**ABPro6GrupalModulo6**

Grupo 4

Claudio Delgado

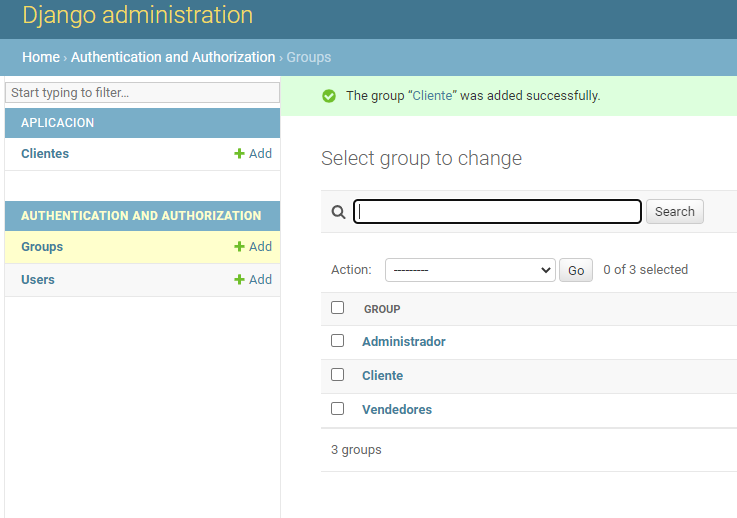
Luis Quintana

Carlos Aguirre

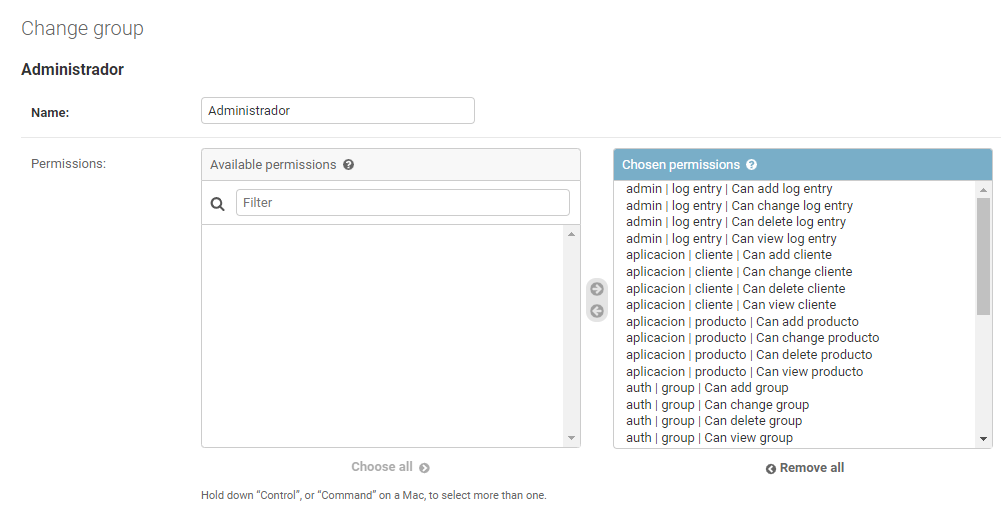
Desarrollo ABPro6

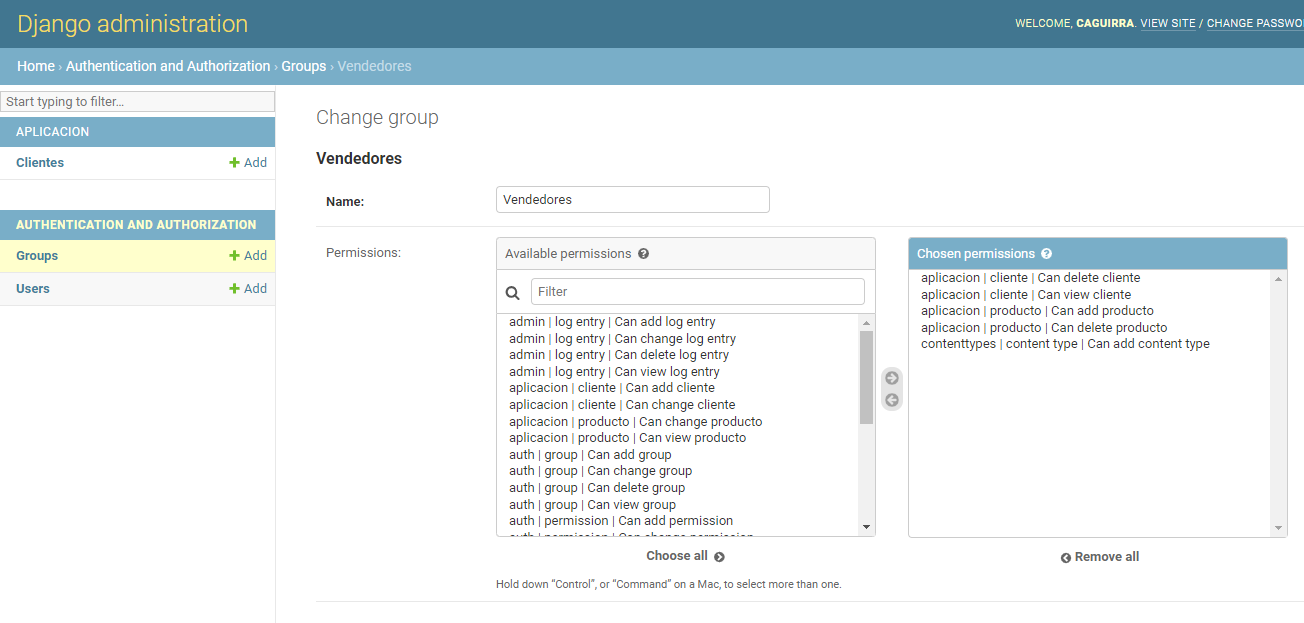
1.- Continuando con su proyecto, desarrolle toda la lógica necesaria para crear tres tipos de