Jose Angel Valadez Barajas

.

Documentacion

Contenido

[Introducción: 2](#_Toc166669796)

[Modelo 3](#_Toc166669797)

[Modulo De Usuarios 5](#_Toc166669798)

[Módulo de permisos y Modulo de perfil de usuario 6](#_Toc166669799)

[Módulo de productos. 7](#_Toc166669800)

[Modulo para permitir registrar las entradas y salidas de inventario. 8](#_Toc166669801)

[Módulo de consumo de datos fuera del sistema. 9](#_Toc166669802)

[Mecanismo de conexión a Front-End 14](#_Toc166669803)

[URL Vistas 15](#_Toc166669804)

[URL API 16](#_Toc166669805)

# Introducción:

El proyecto es para un público principalmente femenino en el ámbito profesional con una

edad entre 25 y 35 años. Esta persona se desarrolla en el ámbito empresarial y le gusta que

los procesos que tiene que hacer sean muy claros, rápidos y puntuales.

Pueden usar el sistema desde su computadora, tableta o teléfono, aunque principalmente

se usará en la computadora, debe poder realizar las tareas en los demás entornos.

# Modelo

1. **Modelo de Rol (Role)**:
   * Este modelo representa los diferentes roles que pueden tener los usuarios en tu sistema.
   * Tiene un campo **name** que almacena el nombre del rol.
   * La función **\_\_str\_\_** devuelve el nombre del rol cuando se imprime una instancia del modelo.
   * La clase tiene un método **default\_role()** que devuelve el rol predeterminado "Consultor" si existe, o lo crea si no.
2. **Modelo de Usuario Personalizado (CustomUser)**:
   * Este modelo representa a los usuarios registrados en tu sistema.
   * Utiliza el modelo **AbstractBaseUser** de Django para personalizar el modelo de usuario.
   * Los campos incluyen **email**, **username**, **first\_name**, **last\_name**, **address** y **role**, que está relacionado con el modelo **Role**.
   * Tiene campos booleanos **is\_active** e **is\_staff** para controlar si el usuario está activo y si tiene permisos de administrador.
   * Utiliza un administrador personalizado (**CustomUserManager**) para manejar la creación de usuarios.
   * Define **USERNAME\_FIELD** como **email**, lo que significa que el email se utiliza para el inicio de sesión en lugar de un nombre de usuario.
3. **Modelo de Producto (Product)**:
   * Este modelo representa los productos en tu sistema.
   * Tiene campos como **name**, **description**, **stock**, **min\_stock** y **price**.
   * **name** y **description** son campos de texto para el nombre y la descripción del producto.
   * **stock** y **min\_stock** son campos enteros que representan la cantidad actual y mínima de stock del producto.
   * **price** es un campo decimal que representa el precio del producto.
4. **Modelo de Entrada de Inventario (InventoryEntry)**:
   * Este modelo representa las entradas de inventario, es decir, cuando se recibe nueva cantidad de un producto.
   * Tiene un campo de clave externa **product** que está relacionado con el modelo **Product**.
   * **quantity\_received** es un campo entero que representa la cantidad de productos recibidos.
   * **date\_received** es un campo de fecha que registra la fecha en que se recibió la entrada de inventario.
5. **Modelo de Salida de Inventario (InventoryExit)**:
   * Este modelo representa las salidas de inventario, es decir, cuando se vende una cantidad de un producto.
   * Tiene un campo de clave externa **product** que está relacionado con el modelo **Product**.
   * **quantity\_sold** es un campo entero que representa la cantidad de productos vendidos.
   * **date\_sold** es un campo de fecha que registra la fecha en que se realizó la salida de inventario.
6. **Modelo de Stock Insuficiente (InsufficientStock)**:
   * Este modelo registra los productos que tienen un stock por debajo del mínimo establecido.
   * Tiene un campo de clave externa **product** que está relacionado con el modelo **Product**.
   * **quantity\_needed** es un campo entero que representa la cantidad necesaria para alcanzar el stock mínimo.
7. **Señales de Stock (update\_product\_stock, update\_product\_stock\_on\_exit, update\_product\_stock\_on\_exit\_delete, check\_insufficient\_stock)**:
   * Estas señales se activan después de que se guardan o eliminan las entradas y salidas de inventario, así como después de guardar o eliminar un producto.
   * Actualizan automáticamente el campo **stock** de un producto cuando se registran entradas o salidas de inventario.
   * Comprueban automáticamente si un producto tiene un stock insuficiente y crean o eliminan un registro en el modelo **InsufficientStock** en consecuencia.

