

# PYTHON

PROYECTO FINAL

Adrián Jiménez Nieto



## ADRIÁN JIMÉNEZ NIETO

RESPONSABLE DEL DESARROLLO DE LA APLICACIÓN

Fecha: 09/03/2023



## DESARROLLO

1. Se nos pide desarrollar una aplicación web con Python para el manejo de una empresa ficticia de suministros informáticos. El proyecto lo he realizado enteramente en Python, HTML, CSS y JavaScript haciendo uso del framework Django, así como la librería 'django-seed' que me permitió poblar la base de datos y 'chart-js' que utilicé para renderizar los gráficos de compra-ventas que se nos exigen.
2. Aunque HTML, CSS Y JavaScript no entran en el scope de este curso me parece muy interesante saber un poco sobre ellos ya que me interesa mucho la parte FrontEnd de la programación Web.
3. He decidido usar el IDE VS-Code para conocer más IDEs y así ampliar conocimiento e investigar qué IDE se ajusta más a mis necesidades.

## INSTALACION

4. Para instalar el proyecto en otros ordenadores es necesario tener instalado al menos el framework 'Django'. Aunque el entorno virtual que yo facilito ya lo tiene instalado, para instalar el framework hay que ir a la consola de comandos (dentro de 'venv') y teclear el comando '**pip install django**'. Una vez instalado podemos teclear '**django-admin.py --version**' para comprobar que se ha instalado correctamente.
5. Para iniciar un proyecto en Django introduciremos el comando '**django-admin startproject**' y el nombre del proyecto. En nuestro, entraremos a la carpeta principal del proyecto llamada 'SuministrosInformaticos' a través de consola y teclearemos el comando '**python manage.py runserver**' para correr el servidor.

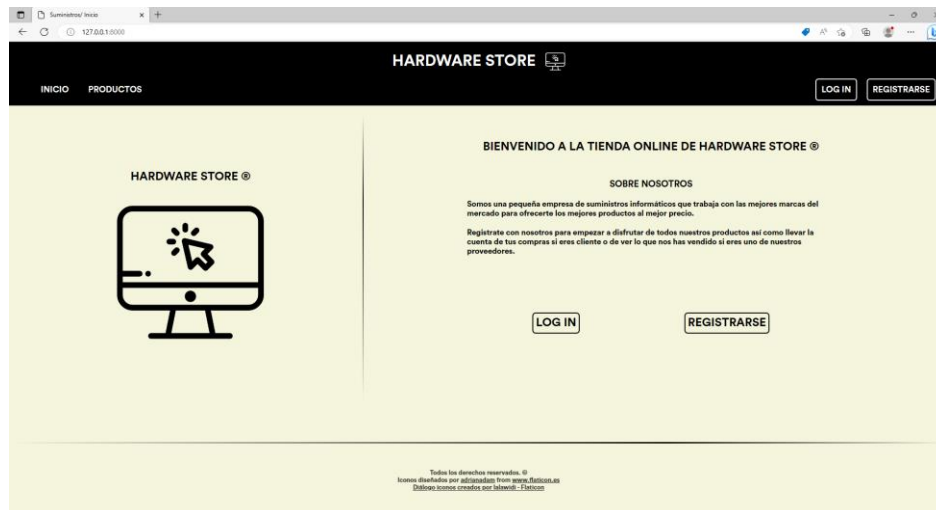
```
(venv) PS D:\SuministrosInformaticos\suministros> python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified some issues:

WARNINGS:
?: (staticfiles.W004) The directory '/var/www/static/' in the STATICFILES_DIRS setting does not exist.

System check identified 1 issue (0 silenced).
April 22, 2023 - 14:48:36
Django version 4.2, using settings 'suministros.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

6. El servidor corre en el puerto 8000 de localhost, para ver la aplicación web iremos a cualquier navegador e introduciremos <http://127.0.0.1:8000/> para ver la pantalla de inicio de nuestra aplicación.



