando los url recuperados.

Para evitar poblemas en la aplicación al trabajar con imágenes, se han definido varias funciones para reajustar el tamaño del mapa de bits una vez decodificado. Se define esta función tanto para decodificar un archivo de la memoria del teléfono a través de su uri correspondiente, como para decodificar un stream obtenido de internet. Esto permitirá cargar los mapas de bits de forma eficiente, ahorrando memoria en el dispositivo y evitando errores por tamaños de imágenes excesivos.

Todos los procesos que requieran un acceso a los recursos de la red se gestionarán utilizando hilos de ejecución asíncronos extendiendo la clase AsyncTask. Algunas de estas tareas mantendrán el interfaz de usuario ocupado mediante un cuadro de diálogo, mientras que otros permitirán al usuario seguir utilizando la aplicación, como por ejemplo las descargas de imágenes.

Detalles

La actividad Detalles obtiene el identificador corresondiente de la habitación seleccionada y un stream codificado de la imagen asignada, la cual será decodificada y mostrada como fondo. Entonces la actividad se comunica con el servicio web y obtiene la última lectura de los datos

realizada por el nodo sensor que se corresponde con el identificador obtenido. Pulsando sobre el fondo se accede a la actividad Gráfico.

Gráfico

La acticidad Gráfico muestra en un webView una gráfica detallada de los datos guardados por el nodo correspondiente durante las últimas lecturas. La gráfica se forma utilizando highcharts, un recurso de dominio público, utilizando los datos obtenidos de la tabla correspondiente por el servicio web.

Modelo de datos

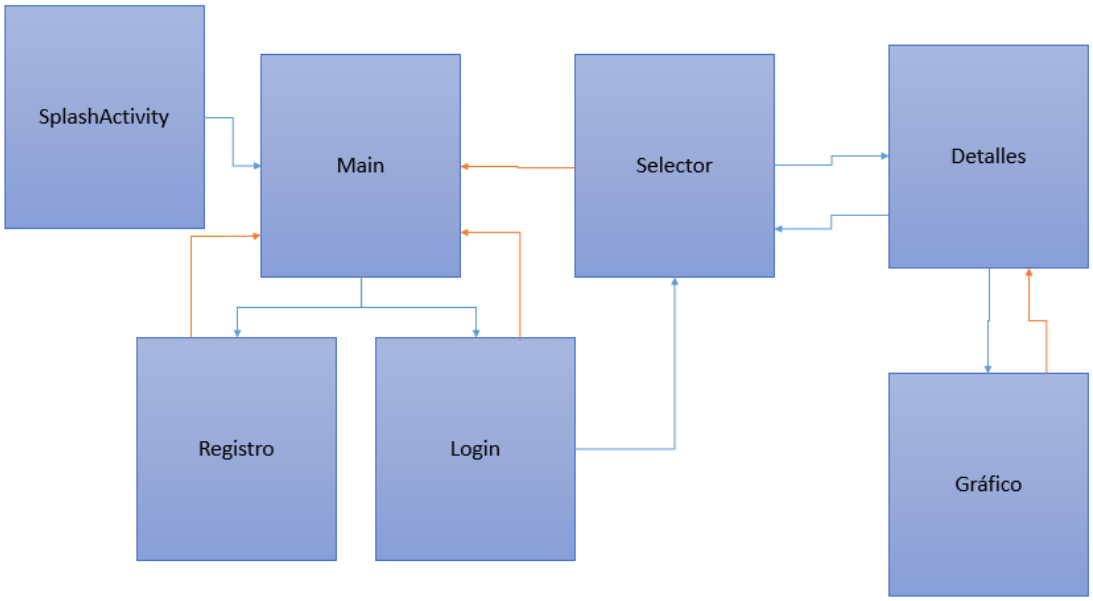
Para hacer funcionar correctamente la aplicación, es necesario la obtención de un dominio en internet en el cual se han creado varios scripts y una base de datos compuesta por tres tablas: Una para almacenar los datos obtenidos por la red de sensores, otra para almacenar los usuarios que se registran en la aplicación, y otra para almacenar las habitaciones de cada uno de los usuarios.

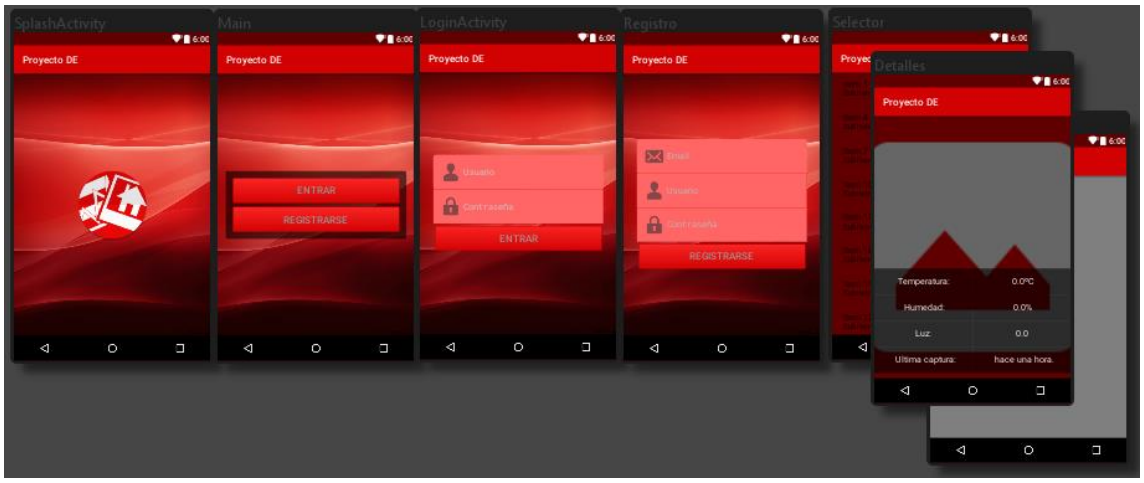
Los script utilizados son los siguientes:

Conexión.php	**Gestiona la conexión con la base de datos, pues contiene todos los datos necesarios para realizar una conexión con la misma: usuario, contraseña y dominio. Los demás scipts utilizados en el presente proyecto incluyen este archivo.**
Add_data.php	Obtiene los datos enviados por el coordinador de la red mediante una petición http get y los guarda en la tabla Lecturas de la base de datos.
Add_rooms.php	Obtiene los datos de un objeto Room desde la aplicación a través de una petición http get y los guarda en la tabla Configuracion de la base de datos.
Clear_user.php	Recibe el valor de un usuario en concreto desde la aplicación mediante una petición http get y borra de la tabla Configuración de la base de datos todos los valores guardados correspondientes a dicho usuario.
Get_data.php	Obtiene un valor src específico desde la aplicación mediante una petición http get y devuelve el último valor almacenado en la tabla lecturas que se corresponda con dicho src.
Get_rooms.php	Recibe el valor de un usuario en concreto desde la aplicación mediante una petición http get y devuelve un objeto JSON con un vector de todos los valores almacenados en la tabla lecturas correspondientes al usuario en cuestión.
Register.php	Recibe los valores correspondientes al registro de un nuevo usuario mediante una petición http post. Tras comprobar que el usuario no existe en la tabla Registros de la base de datos, coloca al usuario dentro de la misma y devuelve un objeto JSON con el código de la respuesta.
Login.php	Recibe los valores de usuario y contraseña (sin codificar) desde la aplicación mediante una petición http post, comprueba que ambos valores son correctos en la tabla Registros de la base de datos y

