Universidad Autónoma de Barcelona

Cargar, exportar y visualizar datos mediante un visor web

Trabajo de Fin de Máster

Irene E. Cabrera Déniz

06/07/2018

Índice

1. Introducción

En el presente documento, se establecerá la estructura seguida durante el periodo de tiempo establecido para la realización del Trabajo de Fin de Máster. Este proyecto se ha realizado durante las prácticas del Máster en la empresa Solucions: Geoinformació i Territori, mediante las peticiones de la misma.

El seguimiento de este trabajo, se ha llevado a cabo por los dos tutores que han figurado durante las prácticas del Máster. Por parte de la empresa, se ha tenido contacto con José Ruiz y, por parte de la universidad con Aitor Alsina. Con el apoyo de estos tutores se han podido resolver dudas y concretar las peticiones de la empresa durante el desarrollo del proyecto.

El proyecto ofertado por la empresa, consiste en intentar estructurar un entorno web con un visor en forma de mapa para poder gestionar los datos sobre los comercios ubicados en Sabadell.

El objetivo final sería el de ofrecer los datos publicados para poder venderlos mediante una plataforma de pago online. Sin embargo, este objetivo se tratará como futurible, debido a la situación temporal disponible.

1.1. Objetivos

Los objetivos que se pretenden conseguir con este proyecto, se han decidido estructurar según el espacio temporal en el que se van a realizar. Por lo tanto, se ha dividido el objetivo principal que consiste en la creación de una página web con un visor asociado a una base de datos propia, entre varios pasos o actividades a realizar en cada etapa del proyecto. Estos son:

1. Creación de una base de datos que centralice la información de la empresa.

2. Importación automatizada de la información en la base de datos.

3. Visualización de los datos a través de un visor web con mapas.

4. Realizar busquedas por filtros en el visor web.

5, Exportación de los datos por parte de los usuarios del visor web.

A pesar de esta metodología de trabajo tan organizada, pueden darse lugar a una serie de contratiempos que se van a contemplar en cada uno de los siguientes apartados, en los que se pretende describir cada una de las actividades y sus respectivos pasos realizados durante la cronología de trabajo.

1.2. Cronología

Para la realización de este proyecto, se ha dividido entre 4 subgrupos de actividades que durante su desarrollo se puede entender de forma consecutiva. Para cada uno de los subgrupos se han definido una serie de objetivos mínimos,

deseables y en algunos casos de futuribles, de esta manera se pretende organizar por prioridades las actividades realizadas dentro del espacio temporal de cada subgrupo.

Como se comentaba en la introducción de este documento, este Trabajo de Fin de Máster se ha realizado en al mismo tiempo que se desarrollaban las prácticas del Máster. Estas prácticas han tenido una duración de dos meses, tomando como fecha de inicio el lunes 23 de Abril y finalizando el lunes 25 de Junio, siendo la duración total de 8 semanas.

En cada una de estas semanas se pretenderá completar una serie de objetivos mínimos y actividades, de esta manera se podrá completar el proyecto en el tiempo establecido. Además, para poder realizar un seguimiento del proyecto por parte de la empresa y tener la posibilidad de establecer cambios de cualquier apartado del proyecto, se han establecido reuniones cada dos semanas con el tutor de la empresa.

1.3. Plan de Trabajo

Desde un principio del proyecto, se han planteado una serie de directrices adecuadas para poder desarrollarlo. Este orden es importante, debido a que todos los objetivos planteados por la empresa están relacionados entre sí, es decir, que no se puede continuar el proyecto sin tener los objetivos por orden completados.

Por este motivo, el proyecto comienza con la creación de la base de datos, importa los datos a la misma y establecer la estructura adecuada. Continuando el proceso con el desarrollo del visor web, que abarca el diseño de la página, la visualización de los datos en el mapa y, la utilización de filtros para realizar búsquedas de establecimientos concretos.

2. Análisis de los Requerimientos

En este apartado se realiza un análisis de las peticiones realizadas por la empresa. Desde esta perspectiva se enfoca el proyecto para poder cumplir con las exigencias de la empresa. Para ello, se tendrán en cuenta las peticiones de la empresa y se buscara la manera más eficiente para cumplirlas en su totalidad.

Por otro lado, en el caso de que no se puedan cumplir algunas de las peticiones realizadas, se propondrán otras funcionalidades similares a los requerimientos establecidos. En este punto es importante la comunicación con la empresa mediante un documento de seguimiento del proyecto para que quede constancia de los posibles cambios ejecutados.

2.1. Características del Usuario

Al comienzo del proyecto, hay que tener en cuenta a que tipo de publico va dirigido el mismo, debido a que según el tipo de usuario que participe se podrán aplicar unas u otras funcionalidades. En este caso, podemos analizar dos modelos de usuarios, la empresa y los clientes de esta.

En este caso, la empresa esta interesada en reflejar los datos de los comercios de Sabadell de los que disponen para sus clientes. Debido a que, su actividad es vender dichos datos con la finalidad de ayudar a sus clientes en estudios de mercado e investigación respecto a la zona de la que se hace referencia, por eso es importante que en el visor se pueda reflejar los datos que consideren necesarios la empresa y ocultar al público los otros.

Por este motivo, se encargan de tener acceso al servidor para actualizar la base de datos cada cierto tiempo cuando consideren que ha cambiado la situación comercial del distrito de Sabadell que trabajan, realizar una selección adecuada de los datos que consideran apropiados para mostrarse en el visor en el apartado de descripción de cada uno de los establecimientos que componen la base de datos. o bien, modificar la distribución de los registros que componen la base de datos de esta.

Los clientes, por su lado, deben tener un pleno acceso a los datos que ofrece Solucions: Geoinformació i Territori. Puesto que, pueden utilizar estos datos para sus estudios de mercado o investigaciones, por ello, es importante que el visor ofrezca la posibilidad de realizar filtros por varias categorías, de esta manera se garantiza que cada cliente pueda obtener la información deseada según el sector, por ejemplo, de manera sencilla.

Una característica importante del visor para los clientes es que, cada establecimiento localizado en el mapa ofrece una breve descripción de los datos más importantes considerados por la empresa propulsora de este proyecto. Además, se ofrece la posibilidad desde el visor de descargar los datos en dos formatos distintos, facilitando de esta manera el trabajo del cliente.

2.2. Requerimientos Funcionales

En este apartado, se procede a describir las funcionalidades que debe cumplir el proyecto para alcanzar los objetivos que serán utilizados por los usuarios del mismo, sin tener en cuenta los procesos necesarios para cumplirlos.

En primer lugar, la empresa debe optar a la posibilidad de subir los registros de su base de datos en un formato cómodo para su elaboración, como puede ser el caso de un csv, de esta manera cualquier empleado de esta puede realizar la actualización de los datos que componen al visor sin tener conocimientos estrictos de programación.

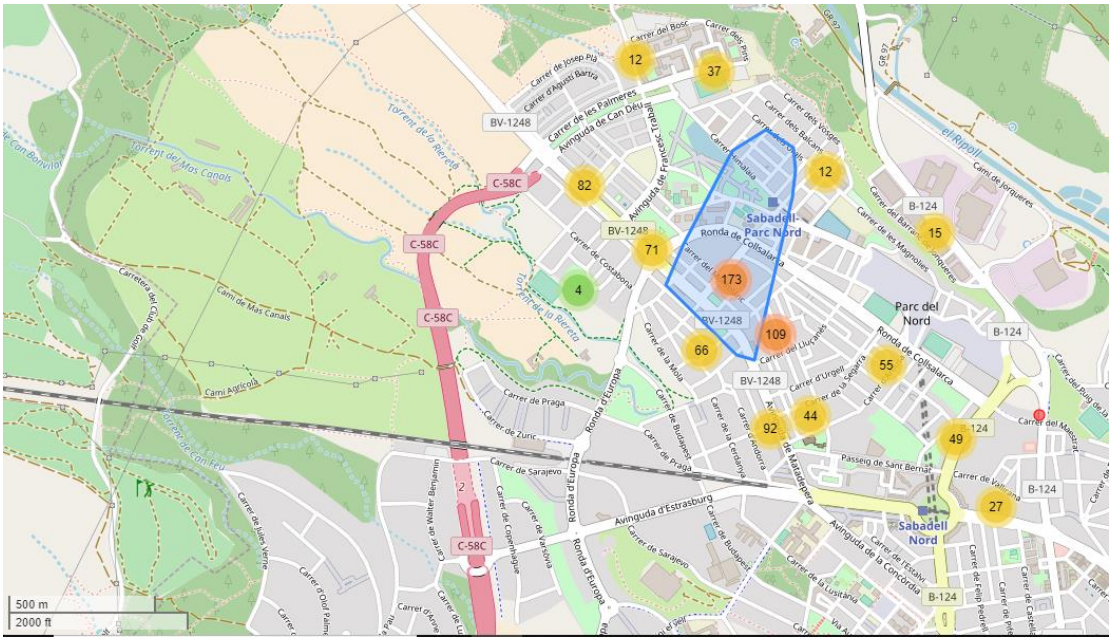
Mientras en el visor, debe cumplir una serie de requisitos mínimos en su diseño. Entre ellos consiste que capa base del mapa sea topográfico, debido a que puede facilitar al cliente localizarse en el territorio cuando realice una búsqueda, además, de contar con la posibilidad de realizar zoom en el mapa y contar con la escala en un lateral.

Por otro lado, es importante el tema de la simbología del visor, por ello, se ha establecido un color para cada uno de los números identificadores de sectore de cada establecimiento, este identificador ha sido creado por la empresa con el objetivo de poder realizar una agrupación de los datos más efectiva, quedando

como resultado cinco colores empleados que son: el uno que corresponde a Comerç al detall, con el color rojo; el dos que corresponde a Serveis, con el color verde; el tres que corresponde a Altres, con el color azul; el cuatro que corresponde a Sense información, con el color gris; y, el cinco que corresponde a En reforma.

También, para que visualmente al entrar en el visor sea más cómoda y limpio el mapa, se ha optado por implantar unos buffers para poder agrupar los datos según la cantidad de establecimientos que se encuentren ubicados en esa manzana. Dependiendo de la cantidad de establecimientos, también podrá varias el numero y el color del círculo, en este caso, emplea una gama de colores que van desde el rojo, pasando en orden de mayor a menos cantidad, por el amarillo y verde.

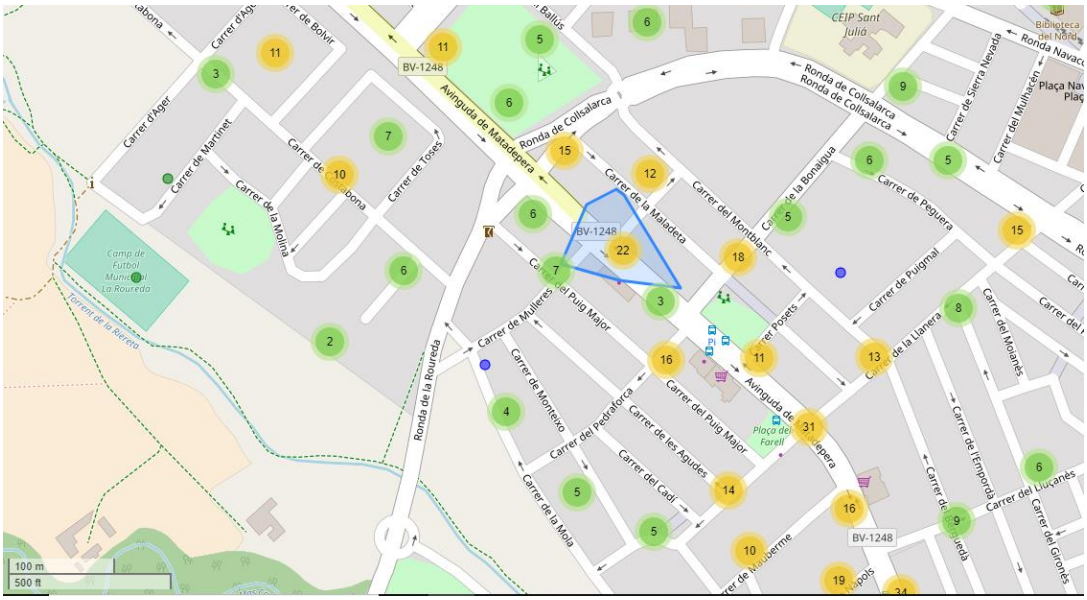
Imagen 1.



Fuente: Visor de elaboración propia.