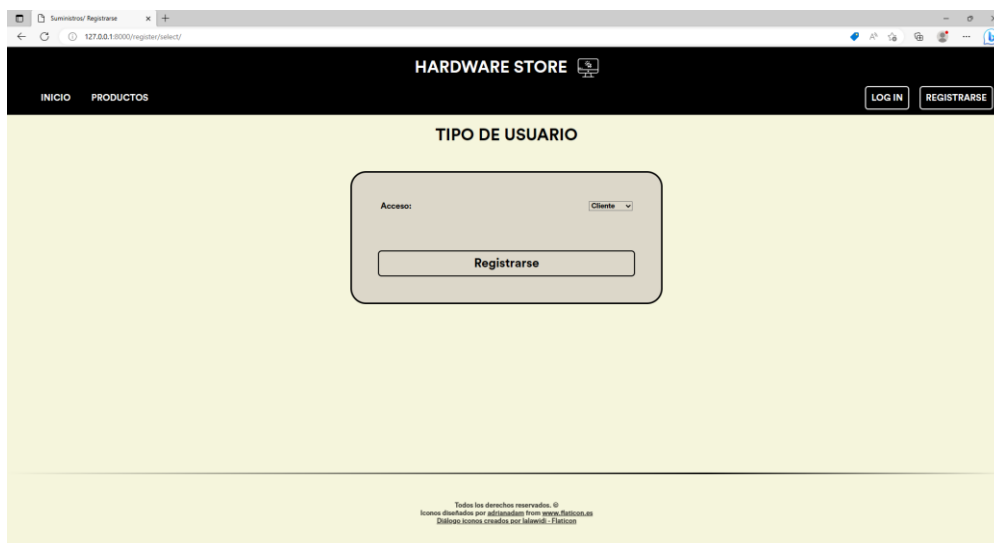
## USUARIOS

7. Lo primero es definir los usuarios que tendrán acceso a la aplicación web, en este caso tendremos 3 tipos de usuarios. El primero, el usuario tipo 'cliente', dicho usuario tendrá la capacidad de ver los productos de la empresa, comprar dichos productos e incluso añadirlos a un carrito de la compra, y ver unas gráficas de la cantidad de productos comprados, tanto por nombre de producto como por fecha en la que se compró. El segundo, el usuario tipo 'proveedor' tendrá las mismas funcionalidades que el tipo 'cliente' pero con la diferencia de que en los gráficos verán los productos que se han vendido (productos que les pertenecen). Finalmente, tendremos el usuario tipo 'staff' que aparte de tener las mismas funcionalidades que los otros dos, en su apartado de gráficas podrán ver todas las ventas y todas las compras de la empresa a los proveedores, tanto por producto como por fecha. Estos últimos, a su vez, tendrán la capacidad de ver un apartado de mensajes de control de stock que les avisará de cuándo hay que pedir más stock de un determinado producto. Cabe mencionar que todos los usuarios tienen la capacidad de editar tanto su perfil como su contraseña.

8. A parte de estos tres tipos de usuarios tendremos un último tipo, el usuario 'superuser' capaz de acceder al panel de administración de Django donde podrá modificar cualquiera de los usuarios mencionados anteriormente.
9. Los usuarios están modelados en base al modelo por defecto 'User' que viene con Django. Todos los modelos heredan de este otro haciendo más fácil el manejo de la autenticación de usuarios y los permisos de los mismos. Así pues en nuestra base de datos tendremos una tabla dónde almacenaremos todos los usuarios (independientemente de su tipo) y tablas secundarias que almacenaran la información correspondiente a cada tipo de usuario. Estas segundas tablas se enlazan a la tabla 'Users' a través de una clave foránea.

## REGISTRO E INICIO DE SESION

10. Una vez estamos en la página principal de nuestra aplicación web, tenemos varias opciones, la primera es registrarnos en el sistema creando un usuario del tipo que queramos (cliente, proveedor o staff). Para ello podemos hacer click tanto en el enlace 'Registrarse' que se encuentra debajo de la descripción de la empresa, o bien hacer click en el enlace que se encuentra en la barra de navegación a la derecha. Una vez entremos en la página de registro se nos pedirá que elijamos el tipo de usuario que queremos crear en un menú desplegable.



11. En función del tipo de acceso que elijamos se nos enviará un formulario u otro que corresponde a los atributos que poseen cada usuario. En este caso vamos a elegir un usuario del tipo 'Cliente'.

### REGISTRARSE

**NOMBRE:**

**APELLIDOS:**

**NOMBRE DE USUARIO:**

Requerido. Aceptados caracteres alfanuméricos y @/./+/-/\_ solo.

**CONTRASEÑA:**

Tu contraseña no puede ser similar al resto de información.  
