Gestor de Laboratorios



Descripción General

El Gestor de Laboratorios, GESLAB FS 1.0, es una aplicación web desarrollada usando el Framework Django y diseñada para llevar un registro y control de instalaciones farmacéuticas, los productos que fabrica o produce, y de su representante o Director, posibilitando hacer gestión sobre los mismos.

Características Principales

- 1. **Gestión de laboratorios:** Permite listar, crear, editar y eliminar laboratorios, pudiendo agregarse nuevos campos de ser necesario. Se captura el nombre del laboratorio, un alias, ciudad, país
 - Vista "Información de Laboratorios"



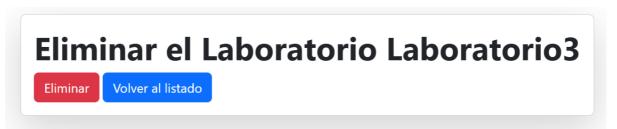
Vista "Agregar Laboratorio"

| Crear Laboratorio |
|----------------------------------|
| Nom lab: Nombre del Laboratorio |
| Alias: |
| City lab: Ciudad del Laboratorio |
| Pais lab: Pais del Laboratorio |
| Registrar Cancelar |

Vista "Editar Laboratorio"



Vista "Eliminar Laboratorio"



- 2. Gestión de Productos Farmacéuticos: Permite listar, crear, editar y eliminar productos. Cada producto sólo puede pertenecer a un Laboratorio. Se puede asignar un año de vencimiento para control de stock. Se captura el nombre, laboratorio, año fabricación, año expiración, precio de costo y precio de venta.
 - Vista "Información de Productos"

Información de Productos Información de Productos de



Vista "Agregar Producto"

| Crear nuevo Producto |
|---------------------------|
| Producto: |
| Lab prod: |
| Selecciona un laboratorio |
| F Fabricación: 2024 > |
| F Expiración: 2024 Y |
| P Costo: |
| P Venta: |
| Crear Cancelar |

Vista "Editar Producto"



Vista "Eliminar Producto"

Eliminar el Producto Producto5

Volver al listado

- 3. **Gestión de Directores:** Permite listar, crear, editar y eliminar Directores, pudiendo agregarse nuevos campos de ser necesario. cada Director, sólo puede pertenecer a un Laboratorio.
 - Vista "Información de Directores"

Información de Directores ■



Vista "Agregar Director"

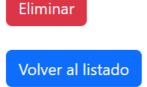


Vista "Editar Director"



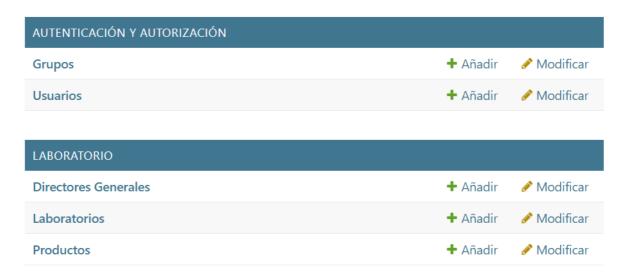
Vista "Eliminar Director"

Eliminar el Director Director2

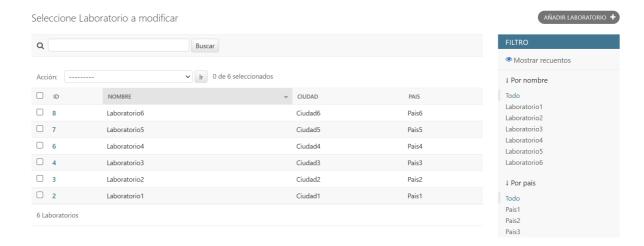


- 4. **Gestión del Sistema (ADMIN):** El sistema permite crear perfiles de usuario, grupos, asignar permisos, etc.
 - Vista "Sitio Administrativo"