usuarios de su aplicación con permisos diferenciados. Reflexionen sobre los permisos que debe tener cada usuario y justifiquen su decisión. Cada grupo debe tener al menos cuatro permisos.

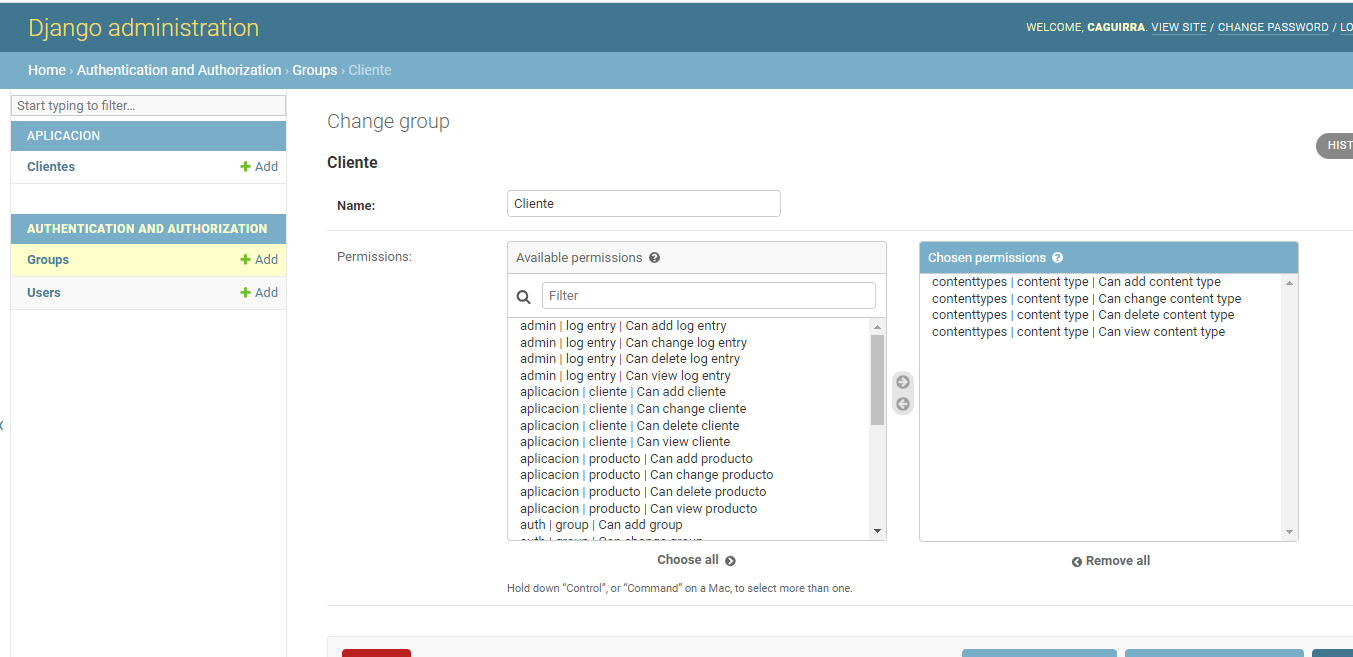
R.- Se crean los tres tipos de usuarios con diferentes permisos en la base de datos.



