Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

Desarrollo de interfaces de usuario para el proyecto "ecoMobility"

Trabajo Fin de Grado

Grado en Ingeniería Informática

Autor: Miguel Ángel Torrentí Pérez

Tutor: Joan Fons i Cors

2018-2019

Resumen

El proyecto ecoMobility intenta evaluar el interés y viabilidad del uso de la Inteligencia Ambiental en el entorno de ciudades inteligentes de modo que permita utilizar técnicas de computación en la aplicación de políticas orientadas a la reducción de la contaminación ambiental de la ciudad. El objetivo del presente estudio, como colaboración al proyecto ecoMobility, es el desarrollo de una interfaz web con la cual se informa al ciudadano de las zonas de la ciudad con circulación restringida o limitada y de posibles rutas alternativas. Este estudio pretende, no solo mostrar esta información, sino ampliarla y modificarla de modo que resulte más funcional y, en definitiva, más eficiente. Para ello hemos creado una interfaz web que, compuesta por una serie de elementos gráficos, facilite la información al usuario y, en definitiva, se integre con otros trabajos desarrollados en el proyecto ecoMobility.

Palabras clave: IoT, ciudades inteligentes, interfaz web, polución, tráfico

Resum

El projecte ecoMobility intenta avaluar l'interès i viabilitat de l'ús de la intel·ligència ambiental arran de les ciutats intel•ligents de manera que permeta emprar tècniques de computació en l'aplicació de polítiques orientades a la reducció de la contaminació ambiental de la ciutat. L'objecte del present estudi, com a col•laboració al projecte ecoMobility, és el desenvolupament d'una interfície web amb la quan s'informa al ciutadà de les zones de la ciutat amb circulació restringida o limitada i de possibles rutes alternatives. Este estudi pretén, no només mostrar esta informació, sinó ampliar-la i modificar-la de manera que resulte més funcional i, en definitiva, més eficient. Per això hem creat una interfície web que, composta per una sèrie d'elements gràfics, facilite la informació a l'usuari i, en definitiva, s'integre amb altres treballs desenvolupats en el projecte ecoMobility.

Paraules clau: IoT, ciutats intel•ligents, interfície web, pol•lució, trànsit

Abstract

The ecoMobility project intends to evaluate the interest and viability of the use of Environmental Awareness around intelligent cities so that it may allow the use of computational techniques in the application of policies oriented towards the reduction of the cities' environmental contamination. The aim of the present study, as a collaboration to the ecoMobility project, is the development of an interface web through which citizens are informed of city zones with restricted or limited circulation and of possible alternative routes. This study intends, not only to show this information, but also amplify and modify it in such a way that it results more functional and, thus, more efficient. For this purpose we have created an interface web that, made up of a series of graphic elements, facilitates information for the user and, in the end, becomes part of other studies developed in the ecoMobility project.

Keywords: IoT, intelligent cities, web interface, pollution, traffic

Tabla de contenidos

Tabla de ilustraciones

1. Introducción

En este TFG vamos a tratar sobre nuestra aportación al proyecto ecoMobility. Este proyecto tiene como principal funcionalidad la reducción de contaminación en ciudades. ecoMobility reconoce las zonas donde más polución hay y redirige a los usuarios por rutas alternativas obteniendo como resultado una disminución de la polución en los puntos más problemáticos de cada ciudad. Dentro del proyecto de ecoMobility nuestro cometido será el desarrollo de una interfaz de usuario. Dicha interfaz utiliza los datos recopilados por sensores situados por toda la ciudad para informar al usuario de las zonas con circulación restringida y de posibles rutas alternativas.

1.1 Motivación

En la actualidad estamos empezando a conocer los efectos del paso del ser humano por el planeta tierra. Uno de los más visibles es la polución en las grandes ciudades. Dicha polución es, en gran medida, provocada por la saturación de vehículos concentrados en determinados puntos de las ciudades. Gracias a las habilidades adquiridas en el Grado de Ingeniería Informática se posibilita la creación de un entorno mediante el cual dar a conocer la situación actual y, a la vez, ofrecer una solución al problema.

Con el paso del tiempo la tecnología ha ido evolucionando a una velocidad increíble. En la actualidad las ciudades poseen sensores que recopilan todo tipo de información. Utilizando esta información y con la tecnología que tenemos a nuestro alcance, el futuro pasa por evolucionar en ciudades inteligentes y, con nuestro proyecto ecoMobility, queremos estar un paso más cerca, afectando a diferentes aspectos de la vida de todo ser humano.

1.2 Objetivos

Como hemos dicho anteriormente, nuestro objetivo es contribuir a una ciudad inteligente mediante la cual mejorar la calidad de vida de los ciudadanos. Con el proyecto de ecoMobility queremos que cualquier ciudadano pueda tener acceso a los siguientes datos: (1) nivel de polución de las diferentes zonas de su ciudad, (2) qué zonas es recomendable evitar y (3) qué zonas están cerradas al tráfico debido a un exceso de polución.

1.3 Estructura

Nuestro TFG está relacionado con el proyecto de ecoMobility y sus funciones, su aportación a la ciudad inteligente, sus efectos en el cambio climático y cómo estos aspectos nos han motivado para trabajar en este proyecto. A través de nuestra interfaz web se pretende contribuir a nuestro objetivo general de crear una ciudad inteligente. Mostraremos el ámbito más amplio de ecoMobility y de las colaboraciones realizadas en otros TFGs.

En el contexto tecnológico, se describen aplicaciones y webs similares a nuestra interfaz web que hemos encontrado en funcionamiento en la actualidad, explicando las diferencias con nuestra propuesta y el porqué de la necesidad de

ésta. A continuación, se analiza el problema, decidiendo crear una aplicación web dada la versatilidad del mismo. Se propone una solución y las fases de desarrollo que se van a seguir en el presente estudio. En nuestro esquema de trabajo planificamos las diferentes tareas a realizar en función del tiempo con el que contamos. Calculamos el hardware y la aportación humana al TFG realizando un presupuesto estimado.

En el siguiente apartado del TFG diseñamos la solución, explicando la arquitectura del sistema, aportando un diagrama de casos de uso, un diagrama de tablas de bases de datos y un diagrama de clases. Una vez explicados los diagramas, indicamos la tecnología utilizada, a la vez que determinamos con qué herramientas y *frameworks* se ha llevado a cabo el desarrollo del TFG. A continuación, describimos el desarrollo de la solución propuesta mediante la utilización de *mockups* y una breve explicación de éstos.

Por último, presentamos las conclusiones, explicando problemas que han surgido, sus soluciones, errores cometidos y cómo evitarlos y lo que hemos aprendido en el transcurso de este TFG tanto personal como profesionalmente.

1.4 Colaboraciones en el proyecto ecoMobility

El proyecto ecoMobility, Figura 1, es un conjunto de servicios orientado a la ciudad inteligente. Su principal objetivo es simplificar la información obtenida por diferentes medios, como pueden ser sensores distribuidos por la ciudad, además de otros conductores y de entidades públicas, y ofrecerla al usuario de una forma comprensiva y eficaz. El proyecto se compone de distintas partes que han sido estudiadas por varios componentes del proyecto y que conforman el conjunto del proyecto ecoMobility.

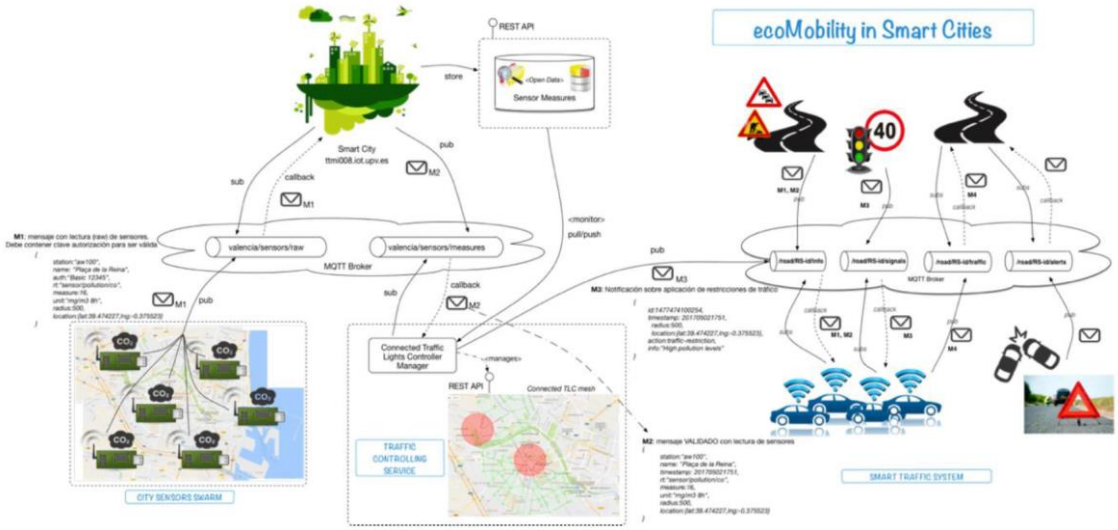


Figura 1. Diagrama del proyecto.

Uno de estos componentes del equipo analiza y diseña una solución IoT basada en Fog/Edge Computing. Según su autor, esta solución resulta en "un sistema aplicable a l'àmbit de les ciutats intel·ligents que permeta la computació ditribuïda de manera seguraen una xarxa de dispositius IoT" [1]. En otro estudio se destaca asimismo la preponderancia de los sistemas IoT en este proyecto. Este

autor se vale también del Fog Computing para el diseño e implementación de una plataforma con la cual optimizar los sistemas IoT [2].

En otra contribución al proyecto, se busca una solución orientada hacia los microservicios. El estudio se apoya en la tecnología Spring-boot a través de la cual se pueden crear microservicios de una forma sencilla. La importancia de los microservicios parece evidente y palpable, ya que, como afirma el autor de este estudio, “segurament els microserveis siguen l’arquitectura del futur” [3].

Otro estudio aborda el problema de la información recibida a través de Internet que no siempre sigue un único modelo de comunicaciones. Con el objetivo de integrar los distintos equipos que transmiten los mensajes, el autor propone un modelo de comunicación entre sensores y una plataforma llamada Fi-ware, plataforma en constante evolución y en la que frecuentemente “suelen cambiar las aplicaciones […]”.No solo esta modificación de aplicaciones, sino que, según este estudio, cambian completamente “los métodos de ejecución de instruccione” [4].

La última aportación que merece ser mencionada es la que desarrolla un módulo de comunicaciones entre ciudad y sus ciudadanos. Para su objetivo, el autor adopta una aplicación funcional en Android con la cual conectarse y recibir información, n tiempo real y en todo momento, sobre el estado ambiental de la ciudad. El autor concluye que “el futuro de las aplicaciones conectadas con la ciudad […] está cada vez más cerca” [5].

2. Contexto tecnológico

En la actualidad, se encuentran dos software webs de características similares: www.eltiempo.es y www.aqicn.org, sitios web en los que la información se ofrece de dos formas diferentes. El primero, www.eltiempo.es, muestra una tabla con diversos datos que van actualizándose a lo largo del día. Dicha tabla consta, en el caso de Valencia, de cinco puntos de toda la zona, un ámbito muy concreto y delimitado de la geografía. En el caso de www.aqicn.org podemos ver un mapa con cuatro sensores repartidos por la zona de Valencia. Dichos sensores muestran datos actualizados periódicamente a lo largo del día, pero no aportan ningún otro tipo de información.

Con respecto al proyecto ecoMobility, nuestra intención no es simplemente mostrar dicha información sino ampliarla y modificarla de forma que resulte más eficiente. El proyecto ecoMobility permite a los usuarios sugerir nuevos sensores ampliando así la red de recopilación de datos. Por otro lado, ecoMobility muestra zonas de más o menos contaminación, dando la opción de tomar una u otra ruta, e incluso cambiar la ruta en caso de que la contaminación de una zona determinada exceda de los límites establecidos previamente, marcando las zonas de exclusión en el mapa de ecoMobility.

3. Análisis del problema

Nos encontramos con un proyecto compuesto por una cantidad de información ilegible para el usuario convencional. Con esto surge el problema de la forma de visualización, existiendo diferentes posibilidades, como puede ser una aplicación de móvil, una página web, texto en crudo o un programa dedicado. Cada una de estas soluciones presenta una problemática diferente: en el caso de la aplicación móvil significaría desarrollar una aplicación para cada sistema operativo, dependiendo del modelo del smartphone o Tablet. El texto en crudo ha sido descartado por la dificultad de lectura que surge en comparación con una presentación interactiva y de forma más visual mediante la utilización de mapas y esquemas. El programa dedicado se ha descartado, al igual que la opción de la aplicación, por la existencia de diferentes sistemas operativos teniendo que crear una interfaz diferente para cada uno de ellos. La opción que se ha decidido adoptar es la página web, dado que se trata de una opción multiplataforma. De esta forma contamos con una tecnología portable entre sistemas operativos, a la vez que tenemos la opción de diseñar una interfaz visualmente atractiva para el usuario; la información es, además, de fácil acceso.

Al tratarse de un diseño web, surge la problemática del requisito de la conectividad para poder visualizar el contenido, pero entendemos que, a día de hoy, con las nuevas tecnologías y el uso común del smartphone (donde el 66% de la población cuenta con una línea de telefonía móvil y sería superior si hablásemos únicamente de países desarrollados)[1], dicho problema se reduce a un porcentaje pequeño de la población que pertenece, en su mayoría, a la tercera edad. Este proyecto está enfocado a una franja de la población comprendida entre los 18 y los 65 años, siendo los mayores de 65 la mayoría del 34% restante de la población [2] que, aunque disponga de un Smartphone, no lo utiliza como herramienta a la hora de conducir.