Tu contraseña debe tener al menos 8 caracteres.  
Tu contraseña no debe ser usada comúnmente.  
Tu contraseña no puede ser solo numérica.

**CONFIRMACION CONTRASEÑA:**

Introduzca la misma contraseña para verificar.

**IDENTIFICACION:**

**DIRECCION DE EMAIL:**

**DIRECCION:**

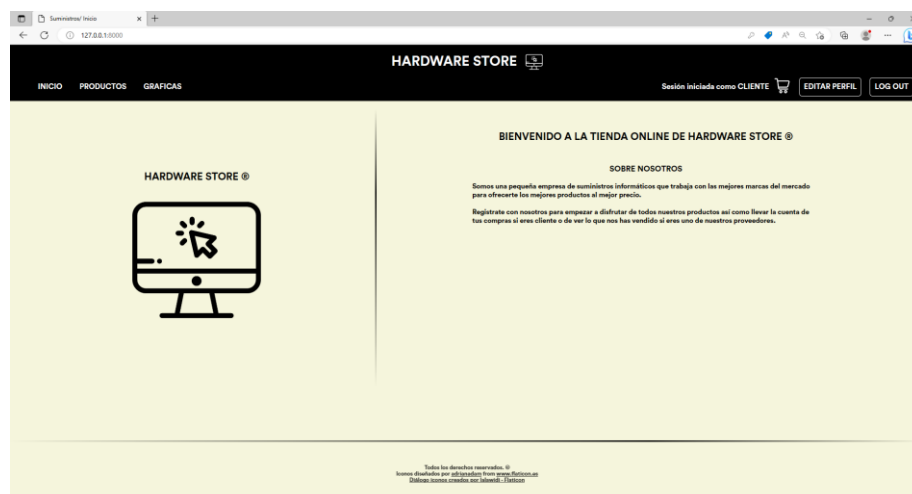
**CÓDIGO POSTAL:**

**AÑO DE NACIMIENTO:**

**Registrarse**

¿Ya tienes una cuenta? Inicia sesión [AQUI](#) !!

12. Este es el formulario de tipo cliente, los tres tipos de usuarios comparten los atributos nombre, nombre de usuario, email, y los dos campos de contraseña. Cuando lo rellenemos, el sistema nos mandará a la página de inicio pero con la sesión ya iniciada.



13. Como vemos, al iniciar la sesión el aspecto de la web cambia un poco. Aparecen nuevos botones en la barra de navegación como el de 'Graficas', el botón del carrito de la compra, otro para editar nuestro perfil y el botón 'LOG OUT' para cerrar la sesión en el sistema. A parte a la izquierda del carrito de la compra aparecerá un texto informativo que nos indicará el nombre de usuario con el que estamos logeados.
14. Una vez cerremos la sesión, para volver a entrar en el sistema es tan sencillo como pulsar el botón 'LOG IN' y rellenar el formulario correspondiente donde se nos pedirá que introduzcamos nuestro nombre de usuario y nuestra contraseña.

## EDICIÓN DE PERFIL

15. Hablemos ahora de la pestaña de edición de perfil, la cual se encuentra en nuestro menú de navegación a la derecha del todo. Si hacemos click en ella se nos abrirá un formulario en el que podemos modificar toda la información del usuario con el que estemos logeados.

**EDITAR USUARIO**

**NOMBRE:**

adrian

**APELLIDOS:**

añskdjf

**NOMBRE DE USUARIO:**

cliente

Requerido. Aceptados caracteres alfanumericos y @/./+/-/\_ solo.

**IDENTIFICACION:**

00000000X

**DIRECCION DE EMAIL:**

adsfjl@ajsfd.com

**DIRECCION:**

No adress

**CÓDIGO POSTAL:**

00000

**AÑO DE NACIMIENTO:**

04/04/2023

**EDITAR PERFIL**

Sí quieres cambiar tu contraseña haz click

[AQUI](#)

*Edición Perfil Cliente*

### EDITAR USUARIO

**NOMBRE:**

**NOMBRE DE USUARIO:**

Requerido. Aceptados caracteres alfanumericos y @/./+/-/\_ solo.

