ÍNDICE

[1 PLANIFICACIÓN DEL PROYECTO 3](#_Toc452742121)

[1.1 Definición de actividades 3](#_Toc452742122)

[ Diagrama de GANTT 5](#_Toc452742123)

[1.2 Estimación de costes 5](#_Toc452742124)

[2 ANÁLISIS DE LA APLICACIÓN 6](#_Toc452742125)

[2.1 Funciones de la aplicación 6](#_Toc452742126)

[2.1.1 Módulo de Administración 7](#_Toc452742127)

[2.1.2 Módulo de Venta 9](#_Toc452742128)

[2.1.3 Otros 9](#_Toc452742129)

[2.2 Requisitos físicos 10](#_Toc452742130)

[2.3 Requisitos lógicos/funcionales 10](#_Toc452742131)

[2.3.1 Tipos de usuarios 10](#_Toc452742132)

[2.3.2 Módulo de Administración 10](#_Toc452742133)

[2.3.3 Módulo de Venta 12](#_Toc452742134)

[2.3.4 Perfil de Usuario 12](#_Toc452742135)

[2.4 Diagramas de casos de uso 13](#_Toc452742136)

[2.4.1 Actores 13](#_Toc452742137)

[2.4.2 Diagramas del Módulo Administración 14](#_Toc452742138)

[2.4.3 Diagramas del Módulo de Venta 17](#_Toc452742139)

[2.4.4 Diagrama del Perfil de Usuario 18](#_Toc452742140)

[2.5 Requisitos no funcionales 18](#_Toc452742141)

[2.6 Mapa Web 19](#_Toc452742142)

[2.6.1 Mapa Web del Módulo de Administración 19](#_Toc452742143)

[2.6.2 Mapa Web del Módulo de Venta 20](#_Toc452742144)

[3 MODELO DE DATOS 21](#_Toc452742145)

[3.1 Esquema Entidad-Relación 21](#_Toc452742146)

[3.2 Descripción de las Tablas 21](#_Toc452742147)

[4 TECNOLOGÍAS UTILIZADAS 24](#_Toc452742148)

[4.1 ¿Por qué usar Software libre? 24](#_Toc452742149)

[4.2 Parte Cliente 25](#_Toc452742150)

[4.2.1 HTML 25](#_Toc452742151)

[4.2.2 CSS 26](#_Toc452742152)

[4.2.3 JavaScript 26](#_Toc452742153)

[4.2.4 Bootstrap 27](#_Toc452742154)

[4.2.5 jQuery 28](#_Toc452742155)

[4.3 Parte Servidor 30](#_Toc452742156)

[4.3.1 Servidor Apache 30](#_Toc452742157)

[4.3.2 PHP 30](#_Toc452742158)

[4.3.3 MySQL 31](#_Toc452742159)

[4.4 Comunicación Cliente-Servidor. Ajax 32](#_Toc452742160)

[4.5 Otros 32](#_Toc452742161)

[4.5.1 CodeIgniter 32](#_Toc452742162)

[4.5.2 Github 33](#_Toc452742163)

[4.5.3 NetBeans 33](#_Toc452742164)

[4.5.4 MySQL Workbench 34](#_Toc452742165)

[4.5.5 FileZilla 34](#_Toc452742166)

[4.5.6 Balsamiq Mockup’s 34](#_Toc452742167)

[4.5.7 Creately UML 34](#_Toc452742168)

[5 ARQUITECTURA DE LA APLICACIÓN 34](#_Toc452742169)

[5.1 Componentes 35](#_Toc452742170)

[5.2 Ventajas y desventajas del uso del patrón MVC 36](#_Toc452742171)

[6 MANUAL DE USUARIO 37](#_Toc452742172)

[6.1 Login 37](#_Toc452742173)

[6.2 Módulo de Administración 37](#_Toc452742174)

[6.2.1 Inicio 37](#_Toc452742175)

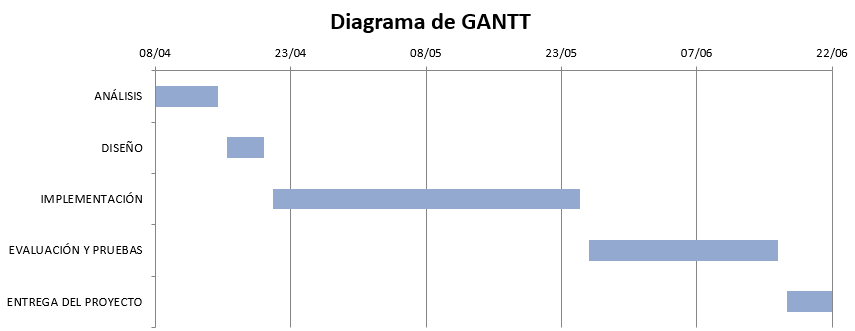
# PLANIFICACIÓN DEL PROYECTO

## Definición de actividades

Las actividades son el resultado final de descomponer el trabajo a realizar en el proyecto en paquetes o fases que puedan ser controlados y gestionados. Al tratarse de un proyecto de desarrollo e implantación de un software web, podemos utilizar algunas de las metodologías más populares dentro del mundo de la ingeniería del software. Por ello se decidió usar el modelo clásico, que es dividir el proyecto en fases secuenciales. Estas fases son: Análisis, Diseño, Implementación y Pruebas.

|  |  |  |
| --- | --- | --- |
| Tarea | Inicio | Fin |
| ANÁLISIS | **08/04/2016** | **15/04/2016** |
| * Estudios previos | 08/04/2016 | 08/04/2016 |
| * Requisitos físicos | 09/04/2016 | 09/04/2016 |
| * Requisitos funcionales | 10/04/2016 | 11/04/2016 |
| * Modelado de datos | 12/04/2016 | 14/04/2016 |
| * Aprobación del cliente | 15/04/2016 | 15/04/2016 |
| DISEÑO | **16/04/2016** | **20/04/2016** |
| * Realización prototipos de baja fidelidad | 16/04/2016 | 18/04/2016 |
| * Selección de tecnologías | 19/04/2016 | 19/04/2016 |
| * Aprobación del cliente | 20/04/2016 | 20/04/2016 |
| IMPLEMENTACIÓN | **21/04/2016** | **25/05/2016** |
| * Configuración plantillas *(Módulo Administración)* | 21/04/2016 | 21/04/2016 |
| * Login *(Módulo Administración)* | 22/04/2016 | 22/04/2016 |
| * Perfil de usuario *(Módulo Administración)* | 23/04/2016 | 23/04/2016 |
| * Menú agregar *(Módulo Administración)* | 24/04/2016 | 29/04/2016 |
| * Menú listar *(Módulo Administración)* | 30/05/2016 | 09/05/2016 |
| * Avisos de productos con stocks bajos | 10/05/2016 | 10/05/2016 |
| * Configuración plantillas *(Módulo Venta)* | 11/05/2016 | 11/05/2016 |
| * Login *(Módulo Venta)* | 12/05/2016 | 12/05/2016 |
| * Perfil de usuario *(Módulo Venta)* | 13/05/2016 | 13/05/2016 |
| * Carrito *(Módulo Venta)* | 14/05/2016 | 17/05/2016 |
| * Proceso de venta *(Módulo Venta)* | 18/05/2016 | 20/05/2016 |
| * Listar facturas *(Módulo Administración)* | 21/05/2016 | 24/05/2016 |
| * Aprobación del cliente | 25/05/2016 | 25/05/2016 |
| EVALUACIÓN Y PRUEBAS | **26/05/2016** | **16/06/2016** |
| * Preparación de pruebas | 26/05/2016 | 26/05/2016 |
| * Realización de pruebas | 27/05/2016 | 30/05/2016 |
| * Evaluación de los resultados de las pruebas | 01/05/2016 | 01/06/2016 |
| * Generar listas de cambios | 02/06/2016 | 02/06/2016 |
| * Aplicar cambios | 03/06/2016 | 10/06/2016 |
| * Aplicar posibles mejoras | 11/06/2016 | 16/06/2016 |
| ENTREGA DEL PROYECTO | **17/06/2016** | **22/06/2016** |
| * Documentación de la aplicación | 17/06/2016 | 20/06/2016 |
| * Presentación de la aplicación | 20/06/2016 | 22/06/2016 |
| * Aprobación del cliente | 22/06/2016 | 22/06/2016 |

* Diagrama de GANTT



## Estimación de costes

Los únicos costes que tenemos que soportar son los recursos humanos utilizado para el desarrollo de la aplicación, ya que el software que utilizaremos será libre.

Para este proyecto se identificaron los roles de las personas que típicamente intervienen en la realización de las distintas actividades como son el jefe de proyecto, analista, diseñador, programador y tester.

Pero en nuestro caso, al ser un trabajo unipersonal, todas las tareas serían realizadas por mi persona por ello, para calcular el coste del proyecto tendríamos en cuenta el tiempo aproximado dedicado a cada tarea, de manera que para cada tarea multiplicaríamos el número de horas estimadas por un precio razonable y ajustado al mercado.

De esta manera, se prevé que se dedique unas 3-4 horas de lunes a viernes y 6-7 horas sábado y domingo, para escoger un valor medio se estima que se echarán en total unas 35 horas semanales (5 horas/día). Cada hora tendrá un coste de 10 €.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tarea | Días | Horas/Día | Precio/h | Coste |
| ANÁLISIS | **7** | **5** | **10 €** | **350 €** |
| * Estudios previos | 1 | 5 | 10 € | 50 € |
| * Requisitos físicos | 1 | 5 | 10 € | 50 € |
| * Requisitos funcionales | 2 | 5 | 10€ | 100 € |
| * Modelado de datos | 3 | 5 | 10 € | 150 € |
| DISEÑO | **4** | **5** | **10 €** | **200 €** |
| * Realización prototipos de baja fidelidad | 3 | 5 | 10 € | 150 € |
| * Selección de tecnologías | 1 | 5 | 10 € | 50 € |
| IMPLEMENTACIÓN | **34** | **5** | **10 €** | **1700 €** |
| * Configuración plantillas *(Módulo Administración)* | 1 | 5 | 10 € | 50 € |
| * Login *(Módulo Administración)* | 1 | 5 | 10 € | 50 € |
| * Perfil de usuario *(Módulo Administración)* | 1 | 5 | 10 € | 50 € |
| * Menú agregar *(Módulo Administración)* | 6 | 5 | 10 € | 300 € |
| * Menú listar *(Módulo Administración)* | 10 | 5 | 10 € | 500 € |
| * Avisos de productos con stocks bajos | 1 | 5 | 10 € | 50 € |
| * Configuración plantillas *(Módulo Venta)* | 1 | 5 | 10 € | 50 € |
| * Login *(Módulo Venta)* | 1 | 5 | 10 € | 50 € |
| * Perfil de usuario *(Módulo Venta)* | 1 | 5 | 10 € | 50 € |
| * Carrito *(Módulo Venta)* | 4 | 5 | 10 € | 200 € |
| * Proceso de venta *(Módulo Venta)* | 3 | 5 | 10 € | 150 € |
| * Listar facturas *(Módulo Administración)* | 4 | 5 | 10 € | 200 € |
| EVALUACIÓN Y PRUEBAS | **20** | **5** | **10 €** | **1000 €** |
| * Preparación de pruebas | 1 | 5 | 10 € | 50 € |
| * Realización de pruebas | 4 | 5 | 10 € | 200 € |
| * Evaluación de los resultados de las pruebas | 1 | 5 | 10 € | 50 € |
| * Generar listas de cambios | 1 | 5 | 10 € | 50 € |
| * Aplicar cambios | 7 | 5 | 10 € | 350 € |
| * Aplicar posibles mejoras | 6 | 5 | 10 € | 300 € |
| ENTREGA DEL PROYECTO | **7** | **5** | **10 €** | **350 €** |
| * Documentación de la aplicación | 4 | 5 | 10 € | 200 € |
| * Presentación de la aplicación | 3 | 5 | 10 € | 150 € |
| COSTE TOTAL DE LA APLICACIÓN | | |  | **3600 €** |

NOTA: No han sido incluidos los días de aprobación con el cliente

# ANÁLISIS DE LA APLICACIÓN

## Funciones de la aplicación

A continuación, se exponen el listado de funciones de nuestro sistema, el cual tendrá dos partes/módulos distintas: Administración y Venta.