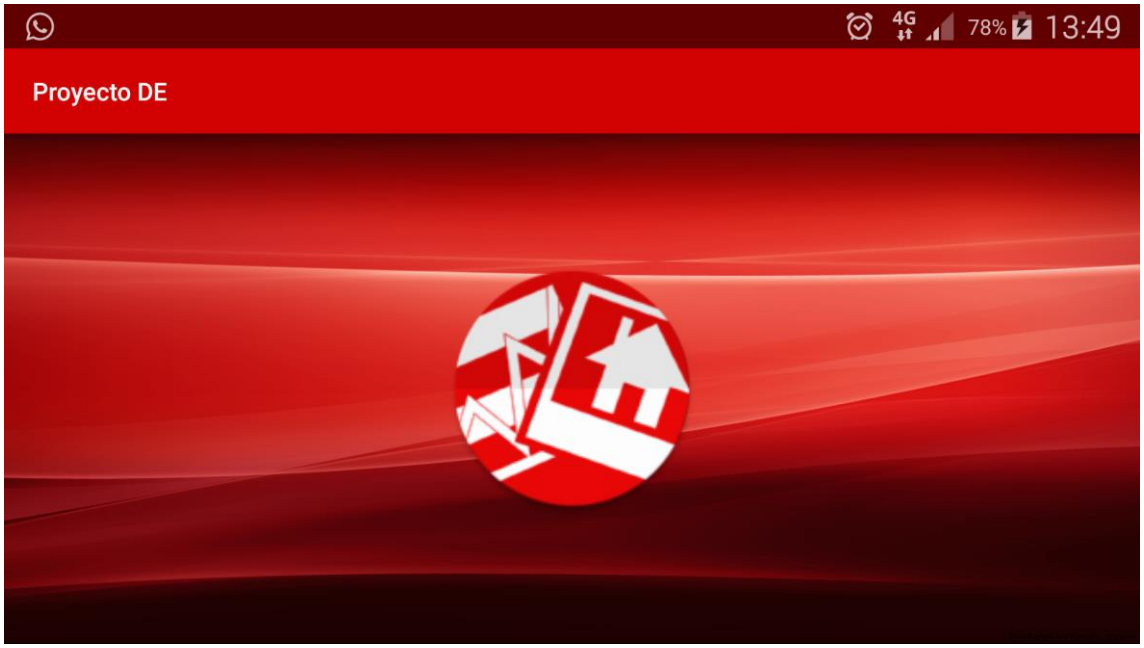
	devuelve un objeto JSON con el código de la respuesta.
Upload.php	Recibe una imagen codificada y la guarda en el servidor web.
Draw.php	Recibe el valor src correspondiente desde le aplicación mediante una petición http get, obtiene los últimos 2000 valores almacenados en la tabla lecturas de la base de datos y formatea una gráfica personalizada mediante unas funciones facilitadas por highcharts.com