3.1 Solución propuesta

Nuestra solución al problema pasa por el desarrollo del proyecto en forma de una interfaz web. Por un lado, como hemos indicado en el anterior punto, trabajar con una interfaz web nos evita tener que desarrollar diferentes aplicaciones o programas dependiendo del sistema operativo que se utilice y, por otro lado, nos da libertad para poder implementar mapas, gráficos, etc.

Una interfaz web es una estructura formada por una serie de elementos gráficos que permite a los usuarios acceder a los contenidos de un sitio web. Es una manera de interactuar con un servidor de suministro, una consola basada en web que se visualiza a través de un navegador.

Nuestra solución consta de las siguientes fases de desarrollo:

- En primer lugar, buscamos la mejor solución para poder ofrecer un producto con el cual los usuarios sean capaces de recibir información útil y de forma fácil, decidiéndonos por la interfaz web. Consideramos las necesidades de los usuarios y nos centramos en crear un producto que resuelva dichas necesidades. Una vez

[1] https://elpais.com/tecnologia/2018/02/27/actualidad/1519725291_071783.html
[2] https://elpais.com/diario/2010/09/19/sociedad/1284847201_850215.html

resueltas estas necesidades, intentamos que el producto cumpla con sus expectativas. Exponemos nuestro proyecto a usuarios potenciales para recibir un *feedback* con respecto a sus expectativas y sus experiencias con un hipotético uso de nuestra interfaz web. Posteriormente planteamos posibles problemas que le puedan surgir al usuario durante la utilización de la interfaz web, teniendo en cuenta el contexto en el cual un usuario decida utilizarlo, el nivel de conocimiento de dicho usuario y la velocidad de acceso a Internet de que disponga.

- A continuación, pasamos a la fase del aspecto visual de la interfaz web. En esta primera fase ponemos énfasis en la consistencia de la visualización de datos, utilizando un diccionario de estilos. Nos centramos en que la asimilación de información sea eficiente por parte del usuario, haciendo hincapié en que el formato sea familiar. Proporcionamos flexibilidad al usuario para controlar la visualización de los datos. Intentamos captar la atención del usuario cuando mostramos información importante o dependiente del tiempo mediante la utilización de la intensidad de información regular e información importante. Resaltamos, asimismo, información mediante subrayado, enmarcado, etc., variación en el tamaño de fuentes para determinar información de más (tamaño más grande) o menos (tamaño más pequeño) importancia, coloreado inverso o señales de audio, utilizando sonidos suaves o estridentes dependiendo de lo que queremos destacar para el usuario.
- Posteriormente nos centramos en el diseño de los diferentes diagramas mediante el uso de un software de desarrollo, en nuestro caso BoUML. Mediante este software, creamos los diagramas de casos de uso, diagramas de clases y diagramas de secuencia.
- A continuación utilizamos el software APACHE para desplegar un servidor HTTP que incluya el lenguaje interpretado PHP. Este proceso nos permite acceder posteriormente a nuestra página web.
- Finalmente, en la última fase de desarrollo implementamos el código resultante de los diagramas para obtener una interfaz web funcional.

3.2 Plan de trabajo

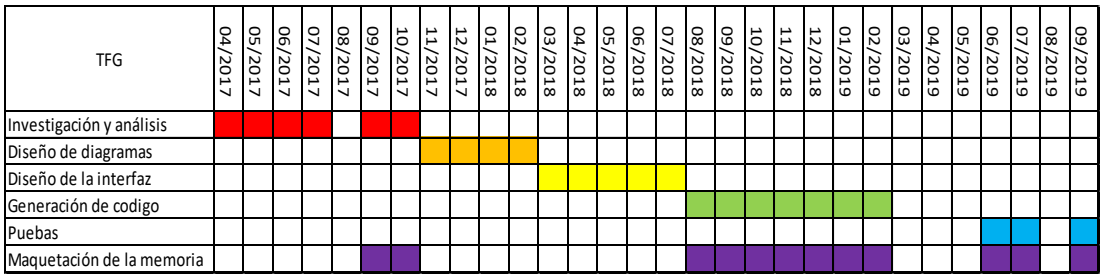


Tabla 1. Diagrama de Gannt.

El plan que hemos seguido se ha basado en seis puntos:

- Investigación y análisis,

- diseño de diagramas,
- diseño de la interfaz web,
- escritura del código de la aplicación,
- realización de pruebas,
- maquetación del documento final.

El trabajo se ha desarrollado en unas 540 horas distribuidas en un periodo comprendido entre abril de 2017 y septiembre de 2019.

3.3 Presupuesto

Servidor dedicado en www.ovh.es/servidores_dedicados/.

Procesador Intel® Xeon® E3-1230v6.

Todos los servidores se entregan con una dirección IPv4 pública.

12 meses * 58,19 + IVA.

Cada servidor dedicado cuenta con un espacio de *backup* gratuito de 500 GB.

Todos los servidores de OVH incluyen un sistema anti-DDoS.

500 Mbit/s, tráfico ilimitado.

Mantenimiento 50+IVA al mes.

4. Diseño de la solución

4.1 Arquitectura del Sistema

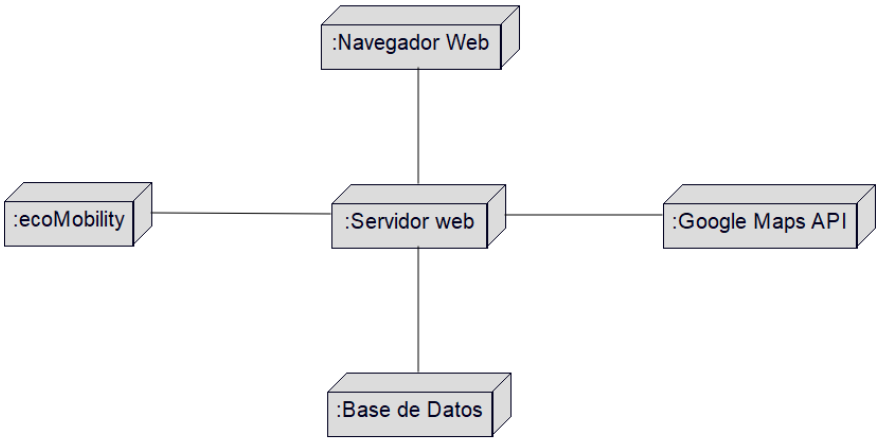


Figura 2. Diagrama de las partes del trabajo.

El presente trabajo está basado en una arquitectura cliente-servidor desglosado en la Figura 2. El cliente actúa como navegador web haciendo peticiones al servidor web. Para obtener los datos, el servidor web hace consultas a ecoMobility, a la API de *GoogleMaps* y a una base de datos privada. La conexión a ecoMobility se realiza a través de un servicio REST devolviéndonos los valores de lectura de los detectores de CO2 ubicados en la ciudad. Por su lado, la API de *GoogleMaps* nos permite ubicarlos mediante coordenadas en un lugar en el mapa. Por último, para poder gestionar los usuarios del servicio web, utilizamos una base de datos del tipo MySQL. En resumen, el usuario hace una petición al servidor web; el servidor web, a su vez, hace las peticiones pertinentes; la aplicación web las interpreta y las devuelve al usuario mediante la interfaz web de la que trata el presente trabajo.

4.2 Diseño detallado

A continuación, mostramos los diferentes diagramas en los que nos hemos basado para desarrollar nuestra interfaz web, diagrama de casos de uso, diagrama de las tablas de la base de datos y diagrama de clases.

4.2.1 Diagrama de casos de uso del rango usuario

En el diagrama de casos de uso del rango usuario, como se puede ver en la Figura 3, considerado rango básico, se observan las acciones que se pueden realizar al utilizar nuestra aplicación web. Un usuario que entre en la página

principal de la aplicación sin estar registrado puede acceder a las siguientes acciones:

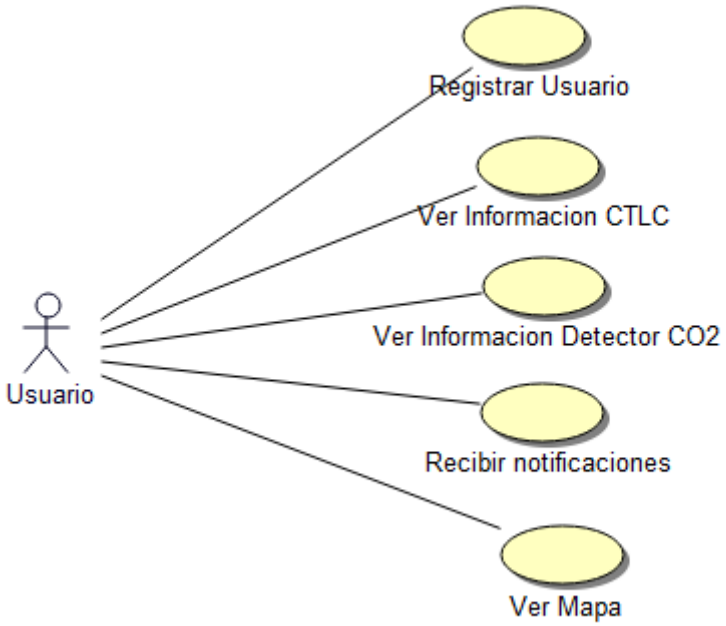


Figura 3. Diagrama de casos de uso de un usuario sin registrar

- Ver la posición en el mapa de los detectores de CO_2 ya existentes.
- Ver la posición de los controladores de luces de los semáforos.
- Registrar un usuario nuevo.
- Ver información CTLC (ver el estado, las coordenadas y si se encuentra o no restringido).
- Ver el mapa con la situación y el estado general de todos los detectores de CO_2 y controladores de luces. Es aquí donde aparecen unos círculos en caso de existir alguna zona restringida.

En el diagrama de casos de uso del rango usuario registrado, en la Figura 4, se observan las acciones que puede realizar un usuario registrado en la página al utilizar nuestra aplicación web. En este caso tiene acceso a todas las acciones anteriores, además de las siguientes:

- Iniciar sesión.
- Ver la lista de CTLCs (listado de controladores de luces de los semáforos).

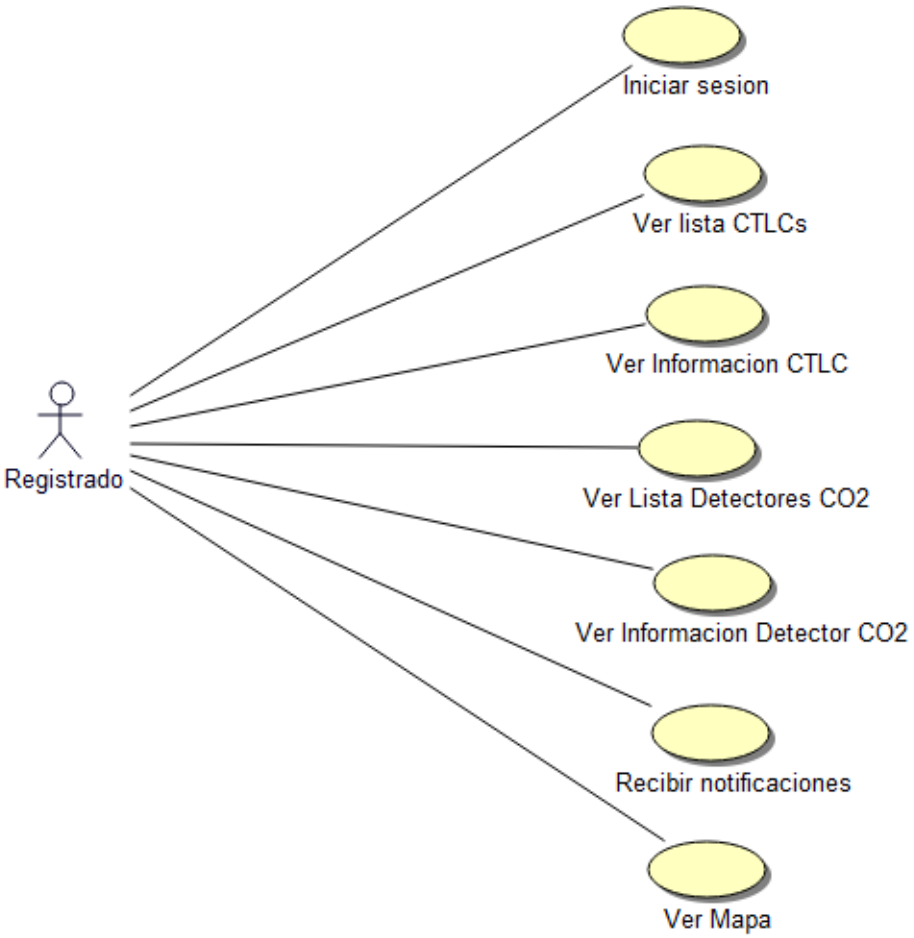


Figura 4. Diagrama de casos de uso de un usuario registrado

- Ver la lista de los detectores de CO2.
- Recibir notificaciones en forma de mail.