### Módulo de Administración

En el módulo de administración se llevará a cabo todo lo relacionado con gestionar nuestros distintos elementos (proveedores, categorías, productos, clientes, usuarios, etc) pudiendo agregar, modificar, listar o buscar.

A este módulo sólo podrá acceder el usuario de tipo *administrador,* identificándose previamente para ello.

#### Gestión de Proveedores

|  |  |  |  |
| --- | --- | --- | --- |
| # | Tarea | Administrador | Empleado |
| 1 | Agregar proveedor | **✓** | ✘ |
| 2 | Dar de baja a un proveedor | **✓** | ✘ |
| 3 | Dar de alta a un proveedor | **✓** | ✘ |
| 4 | Modificar un proveedor | **✓** | ✘ |
| 5 | Listar proveedores | **✓** | ✘ |
| 6 | Ver detalles de un proveedor | **✓** | ✘ |
| 7 | Buscar proveedor por cualquier campo | **✓** | ✘ |

#### Gestión de Categorías

|  |  |  |  |
| --- | --- | --- | --- |
| # | Tarea | Administrador | Empleado |
| 1 | Agregar categoría | **✓** | ✘ |
| 2 | Dar de baja a una categoría | **✓** | ✘ |
| 3 | Dar de alta a una categoría | **✓** | ✘ |
| 4 | Modificar una categoría | **✓** | ✘ |
| 5 | Listar categorías | **✓** | ✘ |
| 6 | Ver detalles de una categoría \*\*\* | **✓** | ✘ |
| 7 | Buscar categoría por cualquier campo | **✓** | ✘ |

#### Gestión de Productos

|  |  |  |  |
| --- | --- | --- | --- |
| # | Tarea | Administrador | Empleado |
| 1 | Agregar producto | **✓** | ✘ |
| 2 | Dar de baja a un producto | **✓** | ✘ |
| 3 | Dar de alta a un producto | **✓** | ✘ |
| 4 | Modificar un producto | **✓** | ✘ |
| 5 | Listar productos | **✓** | ✘ |
| 6 | Ver detalles de un producto | **✓** | ✘ |
| 7 | Buscar productos por cualquier campo | **✓** | ✘ |
| 8 | Buscar productos por categoría | **✓** | ✘ |
| 9 | Ver los productos que se van acabando | **✓** | ✘ |

#### Gestión de Clientes

|  |  |  |  |
| --- | --- | --- | --- |
| # | Tarea | Administrador | Empleado |
| 1 | Agregar cliente | **✓** | ✘ |
| 2 | Dar de baja a un cliente | **✓** | ✘ |
| 3 | Dar de alta a un cliente | **✓** | ✘ |
| 4 | Modificar a un cliente | **✓** | ✘ |
| 5 | Listar clientes | **✓** | ✘ |
| 6 | Ver detalles de un cliente | **✓** | ✘ |
| 7 | Buscar clientes por cualquier campo | **✓** | ✘ |

#### Gestión de Usuarios

|  |  |  |  |
| --- | --- | --- | --- |
| # | Tarea | Administrador | Empleado |
| 1 | Agregar usuario | **✓** | ✘ |
| 2 | Dar de baja a un usuario | **✓** | ✘ |
| 3 | Dar de alta a un usuario | **✓** | ✘ |
| 4 | Modificar otro usuario | ✘ | ✘ |
| 5 | Listar usuarios | **✓** | ✘ |
| 6 | Ver detalles de un usuario \*\*\* | **✓** | ✘ |
| 7 | Buscar usuarios por cualquier campo | **✓** | ✘ |

#### Gestión de Facturas

|  |  |  |  |
| --- | --- | --- | --- |
| # | Tarea | Administrador | Empleado |
| 1 | Ver facturas pendientes | **✓** | ✘ |
| 2 | Ver detalles de una factura | **✓** | ✘ |
| 3 | Marcar como pagada una factura pendiente | **✓** | ✘ |
| 4 | Cambiar el porcentaje de descuento de una pendiente | **✓** | ✘ |
| 5 | Ver facturas pagadas | **✓** | ✘ |
| 6 | Descargar facturas en PDF | **✓** | ✘ |
| 7 | Ver facturas en PDF | **✓** | ✘ |

### Módulo de Venta

En el módulo de venta se llevará a cabo todo lo relacionado con el proceso de venta de productos. A él podrán acceder los dos tipos de usuarios, es decir, *administrador* y *empleado*.

|  |  |  |  |
| --- | --- | --- | --- |
| # | Tarea | Administrador | Empleado |
| 1 | Añadir productos al carrito | **✓** | **✓** |
| 2 | Eliminar productos del carrito | **✓** | **✓** |
| 3 | Ver detalles de un producto | **✓** | **✓** |
| 4 | Finalizar venta | **✓** | **✓** |
| 5 | Elegir el cliente de la venta | **✓** | **✓** |
| 6 | Elegir si paga en el acto si el cliente es de tipo *Mayorista* | **✓** | **✓** |
| 7 | Mostrar albarán | **✓** | **✓** |
| 8 | Mostrar factura | **✓** | **✓** |

### Otros

|  |  |  |  |
| --- | --- | --- | --- |
| # | Tarea | Administrador | Empleado |
| 1 | Loguearse en la aplicación | **✓** | **✓** |
| 2 | Ver su perfil de usuario | **✓** | **✓** |
| 3 | Modificar su perfil de usuario | **✓** | **✓** |
| 4 | Cerrar sesión | **✓** | **✓** |
| 5 | Cambiar de plantilla | **✓** | ✘ |

## Requisitos físicos

La utilización de la aplicación podrá ser desde un ordenador o móvil. Teniendo en cuenta que la aplicación no estará de ‘cara’ al público, se instalará en un servidor local en la tienda, permitiendo el acceso a los ordenadores y móviles de la empresa.

## Requisitos lógicos/funcionales

Los requisitos funcionales de una aplicación definen los comportamientos del sistema y constituyen las especificaciones que definen las distintas funcionalidades que implementa un sistema software. Los requerimientos funcionales pueden ser: cálculos, detalles técnicos, manipulación de datos y otras funcionalidades específicas que se supone, un sistema debe cumplir y que son revisados y aprobados por el cliente. Los requerimientos de comportamiento para cada requerimiento funcional serán los que se muestran en los casos de uso.

### Tipos de usuarios

**Administrador 🡪** Usuario que tendrá todos los privilegios y, por tanto, acceso a todas las funcionalidades que implementa la aplicación. Será el único que tenga acceso al módulo de Administración.

**Empleado 🡪** Usuario que sólo podrá acceder al módulo de Venta.

### Módulo de Administración

* Menú Agregar

Un usuario *Administrador* podrá añadir a la base de datos proveedores, categorías de productos, productos, clientes y usuarios. El proceso de agregación de los diferentes elementos es común, se validan los datos y si son correctos se guardan en la base de datos y ya podrían ser visibles desde el Menú de Listar. Destacando:

1. Para agregar *Productos* se selecciona una imagen, se copia esa imagen a la carpeta *images* del servidor y en la base de datos se guardan el nombre del archivo con su extensión.
2. Para agregar *Usuarios* se rellena en el formulario todos los campos menos la contraseña, que es enviada por email a la dirección de correo electrónico introducida. La contraseña enviada es un campo generado aleatoriamente.
3. Para agregar *Clientes* se elegirá el tipo, que podrán ser *Minorista* o *Mayorista.* Esto se explicará mejor en el siguiente apartado.

* Tipos de Clientes

El cliente *Mayorista* en el proceso de compra podrá elegir si paga en el acto o no, si no es así se generaría su albarán y se guardaría en una factura marcada pendiente, pudiendo pagar ésta más tarde, por lo que una factura podrá tener varios albaranes. En el caso que quiera pagar en el acto se generará su albarán y su factura correspondiente, marcándose ésta como pagada obviamente. Al cliente *Mayorista* se le podrá aplicar un descuento en cada factura.

El cliente *Minorista* será un cliente ‘normal’ que tendrá que pagar en el acto y, se generaría su albarán y su factura correspondiente.

* Menú Listar

El usuario *Administrador* podrá ver todos los elementos guardados en el sistema en formalista paginada, diferenciando obviamente por tipo de elemento (Proveedor, Categoría, Producto, Cliente y Usuario). Teniendo la opción en cada elemento de ver el elemento en detalle, si en la lista no cupieran todos los datos, modificarlo o, darlo de baja o de alta (según su estado).

Destacando que a un usuario sólo se podrá cambiar su estado (baja o alta). No se podrá modificar su cuenta ya que esto sólo lo podrá hacer el propio usuario cuando acceda a su perfil.

Además, se podrá buscar por cualquier campo en cada lista y en la de productos también se podrá mostrar por categoría.

* Gestión de Facturas

En este apartado el usuario *Administrador* podrá ver las facturas pendientes y facturas pagadas.

Por un lado en las facturas pendientes, se podrá ver cada factura en detalle, indicarle un descuento y marcarla como pagada/cobrada, en este caso se guardaría la fecha correspondiente.

Por otro lado en las facturas pagadas, se podrán descargar o verlas en el navegador en formato PDF.

* Configuración de plantillas

El usuario *Administrador* podrá elegir la plantilla que usará la página web, cambiando totalmente su aspecto. En principio, estarán disponibles dos tipos de plantillas para cada módulo (*Administración* y *Venta*) pudiéndose agregar en el futuro más plantillas fácilmente.

La forma en la que podemos cambiar de plantillas es parecida a la utilizada en [Joomla](https://es.wikipedia.org/wiki/Joomla) en su [Gestor de Plantillas](https://www.google.es/search?q=joomla+gestor+de+plantillas&espv=2&biw=1366&bih=643&source=lnms&tbm=isch&sa=X&ved=0ahUKEwiUvq7j8bXMAhUCWRQKHbgWAsgQ_AUIBigB#imgrc=QVXiGRlAAQy6AM%3A).

* Avisos de productos con stocks bajos

En este apartado aparecerán avisos cuando un producto tenga un stock de 10 productos o menos.

* Estadísticas

Este apartado se mostrará en la página principal del módulo y mostrará diversos datos, como el número de ventas por mes actual, mes anterior, semana actual y semana anterior, y un gráfico por el tipo de cliente. También se mostrará un gráfico con las facturas pagadas y no pagadas.

### Módulo de Venta

* Carrito

Se mostrarán todos los productos que están disponibles en la tienda, mostrándolos también por categorías. Pudiendo en cada producto añadirlo al carrito o verlo detalladamente.

En el carrito se podrá elegir la cantidad del producto que se va a comprar, verificando su stock, quitar un producto del carrito o vaciarlo completamente.

* Proceso de venta

Cuando se finaliza la venta se elige el tipo de cliente que realiza la compra. A continuación, explicaremos los posibles escenarios para cada tipo de *cliente:*

* Minorista: se realiza la compra, se genera su albarán y su factura correspondiente, y se pudiendo mostrar éstas en PDF.
* Mayorista: se realiza la compra y si se ha elegido pagar en el acto, se generaría su albarán y su factura correspondiente como en los clientes minoristas. Si no se elige pagar en el acto, se guardaría el albarán en la última factura *no pagada* del cliente y se podrá mostrar el albarán en PDF.