**CIF:**

**DIRECCION DE EMAIL:**

**DIRECCION:**

**CÓDIGO POSTAL:**

**INFORMACIÓN EXTRA:**

**EDITAR PERFIL**

Sí quieres cambiar tu contraseña haz click [AQUÍ](#)

*Edición Perfil Proveedor*

### EDITAR USUARIO

**NOMBRE:**

**NOMBRE DE USUARIO:**

Requerido. Aceptados caracteres alfanumericos y @/./+/-/\_ solo.

**CIF:**

**DIRECCION DE EMAIL:**

**DIRECCION:**

**CÓDIGO POSTAL:**

**INFORMACIÓN EXTRA:**

**EDITAR PERFIL**

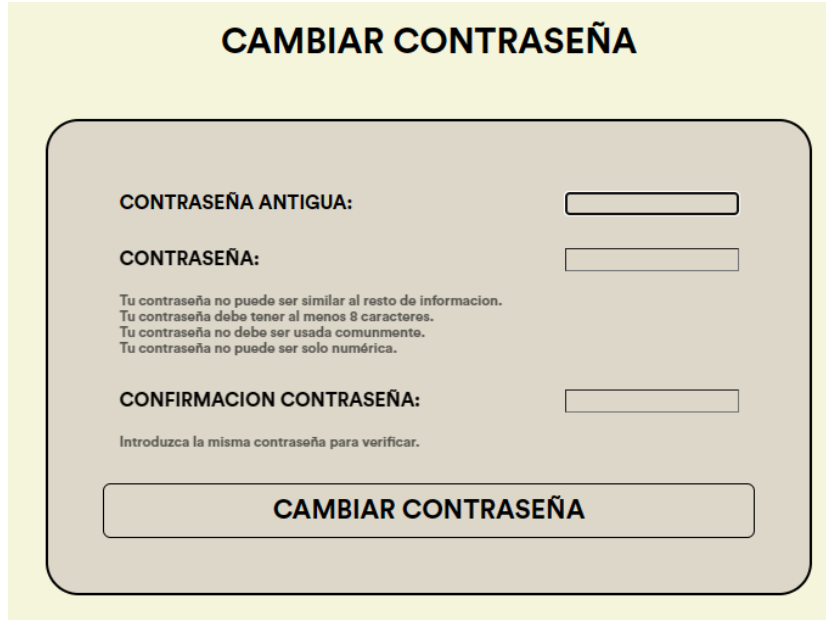
Sí quieres cambiar tu contraseña haz click [AQUÍ](#)

*Edición Perfil Staff*

16. Aquí podremos modificar toda la información relacionada con el usuario excepto el cambio de contraseña, al cual se accede haciendo click en 'AQUÍ' en la esquina inferior derecha del contenedor. Una vez rellenemos el formulario siempre y cuando este sea correcto, bien pulsando enter o el botón de 'EDITAR PERFIL' se nos



solicitará una confirmación a través de un `confirm()` generado con JavaScript.



**CAMBIAR CONTRASEÑA**

**CONTRASEÑA ANTIGUA:**

**CONTRASEÑA:**

Tu contraseña no puede ser similar al resto de informacion.  
Tu contraseña debe tener al menos 8 caracteres.  
Tu contraseña no debe ser usada comunmente.  
Tu contraseña no puede ser solo numérica.

**CONFIRMACION CONTRASEÑA:**

Introduzca la misma contraseña para verificar.