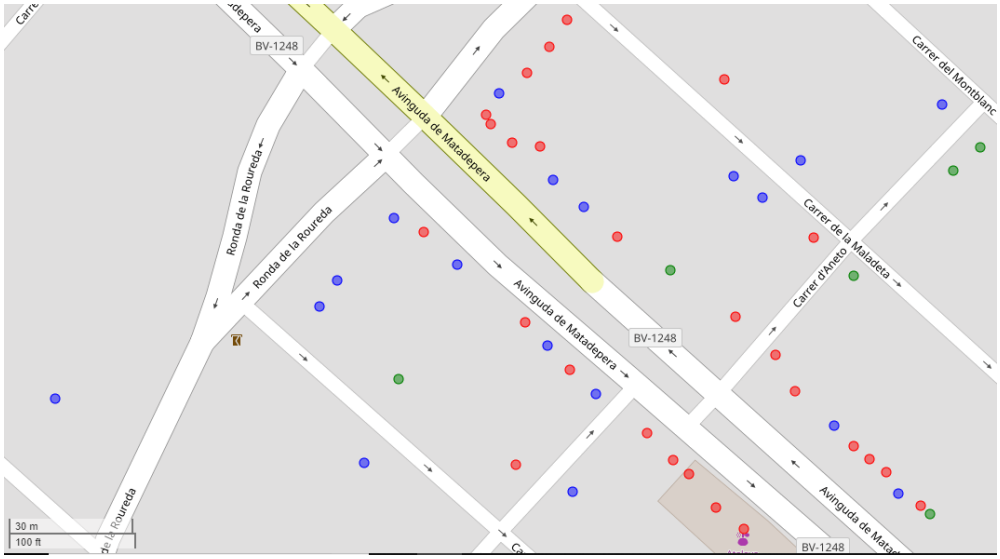
Al emplear el zoom, estos buffers irán cambiando su color y el numero que muestran dentro del círculo, debido a que mientras más se acerque el usuario menos puntos tendrán en su campo de visión.

Imagen 2.



Fuente: Visor de elaboración propia.

Imagen 3.



Fuente: Visor de elaboración propia.

La realización de búsqueda de establecimientos específicos por parte del usuario es importante para la finalidad del proyecto, por conclusión, el visor debe contener varias posibilidades de filtrado de la información que contiene la base de datos conectada a este para que el usuario pueda buscar de manera eficiente según Nom sector, Nom Grup Actividad, Situació (se trata de si el establecimiento se

encuentra en actividad comercial o no), Nom Actividad, Nom Carrer y Numero. Todos ellos, han sido considerados importantes para realizar un filtrado, debido a que describen el establecimiento en su totalidad.

Debido a su objetivo de acercar los datos de los establecimientos del distrito de Sabadell trabajado a sus clientes, para la realización de estudios de mercado en la zona entre otros, es importante facilitar a los usuarios los datos mostrados en el visor para realizar sus trabajos. Como solución a esta propuesta, se realiza una funcionalidad de exportación de los datos en varios formatos distintos, como son el csv y json, dando la posibilidad a los usuarios de no tener que estar en todo momento conectados al servidor.

2.3. Requerimientos No Funcionales

En este apartado se dispone a describir todas las herramientas o procesos utilizados para poder dar lugar a los requerimientos funcionales. En primer lugar, para realización y comprobación del proyecto, se procedió a instalar un servidor local en el portátil de trabajo donde se desarrollaba el proyecto. Esto es necesario para poder realizar modificaciones mientras y comprobaciones mientras se va desarrollando el código necesario para cumplir los objetivos propuestos.

Esta funcionalidad permitía el acceso al gestor de la base de datos Adminer, herramienta clave para poder importar la base de datos al servidor y poder emplearla posteriormente en la creación del visor. Además, permite el diseño de la estructura de la base de datos, un aspecto clave teniendo en cuenta que se dispone a realizar consultas a la base de datos. El hecho de que pueda permitir la lectura de un código php creado expresamente para importar la tabla proporcionada por la empresa en xml y, que posteriormente a sido transformada a csv.

Imagen 4.

Modify	ID	Nom_Establiment	ID_Situacio	Situacio	ID_Sector	Nom_Sector	ID_Grup_Activitat	Nom_grup_activitat
modificar	1	BBVA	1	Actiu	2	Serveis	11	Finances i assegurances
modificar	2	centro dental integral	1	Actiu	2	Serveis	15	Sanitat i assistència
modificar	3	ASSESSORIA ripoll	1	Actiu	2	Serveis	16	Altres
modificar	4	cafe_bar (café León)	1	Actiu	1	Comerç al detall	1	Quotidià alimentari
modificar	6	bollywood	1	Actiu	2	Serveis	16	Altres
modificar	7	roberto	1	Actiu	2	Serveis	16	Altres
modificar	8	PERRUQUERIA agadir	1	Actiu	2	Serveis	16	Altres
modificar	10	pa I café mimos	1	Actiu	1	Comerç al detall	1	Quotidià alimentari
modificar	14	dunay comestibles	1	Actiu	1	Comerç al detall	1	Quotidià alimentari
modificar	19	Remes recycling sl	1	Actiu	3	Altres	17	Altres
modificar	20	Ione computers	1	Actiu	1	Comerç al detall	5	Oci i cultura
modificar	21	tria	1	Actiu	2	Serveis	16	Altres
modificar	22	MINI market	1	Actiu	1	Comerç al detall	1	Quotidià alimentari
modificar	23	bar torreguitart	1	Actiu	1	Comerç al detall	1	Quotidià alimentari
modificar	24	ASSESSORIA consulting d'empreses 2011 sl	1	Actiu	2	Serveis	11	Finances i assegurances
modificar	30	Palma e hijos correduría de seguros sl	1	Actiu	2	Serveis	11	Finances i assegurances
modificar	32	vadepelo	1	Actiu	2	Serveis	16	Altres
modificar	33	tot fet	1	Actiu	2	Serveis	14	Restaurants, bars i hotels (Inclòs hostals, pensions i fonde
modificar	34	kbells	1	Actiu	2	Serveis	16	Altres
modificar	35	CONSTRUCCIONS losla sl	1	Actiu	2	Serveis	16	Altres
modificar	36	BON ambient	1	Actiu	2	Serveis	14	Restaurants, bars i hotels (Inclòs hostals, pensions i fonde
modificar	39	magin magin	1	Actiu	2	Serveis	16	Altres

Fuente: Gestor de Base de Datos Adminer.

A partir de este proceso, la creación del visor se realizó mediante la creación de un código html, asociado a otros dos códigos. Uno de ellos se trata de un código generado para crear un geojson, se optó por este formato de lectura de la base de datos, debido a que era más sencillo en su desarrollo y lectura por parte del html, hay que tener en cuenta en todo momento que la base de datos se compone de 1290 registros.

Al contar con el geojson para el código html, permite una visualización de la base de datos en el visor. A partir de este momento, la manera de representar los datos en el visor web, forma parte de los distintos componentes del código, en este caso se hace referencia a las distintas libreáis Leaflet que proporcionan lenguaje css y javascript, además de algunos componentes en php.

Es necesaria esta combinación de lenguajes, puesto que para poder realizar el apartado de filtros que se puede ejecutar en el visor y visualizar los resultados del mismo, es necesario emplear lenguaje php en un apartado del index.html para que pueda hacer llamadas y consultar al gestor de la base de datos mediante el geojson sin ningún problema.

Por otro lado, se utilizan diversos pluggins de la librería Leaflet aplicando lenguaje de css y javascript, para poder visualizar funciones más sencillas como

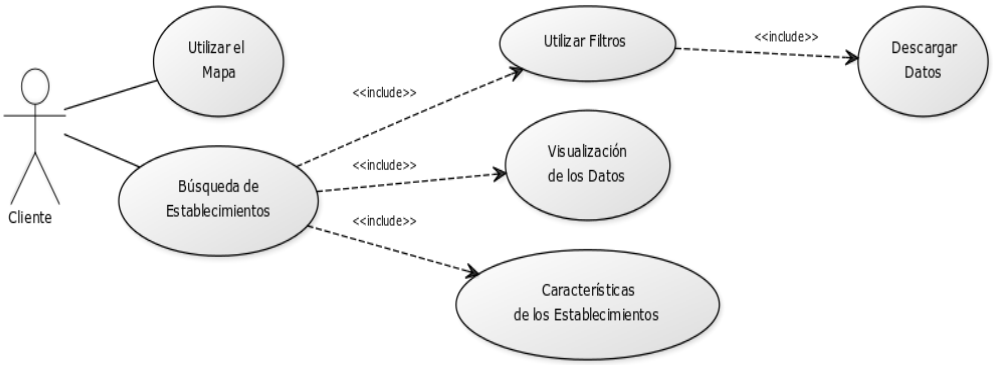
son el zoom, el mapa base y la escala que se pueden apreciar en el visor. Estas son necesarias para que la utilización del visor web sea más cómoda e intuitiva.

2.4. Casos de Uso

En este apartado, se describen mediante un diagrama los usos que pueden dar del proyecto tanto los clientes como la empresa Solucions: Geoinformació i Territori.

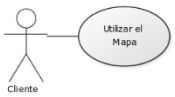
Por parte del cliente podemos encontrar los siguientes usos:

Imagen 5.



Fuente: Diagrama de elaboración propia, mediante la página web yuml.

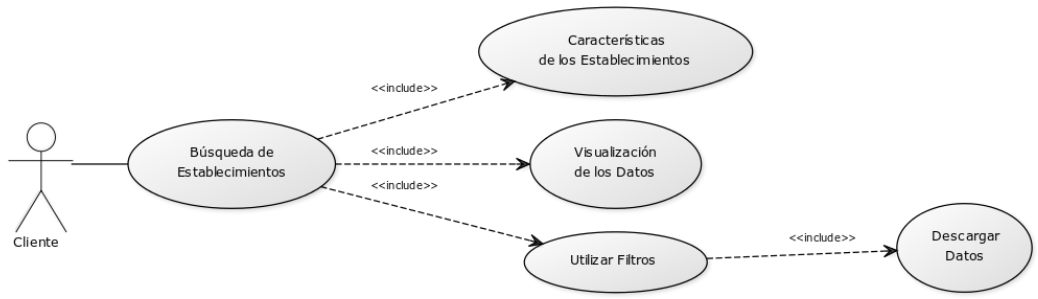
Tabla 1.

CASO DE USO: Utilizar el Mapa	
Autor/a	Irene Cabrera Déniz
Descripción	Mediante esta cartografía se podrá dar la posibilidad al usuario de poder navegar por el mapa para realizar una búsqueda visual. Además, presenta una serie de funcionalidades para facilitar su uso al usuario.
Actores	Clientes
Precondición	Aparece una cartografía base completa del municipio de Sabadell, centrada en el área de trabajo y, que permite interaccionar con ella.
Flujo Principal Cliente	1. El usuario visualiza el mapa y su toponimia 2. Permite el desplazamiento por la cartografía 3. El usuario puede modificar la escala al moverse por el visor
Flujos Alternativos	En el caso de que se aleje de la zona en la que está centrada el visor, se dejara de mostrar establecimientos pero, podrá seguir visualizando la cartografía base. Debido a la base de datos con la que cuenta el visor y a que no se han implantado pluggins limitantes.
Postcondición	Podrán pulsar en el mapa para hacer zoom y obtener una escala diferente. Presenta todos los establecimientos que abarca la base de datos sin ningún tipo de filtro al entrar en el visor. Además, presenta la toponimia de la zona.
Diagrama	

Fuente: Elaboración propia.

Tabla 2.