### Perfil de Usuario

En este apartado cada usuario podrá ver su perfil pudiendo modificar sus datos o cambiar su contraseña.

## Diagramas de casos de uso

Los diagramas de casos de uso se utilizan para ilustrar los requerimientos del sistema al mostrar cómo reacciona a eventos que se producen en su ámbito o en él mismo, siendo los más comunes los empleados para la captura de requisitos funcionales.

En el contexto de ingeniería del software, un caso de uso es una secuencia de interacciones que se desarrollarán entre un sistema y sus actores en respuesta a un evento que inicia un actor principal sobre el propio sistema. Los diagramas de casos de uso sirven para especificar la comunicación y el comportamiento de un sistema mediante su interacción con los usuarios y/u otros sistemas o, dicho de otra forma, un diagrama que muestra la relación entre los actores y los casos de uso en un sistema.

### Actores

Los actores son entidades que pueden interactuar o comunicarse con el mismo. Estos pueden ser usuarios humanos, hardware externo u otros sistemas. Un usuario tiene un nombre y se comunica con el sistema enviando y recibiendo estímulos o eventos.

Los actores representan un rol que pueden desempeñar alguien o algo que interactúa o que necesita intercambiar información con el sistema.

Hay 2 tipos de actores que interactuarán en Shop’s Admin. Por una parte los usuarios *Administradores* y por otra los usuarios *Empleados*, cuyos datos deben haber sido introducidos en el *Menú Agregar Usuario* en el *Módulo de Administración*. [Ver tipos de usuarios.](#_Tipos_de_usuarios)

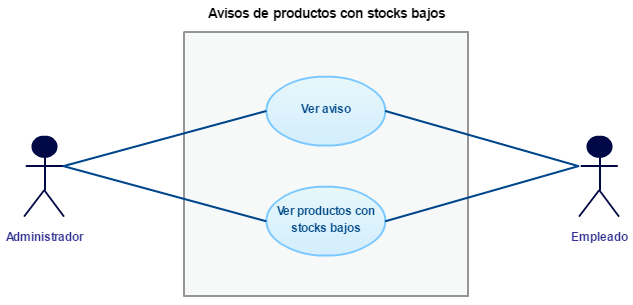
### Diagramas del Módulo Administración











### Diagramas del Módulo de Venta





### Diagrama del Perfil de Usuario



## Requisitos no funcionales

Los requisitos no funcionales son las propiedades o cualidades que nuestro sistema deberá poseer. El conjunto de estos requisitos los podemos dividir en diferentes áreas:

* **Disponibilidad.** Deber ser accesible desde todos los navegadores modernos.
* **Seguridad.** La información estará protegida tanto para visualizarlos como para actualizarlos. Cada usuario contará con un perfil, que le permitirá realizar sólo las operaciones asociadas a dicho perfil.
* **Usabilidad.** Se utilizará un diseño de las páginas del sitio web intuitivo, con una navegación clara, que permita al usuario realizar sus tareas de manera productiva y satisfactoria.
* **Rendimiento.** Será una aplicación web condicionada por diversos factores como la velocidad de acceso a Internet aunque, dado el pequeño volumen de datos a manejar y que accederán a la aplicación pocos usuarios a la vez, a priori éste no será un aspecto problemático.
* **Coste.** Las herramientas, tecnologías y plantillas utilizadas para el desarrollo del sistema son gratuitas.
* **Mantenibilidad.** Se utilizará una arquitectura Model-View-Controller para poder separar el interfaz del usuario, controladores con la lógica del negocio y módulos que transparentemente se conectan a la base de datos. Esto facilitará el análisis de nuevas funcionalidades, los cambios en un futuro y las pruebas.

## Prototipos

Es frecuente que los clientes no sepan lo que quieren, pero cuando ven algo y utilizan prototipos, pronto saben lo que no quieren.

Los prototipos son una representación limitada del software, permite a las partes probarlo en situaciones reales o explorar su uso. Son útiles para discutir y definir ideas.

Estos prototipos han sido diseñados por [Balsamiq Mockup's](#_Balsamiq_Mockup’s).

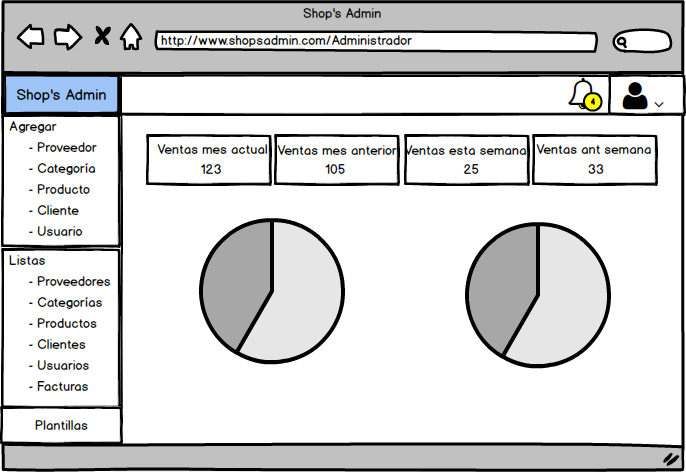
Por una parte, expondremos los prototipos del módulo de Administración y por otra, el módulo de Venta.