# Modulo De Usuarios

El modulo de usuarios se compone de las siguientes vistas:

1. **Vista de inicio de sesión (login\_view)**:
   * Esta vista maneja el inicio de sesión de los usuarios.
   * Obtiene las credenciales del usuario del formulario (**email** y **password**).
   * Si las credenciales son válidas, inicia sesión en el sistema utilizando la función **login** y redirige al usuario al panel de control (**dashboard**).
   * Si las credenciales no son válidas, vuelve a mostrar el formulario de inicio de sesión con un mensaje de error.
2. **Vista de registro (register\_view)**:
   * Esta vista maneja el registro de nuevos usuarios.
   * Procesa el formulario de registro (**CustomUserCreationForm**).
   * Si el formulario es válido, guarda el nuevo usuario en la base de datos y redirige al usuario a la página de inicio de sesión.
   * Si el formulario no es válido, vuelve a mostrar el formulario de registro con los errores.
3. **Vista del panel de control (dashboard\_view)**:
   * Esta vista muestra el panel de control para los usuarios autenticados.
   * Utiliza el decorador **@login\_required** para asegurarse de que solo los usuarios autenticados puedan acceder a esta vista.
   * Obtiene el tipo de usuario (**role\_id**) del usuario actual y lo pasa al template para su visualización.

# Módulo de permisos y Modulo de perfil de usuario

Para la vista de administración de usuarios el módulo de permisos y el módulo de perfil de usuario comparten vista ya que permite administrar los permisos de los usuarios disponibles mediante el uso de “Roles”, para la vista de perfil esta solo usa el módulo de perfil de usuario:

1. **Vista de administración de usuarios (user\_administration)**:
   * Esta vista permite a los administradores ver y administrar a todos los usuarios del sistema.
   * Verifica si el usuario actual tiene permisos de administrador antes de permitir el acceso a esta vista.
   * Utiliza varios formularios (**UserStatusForm**, **UserEditForm**) para cambiar el estado y editar la información de los usuarios.
   * Procesa los formularios enviados para cambiar el estado, la información de los usuarios y sus permisos.
2. **Vista de perfil (profile\_view)**:
   * Esta vista permite a los usuarios ver y actualizar su perfil.
   * Utiliza un formulario personalizado (**CustomUserForm**) para mostrar y actualizar la información del usuario.
   * Procesa el formulario y guarda los cambios.

# Módulo de productos.

El módulo de productos se compone de las siguientes vistas:

1. **Vista de administración de productos (control\_products)**:
   * Esta vista permite a los usuarios administrar los productos en el sistema.
   * Muestra un formulario para agregar nuevos productos y una lista de todos los productos existentes.
   * Procesa formularios para agregar, editar o eliminar productos.
2. **Vista de edición de producto (edit\_product)**:
   * Esta vista permite a los usuarios editar un producto específico.
   * Recibe el **product\_id** como parámetro en la URL y busca el producto correspondiente en la base de datos.
   * Muestra un formulario pre-rellenado con los datos del producto.
   * Actualiza los datos del producto en la base de datos.
3. **Vista de consulta de Stock (inventory\_consult)**
   * Esta vista permite a los usuarios consultar el stock de los productos.
   * Es accesible para el usuario consultor

# Modulo para permitir registrar las entradas y salidas de inventario.

El modulo para registrar las entradas y salidas de inventario se compone por las siguientes vistas:

1. **Vistas para registrar entrada y salida de inventario (register\_inventory\_entry, register\_inventory\_exit)**:
   * Estas vistas manejan el registro de entradas y salidas de inventario, respectivamente.
   * Procesan los formularios para registrar las entradas y salidas de inventario en la base de datos.
2. **Vista de información de inventario (inventory\_information)**:
   * Esta vista muestra información sobre las entradas y salidas de inventario, así como los productos con stock insuficiente.
   * Obtiene todas las entradas, salidas de productos con stock insuficiente de la base de datos y los pasa al template para su visualización.

# Módulo de consumo de datos fuera del sistema.

Este modulo / mecanismo se caracteriza por usar el sistema de API que nos brinda Django para realizar las operaciones que realizaríamos en las vistas. En este modulo encontraras toda la información que se requiere para ejecutar los diferentes end-points dentro de la aplicación.

