Ingeniería Web – Proyecto Web Colaborativo

Titulo: APLICACIÓN WEB PARA LA GESTIÓN DE PEDIDOS

Curso: 2º Grado en Industria Digital (Semestre 2º)

Materia: Ingeniería Web

Estudiantes: Maialen Aguilar Ayuso

Ignacio Chapero Garnica

Álvaro Husillos Echazarreta

Grupo: IW-07

Profesor: Jon Vadillo Romero

Facultad de Ingeniería
UNIVERSIDAD DE DEUSTO

Vitoria - Gasteiz, mayo de 2020

Resumen

La motivación de este proyecto es intentar hacer un caso práctico de una aplicación web real, con el correspondiente aprendizaje que nos ha proporcionado el desarrollarla intentando superar los problemas y adversidades que se nos han presentado.

Es una aplicación con una interacción con el usuario ágil y sencilla. Para su desarrollo en la parte con el cliente, se han desarrollado tecnologías como HTML5, CSS y JavaScript. Mientras que el desarrollo con el servidor se ha implementado usando Djando y por tanto Python.

Descriptores

Django, HTML5, CSS, JavaScript, API Fetch, Python

1. INDICE DE LA MEMORIA

GLOSARIO

JavaScript: Lenguaje de programación interpretado, usado para implementar la lógica tanto del lado del cliente como del lado del servidor.

HTML: HyperText Markup Language. Lenguaje de contenido para el desarrollo de páginas web.

JSON: JavaScript Object Notation. Estándar para intercambio de información.

API: Application Programming Interface. Interfaz de programación entre componentes software.

HTTP: Hypertext Transfer Protocol. Protocolo para acceso a la web.

Front-end: Parte del software que contiene la interfaz de usuario.

Back-end: Parte del software que procesa la entrada desde el front-end.

Framework: Estructura de soporte definida utilizada para organizar y desarrollar un proyecto software.

CAPÍTULO 1.
INTRODUCCIÓN

El presente proyecto ha sido realizado íntegramente por los estudiantes Álvaro Husillos, Maialen Aguilar e Ignacio Chapero. Dicho proyecto tiene como fin la implementación de una aplicación web que ayude a gestionar los pedidos de la cartera de clientes de una empresa de productos tecnológicos. El desarrollo se ha realizado con herramientas avanzadas de programación web.

CAPÍTULO 2.

OBJETIVOS DEL PROYECTO

2.1 TAREAS PRINCIPALES

En cuanto a planificación:

- Visualizar los aspectos a llevar a cabo
- Organizar el equipo
- Priorizar los elementos más importantes
- Planificar los pasos a seguir
- Marcar hitos
- Instalar softwares necesarios
- Ir informando de nuestros logros
- Reuniones una vez acabado el objetivo principal para ponernos al día

En cuanto a programación:

- HTML de la página web
- Dar estilo mediante CSS
- Desarrollo en Python
- Desarrollo en Javascript
- Elección del proyecto a realizar
- Brainstorming
- Definir modelos a desarrollar
- Tipo de vistas y acciones que necesitaremos
- Desarrollo de ideas

CAPÍTULO 3.

ESPECIFICACION DE REQUISITOS DEL SISTEMA

3.1 ALCANCE DEL PROYECTO

El Proyecto comprenderá el desarrollo de las funcionalidades establecidas en el catálogo de requisitos. Cualquier funcionalidad extra solicitada por el cliente, deberá ser añadida a este catálogo para su desarrollo y puesta en marcha. Este Proyecto no contempla la instalación del hardware necesario para el buen funcionamiento de la Aplicación, ni la instalación de BBDD o servidores web.

3.2 CATALOGO DE REQUISITOS

La empresa Deustronics Components S.L. se dedica a la fabricación y venta de componentes electrónicos. Actualmente, toda la información referente a sus pedidos y clientes está digitalizada en formato hojas de Excel, generando errores en la gestión de los pedidos. Por tanto, han decidido implementar una aplicación Web para eliminar la multitud de hojas Excel existentes y solucionar estos problemas.