### Prototipos del Módulo de Administración

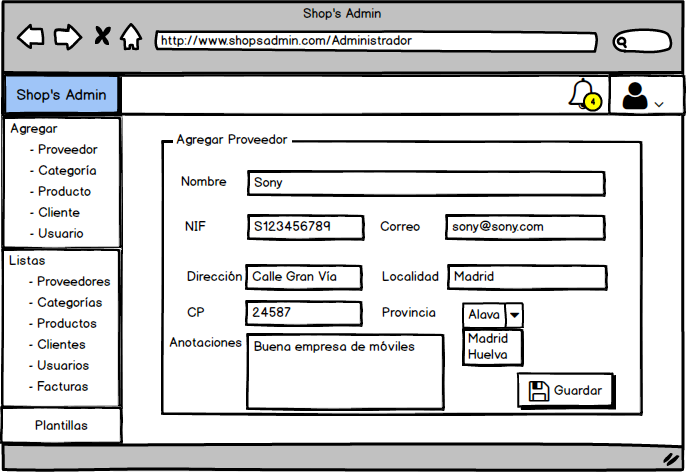
* **Pantalla de Login**



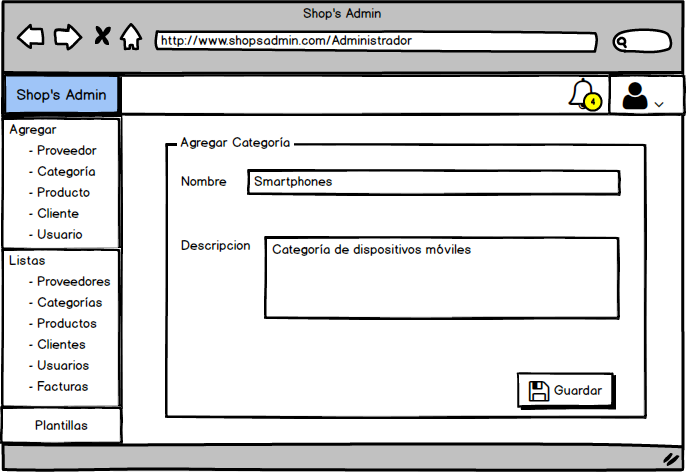
* **Pantalla de Inicio**



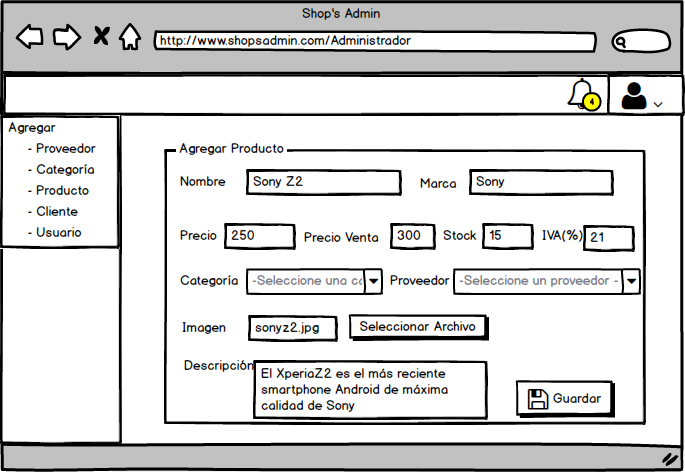
* **Pantalla de Agregar Proveedor**



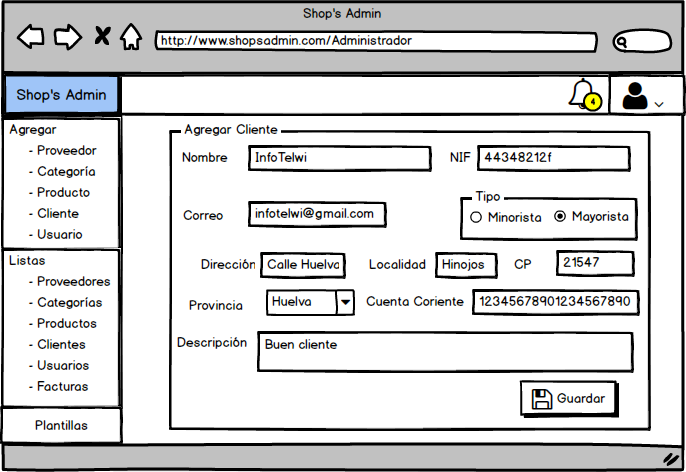
* **Pantalla de Agregar Categoría**



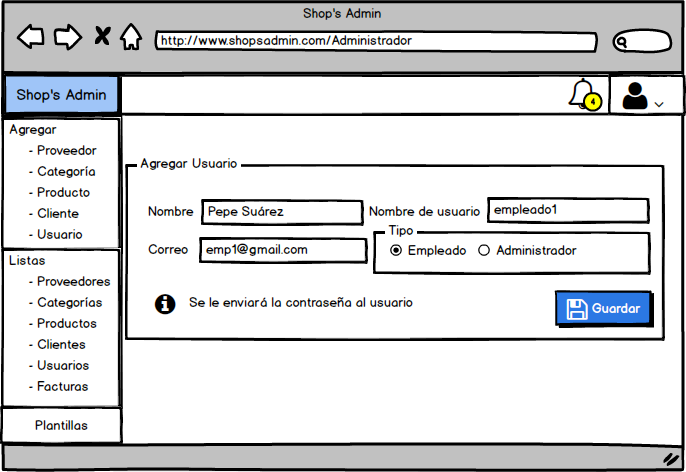
* **Pantalla de Agregar Producto**



* **Pantalla de Agregar Cliente**



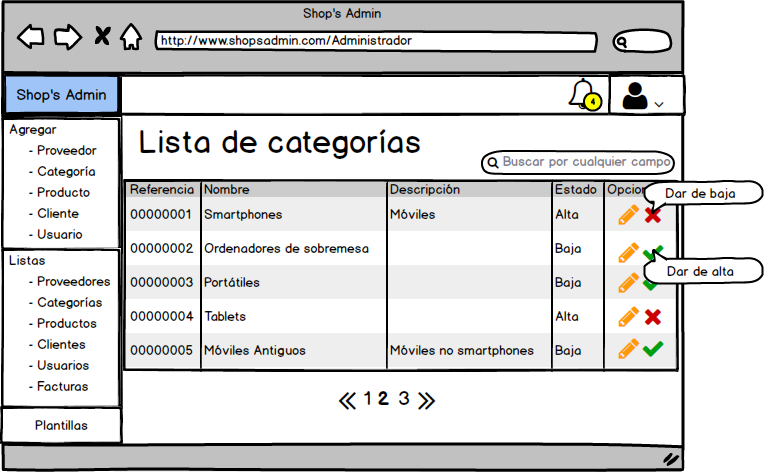
* **Pantalla de Agregar Usuario**



* **Pantalla de Lista de Proveedores**



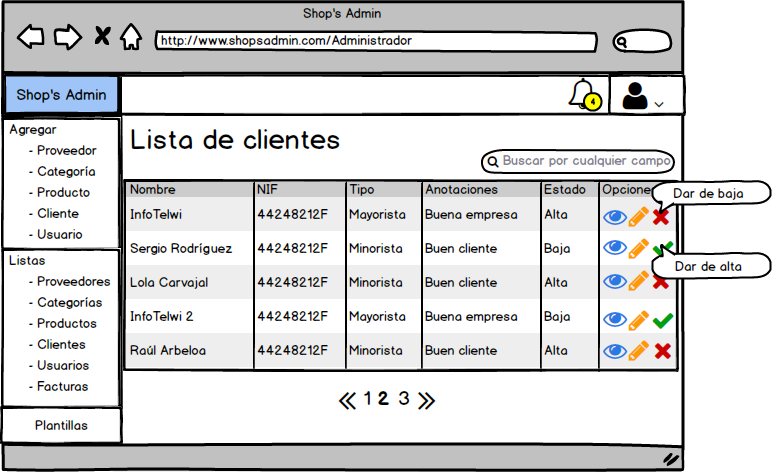
* **Pantalla de Lista de Categorías**



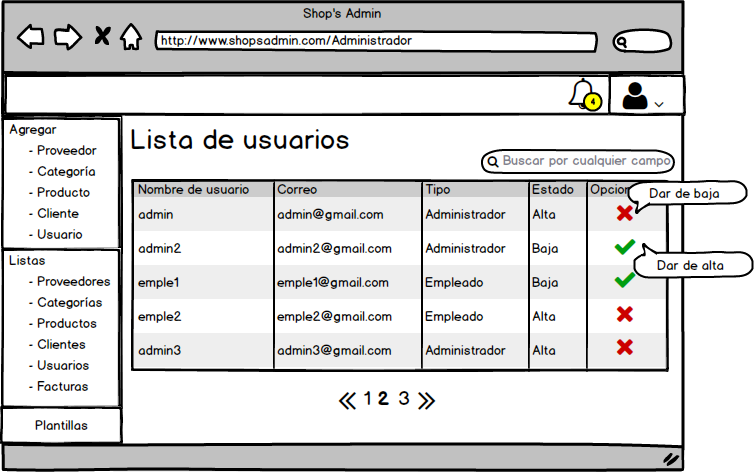
* **Pantalla de Lista de Productos**



* **Pantalla de Lista de Clientes**



* **Pantalla de Lista de Usuarios**



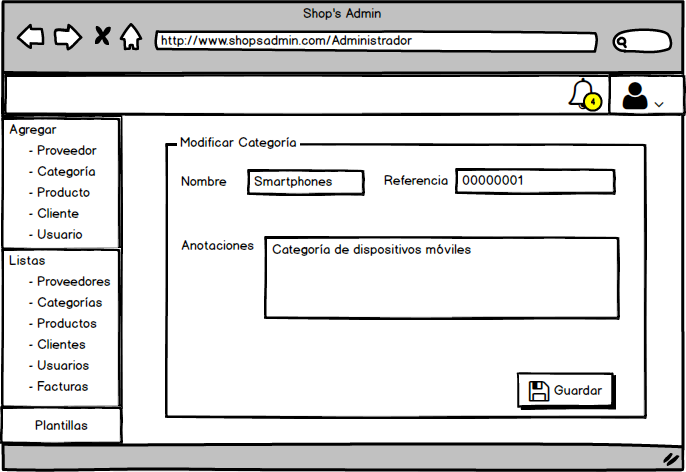
* **Pantalla de ejemplo de alta a un elemento(Proveedor, Categoría, Producto, Cliente o Usuario) pulsando en las opciones en **



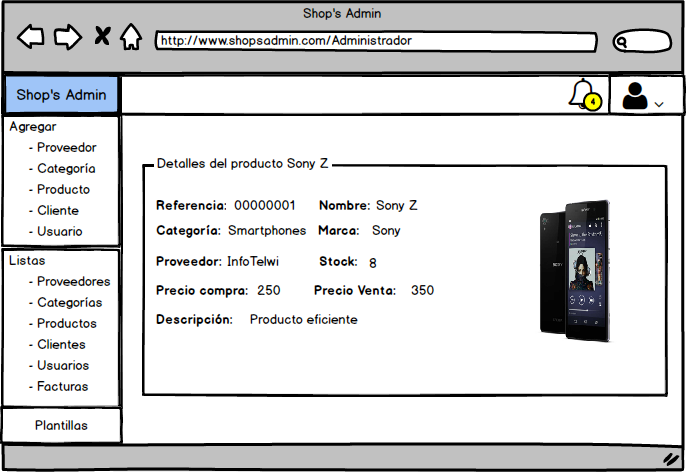
* **Pantalla de ejemplo de baja a un elemento(Proveedor, Categoría, Producto, Cliente o Usuario) pulsando en las opciones en** 



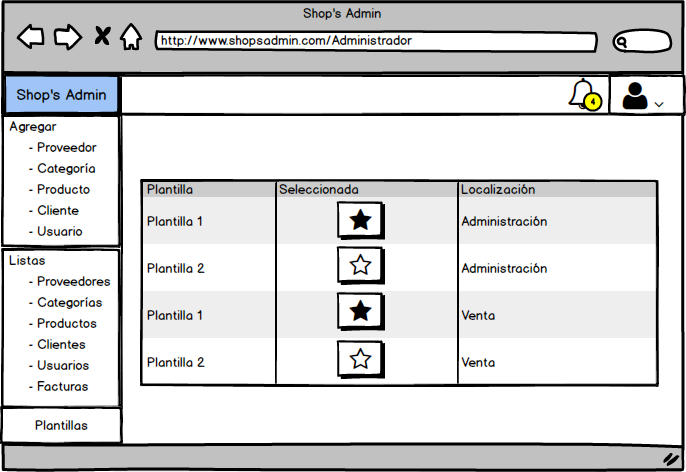
* **Pantalla de ejemplo de modificar un elemento(Proveedor, Categoría, Producto o Cliente) pulsando en las opciones en** 



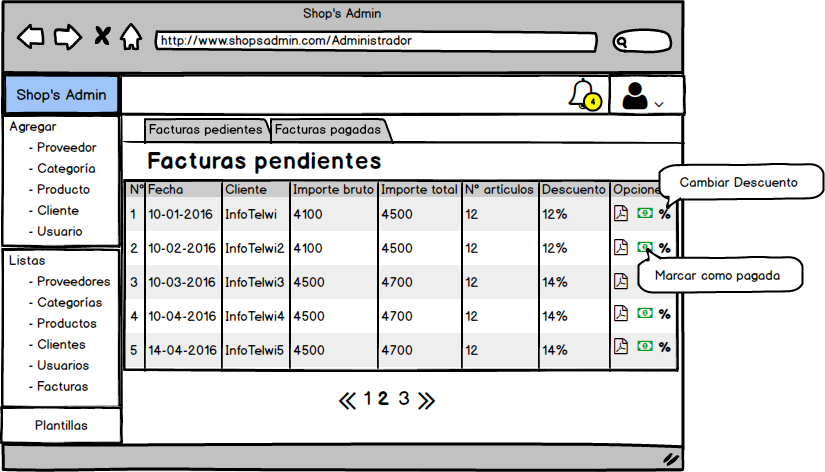
* **Pantalla de ejemplo de ver en detalle un elemento (Proveedor, Producto o Cliente) pulsando en las opciones en **



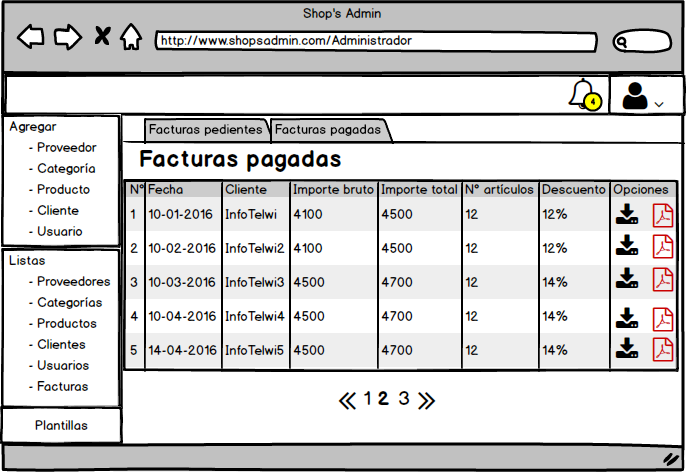
* **Pantalla de Configuración de Plantillas**