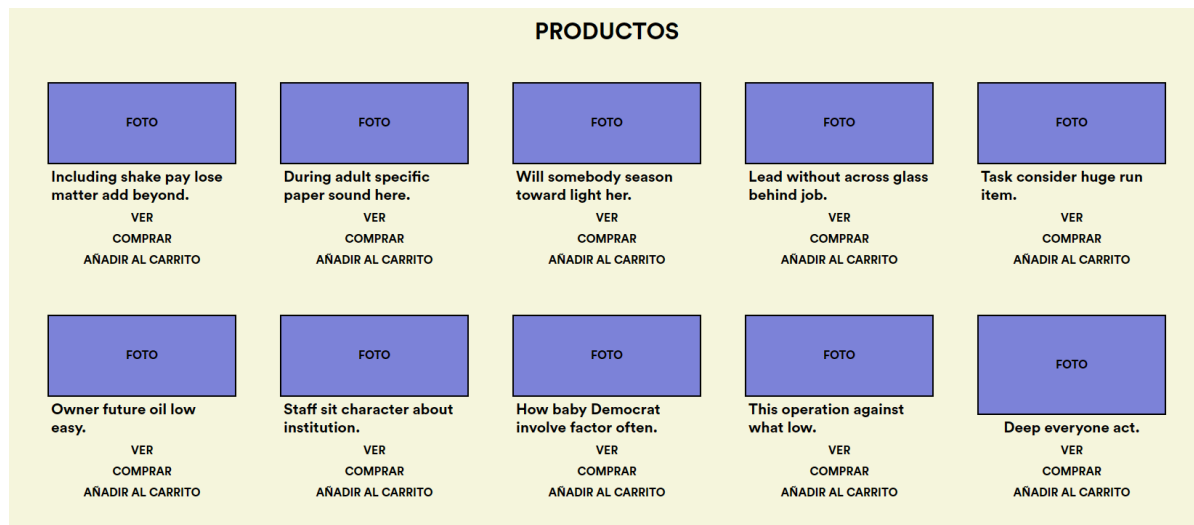
**CAMBIAR CONTRASEÑA**

*Cambio de Contraseña*

17. Para cambiar la contraseña correctamente se nos pedirá que introduzcamos la contraseña antigua (como autenticación), y después que introduzcamos dos veces la nueva contraseña para confirmar.

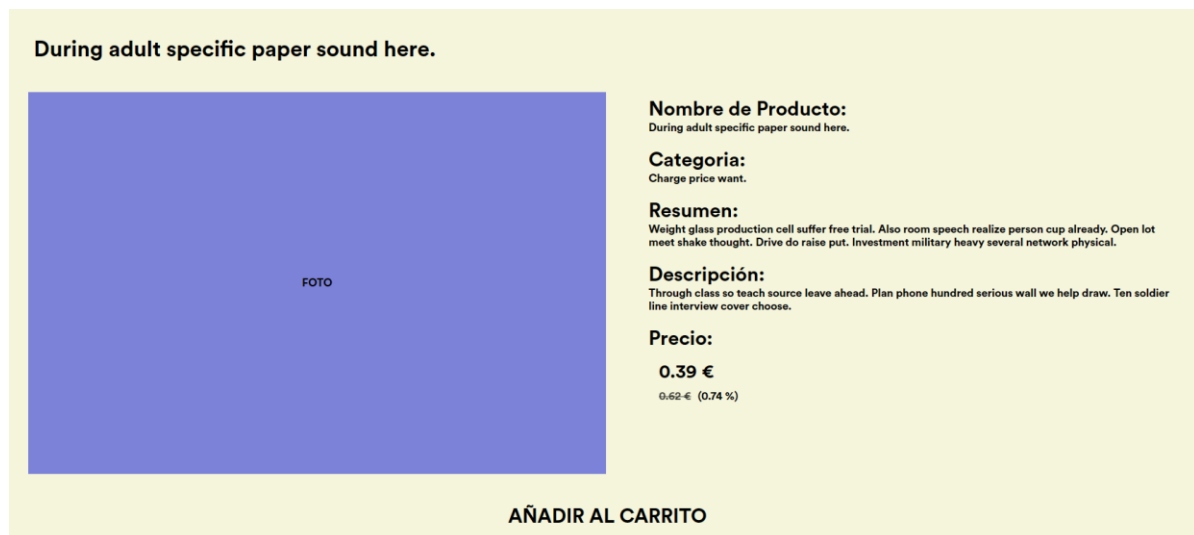
## PRODUCTOS

18. Para empezar a hablar de los productos explicaremos cómo funcionan los modelos y vistas que componen la App 'products'. El modelo es un modelo sencillo dónde se almacena principalmente, el nombre del producto, el stock del mismo, el precio de venta al público, el descuento que tenga en ese momento, su lugar en el almacén, una pequeña descripción y una más detallada, y finalmente tiene un campo de clave foránea que crea una relación de dependencia con un proveedor (el proveedor posee productos).
19. En las vistas, tenemos el CRUD típico para crear, editar, leer y borrar productos. Aparte de un inicio que mostrará los productos en una cuadrícula junto a su nombre, acompañado cada producto de tres botones: uno para ver el producto en detalle, otro para comprar el producto y otro para añadirlo al carrito.