La aplicación debe recoger y gestionar los siguientes datos principales:

- Pedido
  - ✓ Código de referencia del pedido
  - ✓ Fecha
  - ✓ Datos del cliente (CIF, nombre de empresa y datos de contacto) ○ Productos solicitados y cantidad
  - ✓ Precio total
- Productos
  - ✓ Referencia
  - ✓ Precio
  - ✓ Nombre

- ✓ Descripción
- ✓ Categoría
- ✓ Tipos de componentes que contiene el producto

- Componente
  - ✓ Código de referencia
  - ✓ Nombre de modelo ○ Marca

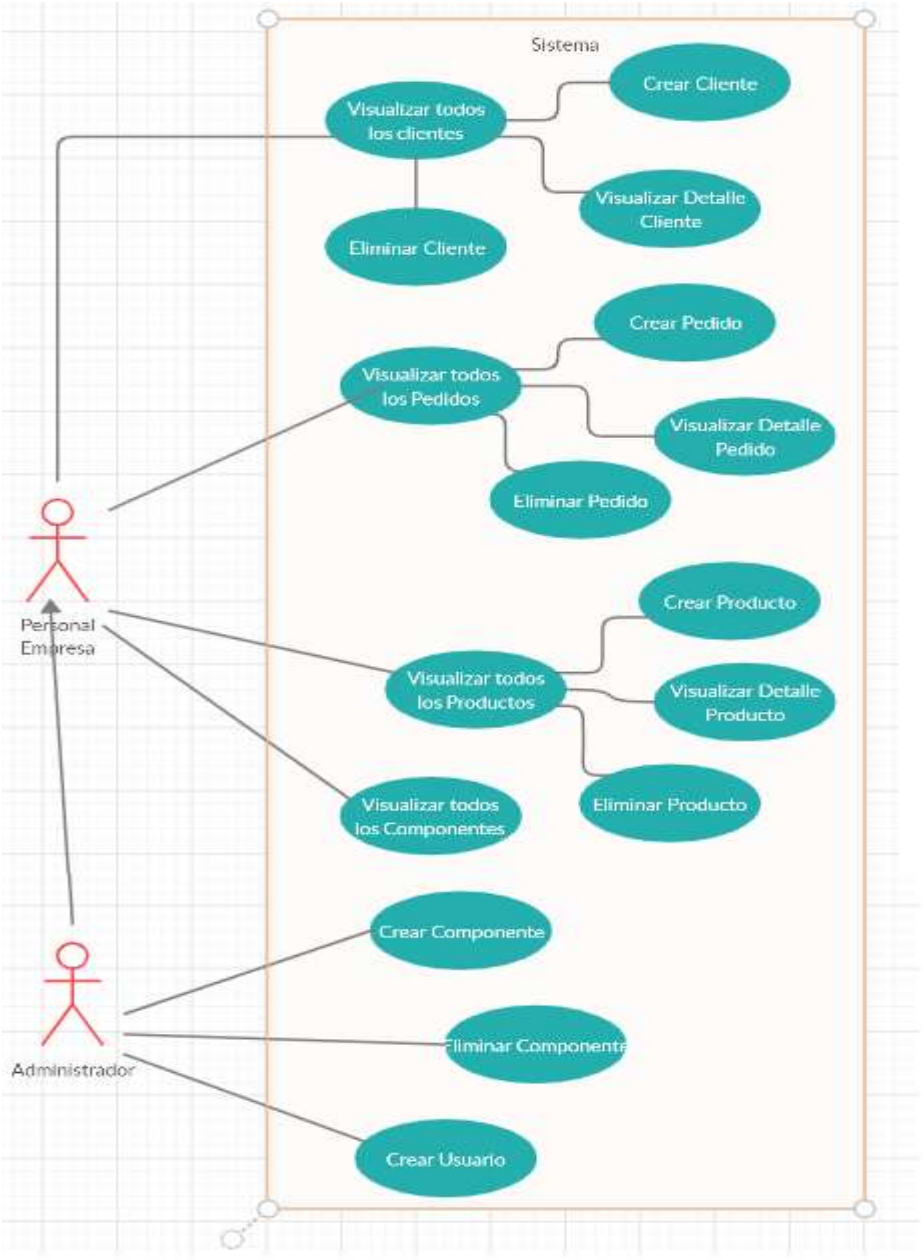
Siendo esta la información principal que se maneja, las funcionalidades más importantes requeridas son:

- Gestión de productos: creación, visualización (listado y detalle) y baja.
- Gestión de pedidos: creación, visualización (listado y detalle) y actualización.

3.3 DESCRIPCION DE REQUISITOS DEL NUEVO SISTEMA

3.3.1 MODELO FUNCIONAL

Las funcionalidades principales de la aplicación, así como los actores que pueden desarrollarlas quedan definidas por el siguiente Diagrama de Casos de Uso:



3.4 DESCRIPCION DE LA INTERFAZ DEL SISTEMA

3.4.1 PERFIL DE LOS USUARIOS

La aplicación está diseñada de forma sencilla para que su navegación sea muy intuitiva. Cuenta con un menú de navegación en la parte izquierda presente en todas las pantallas, así como botones de “vuelta a tras” y “home”, por lo que los únicos conocimientos necesarios son sobre navegación en páginas Web. Por tanto, cualquier tipo de perfil profesional está capacitado para utilizar la aplicación.

CAPÍTULO 4.

ESPECIFICACION DEL DISEÑO

4.1 PRINCIPALES FUNCIONES DEL SOFTWARE

Según los requerimientos de Deustronics Components S.L., las funcionalidades que se han desarrollado en la aplicación son las siguientes:

- Clientes:
  - Visualización del listado de clientes de la empresa
  - Creación de nuevos clientes que serán añadidos a la BBDD.
  - Visualización de todos los datos de un cliente en particular.
  - Visualización del Histórico de pedidos de cada uno de los clientes.
  - Eliminación de clientes de la BBDD.
- Productos:
  - Visualización de todos los productos comercializados, ordenados por categorías.
  - Creación de nuevos productos que serán añadidos a la BBDD.
  - Visualización de todos los datos de un producto en particular.
  - Eliminación de productos descatalogados de la BBDD.
- Componentes:
  - Visualización de todos los componentes que utiliza la empresa.
- Pedidos:

- Visualización de todos los pedidos, tanto en curso como finalizados.
- Creación de nuevos pedidos que serán añadidos a la BBDD.
- Visualización de todos los datos de un pedido en particular.
- Eliminación de pedidos de la BBDD.

Todas estas funciones desarrolladas satisfacen las necesidades de la empresa Deustronics Components S.L. No obstante, se ha añadido alguna funcionalidad extra, que pasamos a detallar:

- **Por seguridad**: puesto que la aplicación maneja toda la información más confidencial de la empresa, se ha decidido que todos los usuarios autorizados deberán tener unas credenciales personales e intransferibles materializadas con Nombre de Usuario y Contraseña. Solo el administrador del sistema puede crear nuevos usuarios, y proporcionar las credenciales a la persona autorizada. Para entrar a la aplicación, cada usuario autorizado deberá introducir sus credenciales. Así mismo, durante la duración de la sesión, se mostrará en pantalla el usuario que está usando la misma.

- **Por usabilidad de la aplicación**: se han implementado dos botones, uno de “Vuelta atrás” y otro de “Home”, para que desde las pantallas de detalles el usuario pueda volver al listado general o a la página principal de la aplicación.