Tomando en cuenta el concepto de ventas desde una página web, debemos centrarnos en los siguientes perfiles. Administrador quien será el poseedor de todos los privilegios dada su condición de administrador del sitio. Por otro lado tenemos al Cliente quien solo tiene permisos jerárquicos de ver, agregar, restar productos y todo lo relacionado a la venta de esta forma se restringen los permisos del comprador. Los vendedores son quienes prestan asistencia a los clientes en caso de tener problemas con las compras, por lo que los permisos jerárquicos son intermedios y acotados a lo relacionado a la supervisión y administración de la venta. Con esa lógica se crean los perfiles de usuario.







2.- Elabore una pequeña guía donde identifique los pasos requeridos para crear usuarios de forma dinámica, sin utilizar la página de administración de django.

R.- Se crea un template que reciba los datos de usuario para registro a través de un formulario creado a través de los que entrega Django, luego el usuario se da por ingresado una vez que se envían los datos a BD. Posteriormente el usuario es redireccionado a otro template donde el usuario debe iniciar sesión con su usuario y contraseña creada con anterioridad, una vez realizado esto el usuario es validado con la BD y es redireccionado al inicio del template y puede ver su contenido.

3.- Complementen el archivo README.txt donde se expliquen las funcionalidades de su aplicación y toda la información necesaria para que cualquier desarrollador pueda levantar su aplicación en un servidor local

R.- Realizado.

4.- Actualice su repositorio en GitHub y envíelo en el archivo comprimido con su proyecto

R.- Realizado.

**ABPro5GrupalModulo6**

Grupo 4

Claudio Delgado

Luis Quintana

Carlos Aguirre

Desarrollo ABPro5

1.- Elaboren un template de login y un template de bienvenida posterior al login. Este último template debe mostrar un mensaje personalizado, saludando a quien haya iniciado sesión.

R.- Realizado.

2.- Si el usuario no ha iniciado sesión, bajo ninguna circunstancia podrá ver la página de bienvenida personalizada. Restringe los permisos correspondientes.

R.- Realizado, protegido con decorador login required.

3.- Elaboren un archivo README.txt donde se expliquen las funcionalidades de su aplicación y toda la información necesaria para que cualquier desarrollador pueda levantar su aplicación en un servidor local.

R.- Realizado.

4.- Actualice su repositorio en GitHub y envielo en el archivo comprimido con su proyecto.

R.- Realizado.

**ABPro4GrupalModulo6**

Grupo 4

Claudio Delgado

Luis Quintana

Carlos Aguirre

Desarrollo ABPro4

1.- Cree un entorno virtual e instalen Django (si ya está creado, utilicen el desarrollado en el ejercicio anterior). Verifiquen la versión de Python y Django.