Vistas

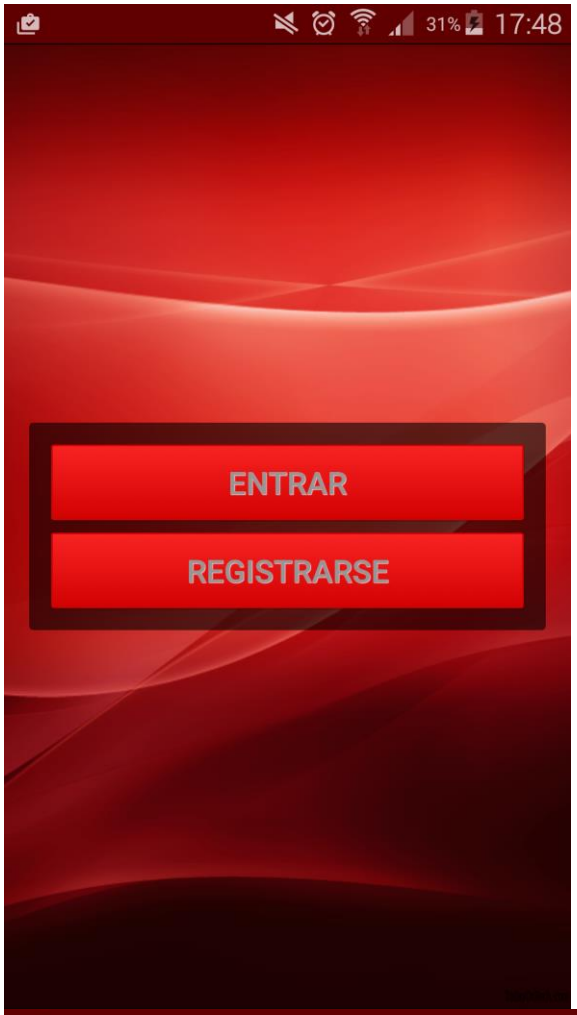


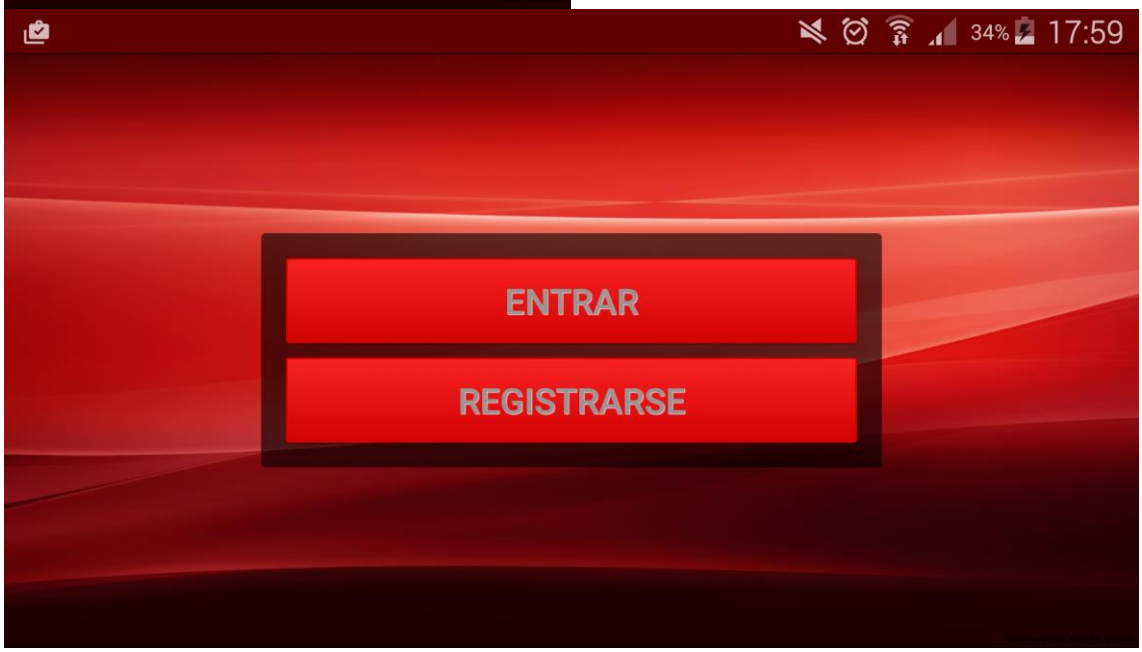


Splash

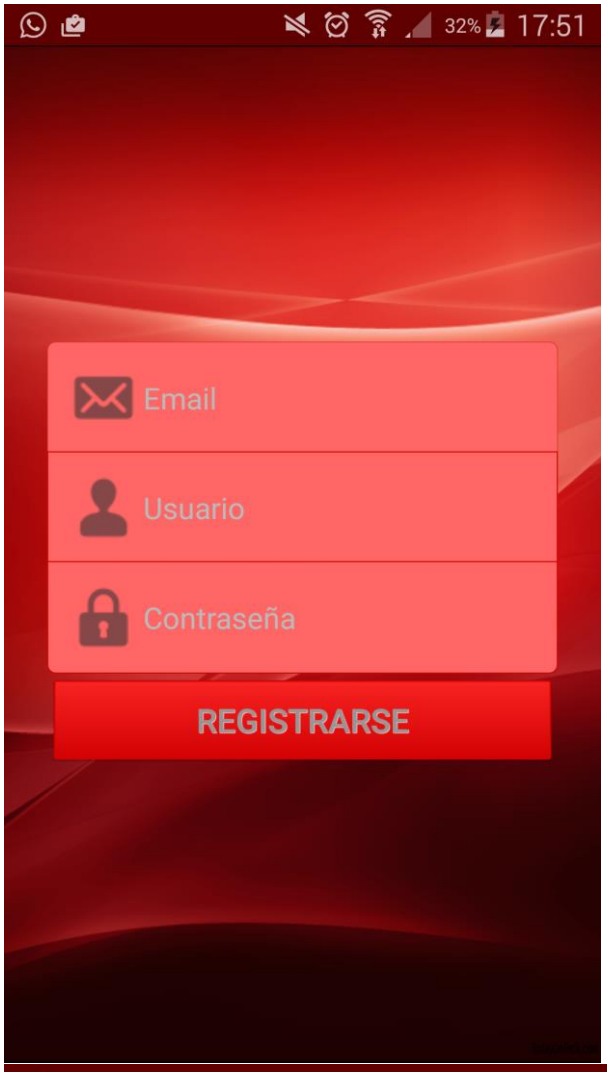


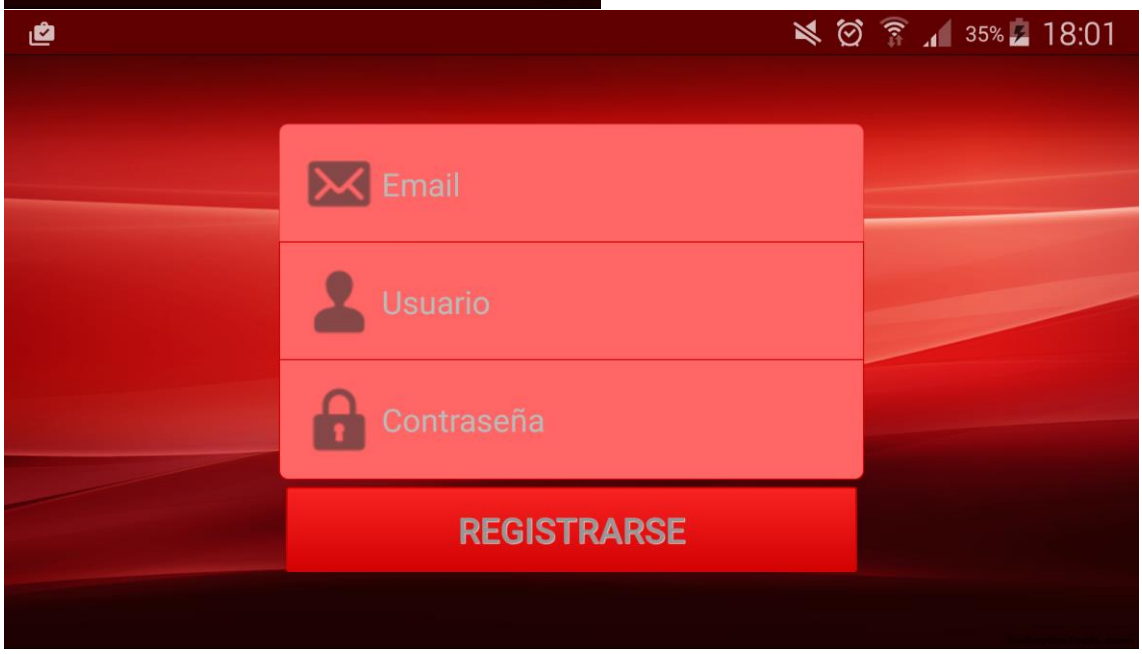
Main



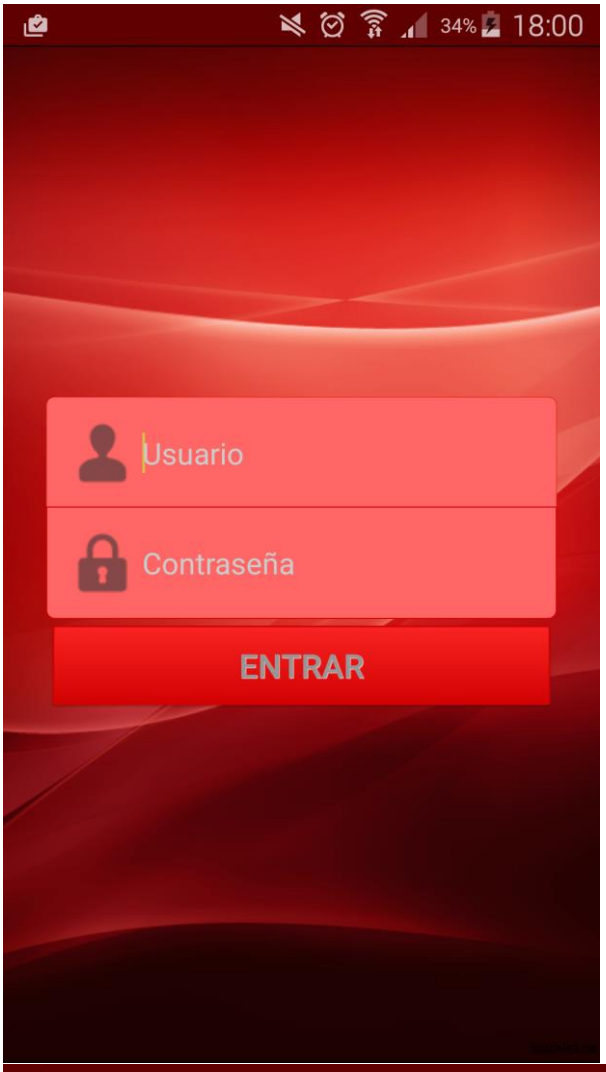


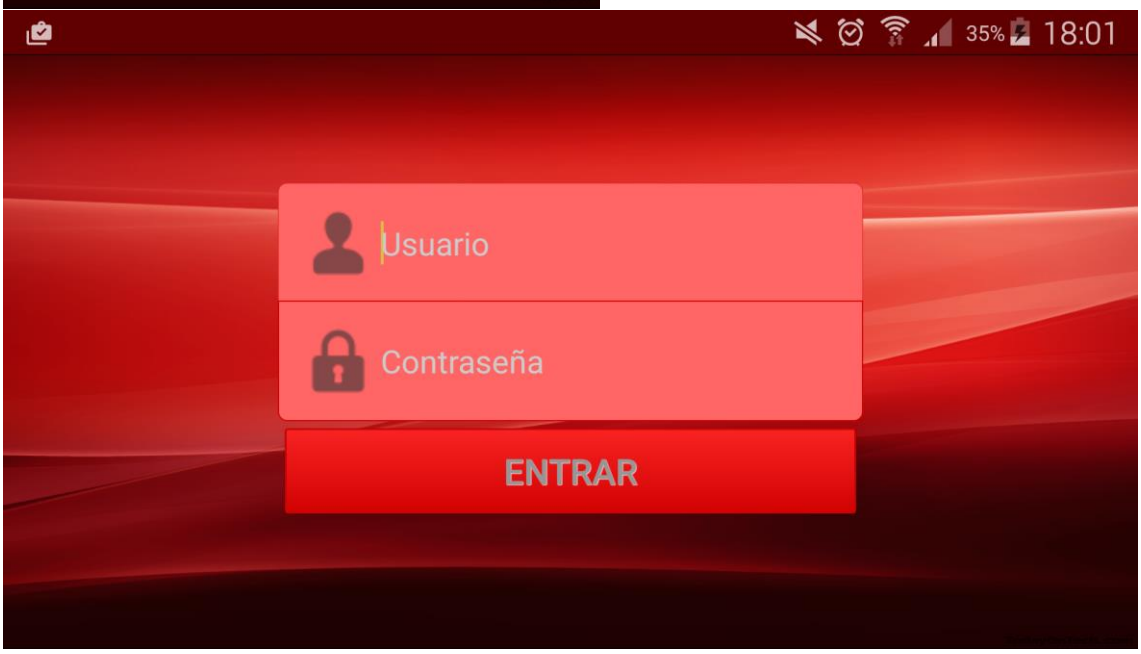
Registro


32%
17:51
Email
Usuario
Contraseña
REGISTRARSE

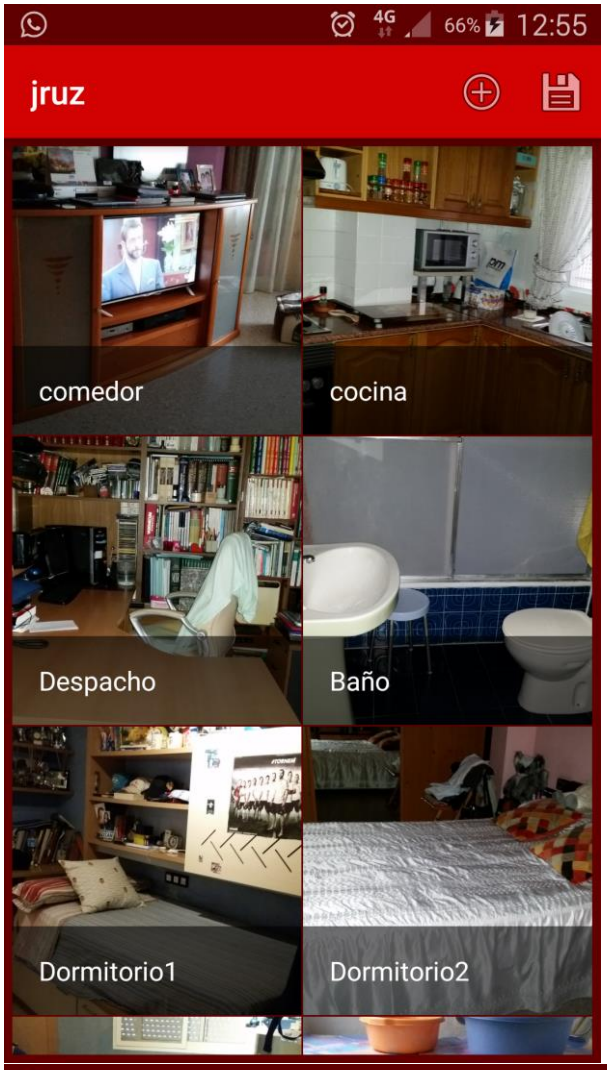

35%
18:01
Email
Usuario
Contraseña
REGISTRARSE

