

# Manual Técnico del Proyecto Django

documentación para el manual técnico

fecha de realización enero 21 – junio 28

[Perfil](#) [Inicio](#) [Inventario](#) [Recetas](#) [Items](#) [Marca](#) [Proveedores](#) [Tipos](#) [Equipos](#) [Bitacora](#) [Permisos](#) [Reporte](#) [Salir](#)

## BIENVENID@ JOSHUA

sistema de Inventario





Este es un sistema de Inventarios para la unidad universitaria de secuenciación masiva y bioinformática donde sirve para tener un control sobre todo los reactivos que entran a la unidad y así llevar un mejor control sobre todos estos reactivos

## 1. Introducción

Este documento es un manual técnico detallado para el proyecto Django, que proporciona una guía completa sobre la configuración, estructura y uso del proyecto. Este proyecto es una aplicación web desarrollada con Django, un framework de alto nivel de Python que facilita el desarrollo rápido y un diseño limpio y pragmático.

### 1.1 Objetivo del Proyecto

El objetivo de este proyecto es proporcionar una base sólida para el desarrollo de una aplicación web que puede ser utilizada para gestionar un inventario. Incluye dos aplicaciones principales:

- **app:** Esta aplicación maneja la lógica principal del inventario, incluyendo la creación, actualización, eliminación y visualización de elementos del inventario.
- **user:** Esta aplicación gestiona la autenticación y autorización de los usuarios, así como la administración de los perfiles de usuario.

### 1.2 Estructura del Proyecto

La estructura del proyecto está organizada de manera que sigue las mejores prácticas de Django, facilitando la escalabilidad y sostenibilidad del código.

- **Directorio Principal:** Contiene el archivo `manage.py`, que es una herramienta de línea de comandos para interactuar con el proyecto Django, y el archivo `db.sqlite3`, que es la base de datos SQLite donde se almacenan los datos de la aplicación.
- **Carpeta Inventario:** Contiene los archivos de configuración del proyecto, incluyendo `settings.py` para la configuración global, `urls.py` para las rutas URL, y `wsgi.py` y `asgi.py` para la configuración del servidor web.
- **Carpetas app y user:** Estas carpetas contienen el código de las aplicaciones individuales. Cada una de estas carpetas sigue una estructura similar con archivos como `models.py` para los modelos de la base de datos, `views.py` para la lógica de las vistas, `forms.py` para los formularios, y `admin.py` para la configuración del panel de administración de Django pero especialmente la carpeta `user` se uso para la creacion de un formulario personalizado para el usuario.
- **Carpeta media:** Utilizada para almacenar los archivos subidos por los usuarios, organizados en subdirectorios como `user`.
- **Carpeta venv:** Contiene el entorno virtual que aísla las dependencias del proyecto, asegurando que las bibliotecas y paquetes utilizados no interfieran con otros proyectos en el mismo sistema.

## 1.3 Tecnologías Utilizadas

Este proyecto utiliza una variedad de tecnologías y herramientas, incluyendo:

- **Python:** El lenguaje de programación principal utilizado para desarrollar el proyecto.
- **Django:** El framework web de alto nivel que facilita el desarrollo rápido y un diseño limpio.
- **PostgreSQL:** La base de datos principal utilizada para almacenar los datos de la aplicación. PostgreSQL es una base de datos relacional avanzada y de código abierto que ofrece un alto rendimiento y escalabilidad.
- **SQLite:** Utilizado en el entorno de desarrollo como una base de datos ligera y fácil de configurar.
- **HTML, CSS y JavaScript:** Tecnologías front-end utilizadas para construir las interfaces de usuario de la aplicación.

## 2. Requisitos Previos

Para trabajar en este proyecto Django, es necesario tener instaladas ciertas dependencias y herramientas en tu entorno de desarrollo.