Vista Indice Productos

20. En la vista del detalle del producto se nos mostrará la foto del producto en grande junto con su nombre, su pequeña descripción, su descripción y el precio del mismo calculado a partir del PVP y el descuento que tenga dicho producto. Finalmente al final de la vista tendremos un botón para añadirlo al carrito.



Vista Detalle Producto



## COMPRAS Y CARRITO DE LA COMPRA

21. Tanto las ventas como las compras funcionan en Apps separadas en nuestro proyecto aunque interaccionan entre sí. Para las ventas tenemos dos modelos: el modelo de venta que almacena qué producto se ha comprado, la cantidad que se ha comprado, el usuario que lo ha comprado y la fecha en la que se compró. Por otro lado almacenamos en otra tabla llamada 'SupplierSale' que almacena el producto que la empresa compra el proveedor, la fecha en la que lo hace y la cantidad de unidades que se compra. Este último lo utilizaremos en la app de control de stock que veremos adelante.
22. Las compras se pueden realizar de dos maneras distinta, o bien en el inicio de productos dándole al botón comprar el cual viene con una confirmación de compra, o comprando el carrito de la compra.
23. La app del carrito de la compra se basa en dos modelos: un modelo 'Cart' que hace pertenece a un usuario a través de una clave foránea y la fecha de creación, por otro lado almacenamos en el modelo 'CartItem' los productos de cada carrito conectando este modelo al otro. Cada ítem del carrito tiene una cantidad.
24. Para añadir productos al carrito se puede hacer desde la página de inicio de productos con el botón añadir al carrito o bien desde la página de detalle de cada producto con el mismo botón situado al final de la vista.