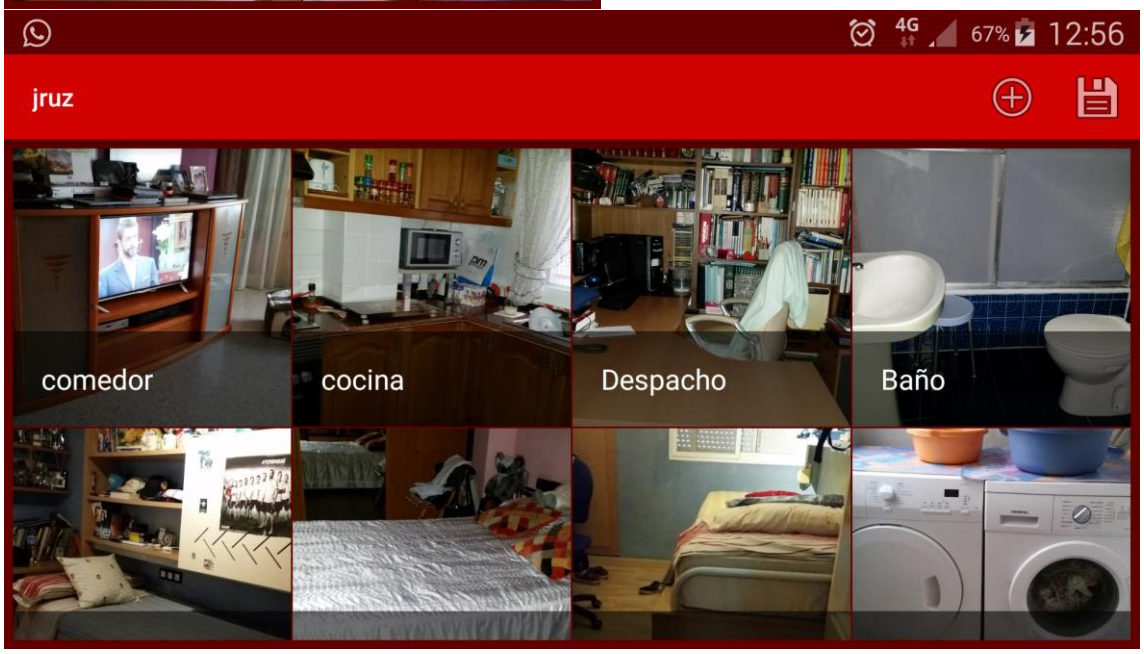
Login


34%
18:00
Usuario
Contraseña
ENTRAR


35%
18:01
Usuario
Contraseña
ENTRAR

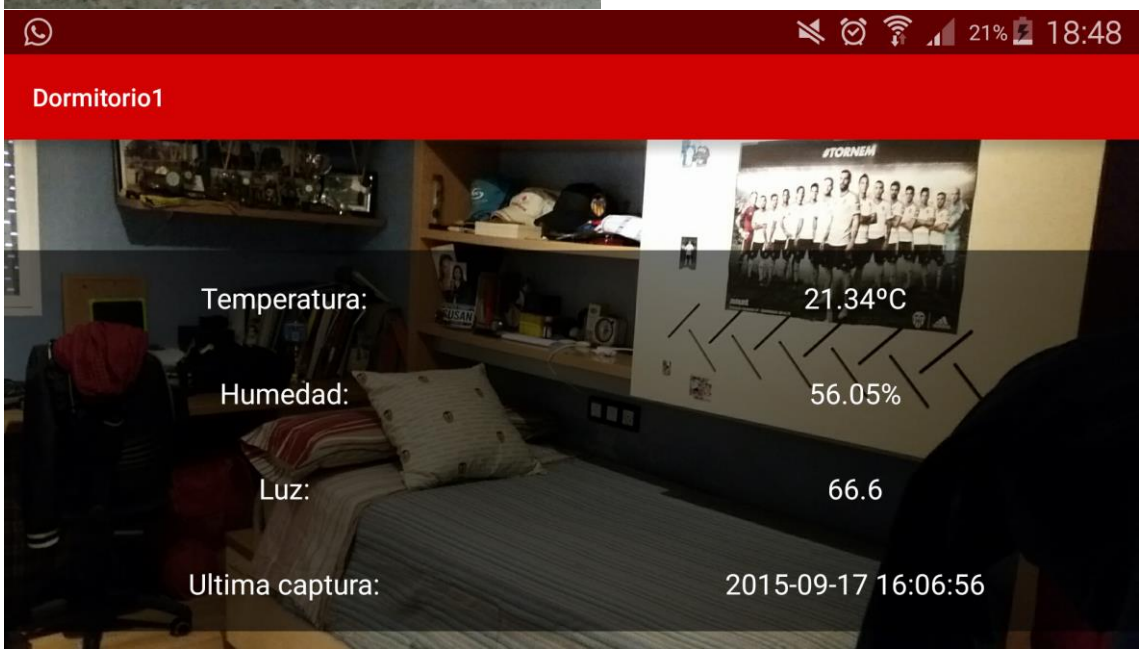
Selector


66% 12:55
jruz
comedor
cocina
Despacho
Baño
Dormitorio1
Dormitorio2

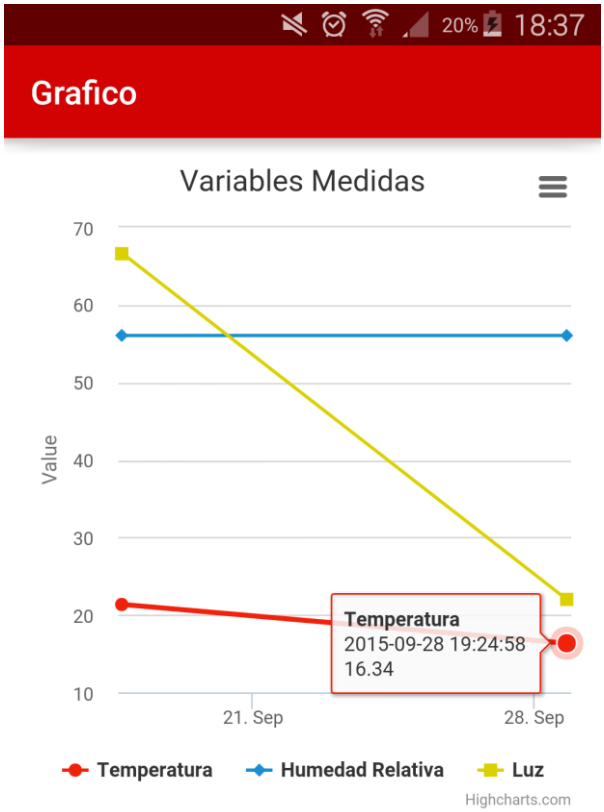

67% 12:56
jruz
comedor
cocina
Despacho
Baño

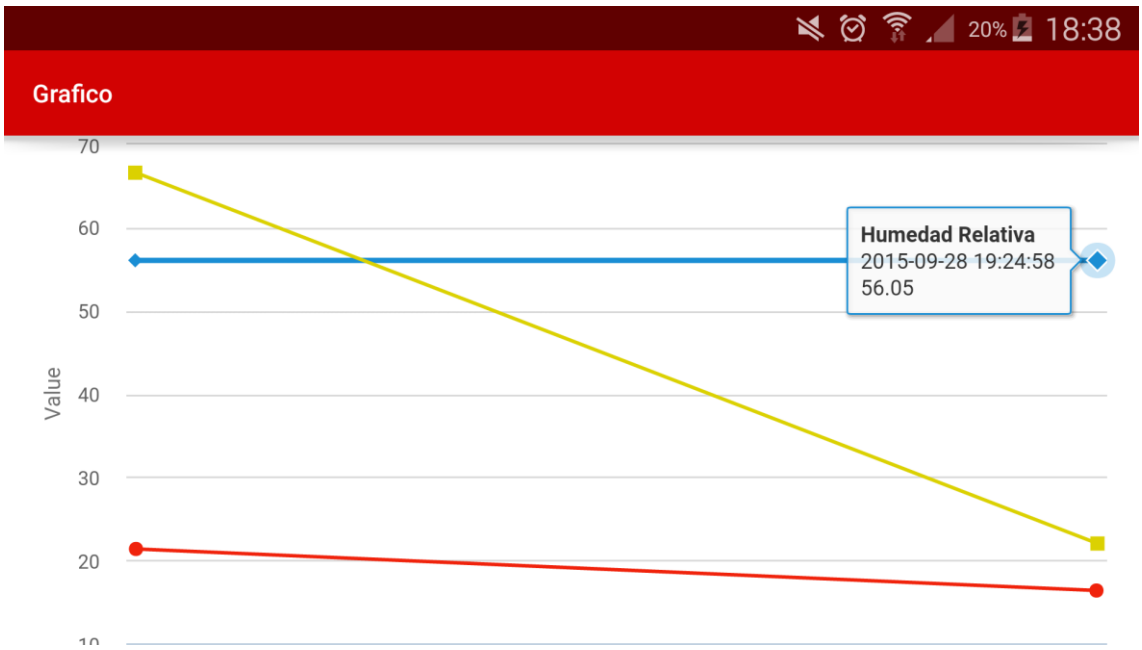
Detalles


20% 18:36
comedor
Temperatura: 16.34ºC
Humedad: 56.05%
Luz: 22
Ultima captura: 2015-09-28 17:24:58


21% 18:48
Dormitorio1
Temperatura: 21.34ºC
Humedad: 56.05%
Luz: 66.6
Ultima captura: 2015-09-17 16:06:56

Grafico


20% 18:37
Grafico
Variables Medidas
70
60
50
40
30
20
10
Value
Temperatura
2015-09-28 19:24:58
16.34
21. Sep
28. Sep
Temperatura
Humedad Relativa
Luz
Highcharts.com


20% 18:38
Grafico
70
60
50
40
30
20
Value
Humedad Relativa
2015-09-28 19:24:58
56.05