* **Pantalla de Lista de Facturas Pendientes**



* **Pantalla de Lista de Facturas Pagadas**



### Prototipos del Módulo de Venta

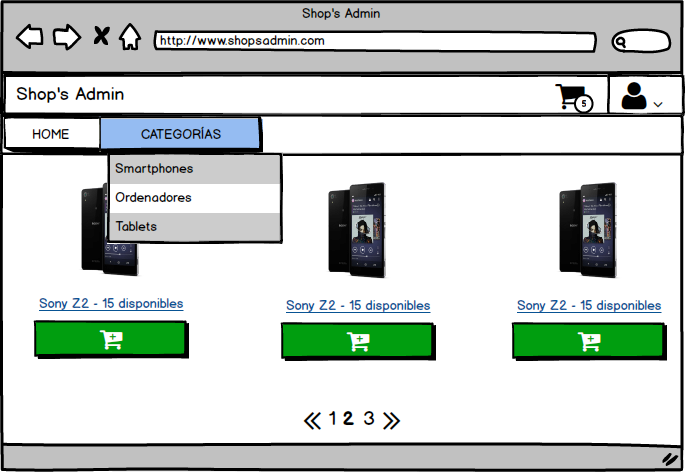
* **Pantalla de Login**



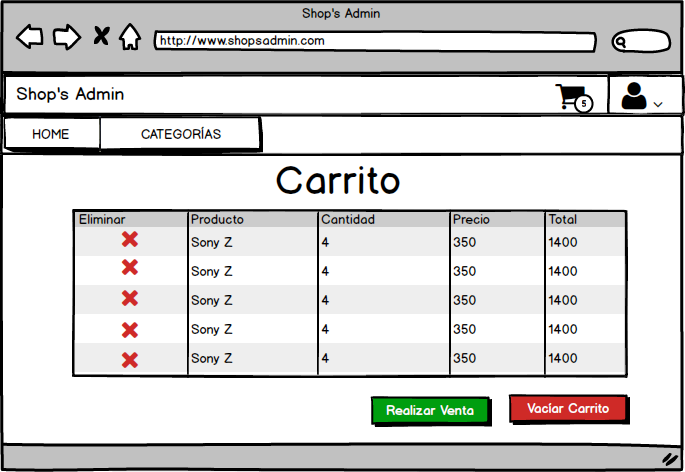
* **Pantalla de Inicio**



* **Pantalla de Categorías**



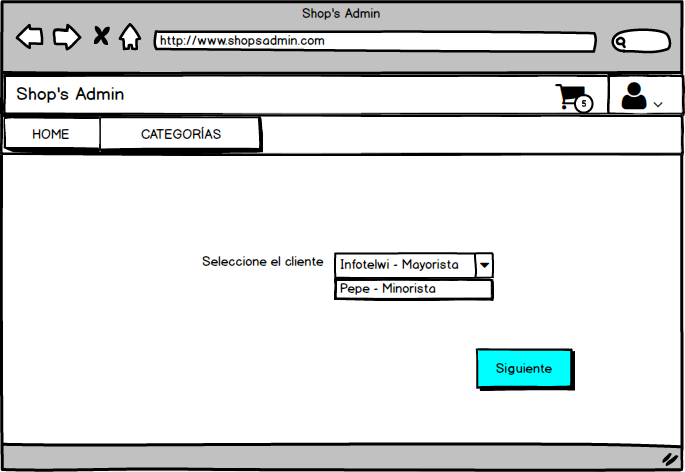
* **Pantalla del Carrito de la Compra**



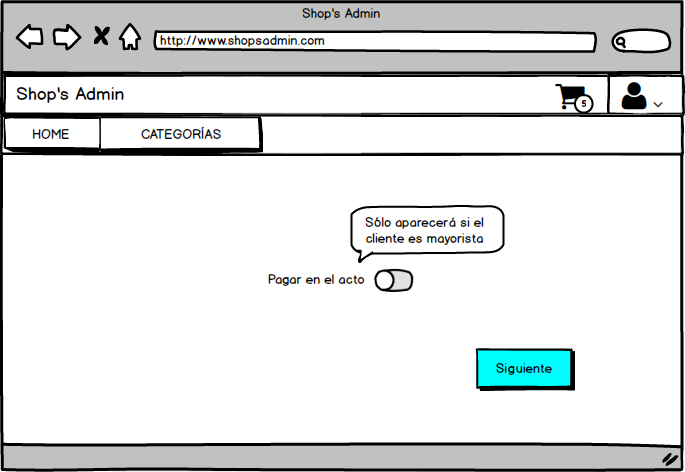
* **Pantalla del carrito de la compra vacío**



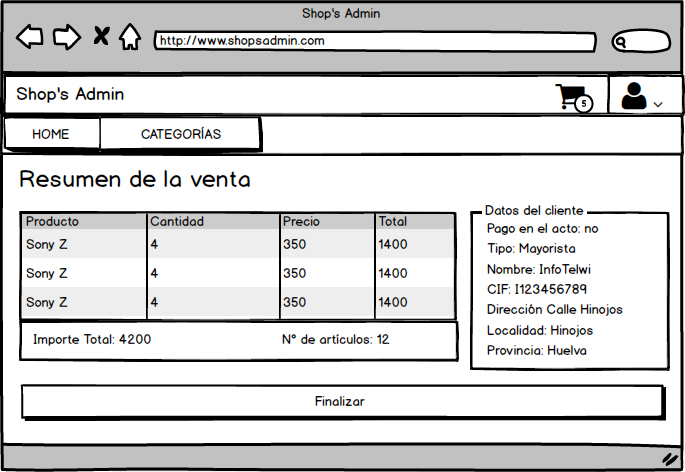
* **Pantalla de Venta – Paso 1**



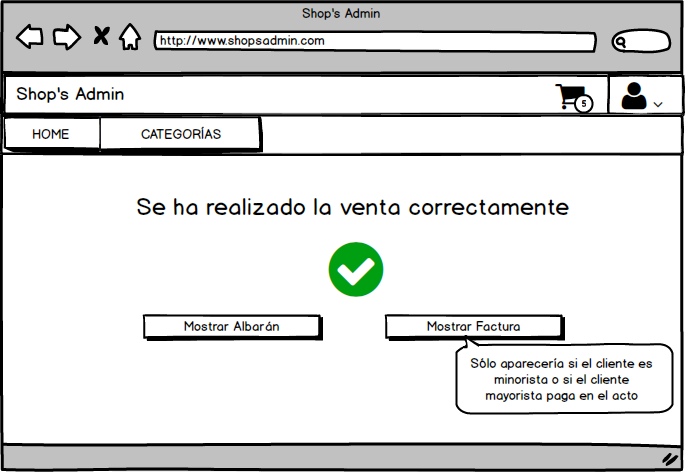
* **Pantalla de Venta – Paso 2**



* **Pantalla de Venta – Paso 3**



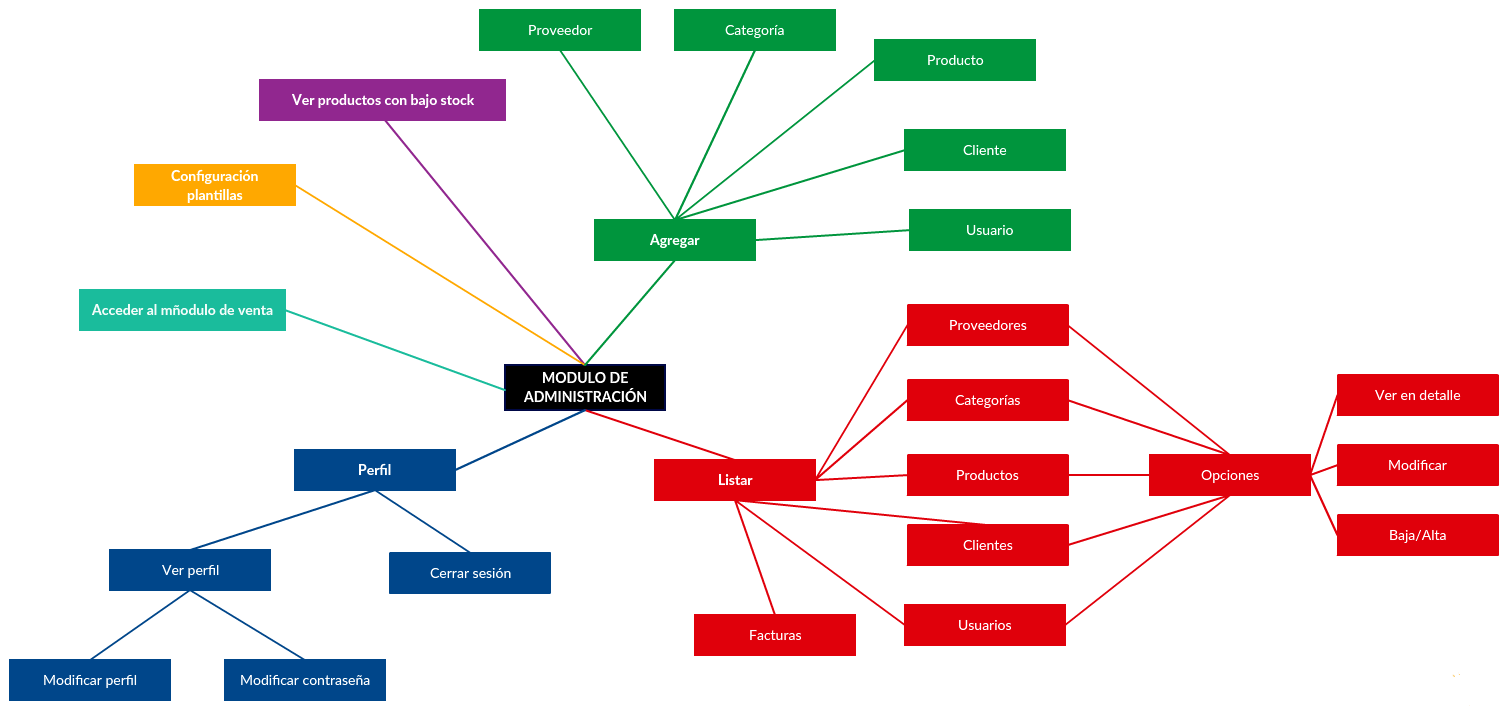
* **Pantalla de Venta – Paso 4**



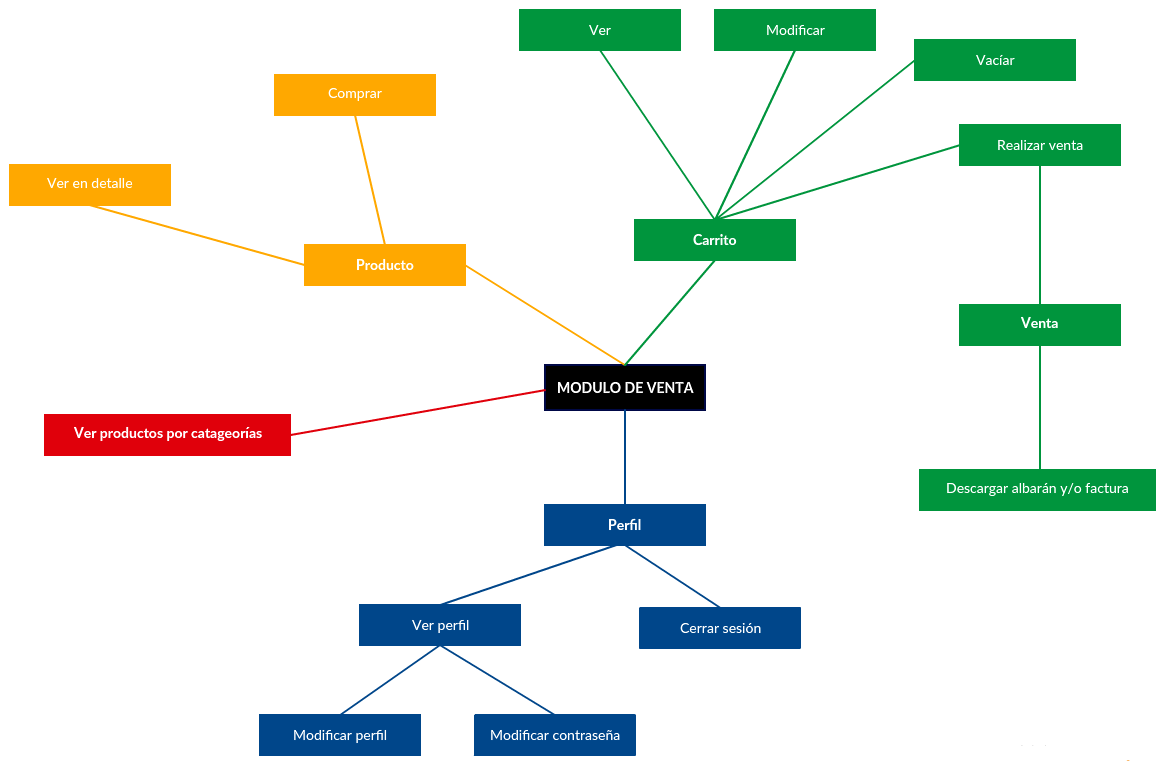
## Mapa Web

El mapa web del portal es un esquema conceptual que permite mostrar de una forma visual las funciones accesibles desde cada una de las páginas web. Representado en un diagrama jerárquico en el que cada nivel del diagrama muestra el número de clicks necesarios para acceder a una determinada página del sitio desde otra cualquiera.