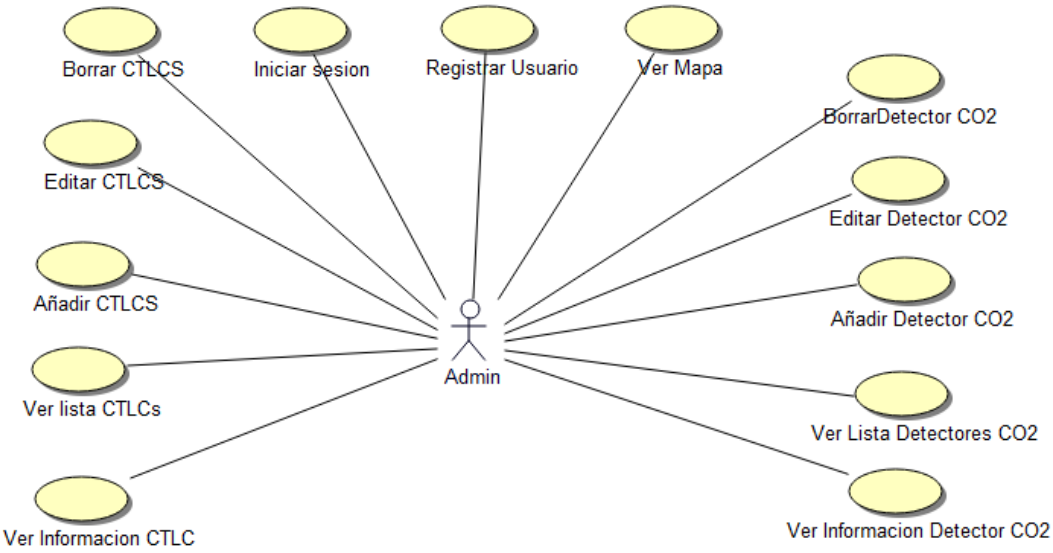


Figura 5. Diagrama de casos de uso de un usuario administrador

En el diagrama de casos de uso del rango administrador, como se puede ver en la Figura 5, se observan las acciones que puede realizar en la página un administrador al utilizar nuestra aplicación web. En este caso puede realizar todas las acciones anteriores, además de las siguientes:

- Añadir, eliminar o editar CTLCs.
- Añadir, eliminar o editar detectores de CO2.
- Añadir, editar, eliminar o buscar usuarios registrados.

4.2.2 Diagrama de las tablas de la base de datos

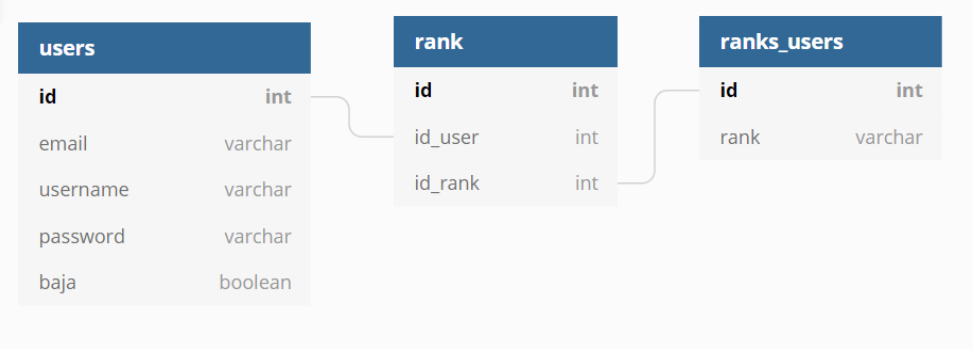


Figura 6. Diagrama de las tablas de la base de datos.

En el diagrama de las tablas de base de datos, en la Figura 6, tenemos tres tablas. En la primera tabla (“users”) almacenamos los usuarios con los siguientes datos: correo electrónico, nombre de usuario, contraseña, un identificador de usuario único y si está dado de baja. La contraseña se guarda en la base de datos cifrada como MD5. La siguiente tabla (“rank”) almacena la información entre el id de usuario único y su id de rango único. Por último, la tercera tabla (“rank_users”) sólo almacena dos registros, un identificador único de rango y el nombre del rango, que representan usuario y administrador.

4.2.3 Diagrama de clases

El diagrama de clases, se puede observar en la Figura 7, está basado en el patrón de diseño MVC (modelo – vista – controlador) que separa los datos y la lógica de la aplicación de su representación. En nuestra aplicación tenemos un controlador llamado “home” el cual puede realizar las siguientes acciones:

- Visualizar página principal.
- Iniciar se session como usuario.
- Registrar un usuario.
- Mostrar la lista de CTLCs.
- Mostrar la información de un CTLC.
- Mostrar la lista de detectores de CO2.
- Mostrar información de un detector de CO2.
- Añadir, editar, borrar o buscar usuarios.

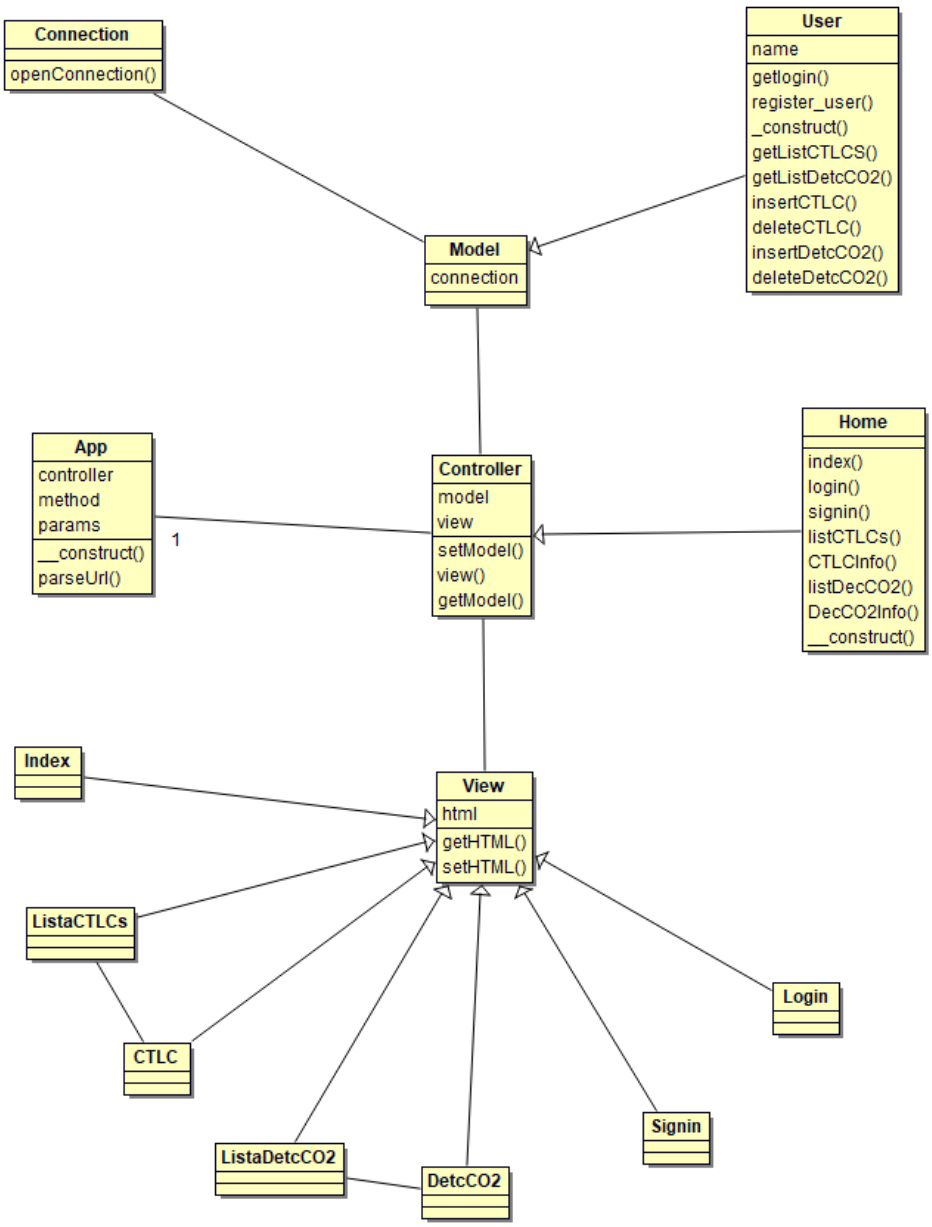


Figura 7. Diagrama de clases

- Añadir, editar o borrar un CTLC.
- Añadir, editar o borrar sensor de CO2.

4.2.4 Mockups del diseño de la interfaz

En la imagen de la Figura 8 a continuación, podemos ver la página principal de nuestra interfaz web a la cual puede acceder cualquier usuario, esté o no registrado. En ella aparecen tres botones (“Inicio”, “Registrarse” e “Iniciar Sesión) y un mapa de la ciudad donde se pueden apreciar los detectores de CO2, los controladores de tráfico, las uniones entre CTLCs y dos muestras de zonas de restricción a la circulación. La figura 8 corresponde al diagrama de casos de uso del rango “usuario” representado en la figura 3.

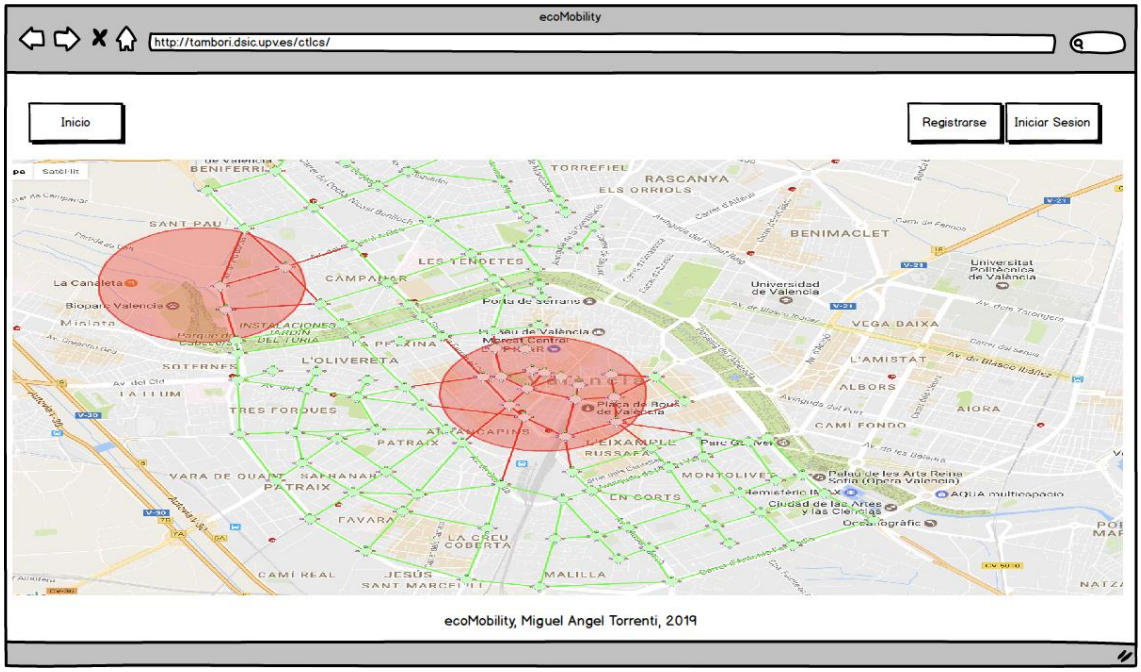


Figura 8. Página principal

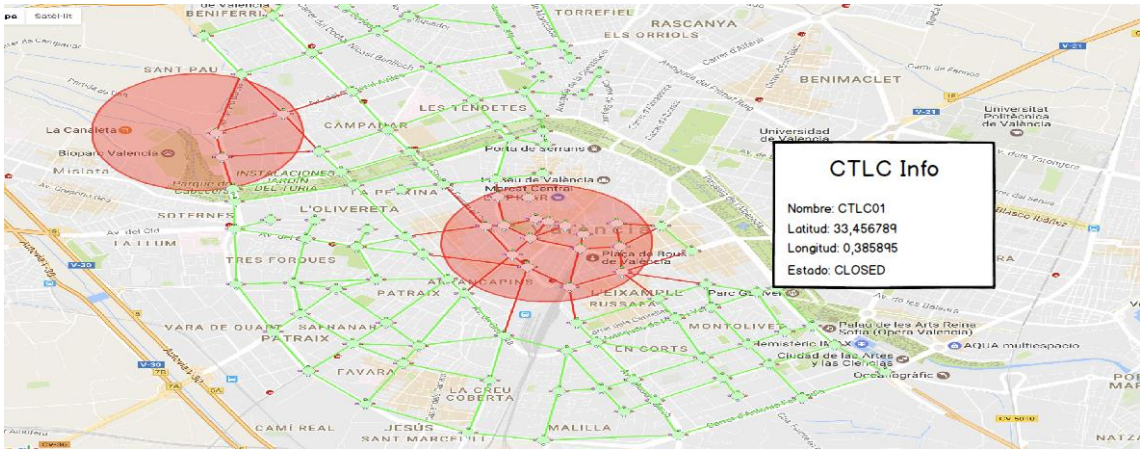


Figura 9. Pantalla de información de un CTLC

En la imagen de la Figura 9 podemos ver lo que ocurre cuando ponemos el ratón sobre un CTLC determinado. Al realizar esta acción aparece el nombre del CTLC, su latitud, longitud y estado. En caso de haber algún tipo de restricción al tráfico, se mostraría de igual forma que en la página principal (Figura 8).

En la imagen a continuación, la Figura 10, podemos ver lo que ocurre cuando ponemos el ratón sobre una zona de restricción determinada del mapa. Esta opción sólo será posible cuando se produzca una restricción; sin embargo, si no hay ninguna zona restringida, no obtendremos información de los detectores. Esta acción nos muestra el nombre del CTLC, junto a su latitud, longitud y estado. En caso de haber algún tipo de restricción al tráfico, se mostrará de igual forma que en la página principal (Figura 8).