25. Una vez entramos al carrito haciendo click en el icono de carrito de la compra situado a la derecha de la barra de navegación. Se nos abrirá otra lista que nos mostrará todos los ítems de nuestro carrito junto a unos botones para ver el detalle y para borrar el ítem del carrito (funcionalidad con confirmación), y finalmente la cantidad de ítems de ese producto que tenemos en el carrito. En caso de no tener un carrito creado o de que nuestro carrito de la compra en ese momento esté vacío se mostrará un mensaje informando de que el carrito de la compra está vacío.

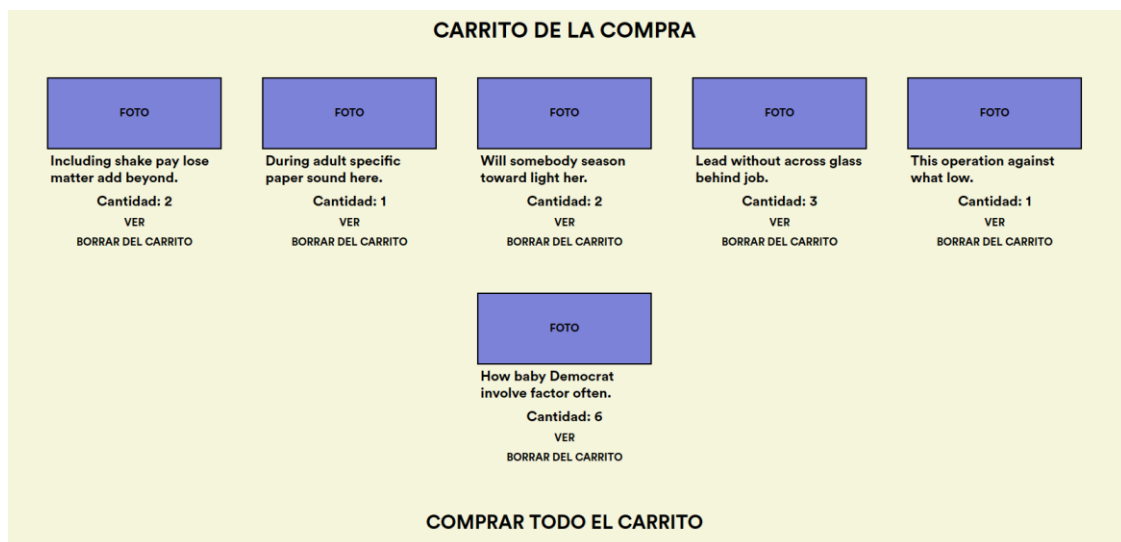
Sesión iniciada como CLIENTE



Icono para acceso al Carrito



Carrito de la compra vacío



Carrito de la compra con ítems

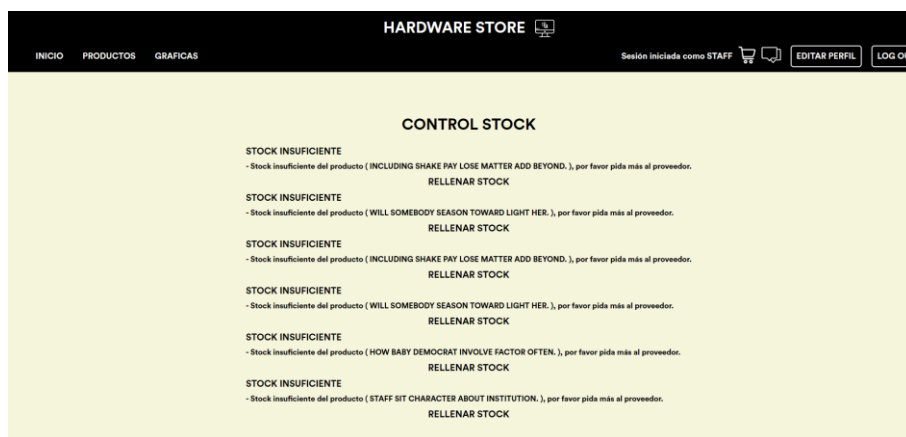
26. Si se intenta comprar un producto del que no hay stock suficiente la compra no se podrá realizar y se mostrará un mensaje por pantalla informando al usuario.

## CONTROL DEL STOCK

27. La app de control de stock sirve para generar mensajes cuando el stock del producto baja del 90% del stock mínimo de ese producto. Esto se realiza en las vistas de las compras con un condicional. Se generan los mensajes cuando la compra hace que el stock baje de este valor primero chequeando que no haya mensajes de control de stock de ese producto en la base de datos, para evitar redundancias.
28. A esta funcionalidad solo tienen acceso los usuarios tipo Staff a los cuales les aparece un icono de mensajes a la izquierda del icono del carrito. El cual lleva a la vista del control de stock donde se nos mostrará el mensaje 'Stock insuficiente' junto al botón para reponer el stock.



*Icono de mensajes control de stock*

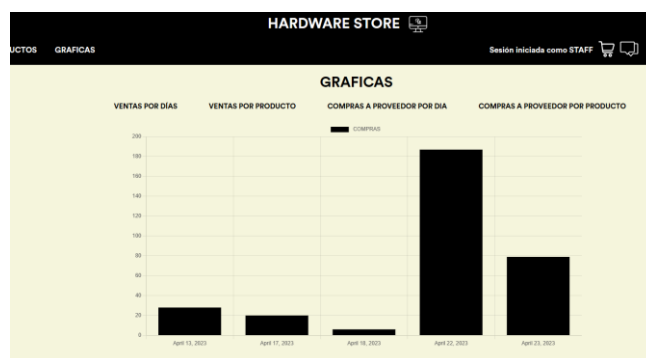


*Vista del control de stock*

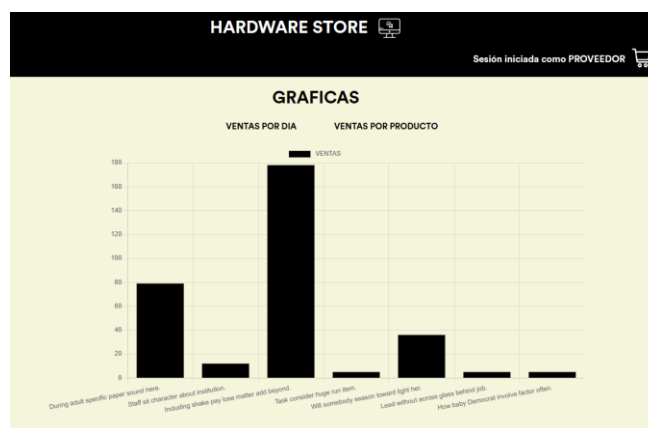
29. Al pulsar sobre el botón rellenar stock se pondrá el stock del producto a su mínimo generando tantas compras a proveedor como productos se hayan comprado. Además se borrará el mensaje de control de stock y finalmente se mostrará por pantalla un mensaje informando al usuario de que el stock ha sido repuesto.