R.- Realizado.

2.- Identifiquen claramente las diferencias entre los métodos GET y POST

R.- El método GET debe ser utilizado en peticiones que no modifican el sistema, como por ejemplo una consulta en el motor de búsqueda del navegador ya que al pasar el parámetro y realizar la búsqueda se ve lo que va en el campo, en cambio en el método POST se pasan valores y esos valores no se muestran al usuario ya que van ocultos y realizan una acción que modifica o crea valores.

3.- Continuando con el trabajo anterior. Elaboren una nueva página con un formulario que guarde información de proveedores que tienen interés en colaborar con su aplicación.

R.- Realizado

4.- Elabora toda la estructura necesaria para guardar información del formulario en su base de datos por defecto.

R.- Realizado

5.- Elabore una pequeña guía que sintetice los pasos necesarios para levantar un formulario desde Django con Forms y Models.

R.- Primero hay que crear un modelo para poder guardar la información de los usuarios, luego se registra en el admin y se importa para que sea reconocida como válida y luego se chequea en el admin. Se crea un archivo forms.py para declarar el formulario desde donde se importa el formulario y se declara la clase, luego se declara en el views y posterior a eso se modifica la página y se mantiene el botón y se aplica un mensaje que llama al formulario de Python y lo transforma a html, luego de esto se puede escribir y almacenar en la base de datos lo escrito en el, adicionalmente se agrega el csrf\_token para escribir en la base de datos.

6.- Actualice su repositorio en GitHub y envielo en el archivo comprimido con su proyecto.

R.- Realizado.

https://github.com/C4rlosAguirre/MODULODJANGO

**ABPro3GrupalModulo6**

Grupo 4

Claudio Delgado

Luis Quintana

Carlos Aguirre

Desarrollo ABPro3

1.- Cree un entorno virtual e instalen Django (si ya está creado, utilicen el desarrollado en el ejercicio anterior). Verifiquen la versión de Python y Django.

R.- Se utiliza el realizado en entrega anterior.

2.- Desarrollen elementos de la página principal elaborada en la actividad anterior. Incorporen aspectos de bootstrap según lo comentado en las cápsulas.

R.- Realizado.

3.- Creen una nueva plantilla, que muestre información de 5 clientes de su plataforma. Cada cliente debe tener al menos 5 atributos. Incorporen elementos gráficos mediante CSS.

R.- Realizado.

4.- Elementos como footer y navbar deben ser compartidos entre las diferentes páginas. Para esto utilice block content.

R.- Realizado.

5.- Elaboren una pequeña guía para que cualquier persona pueda mostrar contenido dinámico desde un archivo sqlite3 a una aplicación web.

R.- Se debe tener un acceso a un panel de control como el admin de Django, una vez dentro del panel puede ver todas las bases de dato creadas y asociadas al panel de administración de django, dentro del panel de control puede ver cada tabla de la base de datos asociada.

Desarrollo ABPro2

1.- Cree un entorno virtual e instalen Django (si ya está creado, utilicen el desarrollado en el ejercicio anterior). Verifiquen la versión de Python y Django.

R.- se utiliza el ya creado.

2.- Ahora desarrollaremos nuestro proyecto utilizando Django. El objetivo es que muestre el mensaje “Bienvenidos a TeLoVendo”. Esta vez, en base a un archivo HTML. Indague en formas de mejorar la estética de su página html.

R.- se realiza lo solicitado en trabajo anterior. La estética del proyecto se mejora usando Bootstrap y CSS.

3.- En esta instancia desarrollaremos una vista, basándonos en la documentación de Django en la página web de Django Project.

R.-

#views.py

from django.shortcuts import render

# Create your views here.

def inicio(request):

return render(request,'aplicacion/index.html')

4.- Elaboren una pequeña guía que pueda ser útil para cualquier persona que quiera mostrar un template con django. Privilegien la simplicidad de las explicaciones, sin perder detalles relevantes. Este ejercicio les será útil en el futuro! Envíelo como continuación al ejercicio ‘Descripción de script de Django’ realizado en el ejercicio anterior.