### 2.1 Herramientas Necesarias

Antes de empezar, asegúrate de tener instaladas las siguientes herramientas en tu sistema:

- **Python:** El lenguaje de programación utilizado para desarrollar el proyecto. Puedes descargarlo desde [python.org](https://python.org).
- **virtualenv:** Una herramienta para crear entornos virtuales en Python, que permite aislar las dependencias del proyecto.
- **pip:** El gestor de paquetes para Python, que se utiliza para instalar las bibliotecas necesarias.

### 2.2 Instalación de Dependencias

Una vez que hayas instalado Python y virtualenv, sigue estos pasos para configurar el entorno de desarrollo del proyecto:

- **Crea y activa un entorno virtual:**

```
python -m venv venv  
source venv/bin/activate # En Windows usa `venv\Scripts\activate`
```

- **Instala las dependencias:** Las dependencias necesarias para este proyecto están listadas en un archivo requirements.txt. Ejecuta el siguiente comando para instalar todas las dependencias:

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

**Aplica las migraciones:** Las migraciones son necesarias para configurar la base de datos.

```
python manage.py migrate
```

**Configura la base de datos:** Este proyecto utiliza PostgreSQL como base de datos principal. Asegúrate de tener PostgreSQL instalado y configurado en tu sistema. Luego, actualiza el archivo settings.py del proyecto con la configuración de tu base de datos PostgreSQL.

Por defecto django coloca una base de datos SQLite para que la puedas usar, pero con la siguiente configuración puedes modificarlo para conectarlo a tu base de datos.

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql',
        'NAME': 'sistema_inventarios',
        'USER': 'uusmb1',
        'PASSWORD': 'G3ZU6e9/',
        'HOST': 'localhost',
        'PORT': '5432',
    }
}
```

Y para que se actualice migre correctamente la base de datos de Django a PostgreSQL usa:

```
python manage.py makemigrations
```

```
python manage.py migrate
```

**Ejecuta el servidor de desarrollo:** Finalmente, ejecuta el servidor de desarrollo para verificar que todo esté configurado correctamente:

```
python manage.py runserver
```

## 2.3 Listado de Dependencias

Dependencias del proyecto y sus versiones:

Package	Version
---------	---------

-----

amqp	5.2.0
------	-------

asarPy	1.0.1
--------	-------

asgiref	3.8.1
---------	-------

billiard	4.2.0
----------	-------

celery	5.4.0
cffi	1.16.0
chardet	5.2.0
click	8.1.7
click-didyoumean	0.3.1
click-plugins	1.1.1
click-repl	0.3.0
Django	5.0.4
kombu	5.3.7
pillow	10.3.0
pip	24.0
prompt_toolkit	3.0.47
psycpg2-binary	2.9.9
pycparser	2.22
python-dateutil	2.9.0.post0
pytz	2024.1
rabbitmq	0.2.0
redis	5.0.6
reportlab	4.2.0
setuptools	65.5.0
six	1.16.0
sqlparse	0.5.0
tzdata	2024.1
vine	5.1.0
wcwidth	0.2.13

### 3. Descripción de Archivos Clave

En esta sección se explicarán los principales archivos y módulos del proyecto Django, detallando su propósito y contenido. Esto incluye los archivos `views.py`, `urls.py`, `forms.py`, `models.py`, y `settings.py`, así como la configuración de archivos estáticos dentro del archivo `settings.py`.

#### 3.2 `urls.py`

El archivo `urls.py` define las rutas URL de la aplicación y las asocia con las vistas correspondientes. Esto permite a Django saber qué vista debe llamar para una URL determinada.

Donde su estructura es `path` ("la URL que tendrá en la web", "la función que tomará del archivo `views.py`", "el nombre para hacer referencia dentro de los templates")