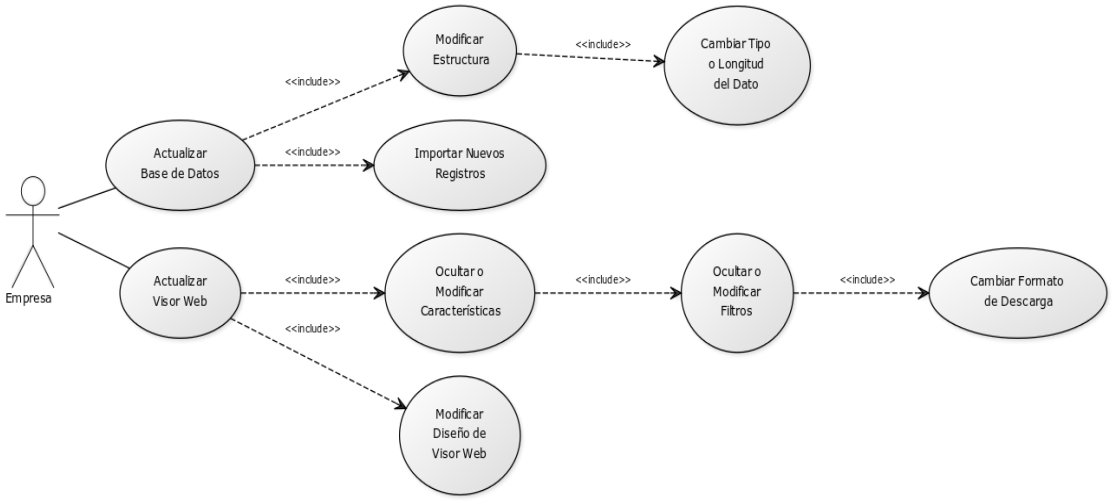
CASO DE USO: Búsqueda de Establecimientos	
Autor/a	Irene Cabrera Déniz
Descripción	El visor web, además de la cartografía inicial, dispone de un apartado configurado por unos filtros de búsqueda. A partir de estas pestañas los usuarios podrán seleccionar diversas características que presentan los establecimientos agrupados en la base de datos, para poder realizar una selección más efectiva según sus preferencias. De esta manera, dan lugar a otras acciones secundarias que consisten en la visualización de los establecimientos y sus características en el mapa, de los comercios que cumplen con los filtros aplicados anteriormente. Además, estos datos pueden ser descargados por el usuario para poder trabajar con ellos en sus estudios sin tener la necesidad de conectarse al visor en todo momento.
Actores	Clientes
Precondición	En la parte inferior derecha del visor, aparece un amplio listado con las diversas características a partir de las que se pueden buscar comercios, debido a que son las mismas que presentan algunos comercios y otros no. Por este motivo, se realiza la búsqueda de manera concreta.
Flujo Principal Cliente	1. El usuario selecciona entre las opciones ofrecidas en las pestañas de los filtros. 2. El motor de condición valida la opción escogida y da diferentes alternativas que contienen el texto escrito. 3. Los elementos que responden a la descripción aparecen repartidos por el mapa, con su respectiva descripción.

	4. El usuario pulsa sobre el símbolo del elemento deseado para poder obtener una descripción del establecimiento más detallada. 5. Mediante los botones, localizados en el apartado de búsqueda por filtros de descarga, el usuario podrá descargar la información de los elementos buscados en dos formatos: csv y geojson.
Flujos Alternativos	Los establecimientos no seleccionados mediante los filtros aplicados, directamente no aparecerán en el visor, de esta manera se pretende evitar confusiones entre los distintos tipos de establecimientos para el usuario. Por otra parte, los establecimientos seleccionados no sufrirán ningún tipo de modificación de su localización, simbología o descripción detallada.
Postcondición	Se origina una distribución en el mapa del visor exclusiva con los establecimientos solicitados, permitiendo la facilidad para trabajar con los datos y localizarse en la realidad. Además, sí los usuarios lo consideran oportuno, podrán solicitar la descarga de los datos en dos formatos diferentes, que son el csv y el jgson.
Diagrama	

Fuente: Elaboración propia.

Mientras, por parte de la empresa, los siguientes:

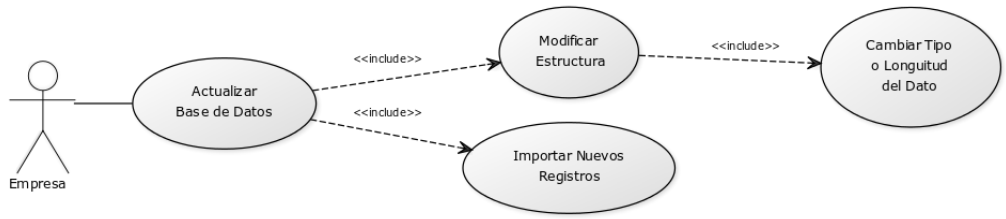
Imagen 6.



Fuente: Diagrama de elaboración propia, mediante la página web yuml.

Tabla 3.

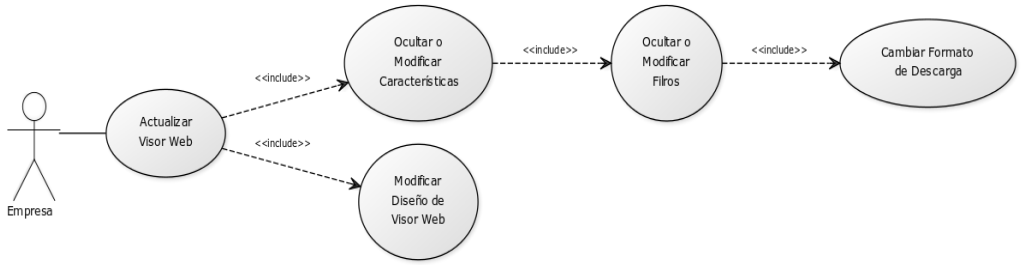
CASO DE USO: Actualizar Base de Datos	
Autor/a	Irene Cabrera Déniz
Descripción	Consiste en la posibilidad de la que disponen los empleados de la empresa para poder aportar la información recaudada mediante el trabajo de campo al visor web. Es el factor principal para el funcionamiento del visor, debido a que sin datos el visor no podrá mostrar ningún tipo de información. Facilita a la empresa herramientas para poder importar datos y que sean visibles en el visor web en cualquier momento, y la posibilidad de su actualización en el futuro.
Actores	Empresa
Precondición	Se debe generar la estructura de la base de datos correctamente, para que no presente errores al importar los datos obtenidos en el trabajo de campo y acumulados en una tabla Excel. Por este motivo en el gestor de la base de datos, deben existir los mismos nombres en las columnas y las mismas características en los registros, que presentan en la tabla de Excel original.
Flujo Principal Empresa	1. Creación de la estructura de la base de datos en el gestor. 2. Modificación de las características de los registros incluidos en la base de datos. 3. Ampliación o actualización de la base de datos mediante la importación de los datos.
Flujos Alternativos	Se podrá modificar en cualquier momento, tanto la estructura del gestor de la base de datos, como el código php creado para actualizar los datos.

	Debido a que, puede existir la posibilidad de que en un futuro la empresa considere que debe aportar más datos a sus clientes mediante el visor.
Postcondición	Generar un código php concreto que permita la importación de datos nuevos para poder actualizar la base de datos y, con ello el visor, sin que sea necesario tener amplios conocimientos de programación por parte del operario.
Diagrama	

Fuente: Elaboración propia.

Tabla 4.

CASO DE USO: Actualizar Visor Web	
Autor/a	Irene Cabrera Déniz
Descripción	Podrán realizar una serie de modificaciones en la visualización y funcionalidades del visor, debido a los códigos generados para su creación. Mediante estos, podrán modificar el diseño del visor, con elementos como a cartografía, la posición de los botones o la simbología. Respecto a la visualización de los datos en el visor web, podrán gestionar las características de los establecimientos que querrán que se visualicen y cuáles no. También, en el apartado de búsqueda por filtros podrán realizar esta modificación para poder ver los nuevos datos.
Actores	Empresa
Precondición	Mediante la variación de los elementos agrupados en el código se podrán realizar las modificaciones. Teniendo en cuenta que según la modificación que queramos realizar, se utilizará leguaje java o php.
Flujo Principal Empresa	1. Modificar el diseño del visor web, según las nuevas funcionalidades que quieran aportar a sus clientes. 2. Decidir las características de cada establecimiento que deseen mostrar en el visor web. 3. Seleccionar los registros que componen los filtros de búsqueda, según hayan sido modificados en la base de datos asociada al visor. 4. Permitir la descarga de los datos mostrados en el visor. Puede llegar a modificar el formato de descarga.
Flujos Alternativos	Ofrece la capacidad de modificar formatos de descarga de los datos, para que los clientes

	puedan trabajar con ellos sin necesidad de conectarse al visor. Proporciona la posibilidad de modificar los datos y la visualización de los mismos, en lo que a características de los establecimientos se refiere.
Postcondición	Permite la posibilidad de ampliar los datos que existen actualmente, teniendo en cuenta las modificaciones realizadas en la base de datos asociada al visor. De esta manera, se logra que los usuarios del visor web tengan acceso a esos nuevos datos y puedan trabajar con ellos.
Diagrama	

Fuente: Elaboración propia.

2.5. Soluciones Existentes

Los objetivos implantados por la empresa para la finalidad del proyecto son bastante concretos y están relacionados unos con otros, por este motivo el desarrollo de estos es muy similar entre ellos.

En primer lugar, para el correcto desarrollo del proyecto, se debe trabajar la base de datos facilitada por la empresa que encarga el proyecto. comenzando con el tratamiento de los datos, es decir, cambiando su formato a csv y corrigiendo errores de escritura que puedan dar problemas al subirlo al gestor de datos.

Además, se procede a la creación de la estructura adecuada en el gestor de base de datos, estableciendo el nombre de las columnas, el tipo de datos y la longitud de los mismos. Se trata de un pao importante, debido a que según como sea la estructura de nuestra base de datos se podrán o no importar los datos correctamente y, posteriormente, afectará en la visualización del contenido.

Es imprescindible, por no decir que el proyecto consiste en esto, aplicar unos conocimientos de programación para poder realizar diversos códigos. Cada uno de estos códigos está desarrollado con un enfoque concreto y algunos se llaman entre ellos para mostrar registros de la Base de Datos.

En conclusión, existen cuatro códigos: el responsable de importar adecuadamente todos los registros que componen la base de datos entregada por parte de la empresa, en archivo xml y transformada a csv, al gestor de base de datos Adminer; el código encargado de transformar estos datos importados en csv a exportarlos en geojson, para que sea posible su posterior visualización en el visor sin problemas; el propio código encargado de desarrollar el visor web, en este caso contiene todas las funcionalidades necesarias para cumplir las peticiones de la empresa; en último lugar, se ha desarrollado un código específico para que el usuario del visor web pueda exportar los datos que considere necesarios.

3. Metodología

En este apartado, se describe los pasos seguidos para la realización del proyecto y los distintos procesos que se han tenido que llevar a cabo para ello. Debido a la correlación entre sí de los objetivos marcados por la empresa, se han establecido una serie de etapas para desarrollar las actividades a realizar.

3.1. Etapas del proyecto

Para cumplir los objetivos propuestos para este Trabajo de Fin de Máster, se ha debido dividir las tareas a realizar en distintas etapas. De esta manera, puede seguirse un orden dentro del proyecto para garantizar el correcto desarrollo del proyecto. Además, de que cada actividad esta correlacionada con la siguiente, es decir, sin llevar a cabo la estructura de la base de datos, por ejemplo, no se puede proceder a la importación de los datos.

En los siguientes sub apartados del documento, se procede a la descripción de los pasos realizados para el desarrollo del proyecto, completando las actividades marcadas para cumplir con los objetivos nombrados en el apartado de introducción.

3.1.1 Definición de la Base de Datos

El resultado final, debe mostrar al usuario del visor la información correspondiente de los establecimientos localizados en el municipio de Sabadell. Por este motivo, se debe realizar una estructura adecuada de los datos proporcionados por la empresa. Estos datos han sido obtenidos por la empresa Solucions: Geoinformació i Territori, que ha recopilado una tabla Excel compuesta por una serie de registros con información importante para sus clientes acerca de los establecimientos de la zona de estudio mediante trabajo de campo.

Se trata de una estructura sencilla, debido a la cantidad de atributos que componen la tabla que se pueden observar en las columnas. Al rellenar estas columnas obtendremos un total de 1920 registros, que corresponde a los atributos

que se poseen sobre los establecimientos. Algunos de ellos, son un código id, el nombre del establecimiento, la dirección, las coordenadas entre otros datos.

Imagen 7.