### Mapa Web del Módulo de Administración



### Mapa Web del Módulo de Venta



# MODELO DE DATOS

El modelo de datos es un recurso orientado a hablar de una Base de Datos que permite describir:

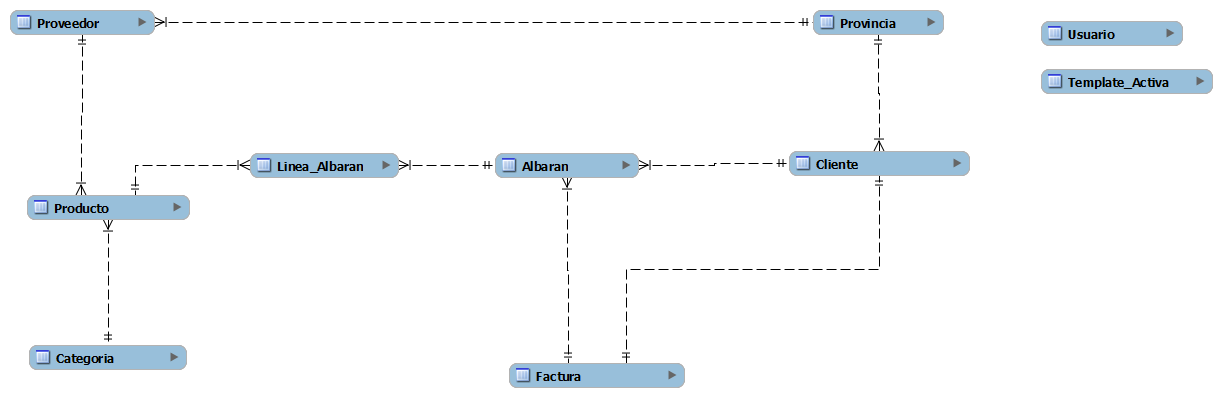
* Las estructuras de datos de la base: El tipo de los datos que hay en la base y la forma en que se relacionan.
* Las restricciones de integridad: Un conjunto de condiciones que deben cumplir los datos para reflejar correctamente la realidad deseada.
* Operaciones de manipulación de los datos: típicamente, operaciones de agregado, borrado, modificación y recuperación de los datos de la base.

## Esquema Entidad-Relación

El diagrama entidad-relación, es un modelo que clasifica los datos en 3 tipos fundamentales: Entidades, atributos y relaciones. Este tipo de diagramas pretende definir un determinado número de objetos de clasificación de datos mediante el reconocimiento intuitivo.

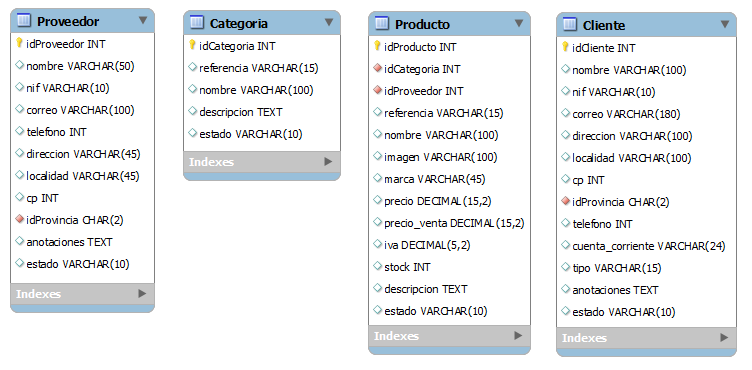
En este contexto una entidad es definida como una colección de objetos del mundo real con propiedades comunes. Un atributo es un elemento de datos que describe una propiedad de una entidad relación o de una relación. Las relaciones definen reglas de correspondencia entre las instancias de las entidades.

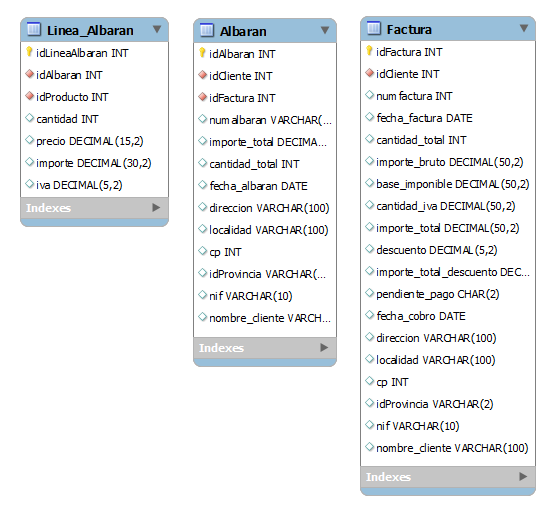
La forma de relacionarse entre entidades es definida por la cardinalidad de la relación que puede ser de uno a uno, de uno a muchos, o de muchos a muchos.

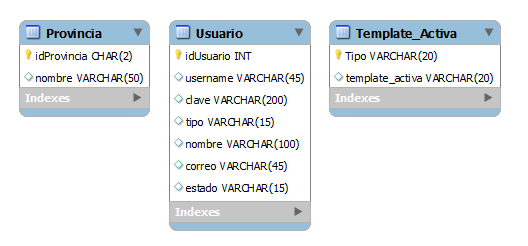


## Descripción de las Tablas

A continuación, se detallan cada una de las tablas y los campos que contienen así como las claves primarias y foráneas.







# TECNOLOGÍAS UTILIZADAS

## ¿Por qué usar Software libre?



En los últimos años hemos venido escuchando cada vez más los términos Software Libre (Free Software) y, más recientemente Software de fuentes abiertas (Open Source Software).

Estos términos se refieren al modelo de desarrollo y de distribución del software desarrollado cooperativamente. En vez de que el código del sistema o de cada uno de los programas sea un secreto celosamente guardado por la empresa que lo produce, éste es puesto a disposición del público, para que puedan modificar, mejorar o corregir.

“Software Libre” se refiere a la libertad de los usuarios para ejecutar, copiar, distribuir, estudiar, cambiar y mejorar el software. De modo más preciso, se refiere a cuatro libertades de los usuarios del software:

* La libertad de usar el programa, con cualquier propósito
* La libertad de estudiar cómo funciona el programa, y adaptarlo a tus necesidades
* La libertad de distribuir copias.
* La libertad de mejorar el programa y hacer públicas las mejoras a los demás, de modo que toda la comunidad se beneficie. El acceso al código fuente es un requisito.

**Razones para usar Software Libre**

* **Libre:** Entre otras cosas eres libre para usar, modificar, regalar o vender los programas de software libre
* **La copia es legal:** es legal repartir software libre a otras personas. Usar un sistema libre evita en gran medida los problemas de la piratería. Si lo natural es compartir tus programas con otras personas, con software libre es legal
* **Abierto:** Se puede usar el código de los programas y modificarlo.
* **Colaborativo:** El modelo de desarrollo de software libre es colaborativo y participativo. Todo lo puedes modificar o criticar.
* **Ayuda:** Existen innumerables grupos de usuarios que se ayudan entre sí a través de Internet. Es decir si te surge un problema es muy probable que a otras personas les ha ocurrido y su ayuda será la más valiosa que puedas encontrar.
* **Auditable:** El software libre se puede inspeccionar al disponer de su código fuente.
* **Robusto frente a los virus:** Existen muy pocos virus para GNU/Linux ya que el problema de los virus se debe al diseño del sistema operativo.
* **Personalizable:** Puedes personalizar toda la interfaz que el sistema le presenta al usuario.
* **Bajo costo:** De estudios realizados para empresas se han encontrado reducciones de hasta un 30% en TCO (Costo total de propiedad).
* **Reutilización de equipos:** GNU/Linux puede ejecutarse perfectamente en equipos que han sido desechados por las nuevas versiones de sistemas operativos propietarios.

## Parte Cliente

### HTML

HTML, sigla en inglés de *HyperText Markup Language* (lenguaje de marcas de hipertexto), hace referencia al lenguaje de marcado para la elaboración de páginas web. Es un estándar que sirve de referencia del software que conecta con la elaboración de páginas web en sus diferentes versiones, define una estructura básica y un código (denominado código HTML) para la definición de contenido de una página web, como texto, imágenes, videos, juegos, entre otros. Es un estándar a cargo del [*World Wide Web Consortium*](https://es.wikipedia.org/wiki/World_Wide_Web_Consortium) ([W3C](https://es.wikipedia.org/wiki/W3C)) o Consorcio WWW, organización dedicada a la estandarización de casi todas las tecnologías ligadas a la web, sobre todo en lo referente a su escritura e interpretación. Se considera el lenguaje web más importante siendo su ya que se ha impuesto en la visualización de páginas web y es el que todos los navegadores actuales han adoptado.

El lenguaje HTML basa su filosofía de desarrollo en la diferenciación. Para añadir un elemento externo a la página (imagen, vídeo, *script*, entre otros.), este no se incrusta directamente en el código de la página, sino que se hace una referencia a la ubicación de dicho elemento mediante texto. De este modo, la página web contiene solamente texto mientras que recae en el navegador web (interpretador del código) la tarea de unir todos los elementos y visualizar la página final. Al ser un estándar, HTML busca ser un lenguaje que permita que cualquier página web escrita en una determinada versión, pueda ser interpretada de la misma forma (estándar) por cualquier navegador web actualizado.

Sin embargo, a lo largo de sus diferentes versiones, se han incorporado y suprimido diversas características, con el fin de hacerlo más eficiente y facilitar el desarrollo de páginas web compatibles con distintos navegadores y plataformas (PC de escritorio, portátiles, teléfonos inteligentes, tabletas, etc. No obstante, para interpretar correctamente una nueva versión de HTML, los desarrolladores de navegadores web deben incorporar estos cambios y el usuario debe ser capaz de usar la nueva versión del navegador con los cambios incorporados.

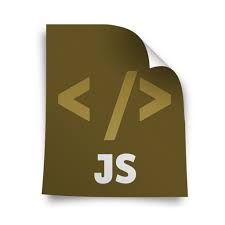
### CSS

Hoja de estilo en cascada o CSS (cascading style sheets) es un lenguaje usado para definir y crear la presentación de un documento estructurado escrito en HTML o XML2 (y por extensión en XHTML). El World Wide Web Consortium (W3C) es el encargado de formular la especificación de las hojas de estilo que servirán de estándar para los agentes de usuario o navegadores.

La idea que se encuentra detrás del desarrollo de CSS es separar la estructura de un documento de su presentación, que es lo recomendado.

La información de estilo puede ser definida en un documento separado o en el mismo documento HTML. En este último caso podrían definirse estilos generales en la cabecera del documento o en cada etiqueta particular mediante el atributo *style*.

### JavaScript

JavaScript (abreviado comúnmente "JS") es un lenguaje de programación interpretado, dialecto del estándar ECMA Script. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico.

Se utiliza principalmente en su forma del lado del cliente, implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas.

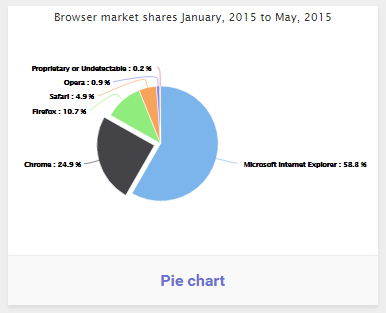
Todos los navegadores modernos interpretan el código JavaScript integrado en las páginas web. Para interactuar con una página web se provee al lenguaje JavaScript de una implementación del Document Object Model (DOM).

Tradicionalmente se venía utilizando en páginas web HTML para realizar operaciones y únicamente en el marco de la aplicación cliente, sin acceso a funciones del servidor. JavaScript se interpreta en el agente de usuario, al mismo tiempo que las sentencias van descargándose junto con el código HTML.

#### Highcharts

[Highcharts](http://www.highcharts.com/) es una biblioteca de gráficos escrito en JavaScript, que ofrece una manera fácil de añadir gráficos interactivos a su sitio web o aplicación web. Highcharts actualmente apoya gráficos lineales, splines, área, área spline, columnas, barras, circular, etc.

*Ejemplo:*

**

### Bootstrap

Twitter Bootstrap es un framework o conjunto de herramientas de software libre para diseño de sitios y aplicaciones web responsive. Contiene plantillas de diseño con tipografía, formularios, botones, cuadros, menús de navegación y otros elementos de diseño basado en HTML y CSS, así como, extensiones de JavaScript opcionales adicionales.