```
urlpatterns = [
    path('admin/', admin.site.urls),
    path('', views.index, name="index"),
    path('accounts/', include('django.contrib.auth.urls')),
    path('salir/', views.salir, name="salir"),
    path('recuperar-contraseña/', views.recuperar_contraseña, name='recuperar_contraseña'),
    path('inventario/', views.inventario),
    path('bitacora/', views.bitacora, name='bitacora'),
    path('marca/', views.marca, name='marca'),
    path('registro-marca/', views.marcaregistro, name='marcaregistro'),
    path('editar-marca/<int:marca_id>/', views.editar_marca, name='editar_marca'),
    path('eliminar-marca/<int:marca_id>/', views.eliminar_marca, name='eliminar_marca'),
    path('proveedores/', views.proveedores),
    path('proveedores-registro/', views.proveedoresregistro, name="r_proveedor"),
    path('editar-proveedor/<int:proveedor_id>/', views.editar_proveedor, name='editar_proveedor'),
    path('eliminar-proveedor/<int:proveedor_id>/', views.eliminar_proveedor, name='eliminar_proveedor'),
    path('insertar/', views.registrar_item, name='insertar_r'),
    path('eliminar_registro/<int:registro_id>/', views.eliminar_registro, name='eliminar_registro'),
    path('editar_registro/<int:registro_id>/', views.editar_registro, name='editar_registro'),
    path('asignar-permisos/', views.asignar_permisos, name='asignar_permisos'),
    path('api/group-permissions/<int:group_id>/', views.get_group_permissions, name='get_group_permissions'),
    path('crear-grupo/', views.crear_grupo, name='crear_grupo'),
    path('perfil/', views.perfil, name='perfil'),
]
```

#### 3.3 `views.py`

El archivo `views.py` contiene las vistas de la aplicación web, esta vista es una función que toma solicitud web y devuelve una respuesta web. Esta respuesta puede ser el contenido HTML de una página web. Una redirección, un error 404, un documento XML, una imagen o cualquier otro tipo de cosa que se necesite.

Las vistas son lógica detrás de la aplicación y se colocan en el archivo `views.py`, un ejemplo la creación de formularios dentro de la aplicación en donde por el método POST recibirá los datos dentro de la función, también dentro de la vista puedes iterar datos y dentro de los templates llamarlos para visualizar estos datos.

```

@permission_required('app.view_auth_permission')
@login_required
def eliminar_usuario(request, username):
    try:
        usuario = User.objects.get(username=username)
    except User.DoesNotExist:
        messages.error(request, f"El usuario '{username}' no existe.")
        return redirect('asignar_permisos')

    if request.method == 'POST':
        if request.POST.get('confirmar') == 'true':
            usuario.delete()
            messages.success(request, f"El usuario '{username}' ha sido eliminado.")
        else:
            messages.warning(request, "No se ha eliminado al usuario.")
            return redirect('asignar_permisos')

    return render(request, 'eliminar_usuario.html', {'usuario': usuario})

```

Dentro del views.py se creo una función por si un usuario ya no forma parte del sistema y pueda ser eliminado obteniendo al usuario y verificando si este existe, en dado caso que no mandara un mensaje de error mostrando que el usuario no existe, pero si el usuario existe podrá ser eliminado y mostrara un mensaje de eliminación exitosa.

Todas estas funciones creadas se usaran para el uso de recetas, el registro de los diferentes módulos, la creación de grupos con permisos, para asignar permisos para los usuarios, etc.

### 3.3 forms.py

El archivo forms.py se utiliza para definir formularios en Django. Los formularios pueden ser utilizados para la entrada y validación de datos por parte del usuario. Django proporciona un módulo llamado forms que facilita la creación y gestión de formularios.

```

class marcaform(forms.ModelForm):
    class Meta:
        model = Marca
        fields = ['nombre', 'descripcion']

class proveedorForm(forms.ModelForm):
    class Meta:
        model = Proveedor
        fields = ['nombre', 'descripcion', 'telefono', 'correo', 'url']

```

Formularios que creas en forms.py especificando las celdas que quieres que muestre.

```
from .forms import marcaform, proveedorForm, ItemForm, TypeForm, LocationForm
```

Dentro del views.py para usarlo debemos hacer referencia a ellos con un import y el nombre del formulario que estarás usando.