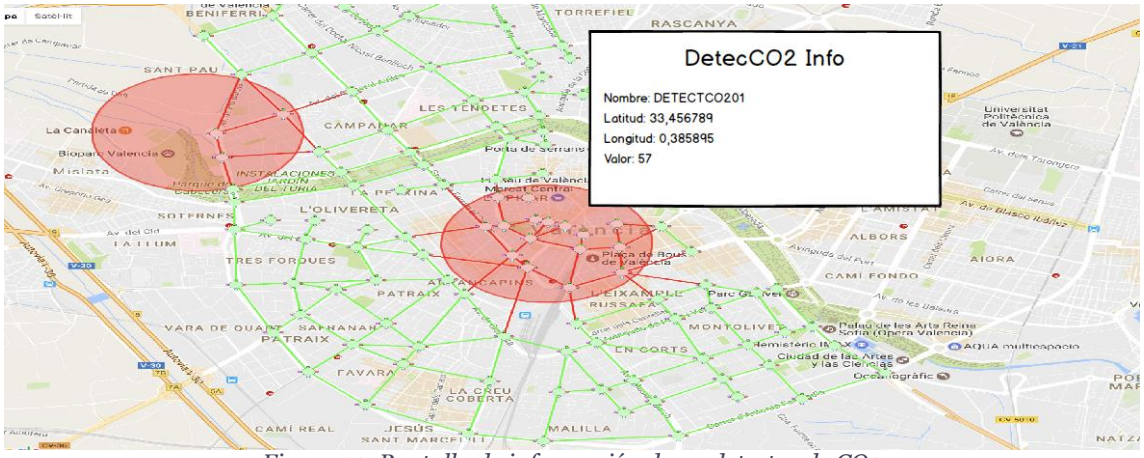


Figura 10. Pantalla de información de un detector de CO2

En la siguiente imagen mostrada en la Figura 11, podemos ver lo que ocurre al darle al botón "registrarse" en la página principal (Figura 8). Al acceder a esta nueva página, aparecen cuatro botones indicativos de las siguientes acciones:

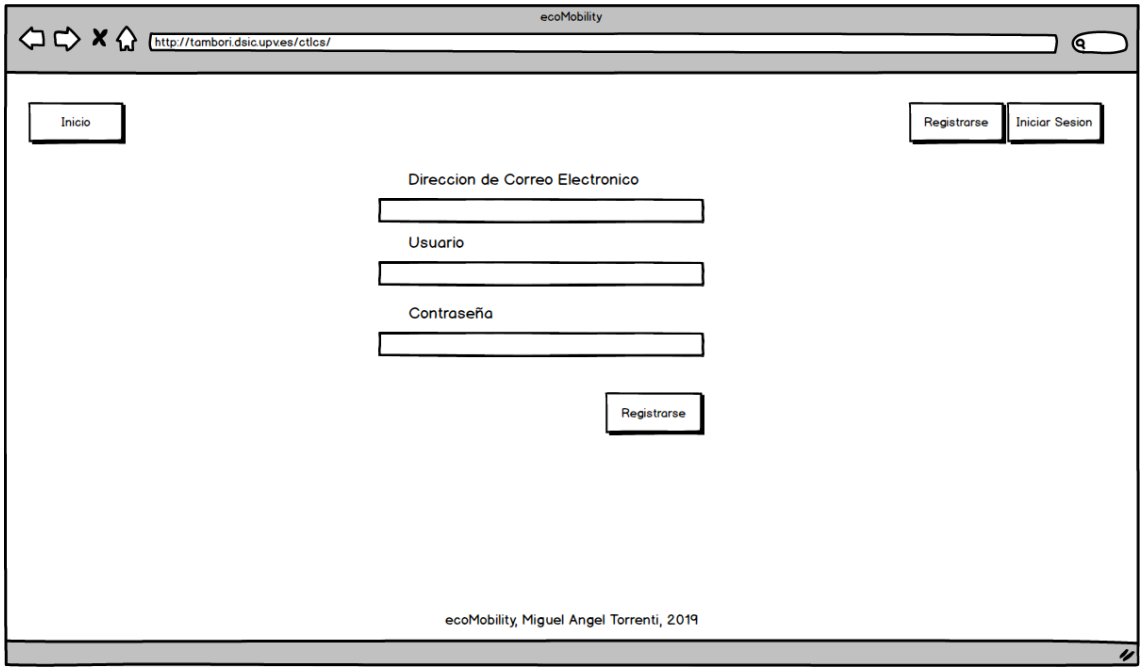


Figura 11. Pantalla de registro de usuario

- "Inicio", para volver a la página principal;
- "Registrarse", localizado en la parte superior derecha, que nos da acceso a la misma página en que estamos;
- "Iniciar sesión", para darnos acceso al login en caso de estar ya registrados;
- "Enviar", localizado en la zona inferior de la pantalla que nos da acceso a la página de inicio de usuario registrado (Figura 13).

Asimismo, tres campos de entrada de texto: "dirección de correo electrónico", "Usuario" y "Contraseña".

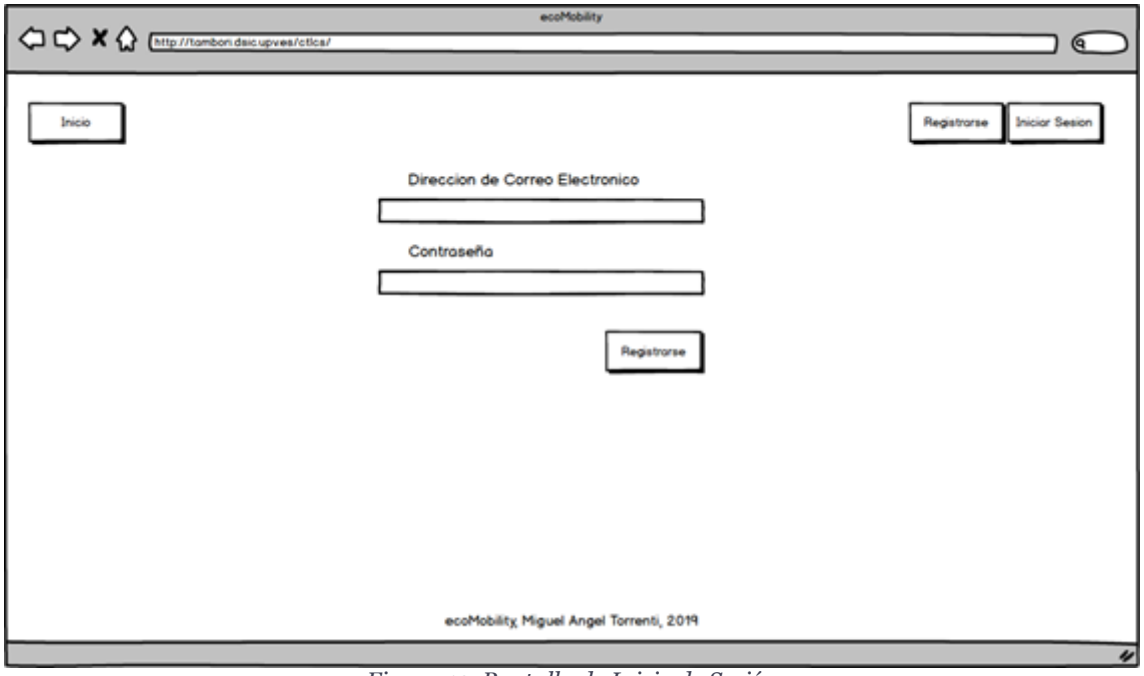


Figura 12. Pantalla de Inicio de Sesión

En la figura 12 vemos lo que ocurre al darle al botón de “Inicio de Sesión” en la página principal (Figura 8). Esta página muestra cuatro botones que posibilitan realizar las siguientes acciones:

- “Inicio”, para volver a la página principal;
- “Registrarse”, situado en la parte superior derecha para dar acceso a la pantalla de registro de usuario;
- “Iniciar sesión”, para darnos acceso a la misma página en la que nos encontramos y,
- “Conectarse”, en la zona inferior de la pantalla, que nos da acceso a la “página de inicio” de usuario registrado.

Esta figura ofrece, asimismo, dos campos de entrada de texto: “Dirección de Correo Electrónico” y “Contraseña”.

En la imagen correspondiente a la Figura 13, una vez iniciada sesión, vemos la página principal de usuario registrado. Aparecen nuevas opciones en forma de cuatro botones

- “Inicio”, que nos lleva a la misma página donde nos encontramos;
- “Lista CTLCs”, que nos da acceso al listado de los CTLCs;
- “Listar DetecCO2”, que da acceso al listado de los detectores de CO2; y
- “Usuario”, que nos da acceso a un desplegable con la opción de “Cerrar Sesión” volviendo a la página principal.

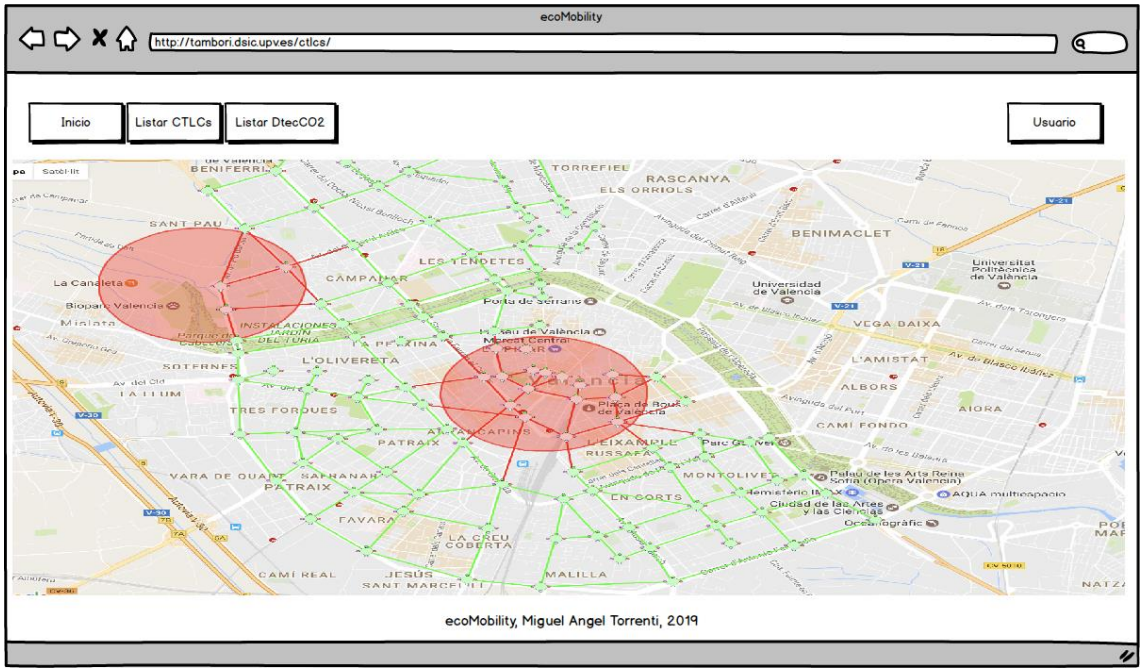


Figura 13. Página principal de Usuario registrado

Además, contamos con el mismo mapa que aparece en la página principal (Figura 8)

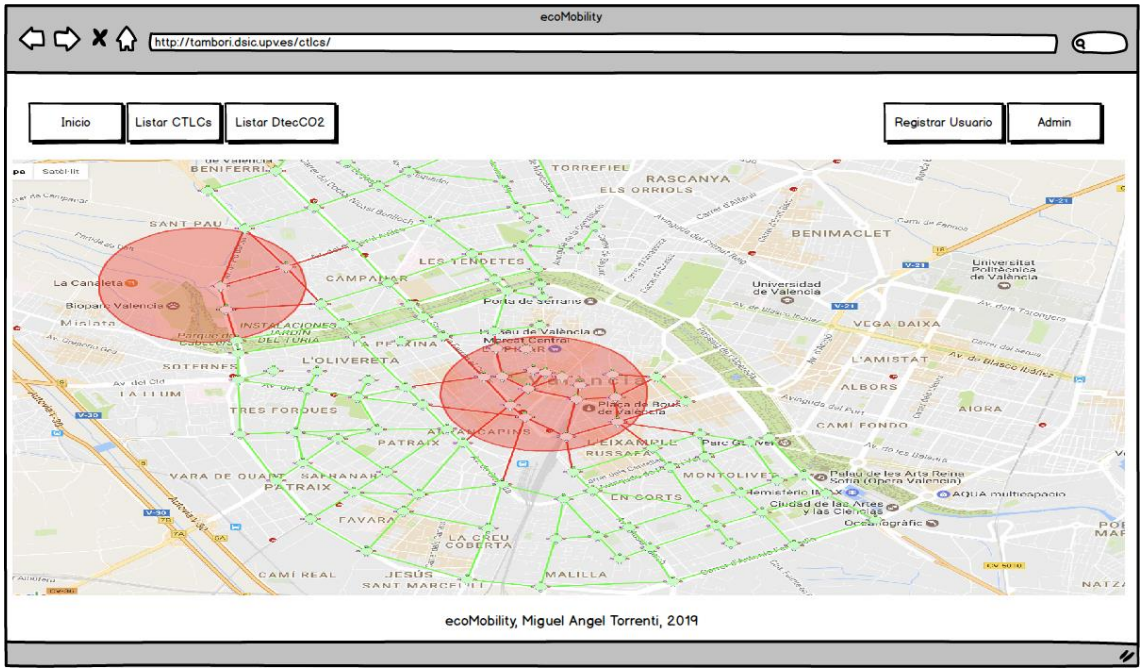


Figura 14. Página Inicial de Administrador

En la imagen de la Figura 14 vemos la página inicial una vez introducidos los datos de un administrador. Aparecen nuevas opciones en forma de botones, en este caso 5:

- "Inicio", que no lleva a la misma página donde nos encontramos;
- "Listar CTLCs", que nos da acceso al listado de los CTLCs;
- "Listar DetecCO2", que da acceso al listado de los detectores de CO2;

- “Registrar usuario”, que da acceso a la página para crear un usuario nuevo; y
- “Admin”, que nos da acceso a un desplegable con la opción de “Cerrar Sesión”, volviendo a la página principal (Figura 8).