## GRAFICOS

30. La funcionalidad de gráficos la poseen los tres tipos de usuarios, pero según el tipo de usuario tendrán acceso a unos gráficos u a otros. Los gráficos están generados con ChartJS el cual nos permite renderizar gráficos en HTML de manera muy sencilla e intuitiva.



Gráficos de Staff



Gráficos de Proveedor



*Gráficas de Cliente*

31. Como se puede observar el usuario tipo Staff tiene la posibilidad de ver las ventas de la empresa tanto por día como por producto, y las compras de la empresa a proveedor tanto por días como por productos. El usuario tipo Proveedor puede ver las compras que ha hecho la empresa de sus productos tanto por día como por producto. Y finalmente el usuario tipo Cliente puede ver sus compras por día y por producto.
32. Para generar los gráficos realizamos unas consultas a las tablas de la base de datos 'Sale' y 'SupplierSale' y procesamos los mismos de manera que juntamos todas las compras/ventas que sean del mismo día, o todas las compra/ventas que sean del mismo producto.

## MENSAJES

33. A lo largo de la documentación del proyecto he ido hablando de unos mensajes que se muestran al usuario. Son unos mensajes generados en las vistas de la aplicación los cuales aparecen durante unos segundos cuando cambiamos de url. Hay mensajes cuando compras productos, cuando compras el carrito, cuando realizas una compra pero hay stock insuficiente, cuando se edita el perfil, etc. Estos mensajes se muestran en nuestra plantilla justo debajo del menú de navegación y están animados usando CSS y JavaScript.



## CONFIRMACIONES

34. También veo necesaria la funcionalidad de confirmación cuando vamos a realizar una acción concreta. Así pues usando JavaScript generamos estas confirmaciones que saltarán en el navegador que estemos usando en ese momento. Dicha funcionalidad la poseen las acciones de: edición de usuario, cambio de contraseña, salida del sistema, compra de producto, compra del carrito y borrado de ítem del carrito.

## ESTRUCTURA DE LA APLICACIÓN WEB

35. Hacemos uso del manejo de plantillas con el que funciona Django, que aparte nos instala Jinja a través del cual podemos escribir código Python dentro de una plantilla HTML. Las plantillas se encuentran en la carpeta 'templates' ordenadas en subcarpetas que hacen referencia a cada aplicación del proyecto.
36. En la subcarpeta 'layout' tenemos almacenado el HTML base de la página en el cual creamos bloques que aprovechamos para crear una herencia de plantillas la cual nos permite tener el código mucho más ordenado.
37. Finalmente en la carpeta 'static' se encuentran los archivos estáticos de los que hace uso la página web. Aquí encontramos las imágenes que hemos usado, el código CSS (que no está modulado) y los archivos JavaScript encargados de las animaciones y las confirmaciones.





## CONCLUSIONES

38. El proyecto ha sido toda una experiencia. He decidido usar Django para ampliar mi campo de conocimiento sobre frameworks, lo cual me llevó un buen tiempo de formación e investigación. Pero he de decir que no me arrepiento para nada, he descubierto un framework sencillo e intuitivo el cual estoy seguro de que usaré más adelante.
39. He dejado la base de datos poblada con los productos generados por el factory faker y cuatro usuarios, un cliente, un proveedor, un staff y un super usuario. Los cuales tienen como nombre de usuario cliente, proveedor y staff respectivamente y todos con la contraseña 'password'. Para que sea más sencillo probar la aplicación y se vean unos gráficos más poblados.
40. Aunque se escapaba del scope del curso también he intentado interesarme por la parte frontend del mismo aprendiendo las bases del CSS y JavaScript.
41. He intentado mantener mi código limpio, intuitivo, modulado y escalable. Para poder mejorar este proyecto más adelante y así poder ponerlo en mi portfolio.
42. También he usado un repositorio de código GIT para familiarizarme con las herramientas que espero utilizar una vez trabaje como desarrollador. Aunque no lo manejo muy bien he intentado hacerlo lo mejor posible.
43. Los iconos utilizados los he cogido de FlatIcon mencionando la autoría en el footer de la página web.
44. El proyecto me ha encantado realizarlo, siento que ha quedado una aplicación competente con potencial para mejora. He aprendido muchísimo con la realización de este proyecto y no espero a poder realizar muchos otros.
45. Muchas gracias por la lectura, espero haber sido claro, ameno y nada redundante.