```
@login_required
@permission_required('app.add_location', raise_exception=True)
def crear_location(request):
    if request.method == 'POST':
        form = LocationForm(request.POST)
        if form.is_valid():
            location = form.save(commit=False)
            location.usuario = request.user
            location.save()
            messages.success(request, 'Se agregó la ubicación correctamente')
            return redirect('listar_locations')
    else:
        form = LocationForm()
    return render(request, 'crear_locations.html', {'form': form})
```

Dentro de las diferentes funciones (backend de la app) estaremos llamando a los formularios que creamos (dentro del archivo views.py de la app) y mediante el metodo post los datos llenados dentro del formulario los va a solicitar para despues guardarlos. Y despues te mandara a por medio del return al html/template que hayas colocado.

### 3.4 models.py

El archivo models.py contiene los modelos de la aplicación, que son clases que representan las estructuras de datos y las tablas de la base de datos. Django utiliza un ORM (Object-Relational Mapping) para interactuar con la base de datos a través de estos modelos.



```

class Type(models.Model):
    nombre = models.CharField(max_length=100)
    descripcion = models.CharField(max_length=255)
    usuario = models.ForeignKey(User, on_delete=models.SET_NULL, null=True, blank=True)

    def delete(self, *args, **kwargs):
        self._usuario = kwargs.pop('usuario', None)
        self._descripcion_personalizada = kwargs.pop('descripcion_personalizada', '')
        super().delete(*args, **kwargs)

    def __str__(self):
        return self.nombre

class Location(models.Model):
    equipo = models.CharField(max_length=100)
    nivel = models.CharField(max_length=50)
    descripcion = models.CharField(max_length=255)
    usuario = models.ForeignKey(User, on_delete=models.SET_NULL, null=True, blank=True)

```

Dentro de los modelos se estarán definiendo las diferentes tablas de la base de datos en donde mediante el class “nombre de la tabla” (models.model) definiremos el nombre de la tabla, y abajo de se pondrá cada dato que contendrá como el nombre, la descripción y en este caso el usuario que se usa en la bitácora.

### 3.5 settings.py

El archivo settings.py contiene la configuración global del proyecto Django. Aquí se definen ajustes como la configuración de la base de datos, las aplicaciones instaladas, y la configuración de archivos estáticos.

```

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'app',
    'user',
]

```

Aplicaciones instaladas que por defecto vendrán unas que funcionen con django a excepción de app (aplicación principal) y user (aplicación para el registro, login de usuarios)

```
LANGUAGE_CODE = 'es'

TIME_ZONE = 'UTC'

USE_I18N = True

USE_TZ = True
```

en este caso también unas configuraciones ya vienen por defecto con django pero puedes modificar el código de lenguaje para que salga en español.

```
STATIC_URL = 'static/'

STATICFILES_DIRS = [
    os.path.join(BASE_DIR, "static"),
]
```

URL para los archivos “static” que de este directorio la aplicación Django es de donde tomara aquellos archivos de estilos (css o scss), archivos js, imágenes, etc.

```
LOGIN_REDIRECT_URL = '/'

MEDIA_URL = '/media/'
MEDIA_ROOT = os.path.join(BASE_DIR, "media")

AUTH_USER_MODEL = 'user.User'

TIME_ZONE = 'America/Mexico_City'
```

configuración para que la dentro de la aplicación muestre la zona horaria de mexico, donde se redirija cuando el usuario inicie sesión y el modelo de user que se usara para el tema de los usuarios.

## 4. Estructura del Proyecto

La estructura del proyecto Django está organizada de manera que facilita el desarrollo modular y escalable de aplicaciones web.