Contamos, además, con el mismo mapa que aparece en la página principal.

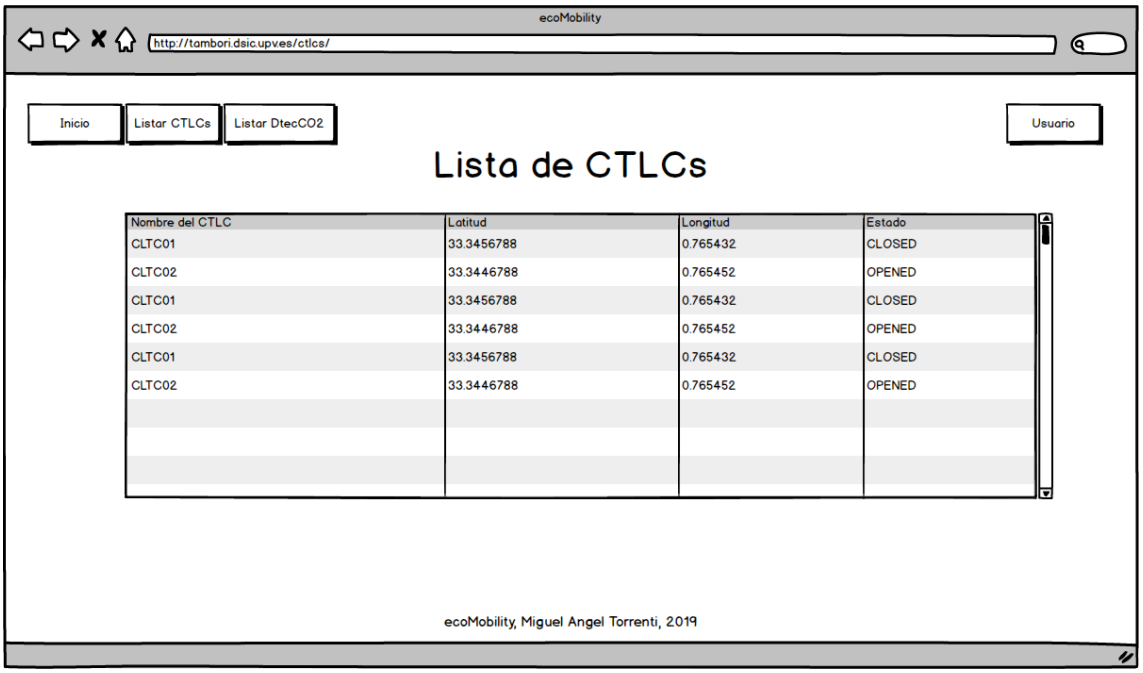


Figura 15. Listado de los CTLCs

En la Figura 15 vemos la lista de CTLCs, que nos aporta información de “nombre”, “longitud”, “latitud” y “estado” de los diferentes CTLCs situados en tu ciudad. Accedemos a ella desde la página principal de usuario registrado (Figura 13) pulsando el botón de “Listar CTLCs”. Aparecen las mismas opciones de botones que en la “Página Principal de Usuario Registrado”.

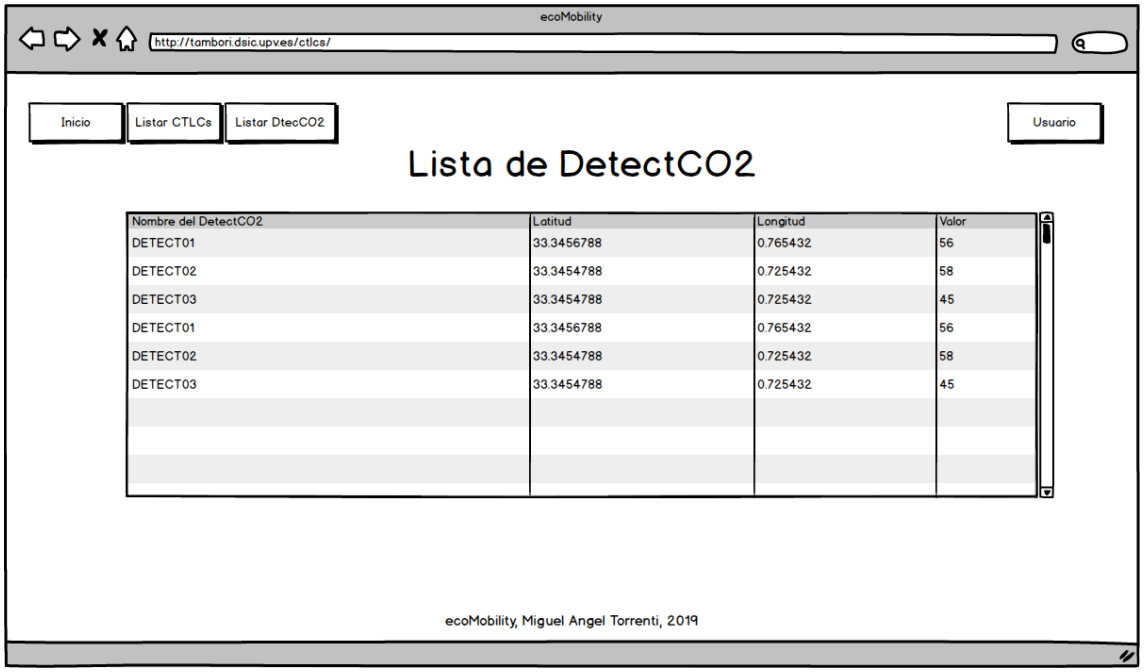


Figura 16. Listado de los Detectores de CO2

Esta imagen de la Figura 16, donde aparece la lista de detectores de CO2, ofrece información sobre "nombre", "longitud", "latitud" y "valor" de los diferentes detectores de CO2 situados en tu ciudad. Accedemos a ella desde la página principal de usuario registrado (Figura 13) pulsando el botón de "Listar DetectCO2". Aparecen las mismas opciones de botones que en "Página Principal de Usuario Registrado".

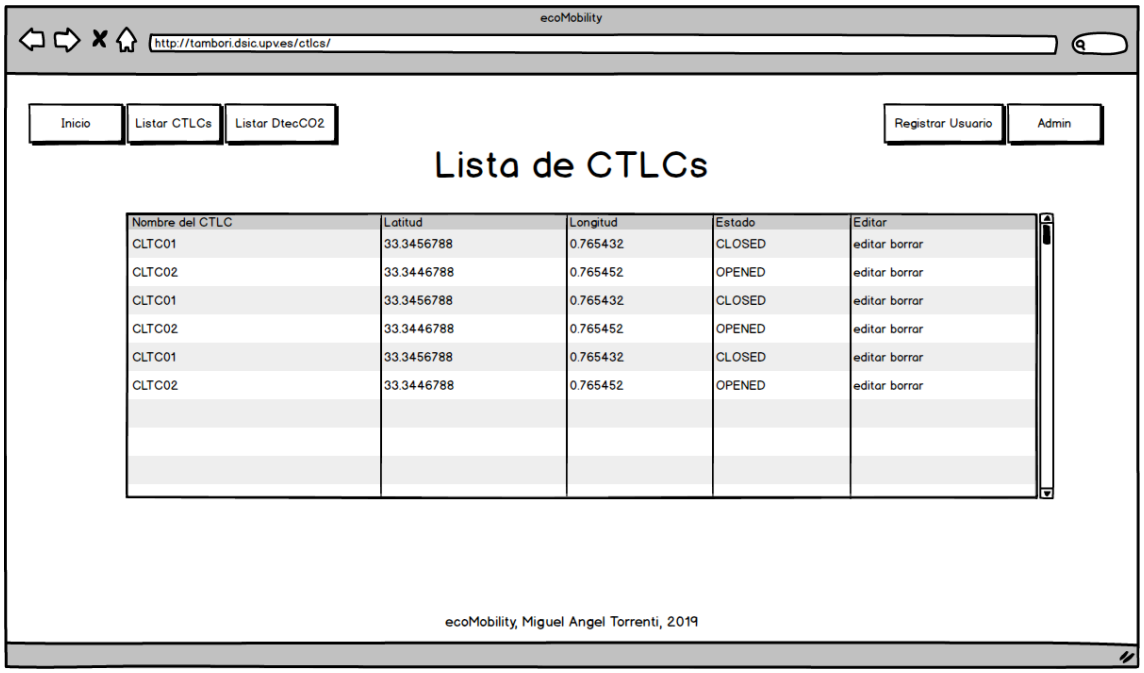


Figura 17. Listado de CTLCs

Esta imagen de la Figura 17, con la lista de CTLCs, nos informa sobre "nombre", "longitud", "latitud", "estado" de los diferentes CTLCs situados en tu ciudad; la opción "editar" nos permite modificar información de los CTLCs o

borrarlos. Accedemos a ella desde la página principal de usuario registrado (Figura 13) pulsando el botón “Listar CTLCs”. Aparecen las mismas opciones de botones que en “Página Inicial de Administrador” (Figura 14).

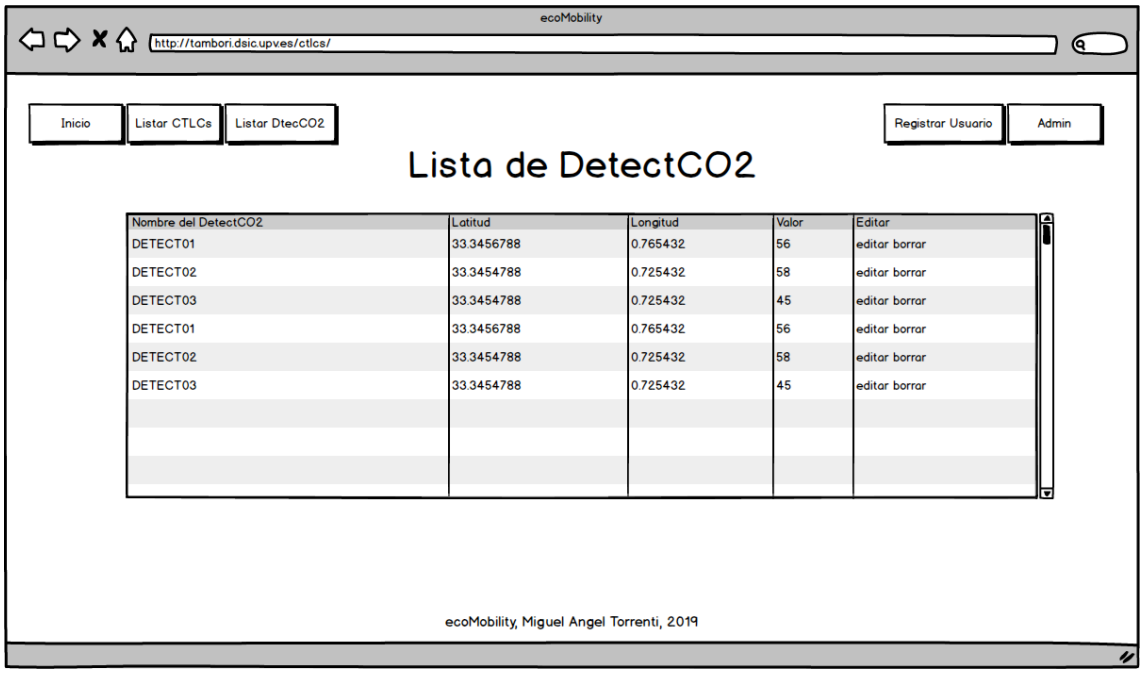


Figura 18. Listado de detectores de CO2

Como podemos ver en esta imagen de la Figura 18, vemos la lista de detectores de COs; esta lista nos da información sobre “nombre”, “longitud”, “latitud”, “valor” de los diferentes CTLCs situados en tu ciudad; y el botón “editar” nos permite modificar información de los detectores de CO2 o borrarlos. Accedemos a ella desde la página principal de usuario registrado (Figura 13) dándole al botón “Listar DetecCO2”. Aparecen las mismas opciones de botones que en “Página Administrador” (Figura 14).

En la imagen de la Figura 19 aparece una ventana modal con dos botones:

- “Cancelar”, para volver a la pantalla anterior de listado de CTLCs; y
- “Modificar”, para validar los cambios introducidos.

También ofrece siete campos de entrada de texto:

- “Nombre”, para diferenciar cada uno de los CTLCs;
- “Longitud” y “Latitud”, para posicionarlo en el mapa; y

“Conexión (01-04)”, para conectarlo con otros CTLCs ya existentes.