Bootstrap 3 se ha creado desde cero pensando en los móviles de manera que en lugar de incluir algunos estilos opcionales para móviles todo ello ya está incluido en el propio Bootstrap. Para ello ha definido un sistema de doce rejillas que permitirá distribuir y adaptar los contenidos de un sitio web para cualquier tipo de dispositivos y navegadores.

Bootstrap tiene un soporte relativamente incompleto paraHTML5yCSS 3, pero es compatible con la mayoría de los navegadores web. La información básica de compatibilidad de sitios web o aplicaciones está disponible para todos los dispositivos y navegadores. Existe un concepto de compatibilidad parcial que hace disponible la información básica. Bootstrap es modular y consiste esencialmente en una serie de hojas de estilo LESS que implementan la variedad de componentes de la herramienta. Los desarrolladores pueden adaptar el mismo archivo de Bootstrap, seleccionando los componentes que deseen usar en su proyecto. Los ajustes son posibles en una medida limitada a través de una hoja de estilo de configuración central. Los cambios más profundos son posibles mediante las declaraciones LESS. El uso del lenguaje de hojas de estilo LESS permite el uso de variables, funciones y operadores, selectores anidados, así como clases mixin.

#### FontAwesome

[FontAwesome](http://fontawesome.io/) es un conjunto de iconos diseñados por Bootstrap que se le pueden modificar su tamaño, color, sombra, y todo lo que se puede hacer con CSS.

### jQuery

jQuery es software libre y de código abierto que se ha convertido en la actualidad en uno de los complementos más esenciales para el desarrollo web, usado en millones de sitios en toda la web, ya que nos facilita en gran medida el desarrollo de aplicaciones enriquecidas del lado del cliente, en JavaScript, compatibles con todos los navegadores.

jQuery no es un lenguaje, sino una serie de funciones y métodos de JavaScript de manera que podremos referirnos a jQuery como un framework o incluso como una API de funciones, útiles en la mayoría de proyectos web. Por tanto, JavaScript es el lenguaje y jQuery es una librería que podremos utilizar opcionalmente para facilitar el desarrollo en JavaScript.

Antes de llegar jQuery los desarrolladores estaban obligados a discriminar entre los diversos navegadores, para ejecutar aquel código JavaScript que funcionaba en cada uno. Con la llegada de jQuery tendríamos la ventaja de que ya no necesitamos preocuparnos del navegador que se utilice pues será la propia librería la que hará el trabajo "sucio" y ejecutará el código que sea compatible con el software del cliente que está accediendo a nuestra web. Para ello usaremos las funciones que jQuery nos proporciona, dentro de un grandísimo abanico de funcionalidades que además se extiende por medio de miles de plugins que ofrece la comunidad para implementar cualquier tipo de comportamiento.

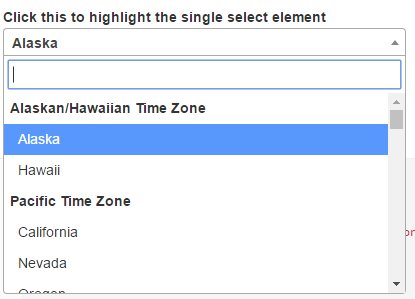
La característica principal de la biblioteca es que permite cambiar el contenido de una página web sin necesidad de recargarla, mediante la manipulación del árbol DOM y peticiones AJAX.

Cabría asimismo citar las siguientes características que jQuery nos aporta: selección de elementos DOM, control de eventos, manipulación de la hoja de estilos CSS, conjunto de efectos y animaciones, obtener información del navegador, operar con objetos y vectores, funciones para rutinas comunes o la de suministrar la compatibilidad con la mayoría de los navegadores disponibles en la actualidad.

#### Select2

[Select2](file:///C:\xampp\htdocs\Proyecto\Alumno\Documentos\Select2) es un plugin de jQuery que permite crear listas desplegables escribibles para poder buscar diferentes elementos

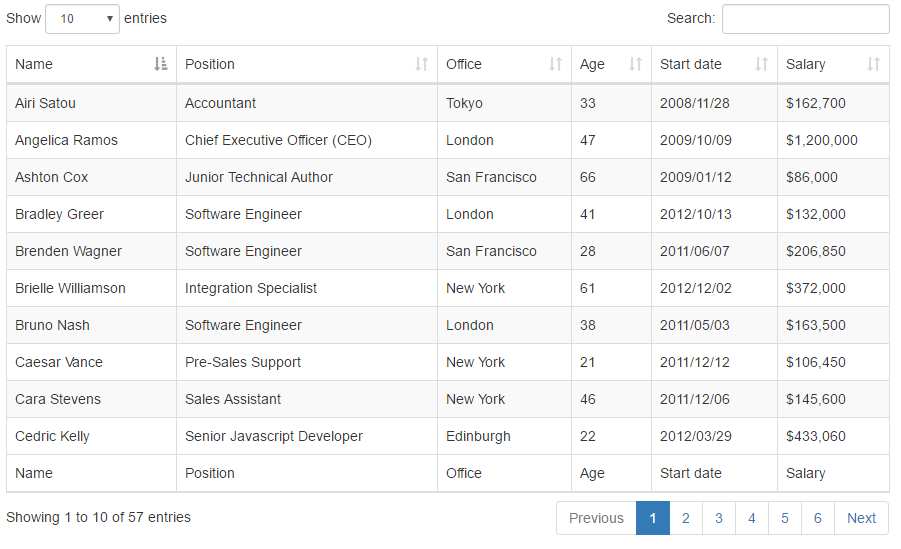
[*Ejemplos*](https://select2.github.io/examples.html)*:*



#### DataTables

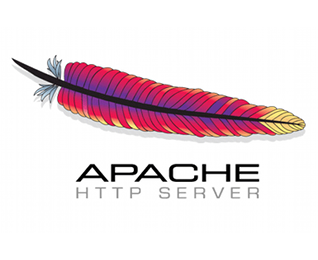
[DataTables](https://datatables.net/) es un plugin de jQuery. Es una herramienta muy flexible, muestra controles para ordenar por columna, paginar, elegir el número de elementos por página,… en cualquier tabla HTML. Además, se puede elegir diferentes de diseños, entre ellos [Bootstrap](#_Bootstrap) o [Foundation](http://foundation.zurb.com/).

[*Ejemplos*](https://datatables.net/examples/):



## Parte Servidor

### Servidor Apache

El Servidor HTTP Apache es un servidor web HTTP de código abierto, para plataformas Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh y otras, que implementa el protocolo HTTP/1.12y la noción de sitio virtual. El servidor Apache se desarrolla dentro del proyecto HTTP Server (httpd) de la Apache Software Foundation.

El proyecto Apache surgió para crear un recurso de libre acceso, robusto y de calidad comercial de un servidor Web HTTP. Este proyecto fue llevado a cabo por voluntarios de todo el mundo. En 1995 una primera versión del servidor resultó ser un gran éxito y en 1999 miembros del grupo Apache formaron el “Apache Software Foundation” que proporcionaría apoyo financiero, legal y organizacional para asegurar el futuro del proyecto, e impulsaría gran cantidad de proyectos de código libre.

El servidor Apache es modular, multiplataforma, de código libre, extensible y resulta muy fácil conseguir ayuda y soporte. Estas características lo convirtieron enseguida en una de las opciones más populares. Desde 1996 fue el más usado llegando al 70% de los sitios en 2005. Apache se muestra como un servidor muy competitivo, rápido, estable y con más ventajas que otros servidores, incluso en entornos con millones de visitas al día. La mayoría de las vulnerabilidades de la seguridad descubiertas y resueltas tan sólo pueden ser aprovechadas por usuarios locales y no remotamente. Sin embargo, algunas se pueden accionar remotamente en ciertas situaciones, o explotar por los usuarios locales malévolos en las disposiciones de recibimiento compartidas que utilizan [PHP](#_PHP) como módulo de Apache.

Apache presenta entre otras características altamente configurables, bases de datos de autenticación y negociado de contenido, pero fue criticado por la falta de una interfaz gráfica que ayude en su configuración.

### PHP

PHP es un lenguaje de programación de uso general de código del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico. Fue uno de los primeros lenguajes de programación del lado del servidor que se podían incorporar directamente en el documento [HTML](#_HTML) en lugar de llamar a un archivo externo que procese los datos. El código es interpretado por un servidor web con un módulo de procesador de PHP que genera la página Web resultante. PHP ha evolucionado por lo que ahora incluye también una interfaz de línea de comandos que puede ser usada en aplicaciones gráficas independientes. Puede ser usado en la mayoría de los servidores web al igual que en casi todos los sistemas operativos y plataformas sin ningún costo.

PHP fue creado por Rasmus Lerdorfen 1995, como un simple conjunto de scripts de Perl para el control de acceso a currículums on-line. El lenguaje fue creciendo en funcionalidad hasta permitir a los usuarios desarrollar sencillas aplicaciones Web dinámicas, llegando en1998 a publicarse la tercera versión de PHP que ya era capaz de competir con productos similares como ASP o JSP.

PHP se considera uno de los lenguajes más flexibles, potentes y de alto rendimiento conocidos hasta el día de hoy, lo que ha atraído el interés de múltiples sitios con gran demanda de tráfico, como Facebook, para optar por el mismo como tecnología de servidor.

La unión entre su potencia y su simplicidad hacen de PHP un lenguaje muy popular, soportado por la mayoría de los proveedores de Internet y usado en millones de dominios.

La base de datos más adecuada para trabajar con PHP es [MySQL](#_MySQL) ya que mySQL y PHP han sido desarrollados teniéndose en cuenta el uno al otro, ambos son [Open Source](#_¿Por_qué_usar) y tienen una amplia comunidad de apoyo, su eficiencia y simplicidad permiten un procesamiento más rápido

### MySQL

MySQL fue desarrollado en 1990 basada en SQL, una simple y pequeña base de datos, a partir de la que se crearía una base de datos más robusta cuya evolución se convertiría en MySQL.

MySQL es un sistema de gestión de bases de datos relacional, multihilo y multiusuario que cuenta en la actualidad con más de seis millones de instalaciones, que se distribuye bajo un sistema de doble licencia. Se puede obtener gratuitamente bajo licencia GPL y en el caso en el que se quiera distribuir una aplicación que no cumpla los términos GPL pero que incluya MySQL existe también una licencia comercial.

Ofrece una gran adaptabilidad permitiendo gran variedad de tipos de datos, y lo que es menos común, la posibilidad de elegir el tipo de tabla en el que guardar los registros, incluso permitiendo diferentes tipos de tablas en una misma base de datos. El motor de bases de datos MyISAM es el proporcionado por defecto y es capaz de manejar eficientemente grandes cantidades de datos, sin embargo existen muchos más tipos cada uno con sus ventajas.

MySQL es una base de datos rápida, fácil de configurar y de aprender a usar, con amplia documentación de apoyo, acceso al código fuente y buena portabilidad. La última versión de mySQL, la versión 5.1 es comparable a cualquier otra base de datos de pago como Oracle, Informix o SQL Server. Al contrario de proyectos como Apache, donde el software es desarrollado por una comunidad pública y los derechos de autor del código están en poder del autor individual, MySQL es patrocinado por una empresa privada, que posee el copyright de la mayor parte del código. Esto es lo que posibilita el esquema de licenciamiento anteriormente mencionado. Además de la venta de licencias privativas, la compañía ofrece soporte y servicios. Para sus operaciones contratan trabajadores alrededor del mundo que colaboran vía Internet.

MySQL es usado en la actualidad por numerosos sitios web grandes y populares, como son Wikipedia, Google, Twitter, Flickr o YouTube.