1. **/api/register/**
   * **Función**: Esta URL se utiliza para registrar nuevos usuarios en el sistema.
   * **Cómo probar en Postman**:
     + Método: POST
     + URL: **http://tu-dominio.com/api/register/**
     + Payload (formato JSON):

{"email":"correo@example.com","username": "nombre\_de\_usuario", "password": "contraseña", "first\_name": "nombre", "last\_name": "apellido", "address": "dirección"}

* + - Autenticación: No se requiere autenticación.

1. **/api/login/**
   * **Función**: Esta URL se utiliza para iniciar sesión en el sistema y obtener un token de acceso.
   * **Cómo probar en Postman**:
     + Método: POST
     + URL: **http://tu-dominio.com/api/login/**
     + Payload (formato JSON):

{"email": "correo@example.com", "password": "contraseña"}

* + - Autenticación: No se requiere autenticación.

1. **/api/profile/**
   * **Función**: Esta URL devuelve los detalles del perfil del usuario autenticado.
   * **Cómo probar en Postman**:
     + Método: GET
     + URL: **http://tu-dominio.com/api/profile/**
     + Autenticación: Se requiere autenticación con token de acceso en el encabezado de autorización (**Bearer token**).
2. **/api/profile/edit/**
   * **Función**: Esta URL se utiliza para editar el perfil del usuario autenticado.
   * **Cómo probar en Postman**:
     + Método: PUT
     + URL: **http://tu-dominio.com/api/profile/edit/**
     + Payload (formato JSON):

{"first\_name": "nuevo\_nombre", "last\_name": "nuevo\_apellido", "address": "nueva\_dirección"}

* + - Autenticación: Se requiere autenticación con token de acceso en el encabezado de autorización (**Bearer token**).

1. **/api/users/**
   * **Función**: Esta URL devuelve una lista de todos los usuarios en el sistema (solo accesible para usuarios con rol de administrador).
   * **Cómo probar en Postman**:
     + Método: GET
     + URL: **http://tu-dominio.com/api/users/**
     + Autenticación: Se requiere autenticación con token de acceso en el encabezado de autorización (**Bearer token**).
2. **/api/users/int:pk/change-role/**
   * **Función**: Esta URL se utiliza para cambiar el rol de un usuario específico (solo accesible para usuarios con rol de administrador).
   * **Cómo probar en Postman**:
     + Método: PUT
     + URL: **http://tu-dominio.com/api/users/<id>/change-role/**
     + Payload (formato JSON):

{ "role\_id": nuevo\_id\_de\_rol }

* + - Autenticación: Se requiere autenticación con token de acceso en el encabezado de autorización (**Bearer token**).

1. **/api/products/create/**
   * **Función**: Esta URL se utiliza para crear un nuevo producto en el sistema (solo accesible para usuarios con rol de administrador).
   * **Cómo probar en Postman**:
     + Método: POST
     + URL: **http://tu-dominio.com/api/products/create/**
     + Payload (formato JSON):

{"name": "nombre\_del\_producto", "description": "descripción\_del\_producto", "stock": cantidad\_de\_stock, "min\_stock": cantidad\_mínima\_de\_stock, "price": precio\_del\_producto}

* + - Autenticación: Se requiere autenticación con token de acceso en el encabezado de autorización (**Bearer token**).

1. **/api/products/int:pk/**
   * **Función**: Esta URL proporciona detalles sobre un producto específico, como su nombre, descripción, stock, stock mínimo y precio.
   * **Cómo probar en Postman**:
     + Método: GET
     + URL: **http://tu-dominio.com/api/products/<id>**
     + Autenticación: Se requiere autenticación con token de acceso en el encabezado de autorización (**Bearer token**).
2. **/api/products/int:pk/stock/**
   * **Función**: Esta URL devuelve la cantidad de stock de un producto específico.
   * **Cómo probar en Postman**:
     + Método: GET
     + URL: **http://tu-dominio.com/api/products/<id>/stock/**
     + Autenticación: Se requiere autenticación con token de acceso en el encabezado de autorización (**Bearer token**).
3. **/api/inventory/entry/create/**
   * **Función**: Esta URL se utiliza para crear una nueva entrada en el inventario.
   * **Cómo probar en Postman**:
     + Método: POST
     + URL: **http://tu-dominio.com/api/inventory/entry/create/**
     + Payload (formato JSON):

{"product":id\_del\_producto,"quantity\_received": cantidad\_recibida}

* + - Autenticación: Se requiere autenticación con token de acceso en el encabezado de autorización (**Bearer token**).