  En los campos de fecha del formulario de pedidos, hemos añadido el siguiente código:

```
widgets = {
        'fecha_pedido': forms.SelectDateWidget,
        'fecha_entrega': forms.SelectDateWidget,
```

```
            'producto': forms.SelectMultiple()
        }
```

Con estos widgets conseguimos que la fecha se nos muestre con tres desplegables, uno de mes, día y año. Mucho más intuitivo que poner la fecha a mano. Con el tercer widget conseguimos que nos muestre una lista con todos los productos, para que podamos seleccionar uno o varios pulsando "Ctrl". En cliente, como sólo se puede seleccionar uno, hemos optado por un select normal.

Las tablas de visualización de listados se han diseñado con scroll vertical, para que sea más amigable su visualización.

En la pantalla de visualización de Pedidos, se ha implementado mediante Javascript un buscador, para que sea más fácil para el usuario visualizar los pedidos Pendientes o Entregados, sin tener necesidad de visualizar los que no le interesen.

- **Por optimización del rendimiento**: en la vista de detalle de un cliente, es posible visualizar el Histórico de Pedidos de este cliente. Para optimizar los recursos del sistema y no recargar toda la página, se ha utilizado la API Fetch de Javascript para hacer una llamada al servidor y nos muestre los datos solicitados. De igual forma, se ha utilizado este método para enviar al servidor los datos de un nuevo cliente.

4.2 DESCRIPCION DEL ENTORNO DE DESARROLLO

El entorno de trabajo que se ha utilizado en la fase de desarrollo es el entorno de desarrollo de Django.

Las principales herramientas que el mismo Django proporciona son un conjunto de scripts de Python para crear y trabajar con proyectos Django, junto con un simple *servidor web de desarrollo* que hemos utilizado para probar de forma local (es decir en nuestro ordenador, no en un servidor web externo) aplicaciones web Django con el explorador web de nuestro ordenador.

Hay otras herramientas periféricas que forman parte del entorno de desarrollo. Utilizamos un IDE para editar código, que en nuestro caso es PyCharm, y una herramienta de gestión del control de fuentes como Git para gestionar con seguridad las diferentes versiones de nuestro código.

El framework Django lo hemos utilizado con Python 3. Además, para la creación de las plantillas se ha utilizado HTML5 Y CSS para crear los estilos de las páginas. Para dotar a la aplicación de algunas funcionalidades extras se ha utilizado Javascript.

4.3 ARQUITECTURA FISICA Y ENTORNO TECNOLOGICO

Sistema Operativo:

Las aplicaciones web Django pueden ejecutarse en casi cualquier máquina donde pueda funcionar el lenguaje de programación Python: Windows, Mac OS X, Linux/Unix, Solaris, por nombrar sólo unos pocos. Casi cualquier computadora debería tener el rendimiento necesario para ejecutar Django durante el desarrollo. En nuestro caso hemos utilizado Windows 10 y es el que recomendamos.

Base de Datos:

Respecto a la base de datos, la recomendada es PostgreSQL, pero también son soportadas MySQL y SQLite3. En nuestro caso hemos utilizado en el entorno de desarrollo SQLite3, que almacena sus datos en un fichero. SQLite está pensado para ser usado como base ligera y no puede soportar un alto nivel de concurrencia. Es sin

embargo una excelente elección para aplicaciones que son principalmente de sólo lectura.