ID	Nom Establiment	ID Situació	Situació	ID Sector	Nom Sector	ID Grup Activitat	Nom Grup Activitat	ID Activitat	Nom Activitat
1	BBVA	1	Actiu	2	Serveis	11	Finances i assegurances	11000	Finances i assegurances
2	centro dental integral	1	Actiu	2	Serveis	15	Sanitat i assistència	15000	Sanitat i assistència
3	ASSESSORIA ripoll	1	Actiu	2	Serveis	16	Altres	16004	Serveis a les empreses i oficin
4	cafe_bar (café León)	1	Actiu	1	Comerç al detall	1	Quotidià alimentari	1001	Begudes
6	bollywood	1	Actiu	2	Serveis	16	Altres	16007	Centres d'estètica
7	roberto	1	Actiu	2	Serveis	16	Altres	16006	Perruqueries
8	PERRUQUERIA agadir	1	Actiu	2	Serveis	16	Altres	16006	Perruqueries
9	clínica dental Dr. SANCHEZ Val	0	Inactiu	4	Sense informació	-	-	-	-
10	pa I café mimos	1	Actiu	1	Comerç al detall	1	Quotidià alimentari	1006	Pa, pastisseria i làctics
11	SN	0	Inactiu	2	Locals buits en lloguer	-	-	-	-
12	CARPI pizza	0	Inactiu	4	Sense informació	-	-	-	-
13	Carlos stylo BARBER shop	0	Inactiu	2	Locals buits en lloguer	-	-	-	-
14	dunay comestibles	1	Actiu	1	Comerç al detall	1	Quotidià alimentari	1000	Resta alimentació
16	SN	0	Inactiu	1	Locals buits en venda	-	-	-	-
17	autoescuela trebol	0	Inactiu	4	Sense informació	-	-	-	-
18	tecni_glass sbd,sl	0	Inactiu	4	Sense informació	-	-	-	-
19	Remes recycling sl	1	Actiu	3	Altres	17	Altres	17000	Altres
20	lone computers	1	Actiu	1	Comerç al detall	5	Oci i cultura	5000	Informàtica
21	tria	1	Actiu	2	Serveis	16	Altres	16112	Altres
22	MINI market	1	Actiu	1	Comerç al detall	1	Quotidià alimentari	1000	Resta alimentació
23	bar torreguitart	1	Actiu	1	Comerç al detall	1	Quotidià alimentari	1001	Begudes
24	ASSESSORIA consulting d'empi	1	Actiu	2	Serveis	11	Finances i assegurances	11000	Finances i assegurances
25	inmuebles 2008	0	Inactiu	4	Sense informació	-	-	-	-
26	PARKING inmuebles	0	Inactiu	4	Sense informació	-	-	-	-
27	J.j orval. puertas automáticas	0	Inactiu	4	Sense informació	-	-	-	-
28	el periodico	0	Inactiu	5	En reforma	-	-	-	-
29	pulidos y abrillantados alicatado	0	Inactiu	4	Sense informació	-	-	-	-
30	Palma e hijos corredurí a de segi	1	Actiu	2	Serveis	11	Finances i assegurances	11000	Finances i assegurances
31	parking\ magenes	0	Inactiu	2	Locals buits en lloguer	-	-	-	-

Fuente: Excel generado por la empresa Solucions: Geoinformació i Territori.

A partir del Excel inicial proporcionado por la empresa, se dispone a modificar su formato para que sea csv. Es elegido este formato debido a que las modificaciones que se realicen en este formato, podrán ser importados y almacenados en el servidor mediante un código php creado para esta función. Además, este código ha tenido que presentar una condición concreta para poder leer sin problemas los acentos que existen en los nombres de los establecimientos y las calles, debido a que por defecto los lee de manera incorrecta y podría generar confusión a los visitantes del visor. Imágenes del código php en el Anexo.

Sin embargo, y debido a que los datos han sido recogidos mediante trabajo de campo, existen datos relacionados con la ubicación de los establecimientos que han dado problemas, debido a escribir mal la dirección, dando lugar a problemas de localización que se han podido solucionar al emplear las coordenadas geográficas para la geo localización de los establecimientos en el mapa del visor web.

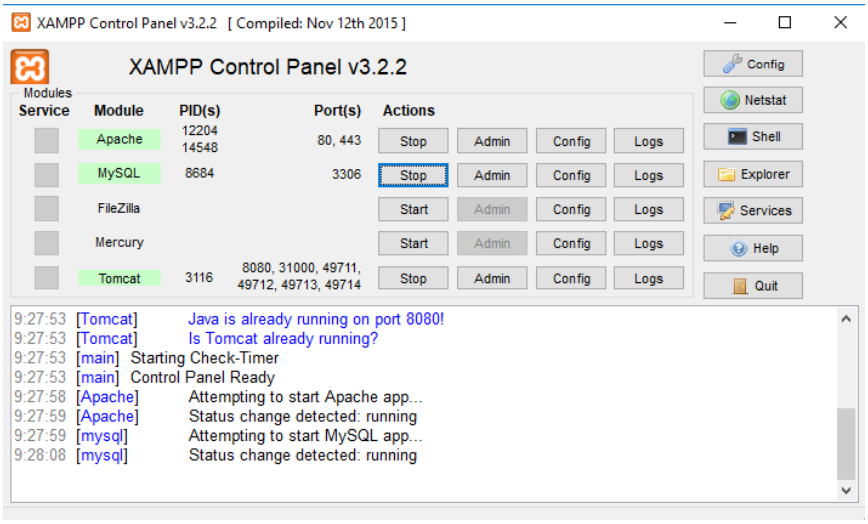
Tras la realización de estas tareas y corrección de errores nombrados mediante la utilización del código y, para poder visualizar posteriormente la base

de datos en el visor, en primer lugar, se debe importar los datos a un servidor. En este proyecto se ha empleado el servidor local, que se utilizará en un portátil personal, mediante la descarga del programa XAMPP que contiene MySQL, Apache y Tomcat. De esta manera, se garantiza el correcto funcionamiento del proyecto y permite la pre visualización, herramienta útil para corregir errores del código desde el punto de vista del usuario.

Aunque, posteriormente se sube el visor y la base de datos creada al propio servidor de la empresa y crearán una dirección concreta desde la página de la empresa para que los usuarios puedan acceder al visor y utilizarlo con mayor comodidad.

Este proceso, cosiste en cambiar simplemente en cambiar la dirección del localhost utilizado hasta el momento, sustituyéndola por la nueva dirección proporcionada por Solucions: Geoinformació i Territori.

Imagen 8.



Fuente: Programa XAMPP.

Mediante las herramientas mostradas en la imagen anterior, se podrá importar los datos aportados por la empresa, al gestor de base de datos llamado Adminer. Desde el Adminer podremos establecer la estructura de la base de datos, para ello usaremos de referencia el archivo csv creado anteriormente. Además de emplear los mismos nombres en las columnas para que no generar errores en el

momento de la importación, es importante determinar el formato adecuado de los registros que corresponderá la tabla.

También, en la estructura de la base de datos alojada en el gestor Adminer, se debe determinar las características y la longitud de los registros que van a ser importados a las misma. Para esto, se debe tener en cuenta los datos que se presentan en la tabla Excel original aportada por la empresa, pues según el número y tipo de caracteres que tengamos en la misma.

Se trata de un punto de este objetivo importante, debido a que según estas características y el tipo de lectura que se determine desde el gestor Adminer, determinará sí en el visor web se podrán o no, leer correctamente los registros mostrados desde la base de datos.

Imagen 9.

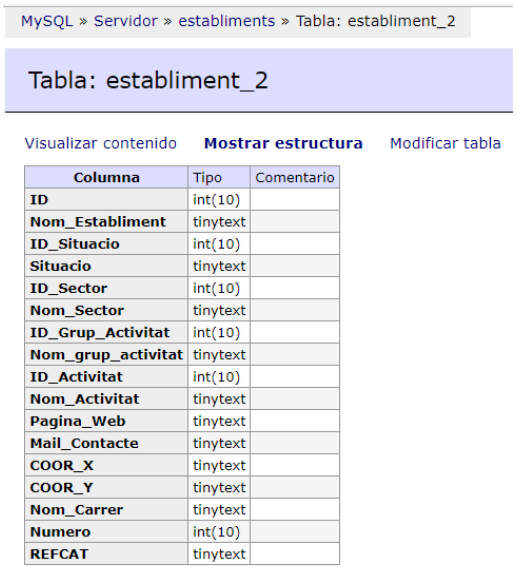

MySQL » Servidor » establiments » Tabla: establiment_2

Tabla: establiment_2

Visualizar contenido **Mostrar estructura** Modificar tabla

Columna	Tipo	Comentario
ID	int(10)	
Nom_Establiment	tinytext	
ID_Situacio	int(10)	
Situacio	tinytext	
ID_Sector	int(10)	
Nom_Sector	tinytext	
ID_Grup_Activitat	int(10)	
Nom_grup_activitat	tinytext	
ID_Activitat	int(10)	
Nom_Activitat	tinytext	
Pagina_Web	tinytext	
Mail_Contacte	tinytext	
COOR_X	tinytext	
COOR_Y	tinytext	
Nom_Carrer	tinytext	
Numero	int(10)	
REFCAT	tinytext	

Fuente: Gestor de Base de Datos Adminer.

Tras realizar estas modificaciones, se podrá emplear el código php creado anteriormente para importar los datos aportados por la empresa Solucions: Geoinformació i Territori y, que posteriormente se podrán visualizar en un visor mediante otra serie de códigos.

Al tener asociado el csv original a este código php, cada vez que se actualice la tabla de datos y se ejecute el código, la base de datos que contiene el servidor

será actualizada automáticamente sin necesidad de que el usuario tenga conocimientos de programación. Objetivo importante, entre los establecidos por la empresa desde el comienzo del proyecto, debido a que no cuentan con personal con conocimientos específicos de programación.

3.1.2. Página de Importación

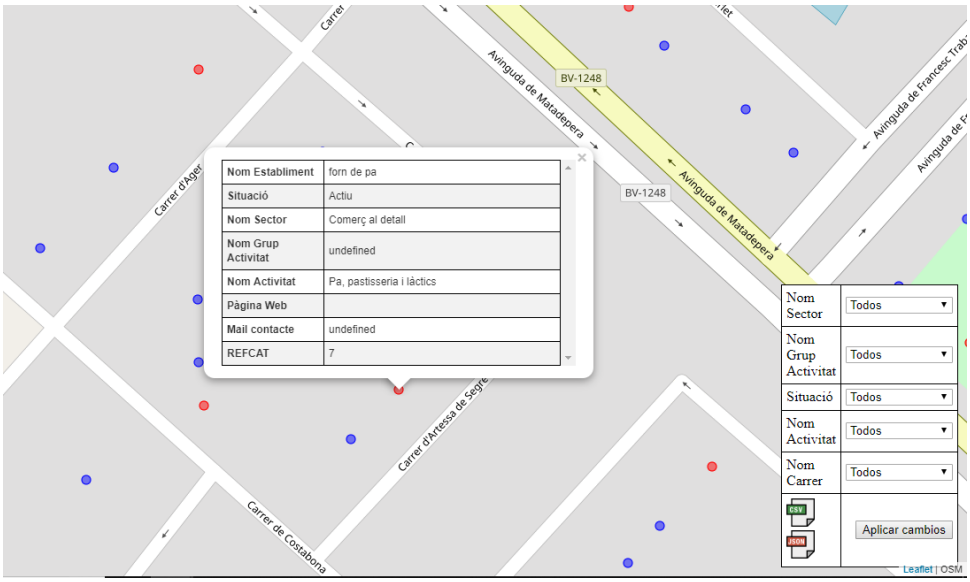
En este apartado, se pretende lograr que, mediante la página web creada para alojar el visor, también sea capaz de cargar los registros actuales o actualizados de la base de datos en formato csv.

Para lograr este objetivo, en primer lugar, se establece la base de datos en un formato jgson para poder facilitar el proceso de localización de los datos en el visor creado posteriormente. Se lleva a cabo el siguiente código php, que nos permite visualizar los datos jgson, se compone de la estructura presentada en el apartado de Anexos.

Este código permite poder organizar los datos dentro del visor según las peticiones de la empresa, es decir, podemos gestionar que estén correctamente colocadas en el mapa por las coordenadas de las que se disponen de cada establecimiento. También, permite añadir los atributos disponibles para describir cada establecimiento recogidos en la base de datos.

Por otro lado, debido a la sencillez de su estructura, está preparado para que cualquier empleado de la empresa con unos conocimientos mínimos, pueda actualizar en un futuro los registros que componen el jgson mediante la simple actualización, tanto de los códigos, como de la base de datos en formato de csv.

Imagen 10.



Fuente: Visor de elaboración propia.

Una vez creado este código, se preparó el visor y para ello, hay que utilizar las librerías de Leaflet y lenguaje php para poder crear todas las funciones solicitadas por la empresa. Muchas de estas funcionalidades, están relacionadas con la búsqueda o agrupación de los datos acogidos en el código jgson descrito en este apartado, por lo tanto, tiene una fuerte relación ambos apartados.