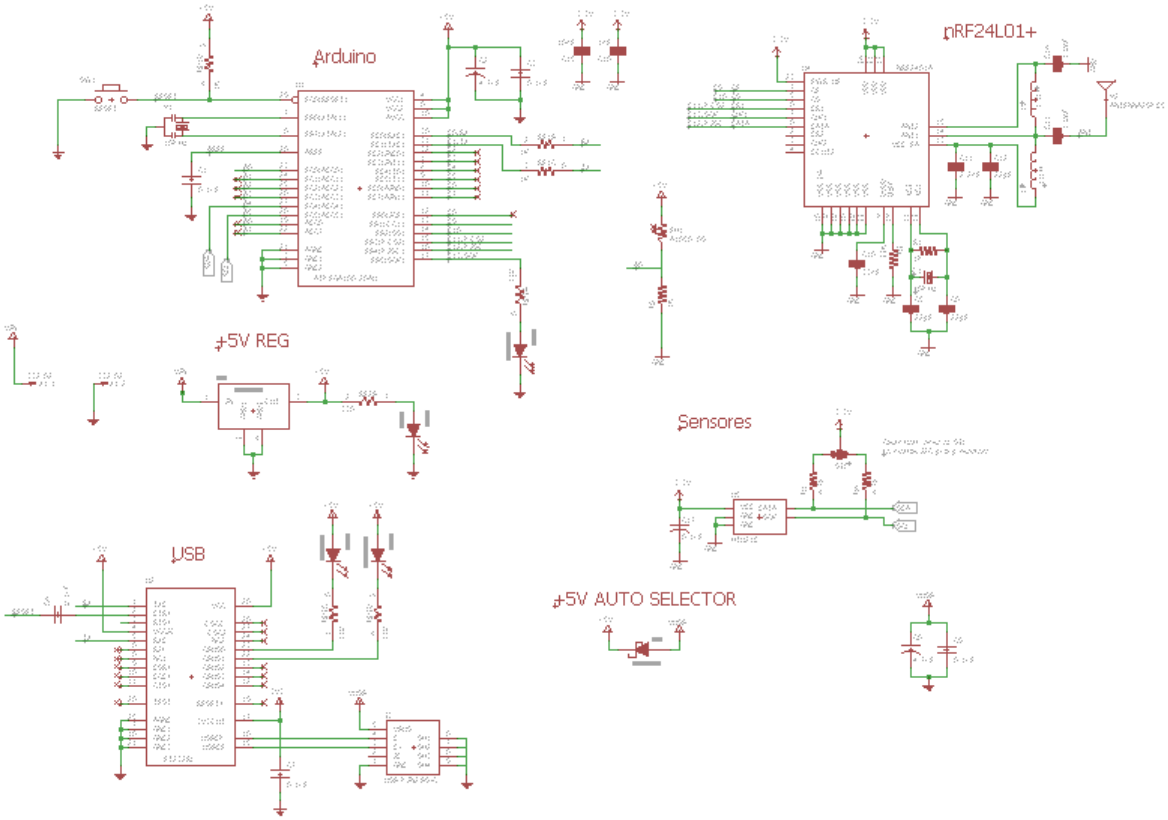
Conclusiones

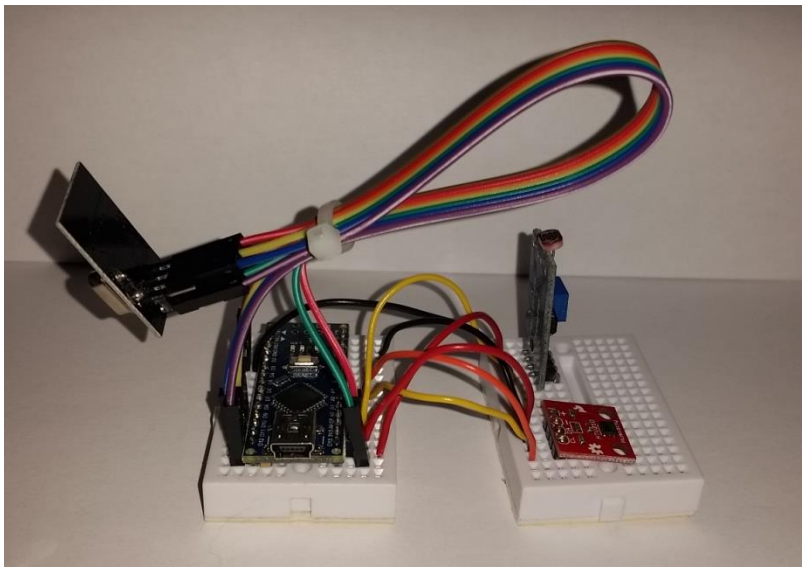
Si bien la aplicación cumple todos los objetivos planteados desde el inicio, aún no ha superado un proceso de verificación lo suficientemente robusto como para poder considerarse terminada. Deben tomarse medidas para evitar la inclusión de espacios en banco en los EditText, ya que estos no son aceptados por la petición http, o bien implementar un método para codificarlos. El tamaño de los mapas de bits se ha fijado en 1024 x 1024 para evitar errores. Se plantea reducir este tamaño o hacerlo depender de las características de cada dispositivo. Se han utilizado dos dispositivos para testear la aplicación y no se han producido OOM. Por último cabría mejorar la estética de la aplicación y añadir transiciones entre las diferentes vistas.