R.- Primero configurar debemos modificar el archivo settings.py para indicarle la nueva aplicación que se va a crear, luego se configura el archivo urls.py del proyecto donde se indica la nueva ruta que se utilizará adicionalmente de la ruta creada por defecto para el Admin, posteriormente se crea al archivo urls.py en la carpeta de la aplicación quitando la ruta del Admin de la carpeta e indicando una ruta a la página principal y por último modificar views.py para poder enrutar las respectivas vistas que voy a obtener o dicho de otra forma la carpeta donde tendremos los templates para mostrar y finalmente en el proyecto agregar un archivo index.html para poder mostrar una página completa en el servidor la cual se guarda en la carpeta templates la cual fue previamente creada y dejado el archivo html.

**ABPro1GrupalModulo6**

Grupo 4

Claudio Delgado

Luis Quintana

Carlos Aguirre

Desarrollo ABPro1

Desarrollo

1.- Como primer ejercicio, levantaremos un servidor con Python, para que imprima el mensaje “Bienvenidos a TeLoVendo” en un navegador. Realícelo solo utilizando Python

D:\DJANGO>python -m http.server 8000

Serving HTTP on :: port 8000 (http://[::]:8000/) ...

::ffff:127.0.0.1 - - [19/Apr/2022 22:12:52] code 404, message File not found

::ffff:127.0.0.1 - - [19/Apr/2022 22:12:52] "GET /templates/aplicacion/ HTTP/1.1" 404 -

::ffff:127.0.0.1 - - [19/Apr/2022 22:13:01] "GET / HTTP/1.1" 200 -

::ffff:127.0.0.1 - - [19/Apr/2022 22:13:06] "GET /aplicacion/ HTTP/1.1" 200 -

::ffff:127.0.0.1 - - [19/Apr/2022 22:13:06] "GET /aplicacion/\_\_pycache\_\_/ HTTP/1.1" 200 -

::ffff:127.0.0.1 - - [19/Apr/2022 22:13:11] "GET /aplicacion/templates/ HTTP/1.1" 200 -

::ffff:127.0.0.1 - - [19/Apr/2022 22:13:12] "GET /aplicacion/templates/aplicacion/ HTTP/1.1" 200 -

::ffff:127.0.0.1 - - [19/Apr/2022 22:13:12] code 404, message File not found

::ffff:127.0.0.1 - - [19/Apr/2022 22:13:12] "GET /w3images/la.jpg HTTP/1.1" 404 –

Se levanta un server en Python y se accede a la página de Bienvenidos a Te Lo Vendo.

2.- Se crea un entorno virtual, se instala Django y se verifica la versión de Django y Python:

python3 -m venv myvenv 🡪crear entorno virtual

python -m pip install --upgrade pip 🡪 actualizar pip install

pip install django==4.0.3 🡪 instalar Django

requeriments.txt 🡪 en este archivo se comentan todos los requerimientos necesarios para trabajar en cada proyecto.

(env) D:\DJANGO\aplicacion>python --version

Python 3.10.2

(env) D:\DJANGO\aplicacion>python -m django --version

4.0.3

3.- Ahora desarrollaremos nuestro primer proyecto utilizando Django. El objetivo es que muestre el mensaje

“Bienvenidos a TeLoVendo”

#settings.py

from pathlib import Path

# Build paths inside the project like this: BASE\_DIR / 'subdir'.

BASE\_DIR = Path(\_\_file\_\_).resolve().parent.parent

# Quick-start development settings - unsuitable for production

# See https://docs.djangoproject.com/en/4.0/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret!

SECRET\_KEY = 'django-insecure-nbtw&c&0s6py5i2\*uo@m9y0$1iymr(\_$tlkx2lrayegyn58rzj'

# SECURITY WARNING: don't run with debug turned on in production!

DEBUG = True

ALLOWED\_HOSTS = []