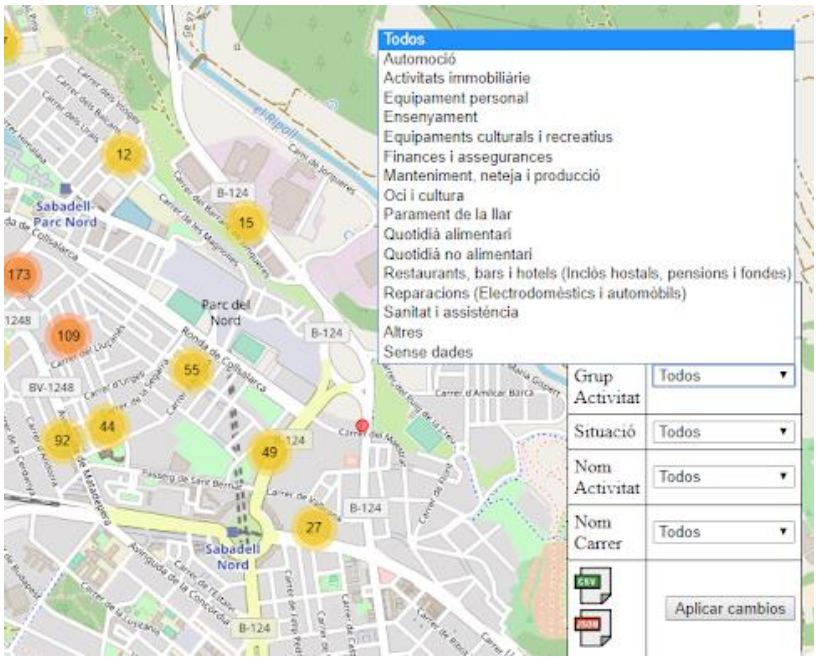
3.1.3. Visor mapa y filtros

Tras realizar el proceso de crear la base de datos e importar los datos al servidor, comienza el proceso de diseño y creación del visor web. Para llevar a cabo este proceso se debe tener en cuenta una serie de peticiones realizadas por la empresa, debido a que establecen un diseño personalizado del servidor que consideran apropiado para representar los datos.

Entre estas peticiones tenemos en cuenta, poder realizar búsquedas de los datos según la situación de uso actual del establecimiento (sí está activo o no), el nombre del sector comercial al que pertenece, el grupo de actividad al que pertenece y, por último, el nombre de la actividad que desarrolla. Este proceso en concreto se realizó con lenguaje php, debido a que sí se empleaba librerías de Leaflet, en el momento de generar la posibilidad de exportar datos daba una serie

de errores, dando lugar a un código combinado entre lenguaje php y librerías Leaflet.

Imagen 11.

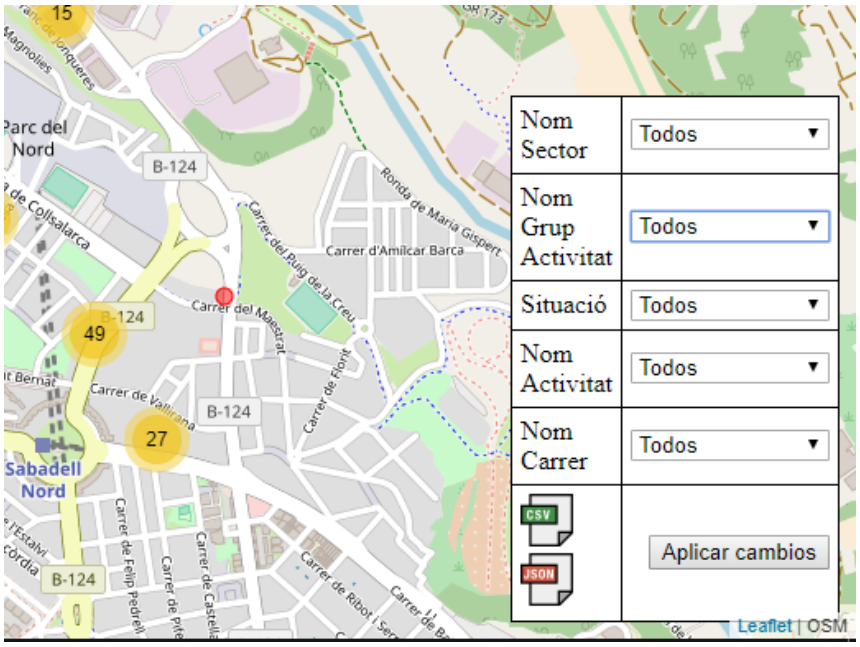


Fuente: Visor de elaboración propia.

Por otro lado, el mapa debe contar con una simbología concreta de puntos, los cuales se diferencian por colores según el ID Sector que ha establecido la empresa a cada comercio según su actividad comercial, según a cuál pertenecen podemos diferenciar cinco colores distintos. Con este diseño, se considera que facilita la asociación de establecimientos y la actividad que realiza.

Además, el mapa cuenta con una serie de características de diseño propias, como puede ser la agrupación de los puntos según la escala a la que se encuentre el usuario. De esta manera, se pretende facilitar la localización del usuario del visor mientras lo utiliza cuando no busca datos muy concretos por sectores o actividad, simplemente por agrupación en una calle, zona o manzana.

Imagen 12.

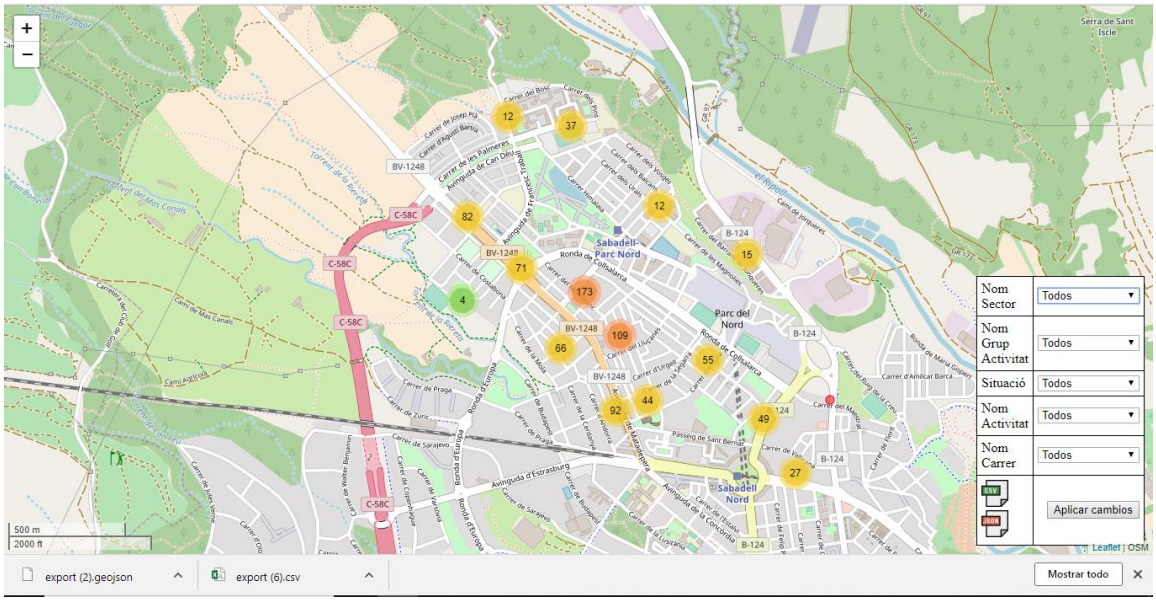


Fuente: Visor de elaboración propia.

3.1.4. Exportación de los datos

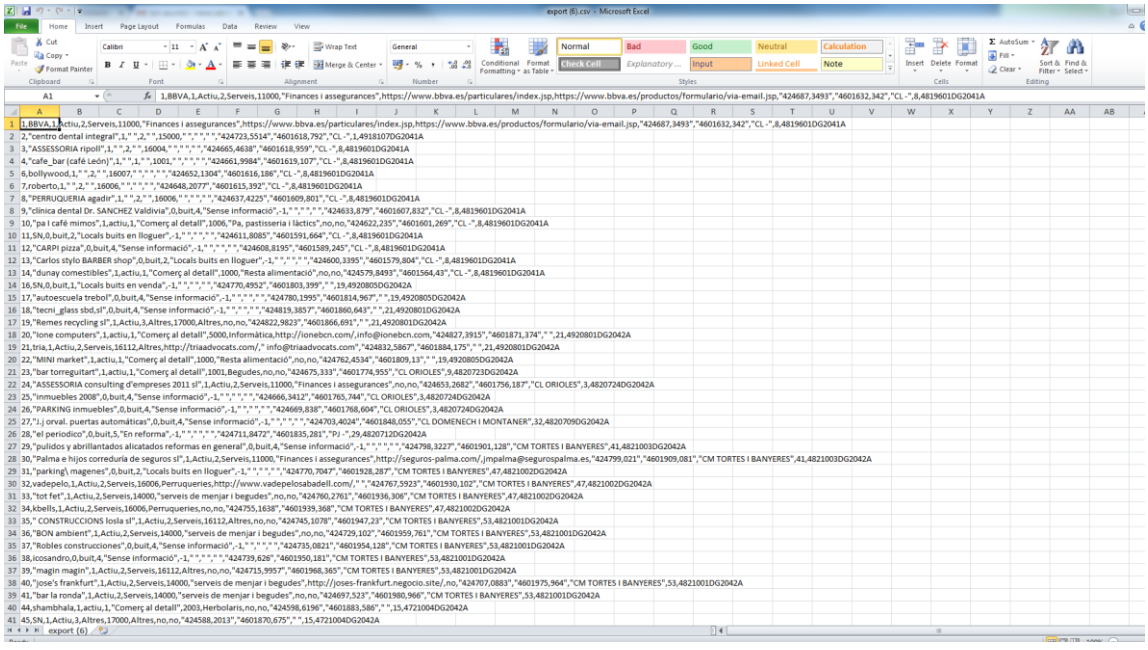
Debido a una de las peticiones realizadas por la empresa, se ha tenido que establecer un código a parte de los dos descritos en los apartados anteriores (los usados para importar la base de datos proporcionada por la empresa al servidor y el necesario para crear un visor para visualizar dichos datos) destinado a permitir que los usuarios que realicen una búsqueda en el visor puedan descargar los datos en formato csv o geojson.

Imagen 13.



Fuente: Visor de elaboración propia.

Imagen 14.



Fuente: Programa Excel del paquete Office 2010.

Respecto al código, se ha empleado lenguaje php debido a que las librerías de Leaflet, daba una serie de errores al intentar establecer formatos concretos de descarga de la información determinada por los filtros. Se ha tratado de uno de los apartados que mayores dificultades ha presentado, debido a que durante el

proceso de creación la empresa decidió realizar una serie de mejoras en base de datos original, generando nuevas columnas con sus respectivos registros.

El código permite descargar todo tipo de combinaciones de búsquedas posibles desde los filtros del visor, además, se consideran apropiados estos formatos para este tipo de datos, debido a que son compatibles con varios programas para poder visualizarlos correctamente. El código está presente en el apartado Anexos de este documento.

4. Resultados

Los resultados que se pretendían conseguir en un comienzo han sido bastante ambiciosos, sin embargo, y tras varias pruebas con distintos leguajes para el visor web y cambios de estructura de la base de datos, se ha podido concluir el proyecto con éxito.

Es cierto, que los objetivos futurables que planteo en u comienzo no se han podido lograr, debido a que se tratan objetivos más relacionados con el comercio o la seguridad de los datos. Aspectos que a pesar de investigar sobre el tema, no ha podido ser desarrollados finalmente.

Sin embargo, los objetivos principales y algunos específicos han sido cumplidos con un éxito bastante positivo y productivo para la finalidad del visor web, que consiste en acercar los datos acerca de establecimientos de Sabadell por parte de la empresa a sus clientes.

Considerando todo lo expuesto a lo largo del documento y, concretamente en este sincero apartado, se obtiene una valoración positiva de los resultados del proyecto debido a todas las funcionalidades que cumple el visor web y, que resulta posible actualizar la base de datos, y por lo tanto el visor, por parte de la empresa sin tener que contar con ninguna persona especialmente cualificada en programación.

5. Conclusiones

Al terminar el proyecto, llega el análisis personal y profesional del resultado obtenido. En este caso concreto, considero que el trabajo realizado durante horas de búsqueda de código, lenguajes de programación y, posteriores pruebas de los mismos, no se encuentran reflejadas en estas páginas ni en el resultado final.