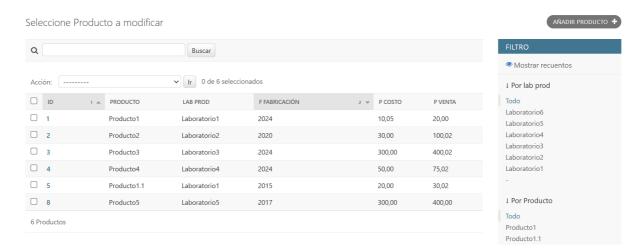
Sitio administrativo



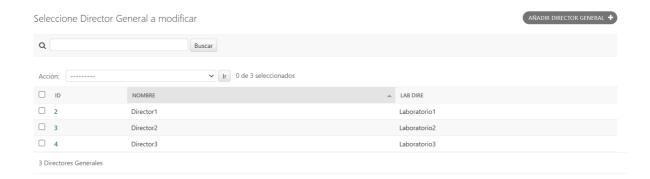
Vista "Gestión Laboratorios"



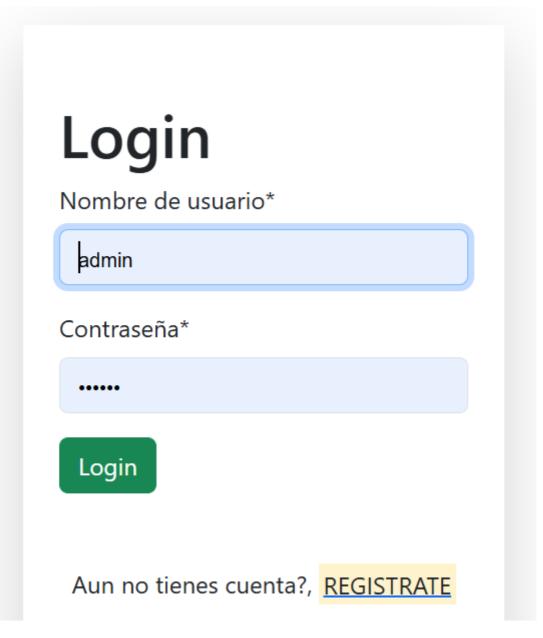
Vista "Gestión Productos"



Vista "Gestión Directores"



- 5. **Reportes:** Se pueden generar los siguientes informes: Información por Laboratorio, Producto y Director.
- 6. **Otras características:** El sistema incluye distintos métodos de búsqueda dinámicas y a través de filtros con selectores.
- 7. **Autenticación y Registro de Usuarios:** El sistema dispone de un módulo para el registro y autenticación de usuarios, otorgando varios niveles de seguridad y segregación de la información.
 - o Vista "Login"



2024-11-30 readme.md

Vista "Registro"

Registro

Ya tienes una Cuenta?, inicia Sesion: LOGIN.

| | Nombre de usuario* |
|--------|---|
| | Requerido. 150 carácteres como máximo. Únicamente letras, dígitos y @/./+/-/_ |
| | Email* |
| | Contraseña |
| | Su contraseña no puede asemejarse tanto a su otra información personal. Su contraseña debe contener al menos 8 caracteres. Su contraseña no puede ser una clave utilizada comúnmente. Su contraseña no puede ser completamente numérica. |
| | Contraseña (confirmación) |
| | Para verificar, introduzca la misma contraseña anterior. Confirmar |
| | tas (queries) utilizando la shell de Django |
| | python manage.py shell |
| 2. Imp | portar los modelos |

```
from laboratorio.models import Laboratorio, DirectorGeneral, Producto
```

- 3. Generar las consultas:
 - 3.1) Obtener todos los objetos: Laboratorio, DirectorGeneral y Productos.
 - 3.1.1) Obtiene y guarda todos los registros de la tabla Laboratorio. Genera un bucle para mostrar los elementos de la variable de uno en uno, son dos líneas de código

```
labs = Laboratorio.objects.all()
  for lab in labs:
    print(lab)
```

3.1.2) Los registros de la tabla DirectorGeneral

```
dires = DirectorGeneral.objects.all()
  for dire in dires:
    print(dire)
```