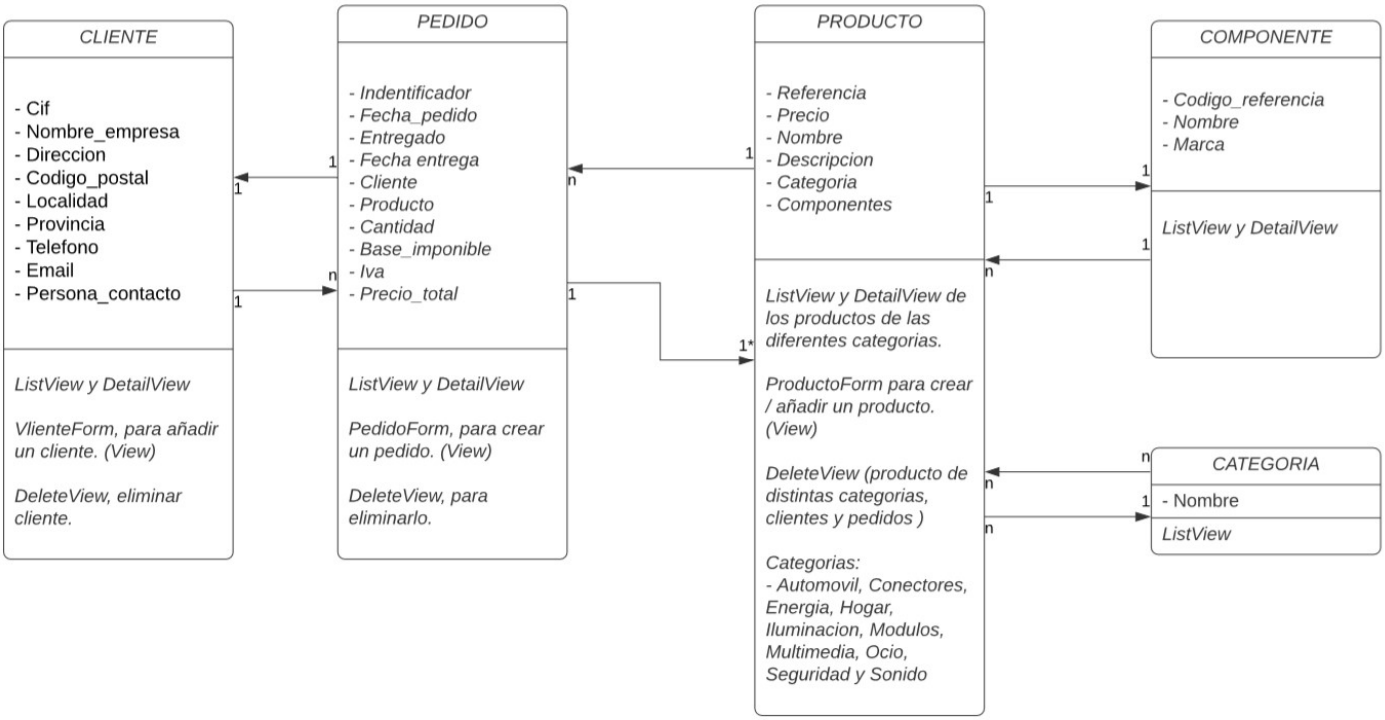
Una vez creados los modelos de datos, Django proporciona una abstracción de la base de datos a través de su API que permite crear, recuperar, actualizar y borrar objetos. También es posible que el usuario ejecute sus propiasconsultas SQL directamente. En el modelo de datos de Django, una clase representa un registro de una tabla en la base de datos y las instancias de ésta serán las tuplas en la tabla.

Servidores:

Como mencionamos en los requisitos, Django incluye un servidor web liviano para realizar pruebas y trabajar en la etapa de desarrollo. En la etapa de producción, sin embargo, se recomienda Apache 2 con mod_python. Aunque Django soporta la especificación WSGI, por lo que puede correr sobre una gran variedad de servidores como FastCGI o SCGI en Apache u otros servidores (particularmente Lighttpd).

4.4 DESCRIPCION DEL DISEÑO

4.4.1 DISEÑO DE LA ESTRUCTURA FÍSICA DE LOS DATOS



4.3.1.1 DEFINICION DE VISTAS

Si analizamos generalmente las vistas que contenemos, estas son las que implementamos ListView, DetailView, DeleteView y View (para formularios).