Esto ocurre porque han sido circunstancias necesarias para llegar a la solución y, por lo tanto, el resultado positivo que se puede mostrar y que está utilizando actualmente la empresa como una herramienta más de relación con sus clientes. Por otro lado, el resultado de estas horas y errores cometidos, también conducen al propio aprendizaje y acercamiento de conceptos más concretos de vistos durante ellas clases del máster, o bien, aspectos completamente nuevos.

Teniendo en cuenta estos aspectos, es compresible que no se hayan podido cumplir los plazos cronológicos establecidos al comienzo de la realización del proyecto, provocando así una enorme dificultad del seguimiento continuo del proyecto por parte de los agentes terceros. Sin embargo, a cada pequeño avance se procuraba mantener informados a todos los agentes implicados, debido a que ha sido necesario el apoyo y resolución de dudas en algunas de las ocasiones de alguno de los agentes.

A pesar de todo esto, considero positivo el resultado obtenido, debido a que cumple los objetivos iniciales y pequeñas peticiones de diseño que se fueron puliendo a lo largo del desarrollo del trabajo, además, de poder reforzar capacidades de resolutivas acerca de la rama de programación. Es cierto que me hubiera gustado no tener que llegar a tantos errores para poder concluir con este proceso pero, dadas las circunstancias y los resultados estoy satisfecha.

6. Anexo

Código para Importar los Datos al Adminer

```php
<!DOCTYPE HTML>
<html>
<body>
<?php
$servername = "localhost";
$username = "phpuser";
$password = "RBGDPFmLq2uUN5tT";
$dbname = "establiments";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

// Read csv
$fila = 0;
$n_inserts = 0;
$n_insert_errors = 0;
$n_updates = 0;
$n_update_errors = 0;
if (($gestor = fopen("datos.csv", "r")) !== FALSE) {
    while (($datos = fgetcsv($gestor, 1000, ";")) !== FALSE) {

        if ($fila != 0) {

            $id = $datos[0];
            $nom_establiment = mysqli_real_escape_string($conn, $datos[1]);
            $idprincipa = $datos[2];
            $nom_princi = mysqli_real_escape_string($conn, $datos[3]);
            $idsector = $datos[4];
            $desc_secto = mysqli_real_escape_string($conn, $datos[5]);
            $idactivita = mysqli_real_escape_string($conn, $datos[6]);
            $n_grupact = mysqli_real_escape_string($conn, $datos[7]);
            $web = mysqli_real_escape_string($conn, $datos[8]);
```

PHP Hypertext Preprocessor file | length : 3.393 lines : 97 | Ln : 1 Col : 16 Sel : 0 | 0 | Windows (CR LF) | UTF-8 | INS

```php
        if ($fila != 0) {

            $id = $datos[0];
            $nom_establiment = mysqli_real_escape_string($conn, $datos[1]);
            $idprincipa = $datos[2];
            $nom_princi = mysqli_real_escape_string($conn, $datos[3]);
            $idsector = $datos[4];
            $desc_secto = mysqli_real_escape_string($conn, $datos[5]);
            $idactivita = mysqli_real_escape_string($conn, $datos[6]);
            $n_grupact = mysqli_real_escape_string($conn, $datos[7]);
            $web = mysqli_real_escape_string($conn, $datos[8]);
            $mail = mysqli_real_escape_string($conn, $datos[9]);
            $COORX = mysqli_real_escape_string($conn, $datos[10]);
            $COORY = mysqli_real_escape_string($conn, $datos[11]);
            $CARRER = mysqli_real_escape_string($conn, $datos[12]);
            $NUMERO = $datos[13];
            $REFCAT = mysqli_real_escape_string($conn, $datos[14]);

            // Check if ID exists
            $sql = "SELECT id FROM establiment WHERE id='" . $id;
            $result = $conn->query($sql);
            if ($result->num_rows == 0) {

                // ID does not exist. Insert row
                $sql = "INSERT INTO establiment (id, nom_establiment, idprincipa, nom_princi, idsector, desc_secto, idactivita, n_grupact, web, mail, COORX, COO
                VALUES (" . $id . ", '" . $nom_establiment . "', " . $idprincipa . ", '" . $nom_princi . "', " . $COORX . ", " . $COORY . ", '" . $CARRER . "',
                if ($conn->query($sql) === TRUE) {
                    $n_inserts++;
                } else {
                    $n_insert_errors++;
                    echo 'Error al insertar el ID=' . $datos[0] . '<br>';
                }

            } else if ($result->num_rows == 1) {

                // ID exists. Update row
```

PHP Hypertext Preprocessor file | length : 3.393 lines : 97 | Ln : 1 Col : 16 Sel : 0 | 0 | Windows (CR LF) | UTF-8 | INS

```
                // ID exists. Update row

                $sql = "UPDATE establiment SET nom_establiment='" . $nom_establiment . "', nom_princi='" . $nom_princi . "', COORX='" . $COORX . "', COORY='" .
                if ($conn->query($sql) === TRUE) {
                    $n_updates++;
                } else {
                    $n_update_errors++;
                    echo 'Error al actualizar el ID=' . $datos[0] . '<br>';
                }

            } else {

                echo 'Error: Hay más de un ID=' . $id . ' en la base de datos.<br>';
            }



        }
        $fila++;
    }
}
fclose($gestor);

// Close connection
$conn->close();


echo '<br>Registros insertados: ' . $n_inserts;
echo '<br>Errores al insertar: ' . $n_insert_errors;

echo '<br>Registros actualizados: ' . $n_updates;
echo '<br>Errores al actualizar: ' . $n_update_errors;
?>
</body>
</html>
```

PHP Hypertext Preprocessor file | length : 3.393 lines : 97 | Ln : 1 Col : 16 Sel : 0 | 0 | Windows (CR LF) | UTF-8 | INS

Código de Geojson

```
<?php
$mysqli = new mysqli("localhost", "phpuser", "RBGDPFmLq2uUN5tT", "establiments");

if ($mysqli->connect_errno)
{
    printf("Falló la conexión: %s\n", $mysqli->connect_error);
    exit();
}
//---------- variable para saber si es exportar o solo consulta del mapa
if( isset($_GET['guardar']) && $_GET['guardar']=="true" )
{
    $filename = "export";
    header('Content-Type: application/geojson');
    header("Content-Disposition: attachment; filename={$filename}.geojson");
    header("Pragma: no-cache");
    header("Expires: 0");
}
//-----------------------------------------------------------------------
$condicionesConsulta = array();
if( isset($_GET['idsector']) && $_GET['idsector']!="" )
{
    $condicionesConsulta[] = 'idsector = '.$_GET['idsector'];
}
if( isset($_GET['nom_princi']) && $_GET['nom_princi']!="" )
{
    $condicionesConsulta[] = 'nom_princi LIKE "%'.$_GET['nom_princi'].'%" ';
}
if( isset($_GET['n_grupact']) && $_GET['n_grupact']!="" )
{
    $condicionesConsulta[] = 'n_grupact LIKE "%'.$_GET['n_grupact'].'%" ';
}

$where = "";
if( !empty($condicionesConsulta) )
{
    $where = " WHERE ".join($condicionesConsulta,' AND ');
}
//echo "SELECT * FROM establiment ".$where.'<br>';
```

PHP Hypertext Preprocessor file | length : 2.495 lines : 100 | Ln : 1 Col : 1 Sel : 0 | 0 | Windows (CR LF) | UTF-8 | INS

```php
//echo "SELECT * FROM establiment ".$where.'<br>';
if ($resultado = $mysqli->query("SELECT * FROM establiment ".$where))
{
        if($resultado->num_rows>0)
        {
            echo '{'.
                    '"type": "FeatureCollection",'.
                    '"name": "establi",'.
                    '"crs": {'.
                      '"type": "name",'.
                      '"properties": {'.
                        '"name": "urn:ogc:def:crs:EPSG::25831"'.
                      '}'.
                    '},'.
                    '"features": [';

            $rows = array();
            $n = 1;
            while($r = mysqli_fetch_assoc($resultado))
            {
                $r['OBJECTID'] = $n;
                $rows[] = '{ '.
                            '"type": "Feature",'.
                            '"properties": '.
                                json_encode(cadena_utf8($r)).
                            ','.
                            '"geometry": '.
                              '{ '.
                                '"type": "Point",'.
                                '"coordinates": ['.str_replace(',', '.', $r['COORX']).', '.str_replace(',', '.', $r['COORY']).'] '.
                              '} '.
                            '}';
                $n++;
            }
            echo join($rows,',');

            echo ']'.
                '}';
```



```php
                            '"geometry": '.
                              '{ '.
                                '"type": "Point",'.
                                '"coordinates": ['.str_replace(',', '.', $r['COORX']).', '.str_replace(',', '.', $r['COORY']).'] '.
                              '} '.
                            '}';
                $n++;
            }
            echo join($rows,',');

            echo ']'.
                '}';

            /* terminar con la base de datos */
            $resultado->close();
        }
        else
        {
            printf('Consulta sin resutado');
        }

}
else
{
    printf('Error en la consulta');
}

function cadena_utf8($listado=array())
{
    $nuevo = array();
    foreach ($listado as $key => $value)
    {
        $nuevo[$key] = utf8_encode($value);
    }
    return $nuevo;
}
?>
```

Código para exportar datos

```php
<?php
$mysqli = new mysqli("localhost", "phpuser", "RBGDPFmLq2uUN5tT", "establiments");

if ($mysqli->connect_errno)
{
    printf("Falló la conexión: %s\n", $mysqli->connect_error);
    exit();
}

$condicionesConsulta = array();
if( isset($_GET['idsector']) && $_GET['idsector']!="" )
{
    $condicionesConsulta[] = 'idsector = '.$_GET['idsector'];
}
if( isset($_GET['nom_princi']) && $_GET['nom_princi']!="" )
{
    $condicionesConsulta[] = 'nom_princi LIKE "%'.$_GET['nom_princi'].'%" ';
}
if( isset($_GET['n_grupact']) && $_GET['n_grupact']!="" )
{
    $condicionesConsulta[] = 'n_grupact LIKE "%'.$_GET['n_grupact'].'%" ';
}

$where = "";
if( !empty($condicionesConsulta) )
{
    $where = " WHERE ".join($condicionesConsulta,' AND ');
}
//echo "SELECT * FROM establiment ".$where.'<br>';
if ($resultado = $mysqli->query("SELECT * FROM establiment ".$where))
{
        if($resultado->num_rows>0)
        {
                $filename = "export";
                header("Content-type: text/csv");
                header("Content-Disposition: attachment; filename={$filename}.csv");
                header("Pragma: no-cache");
                header("Expires: 0");
```

PHP Hypertext Preprocessor file | length : 1.654 lines : 71 | Ln : 1 Col : 1 Sel : 0 | 0 | Windows (CR LF) | UTF-8 | INS

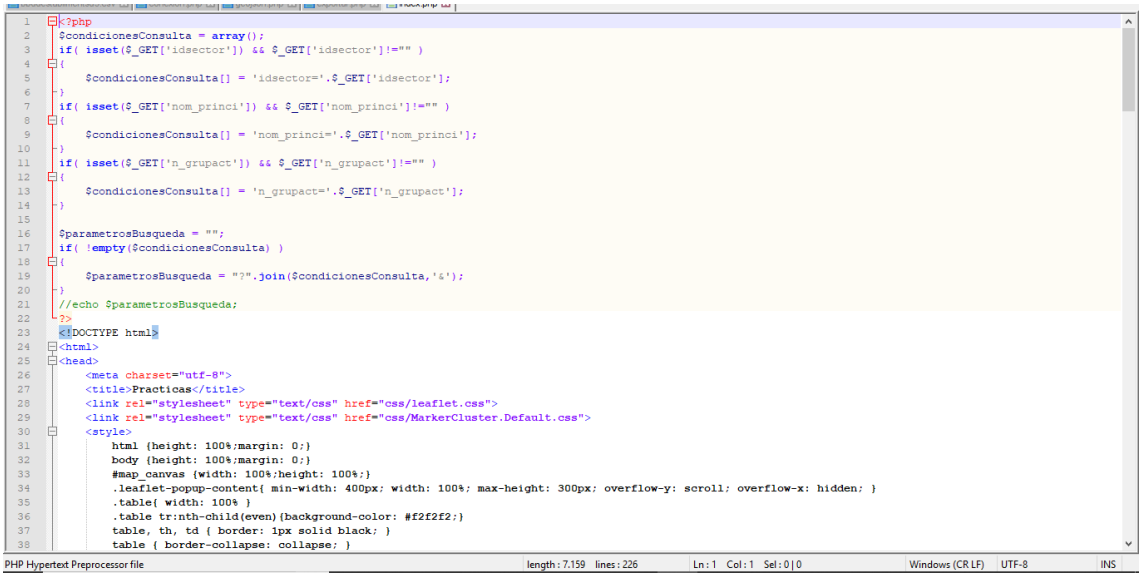
```php
//echo "SELECT * FROM establiment ".$where.'<br>';
if ($resultado = $mysqli->query("SELECT * FROM establiment ".$where))
{
        if($resultado->num_rows>0)
        {
                $filename = "export";
                header("Content-type: text/csv");
                header("Content-Disposition: attachment; filename={$filename}.csv");
                header("Pragma: no-cache");
                header("Expires: 0");
                outputCSV($resultado);
                $resultado->close();
        }
        else
        {
            printf('Consulta sin resutado');
        }

}
else
{
    printf('Error en la consulta');
}

function cadena_utf8($listado=array())
{
    $nuevo = array();
    foreach ($listado as $key => $value)
    {
        $nuevo[$key] = utf8_encode($value);
    }
    return $nuevo;
}
function outputCSV($data)
{
    $outputBuffer = fopen("php://output", 'w');
    while($val = mysqli_fetch_assoc($data))
    {
```