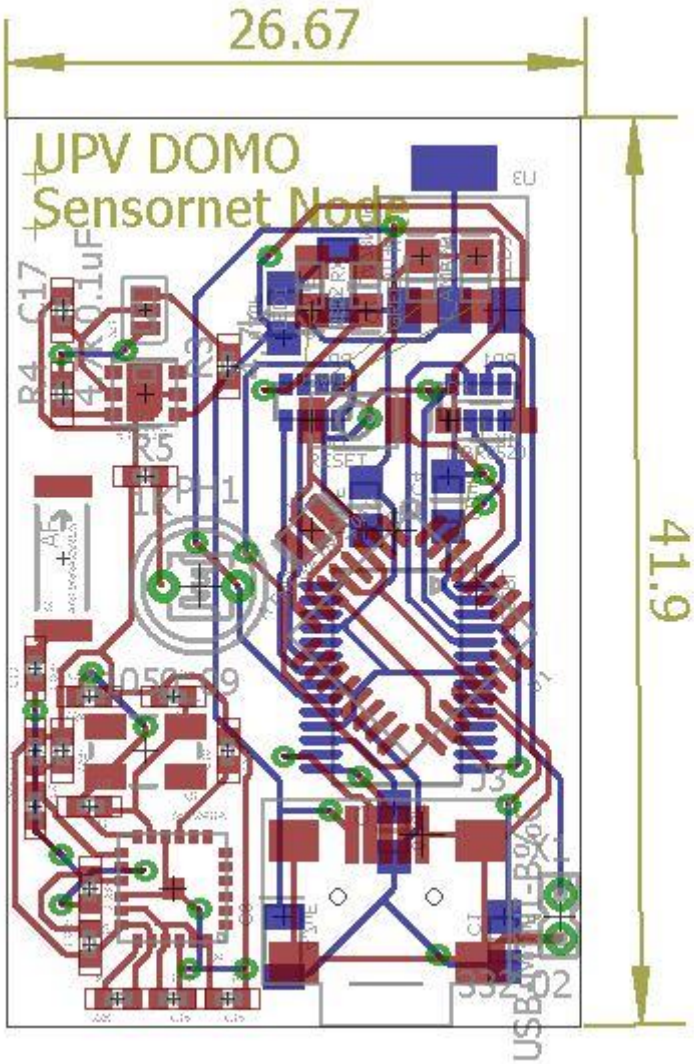
Anexos

Planos y esquemas técnicos

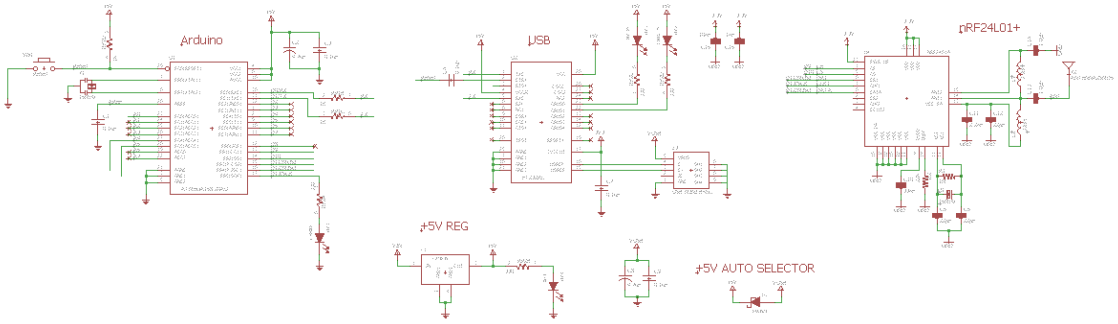
Nodos de la red

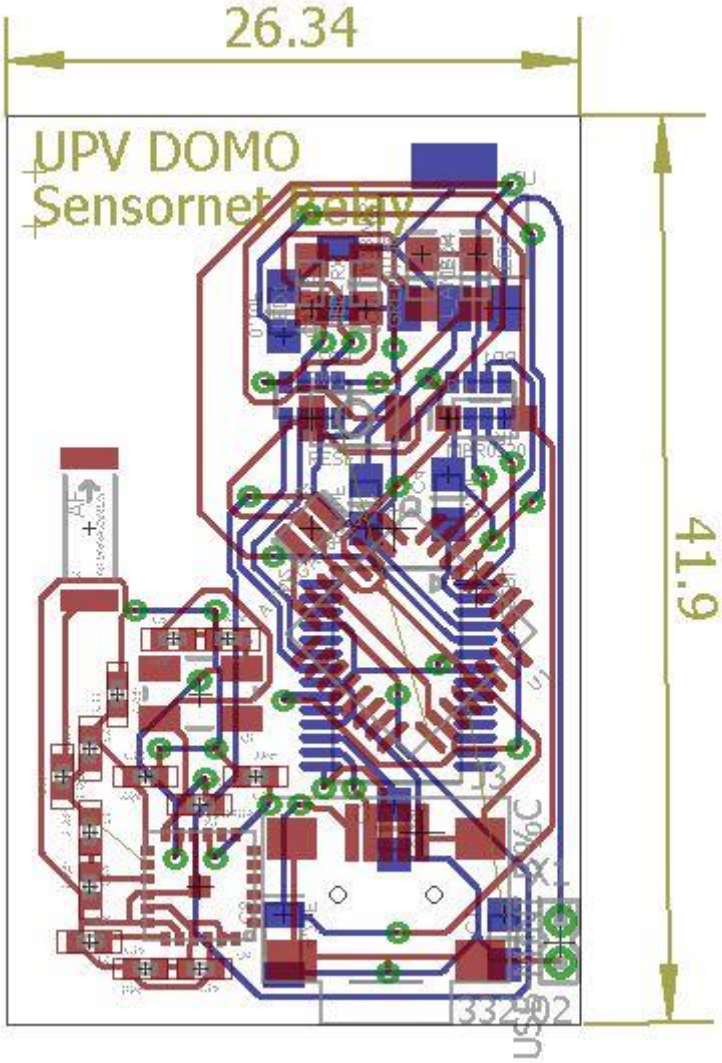




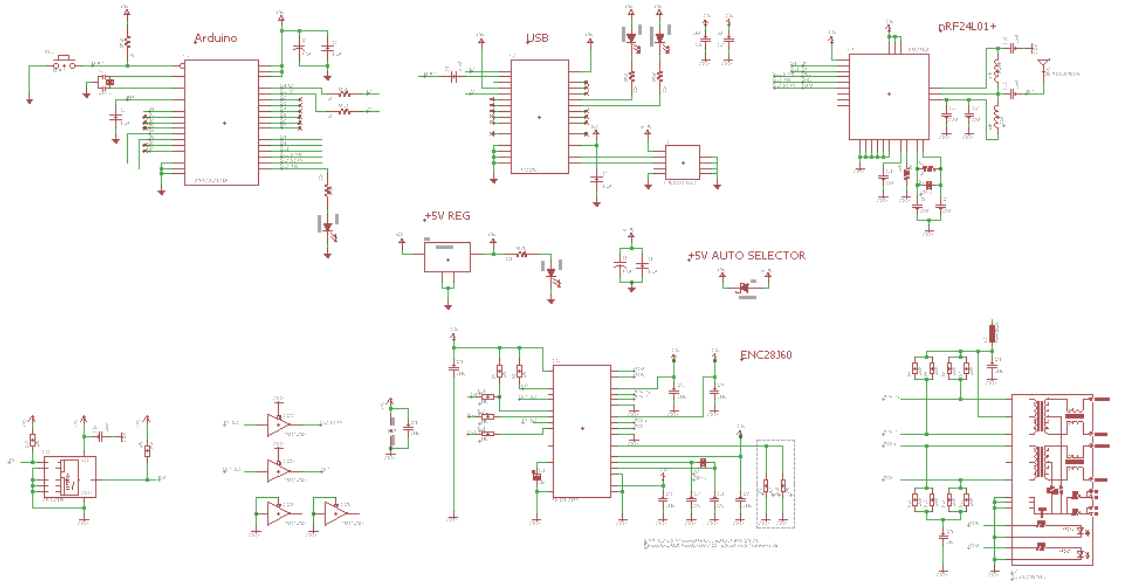

26.67
41.9
UPV DOMO
Sensornet Node

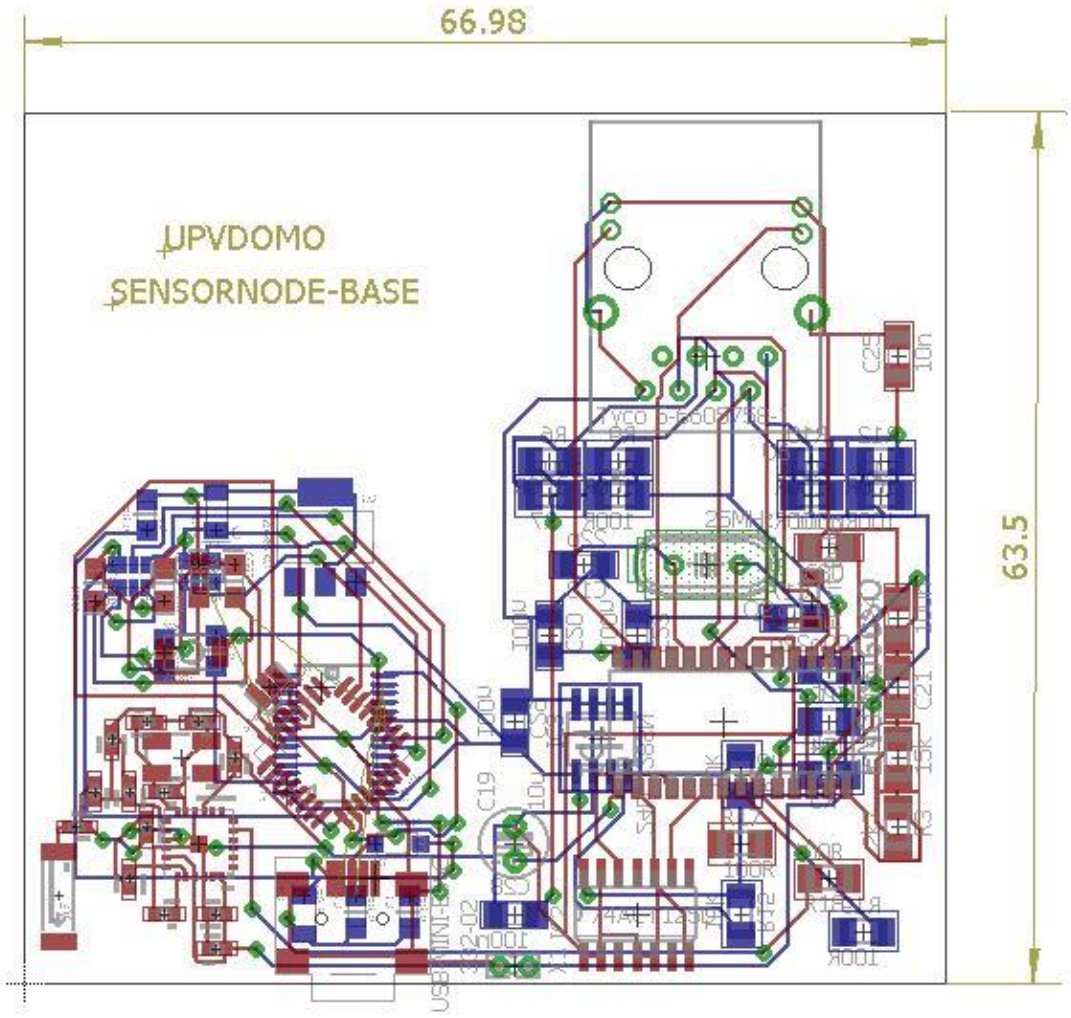
Repetidores




26.34
41.9
UPV DOMO
Sensornet Relay

Coordinador de la red





Presupuesto

Placa Arduino NANO rev3	**1,94€**
Módulo nRF24L01+	0,70€
Módulo sensor de Luz 6495	2,68€
Módulo sensor HTU21D	4,38€
Alimentación	1,82€
TOTAL*	11,52€