# Application definition

INSTALLED\_APPS = [

'django.contrib.admin',

'django.contrib.auth',

'django.contrib.contenttypes',

'django.contrib.sessions',

'django.contrib.messages',

'django.contrib.staticfiles',

'aplicacion',

]

MIDDLEWARE = [

'django.middleware.security.SecurityMiddleware',

'django.contrib.sessions.middleware.SessionMiddleware',

'django.middleware.common.CommonMiddleware',

'django.middleware.csrf.CsrfViewMiddleware',

'django.contrib.auth.middleware.AuthenticationMiddleware',

'django.contrib.messages.middleware.MessageMiddleware',

'django.middleware.clickjacking.XFrameOptionsMiddleware',

]

ROOT\_URLCONF = 'proyecto.urls'

TEMPLATES = [

{

'BACKEND': 'django.template.backends.django.DjangoTemplates',

'DIRS': [],

'APP\_DIRS': True,

'OPTIONS': {

'context\_processors': [

'django.template.context\_processors.debug',

'django.template.context\_processors.request',

'django.contrib.auth.context\_processors.auth',

'django.contrib.messages.context\_processors.messages',

],

},

},

]

WSGI\_APPLICATION = 'proyecto.wsgi.application'

# Database

# https://docs.djangoproject.com/en/4.0/ref/settings/#databases

DATABASES = {

'default': {

'ENGINE': 'django.db.backends.sqlite3',

'NAME': BASE\_DIR / 'db.sqlite3',

}

}

# Password validation

# https://docs.djangoproject.com/en/4.0/ref/settings/#auth-password-validators

AUTH\_PASSWORD\_VALIDATORS = [

{

'NAME': 'django.contrib.auth.password\_validation.UserAttributeSimilarityValidator',

},

{

'NAME': 'django.contrib.auth.password\_validation.MinimumLengthValidator',

},

{

'NAME': 'django.contrib.auth.password\_validation.CommonPasswordValidator',

},

{

'NAME': 'django.contrib.auth.password\_validation.NumericPasswordValidator',

},

]

# Internationalization

# https://docs.djangoproject.com/en/4.0/topics/i18n/

LANGUAGE\_CODE = 'en-us'

TIME\_ZONE = 'UTC'

USE\_I18N = True

USE\_TZ = True

# Static files (CSS, JavaScript, Images)

# https://docs.djangoproject.com/en/4.0/howto/static-files/

STATIC\_URL = 'static/'

# Default primary key field type

# https://docs.djangoproject.com/en/4.0/ref/settings/#default-auto-field

DEFAULT\_AUTO\_FIELD = 'django.db.models.BigAutoField'

#url.py

from django.urls import path, include

from . import views

urlpatterns = [

path('',views.inicio, name='inicio'),

]

#admin.py

from django.contrib import admin

#index.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Django</title>

</head>

<body>

<h1>“Bienvenidos a TeLoVendo”</h1>

</body>

</html>

#server en Django

python manage.py runserver

April 09,2022 - 23:23:02

Django version 4.0.3, using settings 'proyecto.settings'

Starting development server at http://127.0.0.1:8000/

Quit the server with CTRL\_BREAK.

#Identifique los diferentes archivos creados por Django al crear un proyecto y una aplicación.

#django-admin startproject > es el archivo para crear el primer proyecto el cual crea los archivos necesarios para el proyecto.

#luego viene los compoentes que se craen del startproject

#manage.py: utilidad q permite interactuar con el proyecto.

#\_init\_.py: archivo que inidica a python que este archivo debe ser considerado como un paquete de python, está vacío.

#settings.py: ajustes y/o configuración del proyecto, define configuraciones de caracter global.

#urls.py: es donde se apuntan las URL para cada proyecto django, aquí se cargan las distintas urls q sean necesarias.

#asgi.py: es un archivo de entrada para trabajar con servidores ASGI y realizar el deploy o carga.

#wsgi.py: archivo de entrada para trabajar con servidores WSGI y realizar el deploy o carga.