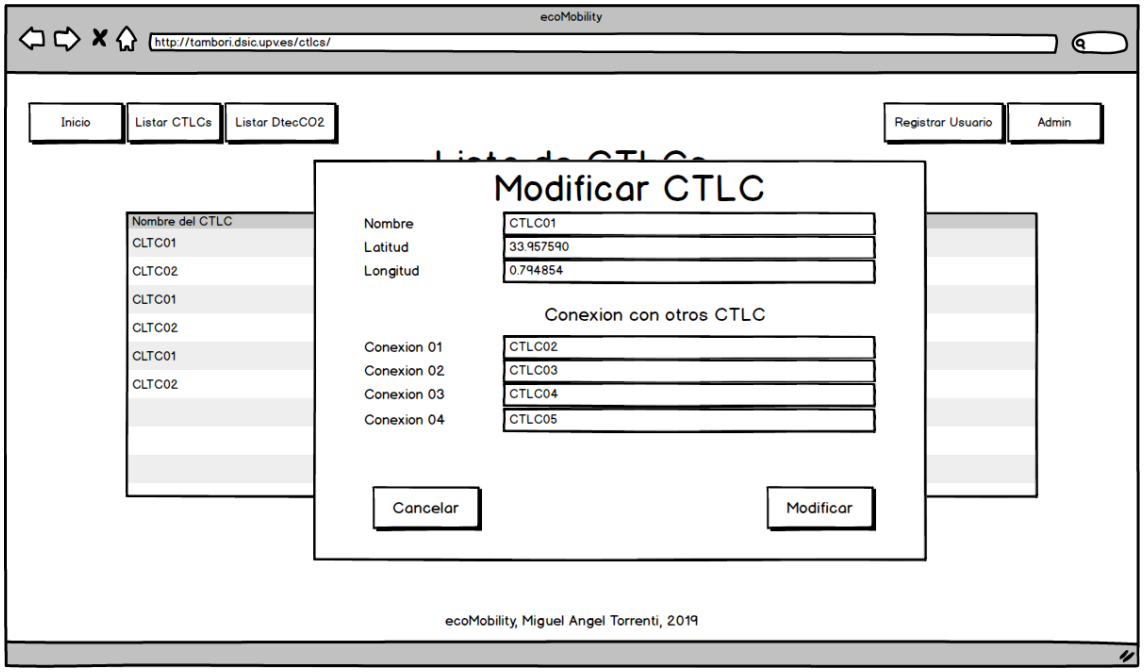


Figura 19. Modificar CTLCs

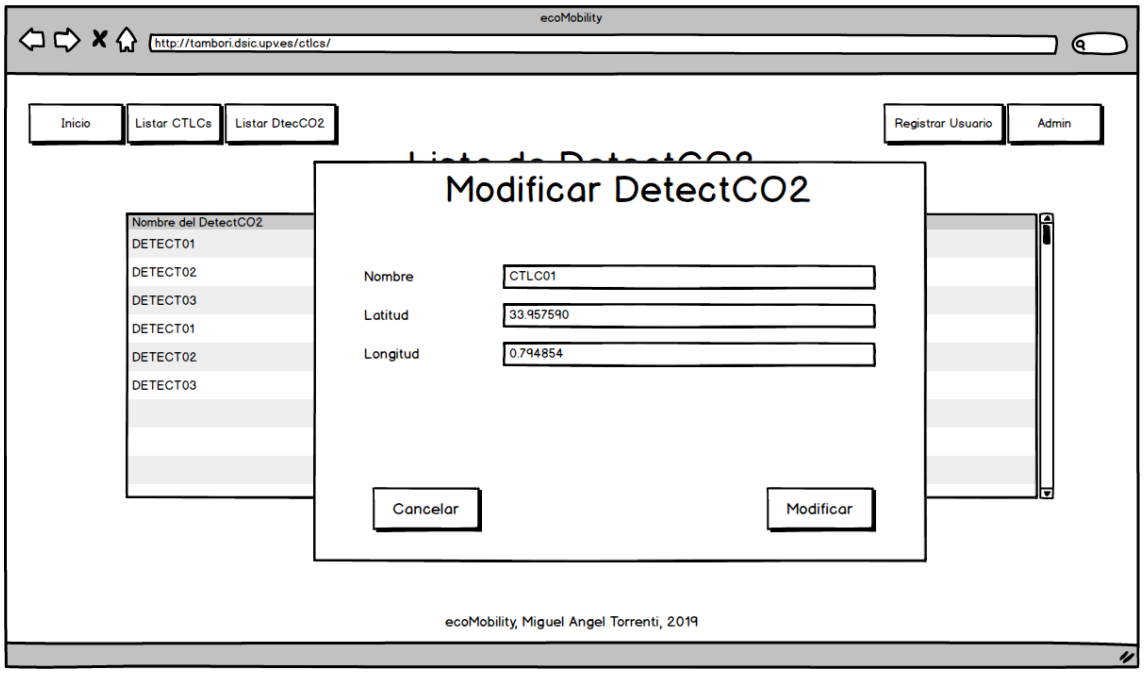


Figura 20. Modificar CO2

De forma parecida a la pantalla anterior, en esta ventana modal mostrada en la Figura 20, aparecen dos botones:

- "Cancelar", que permite volver a la pantalla anterior de "Listado de CO2", y
- "Modificar", para validar los cambios introducidos.
- Contiene, asimismo, siete campos para introducir texto:
- "Nombre", para diferenciar cada uno de los detectores de CO2;

- “Longitud” y “Latitud”, para posicionarlo en el mapa.

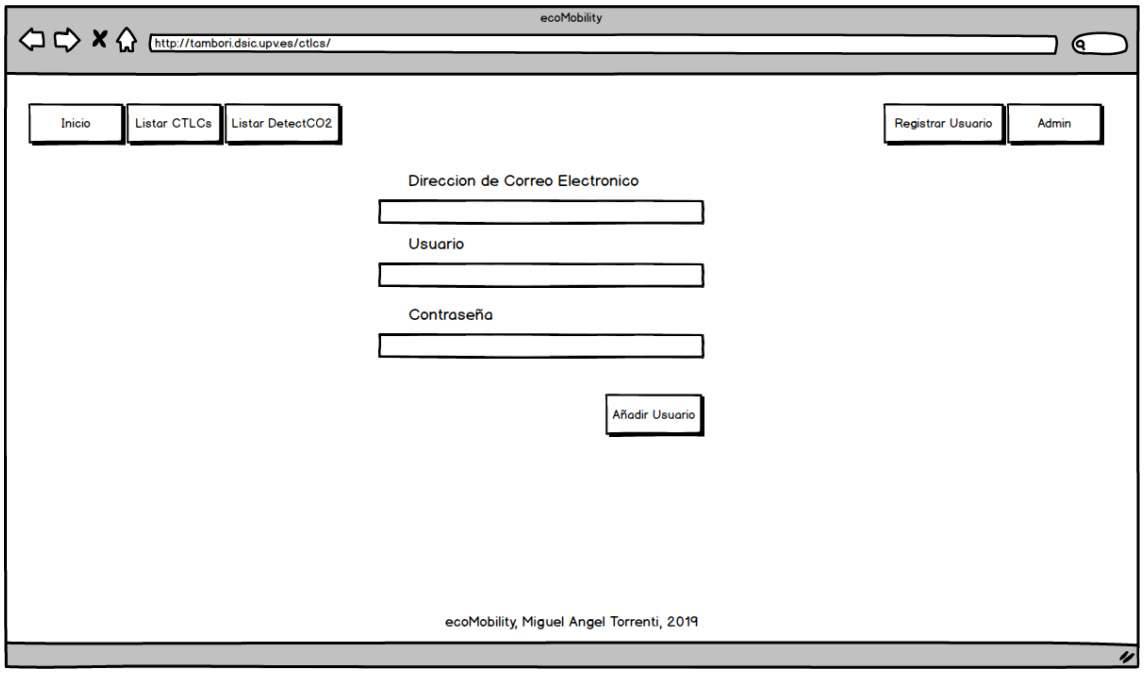


Figura 21. Pantalla de registro de nuevo usuario (Administrador)

En esta imagen, en la Figura 21, podemos ver lo que ocurre al pulsar el botón de “Registrar usuario” en la página principal de administrador (Figura 14):

- “Inicio”, para volver a la página principal;
- “Admin”, ubicado en la parte superior derecha, nos da acceso a la página principal de administrador;
- “Iniciar sesión”, nos da acceso al login en caso de estar ya registrados; y
- “Enviar”, situado en la parte inferior, que nos da acceso a la “página de inicio” de usuario registrado.

Presenta, asimismo, tres campos para la introducción de texto, a saber, “Dirección de correo electrónico”, “Usuario” y “Contraseña”.

4.3 Tecnología utilizada

A continuación enumeramos las diferentes herramientas, tecnologías y *frameworks* que hemos utilizado para desarrollar este trabajo.

4.3.1 Lenguajes de programación

- HTML (Hyper Text Markup Lenguage), que es un lenguaje de etiquetas que se utiliza para visualizar páginas web (*World Wide Web*). HTML fue diseñado inicialmente para describir documentos científicos, pero a lo largo de los años ha ido evolucionando y adaptándose a las nuevas necesidades de diferentes ámbitos tanto científicos como en su uso para aplicaciones [6].

- CSS (*Cascading Style Sheet*), también denominado hoja de estilo en cascada, es un lenguaje utilizado para dar formato a las páginas escritas en HTML y derivados. Sus especificaciones están reguladas por el W3C (World Wide Web Consorcium). Inicialmente se trabajaba en un mismo documento, pero producía malas prácticas; en la actualidad se crea en documentos separados.
    - B*ootstrap* es un *framework* derivado de CSS. Inicialmente concebido para la creación de Twitter, tras su primera *Hack Week*, y tratándose de un software de *open source*, desarrolladores de todos los niveles lo utilizan como guía para herramientas de desarrollo.
- JavaScript, también conocido como lenguaje de script, es un lenguaje interpretado, orientado a objetos con funciones de primera clase. JavaScript, cuyo estándar es ECMAScript, no debe confundirse con Java, ya que se trata de lenguajes diferentes y están registrados como diferentes marcas; se diferencian en sintaxis, semántica y usos, además de que JavaScript es un lenguaje interpretado y Java es semicompilado.
    - JQuery es una librería de JavaScript. Está compuesto por una API de uso fácil que trabaja con múltiples buscadores; JQuery ha cambiado la forma en la que millones de personas escriben JavaScript [7].
    - En Google Maps empleamos diferentes API. En primer lugar, utilizamos el Maps JavaScript API el cual nos permite personalizar mapas con nuestro propio contenido. En segundo lugar, el uso de Directions API nos muestra las direcciones entre múltiples localizaciones. En tercer lugar, usamos Geocoding API para convertir las direcciones en coordenadas geográficas. Por último, utilizamos Roads API para recoger los datos de GPS e información de las carreteras.
- PHP es un leguaje interpretado, parte servidor muy utilizado en el desarrollo web ya que puede ser incrustado en documentos HTML [8] [9].

4.3.2 Software

- Visual Studio Code es un editor de código fuente. Su característica principal es su compatibilidad con varios lenguajes de programación y es, además, personalizable [10]. Gratuito y de código abierto, fue desarrollado por Microsoft e incluye soporte para la depuración, control integrado de Git, resaltado de sintaxis y finalización inteligente de código.
- Git es un software de control de versiones diseñado para soportar tanto proyectos pequeños como proyectos de gran envergadura. Se trata de un software gratuito y de código abierto. Su propósito es llevar el registro de los cambios en archivos y coordinar el trabajo de diferentes personas que realizan modificaciones sobre archivos compartidos [11].
- BoUML es una herramienta gratuita que te permite crear modelos de tres tipos: (1) diagramas de estructura (diagrama de clases, de objetos, de

componentes, de estructuras compuestas, de paquetes y de despliegue), (2) diagramas de comportamiento (diagramas de casos de uso, de actividad y de máquinas de estado) y (3) diagramas de interacción (diagramas de secuencia, de comunicación, de tiempos y de interacción) [12].

- Balsamiq es una herramienta de creación de maquetas. Se trata de una herramienta que crea capturas de pantalla esquematizadas imitando una web o aplicación de forma que te permite navegar por los diferentes diseños esquematizados de dicha web o aplicación.
- MySQL es un sistema de gestión de bases de datos de tipo relacional. Tiene licencia pública general y licencia comercial. Principalmente utilizado en el desarrollo de webs y es muy popular al tratarse de un sistema de código abierto [13].
- Apache es un servidor colaborativo de desarrollo de software. Su objetivo es crear un código fuente que sea robusto, comercializable, personalizable y de open source para servidores HTTP (web).
- El patrón de arquitectura Modelo-Vista-Controlador (MVC) separa los datos y la lógica de su representación en una aplicación. Para ello, MVC propone el modelo, la vista y el controlador; es decir, por un lado define componentes para la representación de la información y, por otro, para la interacción del usuario. Este patrón de arquitectura de software se basa en las ideas de reutilización de código y la separación de conceptos [14].

4.3.3 Otras tecnologías

- JSON, también conocido como *JavaScript Object Notation*, es un formato ligero de intercambio de datos. Está basado en un subconjunto del Lenguaje de Programación de JavaScript. JSON es un formato de texto que es completamente independiente del lenguaje, pero utiliza convenciones que son ampliamente conocidas por los programadores de la familia de lenguajes C. Se trata de un lenguaje ideal para el intercambio de datos.
- REST te da acceso a una lista de recursos que permite obtener información en función de las peticiones que se relizan. Sus opciones básicas son crear, encontrar, actualizar y borrar.
- HTTP (*Hypertext Transfer Protocol*) es un protocolo de nivel aplicación para sistemas de información. Es un protocolo genérico y sin estado que se puede usar para múltiples tareas, aparte de su uso para hipertexto [15]. Puede usarse, por ejemplo, con servidores de nombres y sistemas de gestión de objetos distribuidos, a través de la extensión de métodos de solicitud, códigos de error y encabezados.