ListView:

Principalmente esta Vista nos mostrara una lista de todos aquellos objetos creados de un modelo. Para ver cómo se implementa pondremos un ejemplo:

```
# DEVUELVE EL LISTADO DE CLIENTES

class ClientesListView(ListView):// Definimos el tipo de vista
```

```
//definimos el modelo a utilizar
    model = Cliente

//definimos la plantilla que mostrara dicha lista
    template_name = 'clientes.html'

//nombre de la lista
 context_object_name = 'lista_clientes'


//aparte de los datos del cliente le diremos que pase el titulo de la página
 def get_context_data(self, **kwargs):

        //cargamos el diccionario context
        context = super(ClientesListView, self).get_context_data(**kwargs)

    //añadimos la información que queremos, en este caso aparecerá como
      Subtitulo.
        context['Titulo_pagina'] = 'Listado de clientes'
        return context
```

DetailView:

En este caso es una vista de tipo genérica que nos mostrara en detalle la información de un objeto. Veamos un ejemplo:

```
                # DEVUELVE DATOS DE UN CLIENTE

class Detalle_ClienteDetailView(DetailView):
    model = Cliente
    template_name = 'Datos_Cliente.html'

    def get_context_data(self, **kwargs):
        context = super(Detalle_ClienteDetailView,
self).get_context_data(**kwargs)
        context['Titulo_pagina'] = 'Datos de cliente'
        return context
```

DeleteView:

La acción de dicha vista será eliminar un objeto. El contexto de esta vista es parecido a las anteriores, el único cambio es que utilizaremos la pk (primarykey) con lo cual añadiremos este apartado.

```
def get_context_data(self, **kwargs):
    context = super(Eliminar_ClientesDeleteView, self).get_context_data(**kwargs)
    pk = self.kwargs.get('pk')
    cliente = Cliente.objects.get()
    context.update({'cliente': cliente})
    return context

def get_success_url(self):
    return reverse('clientes')
```

View (Formularios):

Crearemos formularios para crear/añadir objetos.

Ejemplo pedido:

```
class CrearProductoView(View):
    def get(self, request, *args, **kwargs):
        form = ProductoForm()
        context = {
            'form': form,
            'titulo_pagina': 'Insertar un producto'
        }
        return render(request, 'Insertar_Producto.html', context)

    def post(self, request, *args, **kwargs):
        form = ProductoForm(request.POST)
        if form.is_valid():
            form.save()

            # Volvemos a la lista de productos
            return redirect('categorias')

        return render(request, 'Insertar_Producto.html', {'form':
form})
```

El formulario de cliente es distinto, puesto que como ya comentamos, por optimización de recursos, lo mandamos a través de un método Post de la API Fetch, por lo que la vista queda como sigue:

```
        # AÑADIR CLIENTE NUEVO
@method_decorator(csrf_exempt, name='dispatch')
class CrearClienteView(View):
    def get(self, request, *args, **kwargs):
        form = ClienteForm()
        context = {
            'form': form,
            'titulo_pagina': 'Añadir cliente'
        }
        return render(request, 'Añadir_Cliente.html', context)

    def post(self,request):
        cliente=Cliente()
        cliente.cif=request.POST['cif']
        cliente.nombre_empresa=request.POST['nombre_empresa']
        cliente.direccion=request.POST['direccion']
        cliente.codigo_postal=request.POST['codigo_postal']
        cliente.Localidad=request.POST['Localidad']
```

```
        cliente.Provincia=request.POST['Provincia']
        cliente.telefono=request.POST['telefono']
        cliente.email=request.POST['email']
        cliente.persona_contacto=request.POST['persona_contacto']
        cliente.save()
        return JsonResponse(model_to_dict(cliente))
```

Modelo Cliente:

Para este modelo disponemos de 4 vistas, ListView, DetailView, DeleteView y View (ClienteForm()).

Modelo componente:

En este caso solo tendremos ListView.