PHP Hypertext Preprocessor file | length : 1.654 lines : 71 | Ln : 1 Col : 1 Sel : 0 | 0 | Windows (CR LF) | UTF-8 | INS

```php
}
else
{
    printf('Error en la consulta');
}

function cadena_utf8($listado=array())
{
    $nuevo = array();
    foreach ($listado as $key => $value)
    {
        $nuevo[$key] = utf8_encode($value);
    }
    return $nuevo;
}
function outputCSV($data)
{
    $outputBuffer = fopen("php://output", 'w');
    while($val = mysqli_fetch_assoc($data))
    {
        fputcsv($outputBuffer, $val);
    }
    fclose($outputBuffer);
}
?>
```

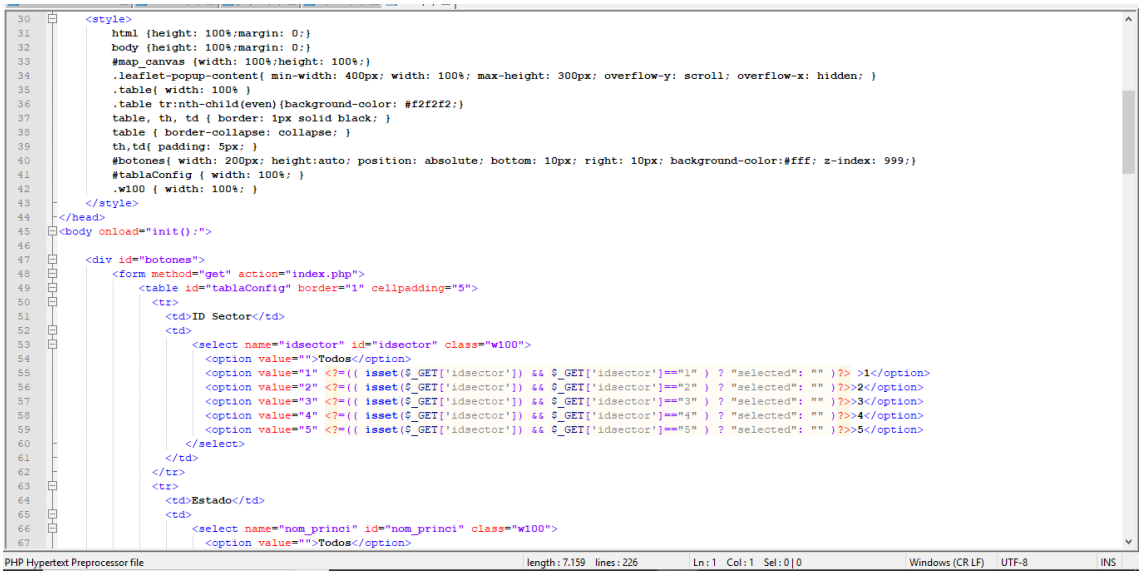
PHP Hypertext Preprocessor file | length : 1.654 lines : 71 | Ln : 1 Col : 1 Sel : 0 | 0 | Windows (CR LF) | UTF-8 | INS

Código del visor



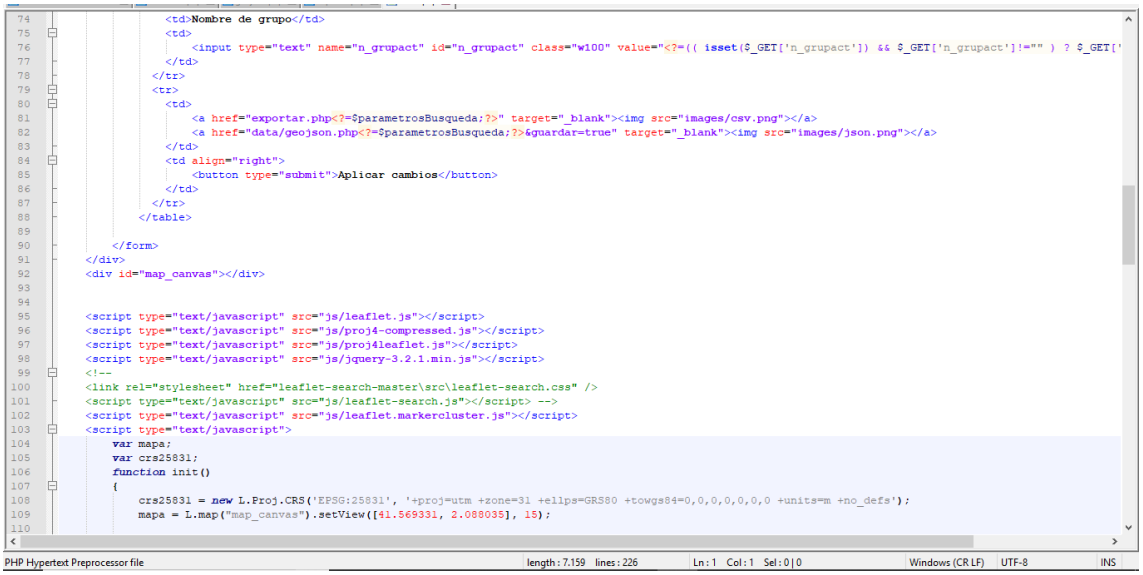
```php
<?php
$condicionesConsulta = array();
if( isset($_GET['idsector']) && $_GET['idsector']!="" )
{
    $condicionesConsulta[] = 'idsector='.$_GET['idsector'];
}
if( isset($_GET['nom_princi']) && $_GET['nom_princi']!="" )
{
    $condicionesConsulta[] = 'nom_princi='.$_GET['nom_princi'];
}
if( isset($_GET['n_grupact']) && $_GET['n_grupact']!="" )
{
    $condicionesConsulta[] = 'n_grupact='.$_GET['n_grupact'];
}

$parametrosBusqueda = "";
if( !empty($condicionesConsulta) )
{
    $parametrosBusqueda = "?".join($condicionesConsulta,'&');
}
//echo $parametrosBusqueda;
?>
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>Practicas</title>
    <link rel="stylesheet" type="text/css" href="css/leaflet.css">
    <link rel="stylesheet" type="text/css" href="css/MarkerCluster.Default.css">
    <style>
        html {height: 100%;margin: 0;}
        body {height: 100%;margin: 0;}
        #map_canvas {width: 100%;height: 100%;}
        .leaflet-popup-content{ min-width: 400px; width: 100%; max-height: 300px; overflow-y: scroll; overflow-x: hidden; }
        .table{ width: 100% }
        .table tr:nth-child(even){background-color: #f2f2f2;}
        table, th, td { border: 1px solid black; }
        table { border-collapse: collapse; }
```

PHP Hypertext Preprocessor file | length : 7.159 | lines : 226 | Ln : 1 | Col : 1 | Sel : 0 | 0 | Windows (CR LF) | UTF-8 | INS



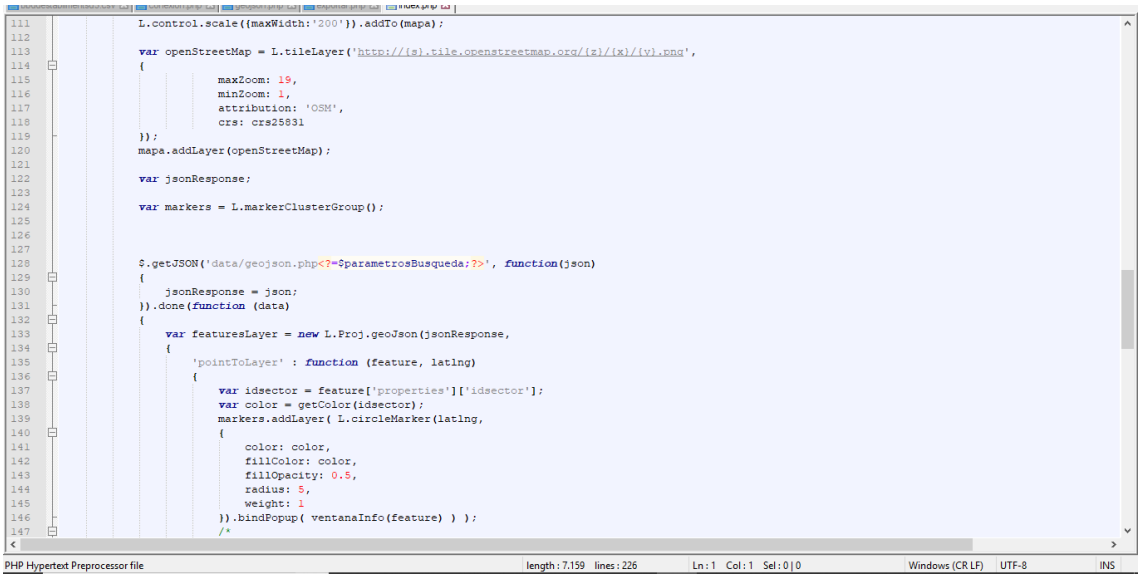
```php
    <style>
        html {height: 100%;margin: 0;}
        body {height: 100%;margin: 0;}
        #map_canvas {width: 100%;height: 100%;}
        .leaflet-popup-content{ min-width: 400px; width: 100%; max-height: 300px; overflow-y: scroll; overflow-x: hidden; }
        .table{ width: 100% }
        .table tr:nth-child(even){background-color: #f2f2f2;}
        table, th, td { border: 1px solid black; }
        table { border-collapse: collapse; }
        th,td{ padding: 5px; }
        #botones{ width: 200px; height:auto; position: absolute; bottom: 10px; right: 10px; background-color:#fff; z-index: 999;}
        #tablaConfig { width: 100%; }
        .w100 { width: 100%; }
    </style>
</head>
<body onload="init();">

    <div id="botones">
        <form method="get" action="index.php">
            <table id="tablaConfig" border="1" cellpadding="5">
              <tr>
                <td>ID Sector</td>
                <td>
                    <select name="idsector" id="idsector" class="w100">
                      <option value="">Todos</option>
                      <option value="1" <?=(( isset($_GET['idsector']) && $_GET['idsector']=="1" ) ? "selected": "" )?> >1</option>
                      <option value="2" <?=(( isset($_GET['idsector']) && $_GET['idsector']=="2" ) ? "selected": "" )?>>2</option>
                      <option value="3" <?=(( isset($_GET['idsector']) && $_GET['idsector']=="3" ) ? "selected": "" )?>>3</option>
                      <option value="4" <?=(( isset($_GET['idsector']) && $_GET['idsector']=="4" ) ? "selected": "" )?>>4</option>
                      <option value="5" <?=(( isset($_GET['idsector']) && $_GET['idsector']=="5" ) ? "selected": "" )?>>5</option>
                    </select>
                </td>
              </tr>
              <tr>
                <td>Estado</td>
                <td>
                    <select name="nom_princi" id="nom_princi" class="w100">
                      <option value="">Todos</option>
```

PHP Hypertext Preprocessor file | length : 7.159 | lines : 226 | Ln : 1 | Col : 1 | Sel : 0 | 0 | Windows (CR LF) | UTF-8 | INS