3.1.3) Los registros de la tabla Productos

```
pds = Producto.objects.all()
  for pd in pds:
    print(pd)
```

3.2) Obtener el laboratorio del Producto cuyo nombre es 'Producto 1'.

```
pd_1 = Producto.objects.get(nom_prod='Producto1') # Primero, obtener el nombre del
Producto 1 (debe ser el mismo)
lab_pd_1 = pd_1.lab_prod # Encontrar el laboratorio por el producto asociado
print(lab_pd_1) # Imprimir el resultado
```

3.3) Listar y ordenar todos los productos por nombre, y muestre los valores de nombre y laboratorio.

```
lista_productos_ordenados = Producto.objects.order_by('nom_prod')  # Primero,
obtener todos los productos ordenados por nombre
for producto in lista_productos_ordenados:  # Bucle para mostrar los
resultados uno a uno
   print(producto.nom_prod, producto.lab_prod)  # se imprimen los elementos
con sus atributos nombre y laboratorio
```

3.4) Imprimir por pantalla los laboratorios de todos los productos.

```
for producto in Producto.objects.all():
    print(producto.lab_prod)
```

Migraciones

Se procedió a actualizar el modelo inicial de datos agregando campos personalizados con el nombre "actualizado_campo".

• Se generan las migraciones luego de modificar los modelos, usando el comando "makemigrations"

```
python python manage.py makemigrations laboratorio --name actualizado_campo
```

• Se aplican las migraciones usando el comando "migrate"

```
python python manage.py makemigrations laboratorio --name actualizado_campo
```

```
Migrations for 'laboratorio':
laboratorio\migrations\0004_actualizado_campo.py
+ Add field alias_lab to laboratorio
+ Add field f_expiracion to producto
laboratorio\migrations\0004_actualizado_campo.py
+ Add field alias_lab to laboratorio
+ Add field f_expiracion to producto
+ Add field alias_lab to laboratorio
+ Add field f_expiracion to producto
```

```
+ Add field f_expiracion to producto
```

Tecnologías Utilizadas

• Framework de Desarrollo: Django 5.1.3.

• Frontend: HTML, CSS, JavaScript, Bootstrap.

• Backend: Python, Django.

• Base de datos: PostgreSQL v16.

• Otras tecnologías: Git, Github.

Instalación y Configuración

1. Clonar el repositorio:

```
https://github.com/jcordovaj/EvalM7.git
```

2. Crear un entorno virtual:

o En Linux/macOS

```
python -m venv venv_lab

source venv_lab/bin/activate
```

o En Windows

```
python -m venv venv_lab
venv_lab\Scripts\activate
```

3. Instalar las dependencias:

```
pip install -r requirements.txt
```

4. Configurar la base de datos:

Crear usuario : userdjangoPassword : userdjango

```
CREATE ROLE userdjango WITH LOGIN

SUPERUSER

CREATEDB

CREATEROLE

INHERIT

NOREPLICATION

BYPASSRLS

CONNECTION LIMIT -1

PASSWORD 'userdjango';
```

o Crear BBDD

Nombre: db_final_orm Usuario: userdjango (superuser) pass: userdjango collation: Spanish_spain.1252

5. Ejecutar la aplicación:

```
python manage.py runserver
```

6. Administrador del Sistema

- Administrador = Admin
- Password = 123456

7. Generación de usuarios de prueba

Se ha empleado la siguiente nomenclatura para crear los usuarios de prueba.

o user: usuarioX (X=Un número entero positivo correlativo)

o correo: usuarioX@usuarioX.com

o pass: clavefacil1234

Contribución (Things-To-Do)

Se puede contribuir con los problemas o nuevas ideas, por favor respetar el estilo de programación y no subir código basura. Puede utilizar: forking del repositorio, crear pull requests, etc. Toda contribución es bienvenida.

Licencia

Proyecto con fines educativos, Licencia MIT

Autor

Jota Córdova - Fun Manager

- Otros colaboradores
 - Nelson Ramirez Tutor