Modelo producto:

Tenemos un ListView y DeleteView para cada categoría y la View para el formulario (ProductoForm()) y crear un nuevo producto.

Modelo pedido:

Al igual que los productos y los clientes tendrán ListView, DetailView, View (formulario) y DeleteView.

CAPÍTULO 5.

MANUAL DE USUARIO

1.- La aplicación se arranca con la url: 127.0.0.1:8000/appGestionPedidos. Esta dirección ejecutará la aplicación, que nos mostrará la página de login. Si intentamos acceder a otra url de la aplicación, por ejemplo 127.0.0.1:8000/appGestionPedidos/home, nos debería dar un error. De esta forma se impide que nadie sin autorización vea la aplicación.

2.- Una vez en la página de login, si no estamos autorizados, o sea, no tenemos usuario y contraseña, no podemos acceder. Para tareas de mantenimiento existe un superuser cuyas credenciales son:

username: admin

password: admin

3.- Una vez logeados, la aplicación nos llevará a la página "home". Aquí tenemos el menú principal, absolutamente intuitivo. Cada opción del menú nos llevará a un listado visualizado en una tabla, salvo catálogo, que nos mostrará las distintas categorías para poder seleccionar, y entonces nos llevará a la tabla.

Clientes:

En este apartado, se mostrará una tabla con los datos de todos nuestros clientes, conteniendo en cada uno de ellos la opción de eliminarlo o mostrarnos todos sus detalles al completo. En la parte inferior de esta tabla existe un botón con el cual podremos crear un cliente nuevo (pulsaremos y nos llevará a otra pantalla para rellenar un formulario con los datos del nuevo cliente).

- **Detalles Cliente:**

  Veremos los detalles de un cliente en forma de listado. En la parte superior derecha, encontraremos un botón de “Histórico de Pedidos” que nos abrirá un Popup con una tabla con todos los pedidos que ha realizado ese cliente. Así mismo, en la parte inferior encontraremos dos botones, uno para volver al listado de clientes y otro para volver a la Página Principal.

- **Eliminar Cliente:**

  Si pulsamos sobre este icono, se nos abrirá una pantalla en la que figurarán los datos del cliente seleccionado, y se nos preguntará por una confirmación del borrado. Si pulsamos sobre el botón “Eliminar”, eliminaremos el cliente definitivamente de la BBDD (**OJO, NO TIENE VUELTA ATRÁS**). Si no deseamos eliminarlo, deberemos pulsar el botón de “Volver” situado abajo a la izquierda.

- **Crear Cliente:**

  Nos aparecerá un formulario con todos los campos en blanco. Es obligatorio rellenar todos los campos para crear un cliente. Una vez pulsemos el botón “Guardar”, el nuevo cliente será registrado en la BBDD.

Pedidos:

En este apartado, se mostrará una tabla con los datos de todos nuestros pedidos, conteniendo en cada uno de ellos la opción de eliminarlo o mostrarnos todos sus detalles al completo. En la parte superior de la tabla, existe un buscador mediante un menú desplegable. Si seleccionamos una de las categorías y pulsamos el botón “Mostrar”, en la tabla aparecerán solamente los pedidos seleccionados.

En la parte inferior de esta tabla existe un botón con el cual podremos crear un pedido nuevo (pulsaremos y nos llevará a otra pantalla para rellenar un formulario con los datos del nuevo pedido).

- **Detalles Pedido:**

  Veremos los detalles de un pedido en forma de listado. Así mismo, en la parte inferior encontraremos dos botones, uno para volver al listado de clientes y otro para volver a la Página Principal.

- **Eliminar Cliente:**

  Si pulsamos sobre este icono, se nos abrirá una pantalla en la que figurarán los datos del pedido seleccionado, y se nos preguntará por una confirmación del borrado. Si pulsamos sobre el botón “Eliminar”, eliminaremos el pedido definitivamente de la BBDD (**OJO, NO TIENE VUELTA ATRÁS**). Si no deseamos eliminarlo, deberemos pulsar el botón de “Volver” situado abajo a la izquierda.