Presupuesto de prototipado por nodo

Placa Arduino NANO rev3	**1,94€**
Módulo nRF24L01+ con Antena	3,39€
Alimentación	1,75€
TOTAL*	7,08€

Presupuesto de prototipado por Repetidor

Placa Arduino MEGA 2560	**7,15€**
Módulo nRF24L01+	0,70€
Módulo ENC28J60	2,59€
Alimentación	1,75€
TOTAL*	12,19€

Presupuesto de prototipado del coordinador

**Estos precios se basan en ofertas encontradas en Aliexpres y similares, en España resulta imposible conseguir estos productos a precios tan reducidos.*

Cantidad	Descripción	Encapsulado	Precio/ud
1	ATMega328p	QFP-32	2,29 €
1	FT23RL	SSOP-28	1,92 €
1	LF33CDT-TR	DPAK-3	0,736 €
1	nRF24L01A	QFN-20	3,39 €
1	Antena SMD	ANTENNA-CHIP5	1,67 €
1	HTU21D	DFN-6	2,45 €
1	Resistencia 1M	C0402	0,0065 €
1	Resistencia 22K	C0402	0,0065 €
1	Resistencia 330	C0402	0,0065 €
2	Resistencia 1K	C0402	0,0065 €
2	Resistencia 4,7K	C0402	0,0065 €
1	Resistencias 1K	RES4NT	0,0065 €
1	Resistencias 330	RES4NT	0,0065 €
5	Condensador 0.1uF	C0402	0,0065 €
2	Condensador 4.7uF	C0402	0,0065 €
3	Condensador 22pF	C0402	0,0065 €
1	Condensador 33nF	C0402	0,0065 €
1	Condensador 2.2nF	C0402	0,0065 €
2	Condensador 1.0pF	C0402	0,0065 €
1	Condensador 10nF	C0402	0,0065 €
1	Condensador 1nF	C0402	0,0065 €
1	Inductor 3.3nH	C0402	0,0065 €
1	Inductor 33nH	C0402	0,0065 €
1	Diodo MBR0520	SOD-123	0,0396 €
1	LED Rojo	LED0805	0,131 €
1	LED verde	LED0805	0,131 €
1	LED Ambar	LED0805	0,131 €
1	LED Azul	LED0805	0,131 €
1	Mini USB Hembra B	USB-MINI-B_2	0,738 €
1	Fotocélula A1050_09	T0-46-A10XX	0,339 €
1	Cristal 16MHz	CRYSTAL-SMD-5X3.2	0,965 €
1	Resonador 16Mhz	PBRC-H	1,86 €
1	Botón PB157	157SW	0,0396 €
1	Conector hembra 2p	332-02	0,0035 €
TOTAL*			19,148€

Balance de Materiales por Nodo

Cantidad	Descripción	Encapsulado	Precio/ud
1	ATMega328p	QFP-32	2,29 €
1	FT23RL	SSOP28	1,92 €
1	LF33CDT-TR	DPAK-3	0,736 €
1	nRF24L01A	QFN-20	3,39 €
1	Antena SMD	ANTENNA-CHIP5	1,67 €
1	Resistencia 1M	C0402	0,0065 €
1	Resistencia 22K	C0402	0,0065 €
1	Resistencia 330	C0402	0,0065 €
1	Resistencias 1K	RES4NT	0,0065 €
1	Resistencias 330	RES4NT	0,0065 €

5	Condensador 0.1uF	C0402	0,0065 €
2	Condensador 4.7uF	C0402	0,0065 €
3	Condensador 22pF	C0402	0,0065 €
1	Condensador 33nF	C0402	0,0065 €
1	Condensador 2.2nF	C0402	0,0065 €
2	Condensador 1.0pF	C0402	0,0065 €
1	Condensador 10nF	C0402	0,0065 €
1	Condensador 1nF	C0402	0,0065 €
1	Inductor 3.3nH	C0402	0,0065 €
1	Inductor 33nH	C0402	0,0065 €
1	Diodo MBR0520	SOD-123	0,0396 €
1	LED Rojo	LED0805	0,131 €
1	LED verde	LED0805	0,131 €
1	LED Ambar	LED0805	0,131 €
1	LED Azul	LED0805	0,131 €
1	Mini USB Hembra B	USB-MINI-B_2	0,738 €
1	Cristal 16MHz	CRYSTAL-SMD-5X3.2	0,965 €
1	Resonador 16Mhz	PBRC-H	1,86 €
1	Botón PB157	157SW	0,0396 €
1	Conector hembra 2p	332-02	0,0035 €
TOTAL*			16,398€

Balance de materiales por Repetidor

Cantidad	Descripción	Encapsulado	Precio/ud
1	ATMega328p	QFP-32	2,29 €
1	FT23RL	SSOP-28	1,92 €
1	LF33CDT-TR	DPAK-3	0,736 €
1	ENC28J60	SSOP-28	2,18 €
1	MC74ACT125D	SOIC-14	0,161 €
1	EEPROM 24LC128SN	SOIC-08	0,471 €
1	nRF24L01A	QFN-20	3,39 €
1	Antena SMD	ANTENNA-CHIP5	1,67 €
1	Resistencia 1M	C0402	0,0065 €
1	Resistencia 22K	C0402	0,0065 €
1	Resistencia 330	C0402	0,0065 €
1	Resistencia 2K7	C0402	0,0065 €
4	Resistencia 10K	C0402	0,0065 €
11	Resistencia 100	C0402	0,0065 €
2	Resistencia 15K	C0402	0,0065 €
1	Resistencias 1K	RES4NT	0,0065 €
1	Resistencias 330	RES4NT	0,0065 €
13	Condensador 0.1uF	C0402	0,0065 €
2	Condensador 4.7uF	C0402	0,0065 €
1	Condensador 10uF	C0402	0,0065 €
5	Condensador 22pF	C0402	0,0065 €
1	Condensador 33nF	C0402	0,0065 €
1	Condensador 2.2nF	C0402	0,0065 €
2	Condensador 1.0pF	C0402	0,0065 €
2	Condensador 10nF	C0402	0,0065 €

1	Condensador 1nF	C0402	0,0065 €
1	Inductor 3.3nH	C0402	0,0065 €
1	Inductor 33nH	C0402	0,0065 €
1	Ferrita	SM1206	0,0065 €
2	Diodo MBR0520	SOD-123	0,0396 €
1	LED Rojo	LED0805	0,131 €
1	LED verde	LED0805	0,131 €
1	LED Ambar	LED0805	0,131 €
1	LED Azul	LED0805	0,131 €
1	Mini USB Hembra B	USB-MINI-B_2	0,738 €
1	Cristal 16MHz	CRYSTAL-SMD-5X3.2	0,965 €
1	Cristal 25MHz	HC49/S	0,516 €
1	Resonador 16Mhz	PBRC-H	1,86 €
1	Botón PB157	157SW	0,0396 €
1	Conector hembra 2p	332-02	0,0035 €
TOTAL*			20,623€

Balance de materiales de la Base

**La especificación de precios se asume para un pedido de entre 100 y 499 unidades en Farnell. Para pedidos de menor o mayor cantidad el precio varía. El precio entre componentes pasivos también puede variar en función de la tolerancia escogida. Se recomienda que no sobrepase el 10%.*

Manual de usuario

La aplicación ha sido diseñada con el fin de ser intuitiva y fácil de utilizar para usuarios de cualquier perfil. En la primera versión de la misma, será necesario tomar en cuenta las siguientes consideraciones:

- Evitar utilizar espacios en blanco en cualquiera de los EditText.
- Utilizar el valor “aaaa” para el campo src de cada habitación.
- Se recomienda estar conectado a internet o disponer de una tarifa de datos para evitar tiempos de carga prolongados.