1. **/api/inventory/exit/create/**
   * **Función**: Esta URL se utiliza para crear una nueva salida en el inventario.
   * **Cómo probar en Postman**:
     + Método: POST
     + URL: **http://tu-dominio.com/api/inventory/exit/create/**
     + Payload (formato JSON):

{"product": id\_del\_producto, "quantity\_sold": cantidad\_vendida}

* + - Autenticación: Se requiere autenticación con token de acceso en el encabezado de autorización (**Bearer token**).

1. **/api/inventory/entries/**
   * **Función**: Esta URL devuelve una lista de todas las entradas de inventario.
   * **Cómo probar en Postman**:
     + Método: GET
     + URL: **http://tu-dominio.com/api/inventory/entries/**
     + Autenticación: Se requiere autenticación con token de acceso en el encabezado de autorización (**Bearer token**).
2. **/api/inventory/exits/**
   * **Función**: Esta URL devuelve una lista de todas las salidas de inventario.
   * **Cómo probar en Postman**:
     + Método: GET
     + URL: **http://tu-dominio.com/api/inventory/exits/**
     + Autenticación: Se requiere autenticación con token de acceso en el encabezado de autorización (**Bearer token**).
3. **/api/inventory/insufficient/**
   * **Función**: Esta URL devuelve una lista de todos los productos con stock insuficiente.
   * **Cómo probar en Postman**:
     + Método: GET
     + URL: **http://tu-dominio.com/api/inventory/insufficient/**
     + Autenticación: Se requiere autenticación con token de acceso en el encabezado de autorización (**Bearer token**).
4. **/api/users/int:pk/**
   * **Función**: Esta URL se utiliza para editar los detalles de un usuario específico (solo accesible para usuarios con rol de administrador).
   * **Cómo probar en Postman**:
     + Método: PUT
     + URL: **http://tu-dominio.com/api/users/<id>/**
     + Payload (formato JSON):

{"email": "nuevo\_email@example.com","username": "nuevo\_username","first\_name":"nuevo\_nombre","last\_name": "nuevo\_apellido","address": "nueva\_dirección"}

* + - Autenticación: Se requiere autenticación con token de acceso en el encabezado de autorización (**Bearer token**).

# Mecanismo de conexión a Front-End

Este mecanismo se caracteriza por presentar las vistas usando la interfaz que Django proporciona para generar las plantillas y presentarlas en el front de una manera eficaz.

Algunas capturas son:

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

# URL Vistas

1. **/login/**
   * Vista asociada: **login\_view**
2. **/register/**
   * Vista asociada: **register\_view**
3. **/dashboard/**
   * Vista asociada: **dashboard\_view**
4. **/profile/**
   * Vista asociada: **profile\_view**
5. **/user-administration/**
   * Vista asociada: **user\_administration**
6. **/control\_products/**
   * Vista asociada: **control\_products**
7. **/editar\_producto/int:product\_id/**
   * Vista asociada: **editar\_producto**
8. **/get\_product\_stock/int:product\_id/**
   * Vista asociada: **get\_product\_stock**
9. **/new\_inventory/**
   * Vista asociada: **register\_inventory\_entry**
10. **/sales/**
    * Vista asociada: **register\_inventory\_exit**
11. **/inventory\_information/**
    * Vista asociada: **inventory\_information**
12. **/inventory\_consult/**

* Vista asociada: **inventory\_consult**

# URL API

1. **/api/register/**

* Vista asociada: **register**

2. **/api/login/**

* Vista asociada: **loginapi**

3. **/api/profile/**

* Vista asociada: **ProfileDetailView**

4. **/api/profile/edit/**

* Vista asociada: **ProfileEditView**

5. **/api/users/**

* Vista asociada: **UserListAPIView**

6. **/api/users/<int:pk>/**

* Vista asociada: **UserEditAPIView**

7. **/api/users/<int:pk>/change-role/**

* Vista asociada: **UserRoleChangeAPIView**

8. **/api/products/create/**

* Vista asociada: **ProductCreateAPIView**

9. **/api/products/<int:pk>/**

* Vista asociada: **ProductDetailAPIView**

10. **/api/products/<int:pk>/stock/**

* Vista asociada: **ProductStockAPIView**

12. **/api/inventory/entry/create/**

* Vista asociada: **InventoryEntryCreateAPIView**

13. **/api/inventory/exit/create/**

* Vista asociada: **InventoryExitCreateAPIView**

14. **/api/inventory/entries/**

* Vista asociada: **InventoryEntryListAPIView**

15. **/api/inventory/exits/**

* Vista asociada: **InventoryExitListAPIView**

16. **/api/inventory/insufficient/**

* Vista asociada: **InsufficientStockListAPIView**