5. Desarrollo de la solución propuesta

Se inicia el desarrollo de la solución propuesta mediante una serie de pasos. El primero de ellos se refiere a la parte lógica que se desarrolla, para el modelo MVC, en PHP. [16] En esta parte lógica se crean las clases con las cuales poder obtener los datos de la base de datos o del servidor REST.

A partir de la obtención de los datos se maqueta, se maqueta la capa de la interfaz para poder visualizarlos. El posicionamiento de los elementos, sin embargo, se presenta problemático y de difícil solución. Por ejemplo, conseguir que el tamaño de los márgenes superior e inferior de la aplicación se mantengan fijos independientemente del modo de visualización de nuestra interfaz web, ha resultado de difícil solución. Viendo la imposibilidad de mantener fijos dichos márgenes hemos optado por dar una dimensión fija a la altura del mapa.

En el diseño y creación de la interfaz surgió un segundo problema relacionado con la división de la interfaz en tres partes: encabezamiento, cuerpo y pie de página, precisamente porque, dependiendo de la página que se visualice, tiene que cargar unas librerías u otras. Este problema no se ha podido solucionar por el momento, pero pensamos seguir en esta investigación para futuro trabajos.

A continuación empezamos las pruebas de tiempos de respuesta; es decir, el tiempo que tarda la web en responder, lo cual depende, no sólo del tiempo de conexión a la API de Google, sino también al número y tamaño de las imágenes que introduzcamos, etc; también depende de las pruebas de visualización en diferentes dispositivos, tanto móviles como PCs. Asimismo, las pruebas de seguridad para comprobar que no existe ninguna vulnerabilidad que permita a una persona ajena al proyecto acceder a la parte privada de nuestra interfaz web y, en especial, a la base de datos. También hay que realizar pruebas de usuario; por ejemplo, se le da acceso a un usuario corriente y ajeno al proyecto de la interfaz web y se observa lo que ocurre. Tuvimos especial precaución en que la información llegara a los usuarios de forma legible y fiable.

En la creación de la interfaz web seguimos una arquitectura de desarrollo de software, llamada MVC, para separar la capa lógica de la aplicación de la capa visual. Hay que puntualizar que, además del uso de MVC, también utilizamos la arquitectura de cliente-servidor, la cual nos proporciona ciertas ventajas en la estructuración de la interfaz web, como por ejemplo, la integración entre diferentes sistemas compartiendo información. Favorece, asimismo, el uso de interfaces gráficas interactivas, siendo el resultado mucho más atractivo para el usuario.

En la imagen a continuación, en la Figura 22, podemos ver el esquema de directorios y ficheros de nuestra interfaz web, además de las conexiones que hacemos a distintos servidores para poder trabajar con la API de Google Maps, con el framework JQuery y con el framework Bootstrap [7] [17]. El esquema se puede dividir en tres bloques, el primero de los cuales se presenta en la siguiente figura:

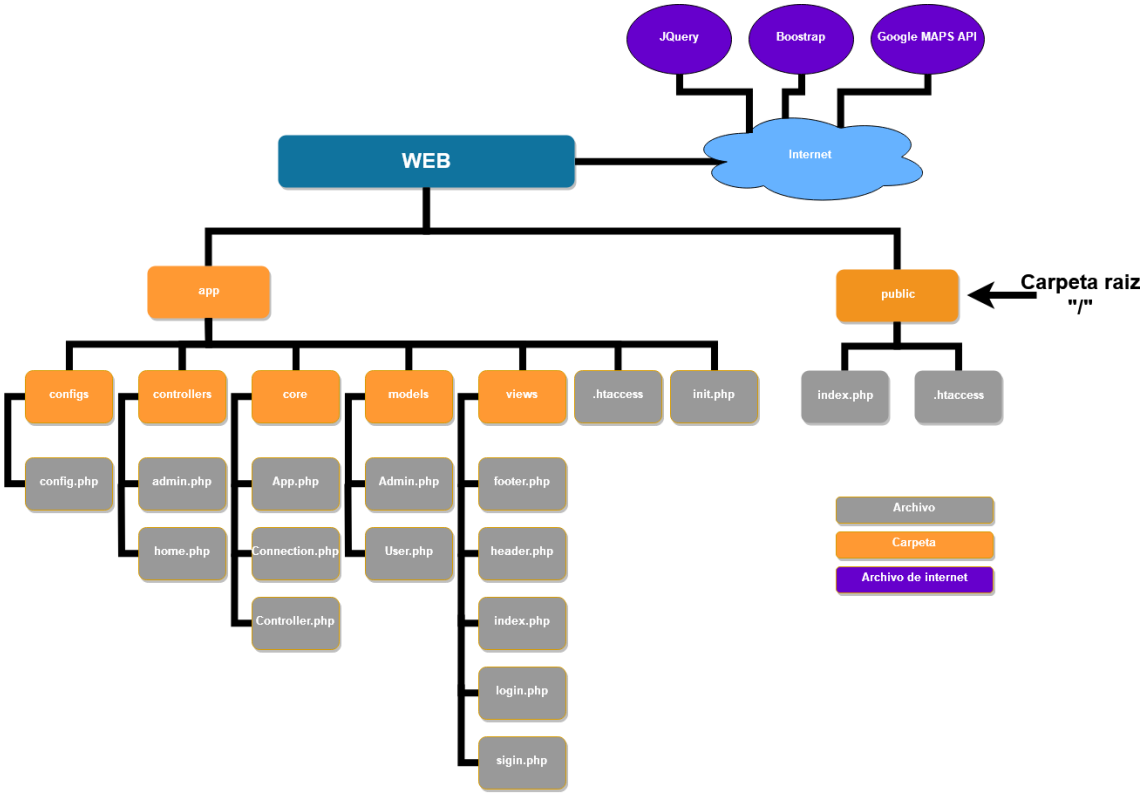


Figura 22. Diagrama de la estructura de ficheros del trabajo

Esta primera parte es la sección pública, como se muestra en la Figura 23, que está formada por dos ficheros: “index.php”, que sirve para iniciar la sesión de PHP e iniciar nuestra aplicación, y “.htaccess”, que se utiliza para transformar una petición de tipo GET, con variables incluidas, en una dirección legible y clara para el usuario[3] [18].

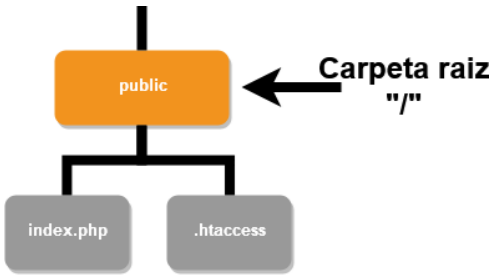


Figura 23. Diagrama de la carpeta raíz del servidor web

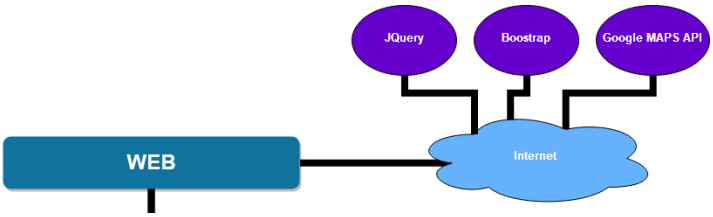


Figura 24. Diagrama de conexiones con APIs y frameworks.

[3] Un ejemplo de la función del fichero “.htaccess” sería: http://localhost/index.php¿url=pepe&producto=maria&color=verde que, gracias a este fichero, se traduce en http://localhost/pepe/maria/verde .

La segunda parte, mostrada en la Figura 24, es la encargada de hacer las conexiones a las APIs y a los frameworks que no están alojados en nuestro servidor. En este caso utilizamos la versión 3.4.1 de JQuery, la cual nos sirve para hacer peticiones, de tipo AJAX al servidor REST de ecoMobility, y para realizar efectos visuales relacionados con Bootstrap 4.3.1. La API de Google maps nos posiciona en el mapa los controladores de semáforos (CTLCs), los detectores de contaminación (detectores de CO2) y nos determina las rutas dependiendo de las restricciones de tráfico en tiempo real.

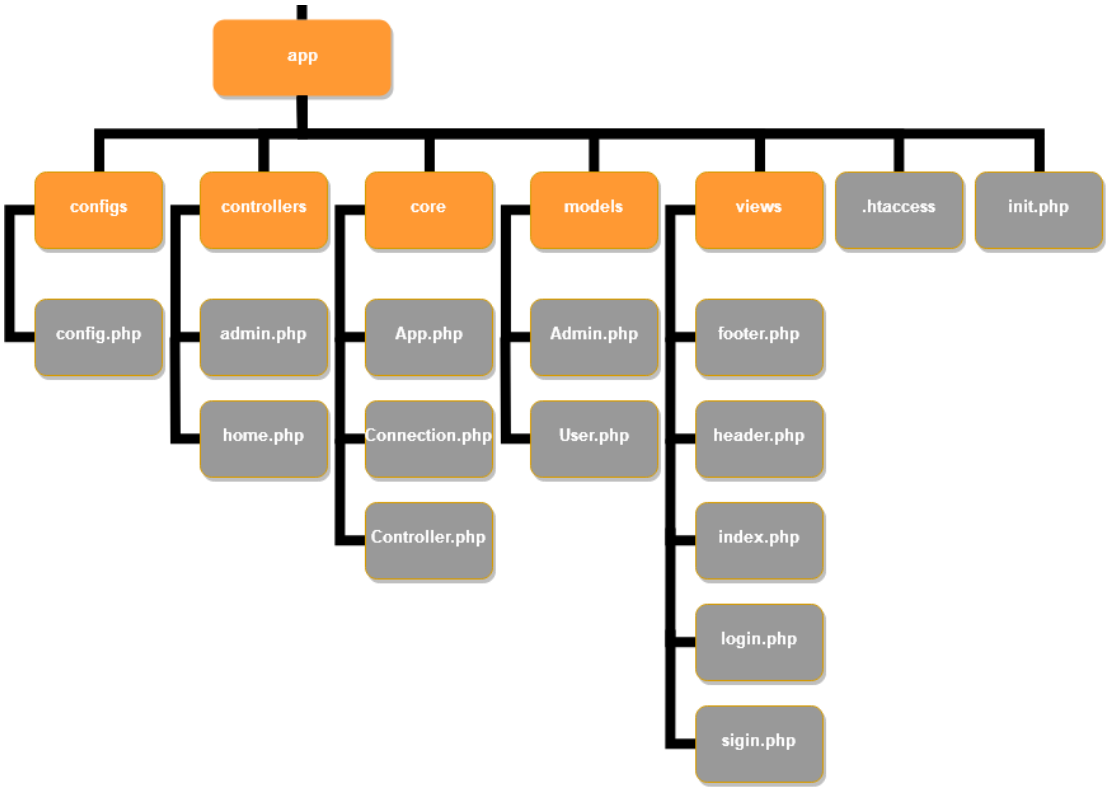


Figura 25. Diagrama de los ficheros y carpetas de la app

La tercera parte del esquema, en la imagen Figura 25, está distribuida en varias carpetas. Una carpeta principal, llamada “app”, tiene dos ficheros “.htaccess” e “init.php”. El fichero “.htaccess” impide que se indexen todas las carpetas y ficheros en caso de fallo de seguridad a la hora de suministrar recursos web. De no existir este archivo, el usuario podría acceder a determinados archivos y carpetas que queremos mantener ocultos. El fichero “init.php” carga el contenido de la subcarpeta “core”.

Dentro de la carpeta “app” encontramos cinco subcarpetas. La primera de estas subcarpetas es la llamada “core” que está compuesta por tres archivos: “App.php”, que carga la aplicación; “Controller.php”, que carga los controladores; y “Connections.php”, que establece la conexión con nuestra base de datos. La segunda subcarpeta es “configs”, que a su vez tiene un archivo, llamado “config.php”, que se encarga de la configuración y conexión a la base de datos. La tercera subcarpeta se llama “controllers” y está compuesta por dos archivos: “admin.php”, que se encarga de controlar las funciones que puede realizar el usuario administrador. El segundo archivo “home.php” se encarga de controlar

las acciones que pueden realizar los demás usuarios. La cuarta subcarpeta, llamada “models”, está compuesta por dos archivos: “Admin.php”, que se encarga de obtener los datos de la base de datos para los usuarios con permisos de administrador, y “User.php”, que se encarga de obtener los datos de la base de datos para los usuarios registrados. La quinta subcarpeta se llama “Views” y está compuesta por cinco archivos. El primero de estos archivos es “footer.php”, que se encarga de visualizar el pie de página de nuestra interfaz web, cargar la API de Google Maps y cargar scipts, en JavaScript, de Bootstrap (efectos visuales de la interfaz web). El segundo archivo se llama “header.php”, que se encarga de visualizar la cabecera de la página con el menú, cargar el framework de Bootstrap y cargar el framework de JQuery. El tercer archivo se llama “index.php” y se encarga de mostrar la página principal (figura 8) y las funciones del mapa: visualizar la información de los CTLCs, los detectores de CO_2 y las conexiones de los CTLCs. La mayor parte de este archivo “index.php” está escrito en JavaScript. El cuarto archivo es “login.php”, que se encarga de mostrar la página de acceso para usuarios ya registrados en nuestra base de datos. El quinto archivo es “signin.php”, que permite el registro a nuevos usuarios en nuestra base de datos.