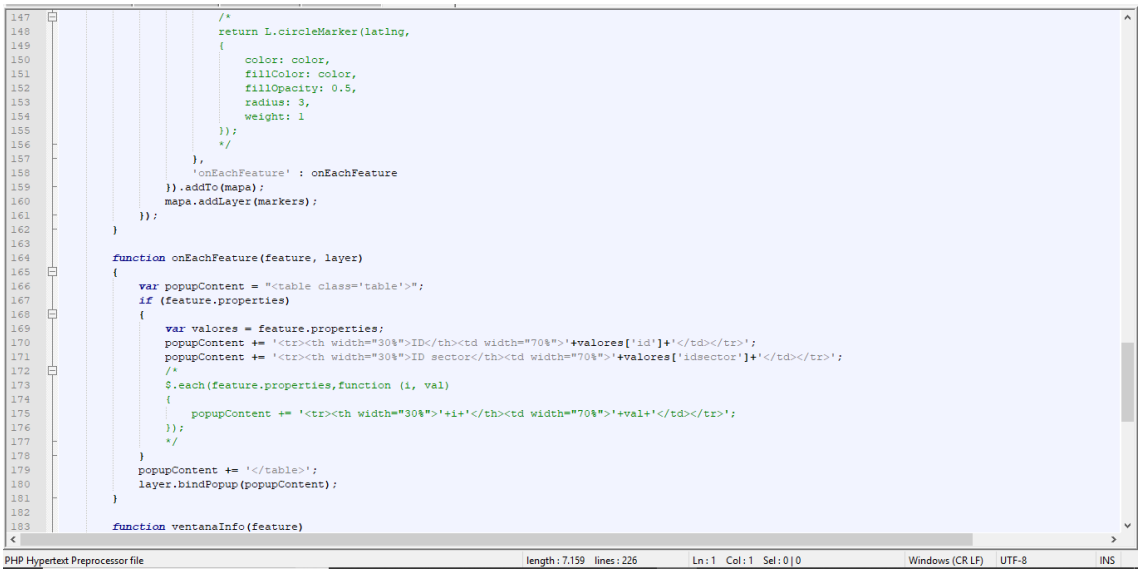
```
                <td>Nombre de grupo</td>
                <td>
                    <input type="text" name="n_grupact" id="n_grupact" class="w100" value="<?=(( isset($_GET['n_grupact']) && $_GET['n_grupact']!="" ) ? $_GET['
                </td>
            </tr>
            <tr>
                <td>
                    <a href="exportar.php<?=$parametrosBusqueda;?>" target="_blank"><img src="images/csv.png"></a>
                    <a href="data/geojson.php<?=$parametrosBusqueda;?>&guardar=true" target="_blank"><img src="images/json.png"></a>
                </td>
                <td align="right">
                    <button type="submit">Aplicar cambios</button>
                </td>
            </tr>
        </table>

    </form>
</div>
<div id="map_canvas"></div>


<script type="text/javascript" src="js/leaflet.js"></script>
<script type="text/javascript" src="js/proj4-compressed.js"></script>
<script type="text/javascript" src="js/proj4leaflet.js"></script>
<script type="text/javascript" src="js/jquery-3.2.1.min.js"></script>
<!--
<link rel="stylesheet" href="leaflet-search-master\src\leaflet-search.css" />
<script type="text/javascript" src="js/leaflet-search.js"></script> -->
<script type="text/javascript" src="js/leaflet.markercluster.js"></script>
<script type="text/javascript">
    var mapa;
    var crs25831;
    function init()
    {
        crs25831 = new L.Proj.CRS('EPSG:25831', '+proj=utm +zone=31 +ellps=GRS80 +towgs84=0,0,0,0,0,0,0 +units=m +no_defs');
        mapa = L.map("map_canvas").setView([41.569331, 2.088035], 15);

```

PHP Hypertext Preprocessor file | length : 7.159 | lines : 226 | Ln : 1 | Col : 1 | Sel : 0 | 0 | Windows (CR LF) | UTF-8 | INS



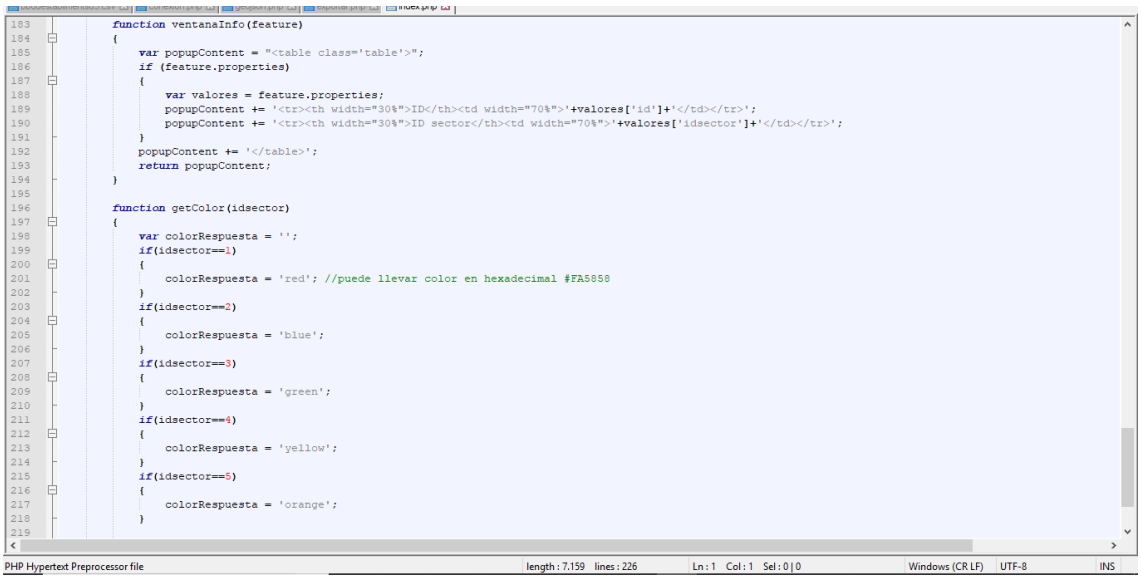
```
        L.control.scale({maxWidth:'200'}).addTo(mapa);

        var openStreetMap = L.tileLayer('http://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png',
        {
                maxZoom: 19,
                minZoom: 1,
                attribution: 'OSM',
                crs: crs25831
        });
        mapa.addLayer(openStreetMap);

        var jsonResponse;

        var markers = L.markerClusterGroup();


        $.getJSON('data/geojson.php<?=$parametrosBusqueda;?>', function(json)
        {
            jsonResponse = json;
        }).done(function (data)
        {
            var featuresLayer = new L.Proj.geoJson(jsonResponse,
            {
                'pointToLayer' : function (feature, latlng)
                {
                    var idsector = feature['properties']['idsector'];
                    var color = getColor(idsector);
                    markers.addLayer( L.circleMarker(latlng,
                    {
                        color: color,
                        fillColor: color,
                        fillOpacity: 0.5,
                        radius: 5,
                        weight: 1
                    }).bindPopup( ventanaInfo(feature) ) );
                    /*
```

PHP Hypertext Preprocessor file | length : 7.159 | lines : 226 | Ln : 1 | Col : 1 | Sel : 0 | 0 | Windows (CR LF) | UTF-8 | INS



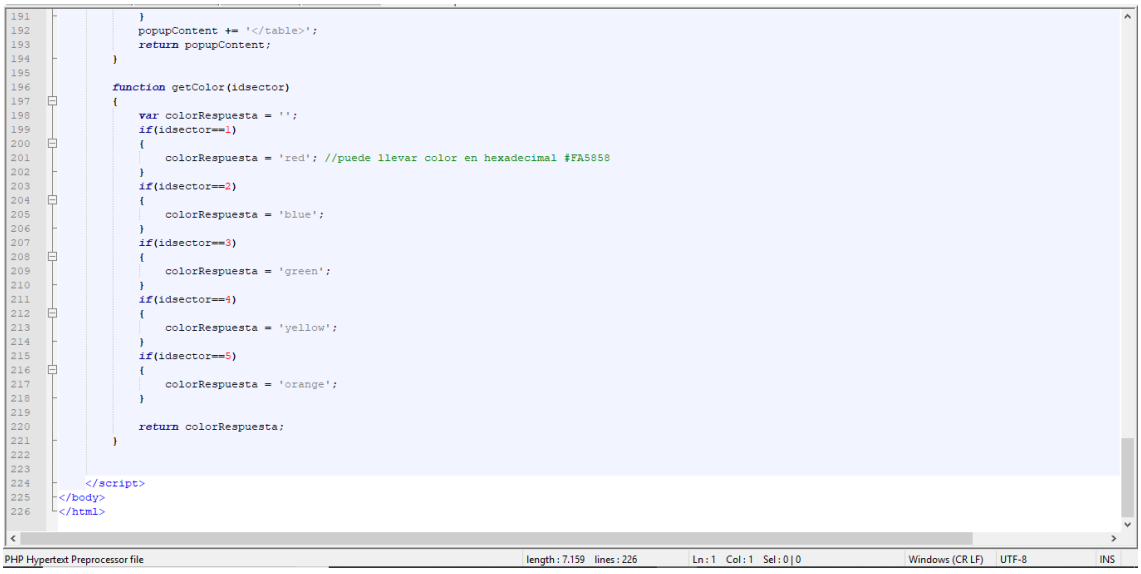
```
                    /*
                    return L.circleMarker(latlng,
                    {
                        color: color,
                        fillColor: color,
                        fillOpacity: 0.5,
                        radius: 3,
                        weight: 1
                    });
                    */
                },
                'onEachFeature' : onEachFeature
            }).addTo(mapa);
            mapa.addLayer(markers);
        });
    }

    function onEachFeature(feature, layer)
    {
        var popupContent = "<table class='table'>";
        if (feature.properties)
        {
            var valores = feature.properties;
            popupContent += '<tr><th width="30%">ID</th><td width="70%">'+valores['id']+'</td></tr>';
            popupContent += '<tr><th width="30%">ID sector</th><td width="70%">'+valores['idsector']+'</td></tr>';
            /*
            $.each(feature.properties,function (i, val)
            {
                popupContent += '<tr><th width="30%">'+i+'</th><td width="70%">'+val+'</td></tr>';
            });
            */
        }
        popupContent += '</table>';
        layer.bindPopup(popupContent);
    }

    function ventanaInfo(feature)
```

PHP Hypertext Preprocessor file | length : 7.159 | lines : 226 | Ln : 1 | Col : 1 | Sel : 0 | 0 | Windows (CR LF) | UTF-8 | INS



```
183         function ventanaInfo(feature)
184         {
185             var popupContent = "<table class='table'>";
186             if (feature.properties)
187             {
188                 var valores = feature.properties;
189                 popupContent += '<tr><th width="30%">ID</th><td width="70%">'+valores['id']+'</td></tr>';
190                 popupContent += '<tr><th width="30%">ID sector</th><td width="70%">'+valores['idsector']+'</td></tr>';
191             }
192             popupContent += '</table>';
193             return popupContent;
194         }
195
196         function getColor(idsector)
197         {
198             var colorRespuesta = '';
199             if(idsector==1)
200             {
201                 colorRespuesta = 'red'; //puede llevar color en hexadecimal #FA5858
202             }
203             if(idsector==2)
204             {
205                 colorRespuesta = 'blue';
206             }
207             if(idsector==3)
208             {
209                 colorRespuesta = 'green';
210             }
211             if(idsector==4)
212             {
213                 colorRespuesta = 'yellow';
214             }
215             if(idsector==5)
216             {
217                 colorRespuesta = 'orange';
218             }
219
```

PHP Hypertext Preprocessor file | length : 7.159 lines : 226 | Ln : 1 Col : 1 Sel : 0 | 0 | Windows (CR LF) | UTF-8 | INS



```
191             }
192             popupContent += '</table>';
193             return popupContent;
194         }
195
196         function getColor(idsector)
197         {
198             var colorRespuesta = '';
199             if(idsector==1)
200             {
201                 colorRespuesta = 'red'; //puede llevar color en hexadecimal #FA5858
202             }
203             if(idsector==2)
204             {
205                 colorRespuesta = 'blue';
206             }
207             if(idsector==3)
208             {
209                 colorRespuesta = 'green';
210             }
211             if(idsector==4)
212             {
213                 colorRespuesta = 'yellow';
214             }
215             if(idsector==5)
216             {
217                 colorRespuesta = 'orange';
218             }
219
220             return colorRespuesta;
221         }
222
223
224     </script>
225 </body>
226 </html>
```

PHP Hypertext Preprocessor file | length : 7.159 lines : 226 | Ln : 1 Col : 1 Sel : 0 | 0 | Windows (CR LF) | UTF-8 | INS