- **Crear Pedido:**

  Nos aparecerá un formulario con todos los campos en blanco. Es obligatorio rellenar todos los campos para crear un pedido.

Catálogo:

En este apartado, se mostrará una tabla con los datos de todos nuestros productos, ordenados por categoría, conteniendo en cada uno de ellos la opción de eliminarlo o mostrarnos todos sus detalles al completo. En la parte inferior de esta tabla existe un botón con el cual podremos crear un producto nuevo (pulsaremos y nos llevará a otra pantalla para rellenar un formulario con los datos del nuevo producto).

- **Detalles Producto:**

  Veremos los detalles de un producto en forma de listado. Así mismo, en la parte inferior encontraremos dos botones, uno para volver al listado de categorías y otro para volver a la Página Principal.

- **Eliminar Producto:**

  Si pulsamos sobre este icono, se nos abrirá una pantalla en la que figurarán los datos del producto seleccionado, y se nos preguntará por una confirmación del borrado. Si pulsamos sobre el botón "Eliminar", eliminaremos el producto definitivamente de la BBDD (**OJO, NO TIENE VUELTA ATRÁS**). Si no deseamos eliminarlo, deberemos pulsar el botón de "Volver" situado abajo a la izquierda.

- **Crear Producto:**

  Nos aparecerá un formulario con todos los campos en blanco. Es obligatorio rellenar todos los campos para crear un producto.

Componentes:

En este apartado, se mostrará una tabla con los datos de todos nuestros componentes.

4.- En la esquina superior derecha, se mostrará en todo momento el usuario que está haciendo uso de la aplicación. Es importante Cerrar la Sesión mediante el botón colocado a tal efecto debajo del usuario actual. Si no cerramos la sesión, dejamos abierta una brecha de seguridad.

CAPÍTULO 6.

INCIDENCIAS DEL PROYECTO Y CONCLUSIONES

Algunos de los inconvenientes más destacados con los que nos hemos ido encontrando han sido los siguientes:

- Uno de los compañeros dio la idea de hacer un registro para poder acceder al proyecto, esta acción requería de conocimientos un poco más avanzados que los que teníamos, pero lo solventamos haciendo una ardua investigación en Google
- Después de la primera entrega, revisando las funcionalidades de nuestro proyecto, nos dimos cuenta de que nuestras clases DeleteView habían dejado de hacer su función. Nuestra reacción fue inmediata, nos pusimos a revisar el código y descubrimos para nuestra sorpresa que el código había sido modificado. Lo corregimos y volvió a funcionar correctamente.
- A medida que íbamos programando funciones en un único archivo JS, nos dimos cuenta de que algunas dejaban de funcionar. Entonces entendimos que, en lugar de un único archivo, igual era recomendable tener más de uno, en función de la vista donde se aplicasen esas funciones.
- La implementación de la función Post ha sido bastante difícil hasta conseguir el resultado esperado. Cuando ya pensábamos que funcionaba correctamente nos dábamos cuenta de que se nos modificaban otras cosas. Finalmente, conseguimos obtener el resultado esperado.
- Hemos tenido diversos problemas con Github. La última parte del desarrollo Maialen no podía visualizar correctamente la aplicación, a pesar de tener el mismo código que el resto. Hicimos pruebas con distintos navegadores, borramos la caché de los mismos, pero no conseguimos solucionar el problema. Por tanto, todo el código desarrollado por Maialen era implementado y probado por Alvaro.

- También nos hemos percatado de que algunas de las funcionalidades no funcionan correctamente con Chrome. Investigando información, sí parece que algunos aspectos cambian en función del Navegador, por lo que para la correcta utilización de esta es altamente recomendable usar Firefox.