6. Conclusiones

Unas de las dificultades con que se encuentra el proyecto ecoMobilty es la diseminación de la información con la política ambiental a cada uno de los ciudadanos con respecto a zonas de mayor o menor contaminación, qué sectores evitar. Para ello nos servimos de la inteligencia ambiental de modo que permita utilizar técnicas de computación para alcanzar soluciones viables. El presente trabajo tiene el objetivo principal agilizar esta información para que llegue al ciudadano de una forma eficiente. Para ello hemos creado una interfaz web mediante la cual poder alcanzar este objetivo.

Nos hemos servido de diversas técnicas para la realización de este trabajo, principalmente relacionadas con el lenguaje computacional web, diversas aplicaciones de software, así como combinándolas con los conocimientos adquiridos a lo largo de mis estudios. El principal escollo con el que nos hemos encontrado se relaciona con el recálculo de rutas para evitar zonas de tráfico restringido. Se ha intentado solucionar mediante asignaciones alternativas de puntos determinados fuera de las zonas de restricción. Si bien es verdad que con otras APIs sería factibles encontrar una solución, no obstante, nos hemos decidido por Google Maps por ser más intuitivo y amigable. Sin embargo, es un tema que merece ser estudiado más en profundidad y al que pensamos ampliarlo en un futuro.

Como hemos comentado anteriormente, nos encontramos con el problema de los márgenes superior e inferior de nuestra página principal. Siendo nuestra intención que fuera el mapa el que se adaptase al tamaño del dispositivo, hemos tenido que optar por un mapa de altura fija y márgenes superior e inferior adaptables según dispositivo del usuario. Los conocimientos adquiridos en las asignaturas de la rama de tecnologías de la información nos han facilitado la elección de realizar este proyecto mediante una interfaz web. Nos hemos centrado en la consistencia de la visualización de los datos de modo que esta información sea clara y diáfana para el usuario. En la integración de dicha información hemos ampliado nuestro conocimiento de JQuery aprendiendo a introducir información en un mapa de Google Maps. Además, hemos aprendido nuevos usos y aplicaciones de Bootstrap a la hora de esquematizar y organizar nuestra interfaz web. Otra técnica importante que hemos aprendido es el uso de MVC, facilitándonos la legibilidad de la información. Por último, pero no menos importante, hemos aprendido a implementar la tecnología REST en nuestro proyecto aprovechando su sistema de recursos.

La aventura de adentrarse en un tema tan complejo como todo lo relacionado con la educación medioambiental y el estudio previo sobre el tema, me ha ampliado enormemente los conocimientos no solo sobre las ciudades inteligentes, sino también mi preocupación por los problemas ambientales de hoy. Los conocimientos que hemos adquirido en el transcurso de este estudio y las soluciones alcanzadas en el mismo son un empuje el cual nos ha creado un interés adicional en la tecnología aplicada a las ciudades inteligentes.

7. Bibliografía

[1] D. Pedrós i Oliver, «Smart ecoMobility. Diseny d’una solución IoT basada en Fog/Edge computing aplicada a l’àmbit del trànsit sostenible a les ciutats intel•ligents,» 2018. [En línea]. Disponible: https://riunet.upv.es/handle/10251/107000.

[2] F. Martínez González, «Desarrollo de servicios IoT mediante IFTT. Aplicando los principos del Fog Computing al proyecto ecoMobility,» 2018. [En línea]. Disponible: https://riunet.upv.es/handle/10251/110994.

[3] A. Tallá i Vivó, «Diseny d’una solució basada en microserveis aplicada al Project ecoMobility,» 2018. [En línea]. Disponible: https://riunet.upv.es/handle/10251/110837.

[4] H. Porta Albentosa, «Smart ecoMobility: Movilidad sostenible en Ciudades Inteligentes,» 2018. [En línea]. Disponible: https://riunet.upv.es/handle/10251/111169.

[5] J. Madrigal García, «Desarrollo de un módulo de comunicaciones entre una ciudad y sus ciudadanos en el ámbito de la Internet de las Cosas. Aplicación práctica al proyecto de ecoMobility,» 2018. [En línea]. Disponible: Estudio accesible en https://riunet.upv.es/handle/10251/107286.

[6] W3 Consortium, «Leading the web to its full potential,» [En línea]. Disponible: https://www.w3.org. [Último acceso: 2019].

[7] D. DeBurr, Build Gamified Websites with PHP and jQuery, Packt Publishing, 2013.

[8] L. J. Mitchell, PHP Web Services, O’Reilly, 2013.

[9] W. Sanders, Learning PHP Design Patterns, O'Reilly, 2014.

[10] A. Del Sole, Visual Studio Code Succinctly, Syncfusion Inc., 2016.

[11] S. Chacon and B. Straub, Pro Git, Apress, 2014.

[12] K. Hamilton y M. R., Learning UML 2.0, O'Reilly, 2006.

[13] M. Delisle, Creating your MySQL Database: Practical Design Tips and Techniques., Packt Publishing, 2006.

[14] C. Pitt, Pro PHP MVC, Apress, 2012.

[15] P. Oppenheimer, Top-Down Network Design, Cisco Press, 2011.

[16] Codecourse, «Build a PHP MVC Application,» 2014. [En línea]. Disponible: https://www.youtube.com/user/phpacademy/search?query=Build+a+PHP+MVC+Application. [Último acceso: 2019].

[17] D. Cochran y I. Whitley, Bootstrap Site Blueprints, Packt Publishing, 2014.

[18] Apache Software Fundation, «Apache HTTP Server Documentation Version 2.4,» 2015. [En línea]. Disponible: https://archive.apache.org/dist/httpd/docs/httpd-docs-2.4.16.en.pdf.

8. Anexos

8.1 Instalación en un servidor Linux 18.04 la pila LAMP

La pila LAMP es un grupo de diferentes programas para habilitar a un servidor de servicios de páginas web dinámicas. Su término es el acrónimo de sistema operativo Linux, el servidor web Apache, los datos se almacenan en una base de datos MySQL, y el contenido dinámico es procesado mediante PHP.

Primero, instalamos Apache usando el administrador de paquetes, `apt`:

- `sudo apt update`
- `sudo apt install apache2`

Con el comando `sudo`, las operaciones son ejecutadas con los privilegios de superusuario.

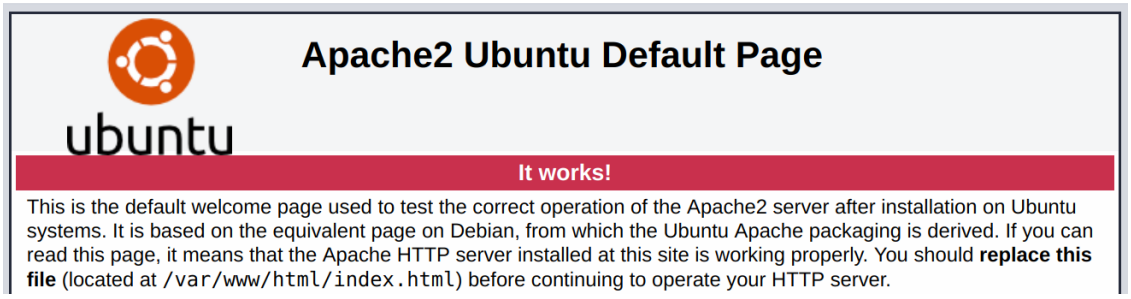
Para permitir el tráfico de entrada HTTP y HTTPS en el servidor se ejecuta el siguiente comando:

- `sudo ufw allow in "Apache Full"`

Escribiendo la dirección IP en el navegador de internet de la siguiente forma:

```
http://direccion_ip_servidor_apache
```

Entonces veremos la página web predeterminada de Apache:



Después de instalar el servidor web, instalamos MySQL. MySQL nos da acceso a las bases de datos donde guardamos información.

- `sudo apt install mysql-server`

Cuando la instalación esté completa, debemos ejecutar un archivo de comandos de seguridad que viene preinstalado con MySQL:

- `sudo mysql_secure_installation`

En este punto, él sistema de bases de datos se encuentra configurado y podemos seguir con la instalación de PHP, el componente final de la pila LAMP.

PHP es el componente de la configuración que procesa código para desplegar contenido dinámico. Puede ejecutar archivos, conectarse a las bases de datos MySQL para obtener información, y manejar la visualización del contenido procesado sobre el servidor web.

Para instalar PHP usamos el comando `apt`.

- `sudo apt install php libapache2-mod-php php-mysql`

A continuación, debemos reiniciar el servidor Apache para que los cambios sean reconocidos:

- `sudo systemctl restart apache2`

También podemos verificar el estado del servicio `apache2` utilizando `systemctl`:

- `sudo systemctl status apache2`

Para poder evaluar si el sistema está configurado de manera correcta para utilizar PHP, creamos un archivo llamado `info.php`.

- `sudo nano /var/www/html/info.php`

Esto crea un archivo en blanco, en el cual, escribimos el siguiente código PHP, dentro del archivo de texto:
info.php

```
<?php
phpinfo();
?>
```

Grabamos y cerramos.

Ahora ya puedes probar si tu servidor web se encuentra habilitado para desplegar correctamente contenido generado por este archivo de comandos PHP. Para hacerlo, visita una página web específica en tu navegador, necesitarás tu dirección pública de nuevo.

La dirección que deberás visitar es:

```
http://direccion_ip_servidor_apache/info.php
```

La página que debemos ver debe ser similar la siguiente:

PHP Version 7.2.3-1ubuntu1

System	Linux LAMP-1804-test 4.15.0-15-generic #16-Ubuntu SMP Wed Apr 4 13:58:14 UTC 2018 x86_64
Build Date	Mar 14 2018 22:03:58
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/7.2/apache2
Loaded Configuration File	/etc/php/7.2/apache2/php.ini
Scan this dir for additional .ini files	/etc/php/7.2/apache2/conf.d
Additional .ini files parsed	/etc/php/7.2/apache2/conf.d/10-mysqlnd.ini, /etc/php/7.2/apache2/conf.d/10-opcache.ini, /etc/php/7.2/apache2/conf.d/10-pdo.ini, /etc/php/7.2/apache2/conf.d/20-calendar.ini, /etc/php/7.2/apache2/conf.d/20-ctype.ini, /etc/php/7.2/apache2/conf.d/20-curl.ini, /etc/php/7.2/apache2/conf.d/20-exif.ini, /etc/php/7.2/apache2/conf.d/20-fileinfo.ini, /etc/php/7.2/apache2/conf.d/20-ftp.ini, /etc/php/7.2/apache2/conf.d/20-gd.ini, /etc/php/7.2/apache2/conf.d/20-gettext.ini, /etc/php/7.2/apache2/conf.d/20-iconv.ini, /etc/php/7.2/apache2/conf.d/20-intl.ini, /etc/php/7.2/apache2/conf.d/20-json.ini, /etc/php/7.2/apache2/conf.d/20-mysqli.ini, /etc/php/7.2/apache2/conf.d/20-pdo_mysql.ini, /etc/php/7.2/apache2/conf.d/20-phar.ini, /etc/php/7.2/apache2/conf.d/20-posix.ini, /etc/php/7.2/apache2/conf.d/20-readline.ini, /etc/php/7.2/apache2/conf.d/20-shmop.ini, /etc/php/7.2/apache2/conf.d/20-sockets.ini, /etc/php/7.2/apache2/conf.d/20-sysvmsg.ini, /etc/php/7.2/apache2/conf.d/20-sysvsem.ini, /etc/php/7.2/apache2/conf.d/20-sysvshm.ini, /etc/php/7.2/apache2/conf.d/20-tokenizer.ini, /etc/php/7.2/apache2/conf.d/20-xmlrpc.ini

9. Glosario de términos

Ajax: (Asynchronous JavaScript and XML) Técnica de desarrollo web para crear aplicaciones interactivas. Mantiene conexiones asíncronas con el servidor.

Android: Sistema Operativo de teléfonos inteligentes (Smartphones).

Apache: Servidor web.

API: Conjunto de funciones y procedimientos para desarrollar programas como una capa de abstracción.

Backup: Copia de seguridad de datos almacenada en un dispositivo o formato diferente al original.

BoUML: Programa de desarrollo de diagramas.

CTLC: (Control Traffic Light Communications) Controlador de luces de tráfico.

DELETE: Método REST para eliminar un recurso.

ECMAScript: Especificación de un lenguaje de programación.

Feedback: Reacción, respuesta u opinión que nos da un interlocutor.

Fi-Ware: Plataforma impulsada por la Unión Europea para el desarrollo y despliegue global de aplicaciones de Internet.

Framework: Conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática.

GET: Método REST para obtener un recurso.

GIT: Software de control de versiones.

Google Maps: API para desarrollo de mapas personalizados.

GPS: (Global Positioning System) Sistema de posicionamiento global.

HTTP: (Hypertext Transfer Protocol) Protocolo de transferencia de páginas web.

Login: Acción de iniciar sesión en una determinada aplicación.

MD5: Método de cifrado criptográfico de 128 bits.

Mockup: Maqueta o bosquejo de una interfaz.

MySQL: Sistema de base de datos relacional.

Open Source: Software cuyo código fuente es de dominio público.

PATCH: Método REST para modificar un recurso.

PC: (Personal Computer) Ordenador personal.

PHP: (Hypertext Preprocessor) Lenguaje parte servidor para el preprocesado de texto plano.

Portable: Software que no necesita de instalación para su manejo.

POST: Método REST para obtener un recurso.

PUT: Método REST para incluir un recurso.

REST: Interfaz de sistemas que utiliza HTTP para obtener datos en varios formatos.

Smartphone: Teléfono con sistema operativo que nos permite acceder a diversas utilidades.

Software: Conjunto de programas y rutinas que permiten a un ordenador realizar determinadas tareas.

Tablet: Dispositivo electrónico con pantalla más grande que un Smartphone.

User: Usuario de una determinada aplicación.