## Comunicación Cliente-Servidor. Ajax

AJAX, acrónimo de Asynchronous JavaScript And XML (JavaScript asíncrono y XML), es una técnica de desarrollo web para crear aplicaciones interactivas o RIA (Rich Internet Applications). Estas aplicaciones se ejecutan en el cliente, es decir, en el navegador de los usuarios mientras se mantiene la comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre las páginas sin necesidad de recargarlas, mejorando la interactividad, velocidad y usabilidad en las aplicaciones.

Es una tecnología asíncrona, en el sentido de que los datos adicionales se solicitan al servidor y se cargan en segundo plano sin interferir con la visualización ni el comportamiento de la página. JavaScript es el lenguaje interpretado (scripting language) en el que normalmente se efectúan las funciones de llamada de Ajax mientras que el acceso a los datos se realiza mediante XMLHttpRequest, objeto disponible en los navegadores actuales. En cualquier caso, no es necesario que el contenido asíncrono esté formateado en XML.

## Otros

### CodeIgniter

[CodeIgniter](https://www.codeigniter.com/) es un **framework** PHP para la creación rápida de aplicaciones web, y por tanto, un programa o aplicación web desarrollada en PHP para la creación de cualquier tipo de aplicación web bajo PHP. Es un producto de código libre, libre de uso para cualquier aplicación.

Como cualquier otro framework, CodeIgniter contiene una serie de librerías que sirven para el desarrollo de aplicaciones web y además propone una manera de desarrollarlas que debemos seguir para obtener provecho de la aplicación, marcando una manera específica decodificar las páginas web y clasificar sus diferentes scripts, que sirve para que el código esté organizado y sea más fácil de crear y mantener. CodeIgniter implementa el proceso de desarrollo llamado Modelo-Vista-Controlador (MVC), que es un estándar de programación de aplicaciones, utilizado tanto para hacer sitios web como programas tradicionales. Este sistema tiene sus características, que veremos en artículos siguientes.

### Github

Un **control de versiones** que permite gestionar y controlar los diversos cambios que se realizan sobre los elementos de la aplicación o de alguna de las opciones de configuración de la misma.

En el desarrollo de la aplicación he utilizado como herramienta de control de versiones [GitHub](https://github.com). Github es uno de los sistemas de control de versiones más populares entre los desarrolladores, un excelente servicio de alojamiento de repositorios de software con este sistema, que ofrece hoy en día un conjunto de características muy útiles para el trabajo en equipo, no en vano, es el servicio elegido por proyectos de software libre como jQuery, reddit, Sparkle, curl, Ruby on Rails, node.js, ClickToFlash, Erlang/OTP, CakePHP, Redis, y otros muchos. Además, algunas de las grandes empresas de Internet, como Facebook, alojan ahí sus desarrollos públicos, tales como el SDK, librerías, ejemplos, etc.

Al alojar el código, podemos decidir si éste será público o su visibilidad estará limitada a un grupo de usuarios determinado que serán los únicos que tendrán permiso para acceder al mismo y subir cambios. Asimismo, GitHub posee un visor de código mediante el cual, a través del navegador, podremos consultar en un instante el contenido de un determinado fichero, con su resaltado de sintaxis correspondiente para el lenguaje en el que esté escrito. Por supuesto, se puede navegar por cualquiera de las versiones del mismo, de modo que podemos ver el contenido de un fichero antes de que se dieran varias actualizaciones de cambios. Seleccionar un determinado commit, tag o rama es cuestión de pocos clicks.

Por último, para sincronizar los archivos se ha utilizado la [aplicación de escritorio de GitHub](https://desktop.github.com/) y el repositorio es [**éste**](https://github.com/isacm94/Proyecto/).

### NetBeans

[NetBeans](https://netbeans.org/)es un **entorno de desarrollo integrado (IDE)**libre, hecho principalmente para el lenguaje de programación Java. Existe además un número importante de módulos para extenderlo. NetBeans IDEes un producto libre y gratuito sin restricciones de uso.

NetBeans es un proyecto de código abierto de gran éxito con una gran base de usuarios, una comunidad en constante crecimiento. Sun MicroSystems fundó el proyecto de código abierto NetBeans en junio de 2000 y continúa siendo el patrocinador principal de los proyectos.

La plataforma NetBeans permite que las aplicaciones sean desarrolladas a partir de un conjunto de componentes de software llamados *módulos*. Un módulo es un archivo Java que contiene clases de java escritas para interactuar con las APIs de NetBeans y un archivo especial (manifest file) que lo identifica como módulo. Las aplicaciones construidas a partir de módulos pueden ser extendidas agregándole nuevos módulos. Debido a que los módulos pueden ser desarrollados independientemente, las aplicaciones basadas en la plataforma NetBeans pueden ser extendidas fácilmente por otros desarrolladores de software.

Entre sus características se encuentra un sistema de proyectos basado en Ant, control de versiones y refactoring, pudiendo desarrollar en Java EE, PHP, C/C++, HTML, XML, CSS, JavaScript, etc.

### MySQL Workbench

[MySQL Workbench](https://www.mysql.com/products/workbench/) es una herramienta visual de diseño de bases de datos que integra desarrollo de software, administración de bases de datos, diseño de bases de datos, creación y mantenimiento para el sistema de base de datos MySQL.

### FileZilla

[FileZilla](https://filezilla-project.org/), para la transferencia de archivos al servidor.

### Balsamiq Mockup’s

[Balsamiq Mockup’s](https://balsamiq.com/) utilizado para el desarrollo de prototipos de baja fidelidad.

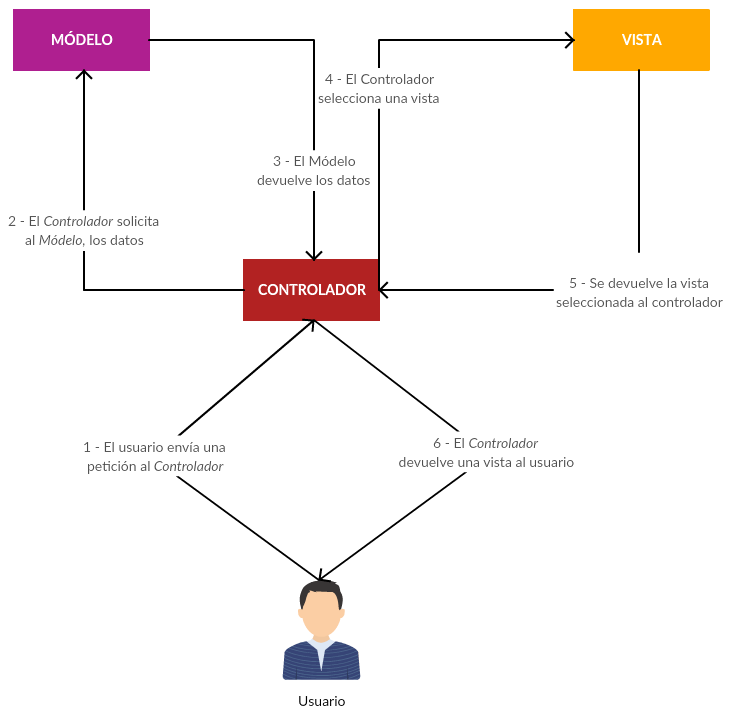
### Creately UML

[Creately](http://creately.com/) es una aplicación web que permite crear gráficos [UML](https://es.wikipedia.org/wiki/Lenguaje_unificado_de_modelado) con un diseño moderno y de forma gratuita.

# ARQUITECTURA DE LA APLICACIÓN

El modelo-vista-controlador (MVC) es un patrón de arquitectura de software que separa los datos y la lógica de negocio de una aplicación de la interfaz de usuario y el módulo encargado de gestionar los eventos y las comunicaciones. Para ello MVC propone la construcción de tres componentes distintos que son el modelo, la vista y el controlador, es decir, por un lado define componentes para la representación de la información, y por otro lado para la interacción del usuario.

Este patrón de arquitectura de software se basa en las ideas de reutilización de código y la separación de conceptos, características que buscan facilitar la tarea de desarrollo de aplicaciones y su posterior mantenimiento.



## Componentes

1. **MODELO**: Es la representación de la información con la cual el sistema opera, por lo tanto gestiona todos los accesos a dicha información, tanto consultas como actualizaciones de la base de datos, implementando también los privilegios de acceso que se hayan descrito en las especificaciones de la aplicación (lógica de negocio). Envía a la 'vista' aquella parte de la información que en cada momento se le solicita para que sea mostrada (típicamente a un usuario). Las peticiones de acceso o manipulación de información llegan al *modelo* a través del *controlador.*
2. **CONTROLADOR**: Responde a eventos (usualmente acciones del usuario) e invoca peticiones al *modelo* cuando se hace alguna solicitud sobre la información (por ejemplo, editar un documento o un registro en una base de datos). También puede enviar comandos a su *vista* asociada si se solicita un cambio en la forma en que se presenta el modelo (por ejemplo, desplazamiento o scroll por un documento o por los diferentes registros de una base de datos), por tanto se podría decir que el *controlador* hace de intermediario entre la *vista* y el *modelo.*

## Ventajas y desventajas del uso del patrón MVC

**🡪 Ventajas**

* La implementación se realiza de forma modular.
* Sus vistas muestran información actualizada siempre. El programador no debe preocuparse de solicitar que las vistas se actualicen, ya que este proceso es realizado automáticamente por el modelo de la aplicación.
* Cualquier modificación que afecte al dominio, como aumentar métodos o datos contenidos, implica una modificación sólo en el modelo y las interfaces del mismo con las vistas, no todo el mecanismo de comunicación y de actualización entre modelos.
* Las modificaciones a las vistas no afectan al modelo de dominio, simplemente se modifica la representación de la información, no su tratamiento.
* MVC está demostrando ser un patrón de diseño bien elaborado pues las aplicaciones que lo implementan presentan una extensibilidad y una mantenibilidad únicas comparadas con otras aplicaciones basadas en otros patrones.
* Clara separación entre interfaz, lógica de negocio y de presentación, que además provoca parte de las ventajas siguientes.
* Reutilización de los componentes.

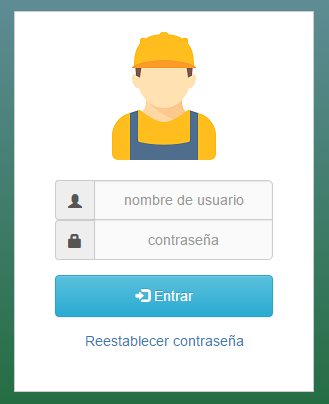
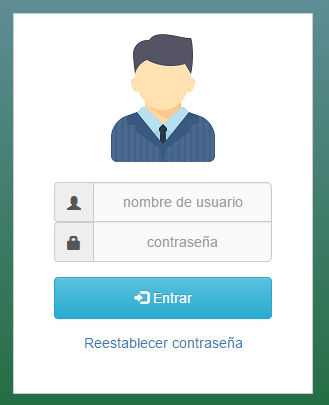
🡪 **Desventajas**

* Tener que ceñirse a una estructura predefinida, lo que a veces puede incrementar la complejidad del sistema. Hay problemas que son más difíciles de resolver respetando el patrón MVC.
* La curva de aprendizaje para los nuevos desarrolladores se estima mayor que la de modelos más simples como Webforms.
* La distribución de componentes obliga a crear y mantener un mayor número de ficheros

# MANUAL DE USUARIO

## Login

El objetivo de esta pantalla es la de crear un sistema de autenticación capaz de identificar a los usuarios que desean acceder a la aplicación por medio de un nombre de usuario y una contraseña. Existirán dos logins, uno para la parte de Administración y otro para la de Venta. Ya que a la parte de Administración sólo podrán acceder los [usuarios](#_Tipos_de_usuarios) de tipo de Administradores.



## Módulo de Administración

### Inicio