```
residencia-proyecto/          # Directorio principal del proyecto
|
|— app/                        # Aplicación principal del proyecto
|   |— admin.py               # Configuración del panel de administración
|   |— apps.py                # Configuración de la aplicación
|   |— forms.py               # Definición de formularios
|   |— __init__.py
|   |— migrations/           # Migraciones de la base de datos
|   |— models.py              # Definición de modelos de base de datos
|   |— __pycache__/
|   |— static/                # Archivos estáticos (CSS, JavaScript, imágenes)
|   |— templates/             # Plantillas HTML
|   |— tests.py               # Pruebas unitarias
|   |— views.py               # Definición de vistas
|
|— db.sqlite3                  # Base de datos SQLite (local)
|
|— Inventario/                 # Configuración del proyecto Django
|   |— asgi.py                 # Punto de entrada para servidores asgi
|   |— __init__.py
|   |— __pycache__/
|   |— settings.py            # Configuración global del proyecto
|   |— urls.py                 # Configuración de rutas URL
|   |— wsgi.py                 # Punto de entrada para servidores wsgi
|
|— manage.py                   # Gestor de comandos Django
```

```

|
|— media/          # Archivos multimedia (subidos por usuarios)
|  |— user/        # Directorio específico para archivos de usuarios
|
|— user/           # Aplicación secundaria (manejo de usuarios)
|  |— admin.py
|  |— apps.py
|  |— forms.py
|  |— __init__.py
|  |— migrations/
|  |— models.py
|  |— __pycache__/
|  |— tests.py
|  |— views.py
|
|— venv/           # Entorno virtual Python (aisla dependencias)
|  |— bin/
|  |— include/
|  |— lib/
|  |— lib64 -> lib/
|  |— pyvenv.cfg

```

## 5. Uso de la Aplicación

En este capítulo se explicará cómo interactuar y utilizar la aplicación desarrollada con Django. Se detallarán los pasos necesarios para ejecutar la aplicación, así como las funcionalidades principales que ofrece a los usuarios.

### 5.1 Ejecución de la Aplicación

Para ejecutar la aplicación Django en un entorno de desarrollo, sigue estos pasos:

1. **Activación del Entorno Virtual:** Si no has activado el entorno virtual previamente, hazlo con el siguiente comando:

```
source venv/bin/activate # En Windows: venv\Scripts\activate
```

2. **Iniciar el Servidor de Desarrollo:** Utiliza `manage.py` para iniciar el servidor de desarrollo de Django:

```
python manage.py runserver
```

Esto iniciará el servidor en `http://127.0.0.1:8000/`. Abre tu navegador y navega a esta dirección para ver la aplicación en funcionamiento.

## 5.2 Funcionalidades Principales

Algunas de las funcionalidades principales que puedes encontrar en la aplicación:

- **Registro y Autenticación de Usuarios:** La aplicación puede incluir formularios de registro y autenticación de usuarios, utilizando vistas y formularios definidos en `views.py` y `forms.py`.
- **Visualización de Datos:** Utiliza las vistas definidas en `views.py` para procesar y presentar datos desde la base de datos. Puedes utilizar plantillas HTML en el directorio `templates/` para renderizar estas vistas.
- **Administración de Datos:** Utiliza el panel de administración de Django (`/admin/`) para gestionar datos de la base de datos. Configura modelos en `models.py` y administra registros usando `admin.py`.
- **Interacción con URL:** Define rutas URL en `urls.py` para asociar URLs específicas con vistas correspondientes. Esto permite la navegación y acceso a diferentes partes de la aplicación.
- **Gestión de Archivos Estáticos y Multimedia:** Configura archivos estáticos (CSS, JavaScript, imágenes) en `settings.py`. Utiliza directorios como `static/` y `media/` para almacenar y servir estos archivos.

## 5.3 Personalización y Ampliación

Para personalizar y ampliar la aplicación:

- **Crear Nuevas Funcionalidades:** Define nuevas vistas, modelos y formularios según los requisitos de la aplicación en `app/`.
- **Integración de Funcionalidades Externas:** Integra bibliotecas externas y aplicaciones Django de terceros utilizando `pip install` y configurándolas en `settings.py`.

- **Optimización y Escalabilidad:** Implementa buenas prácticas de desarrollo, como el uso eficiente de consultas a la base de datos, caché de datos, y técnicas de optimización de rendimiento.

## 5.1 Gestión de Usuarios

### Registro de Usuario

- **Descripción:** Permite a los usuarios crear una cuenta nueva en la aplicación.
- **Acción:** Los usuarios pueden completar un formulario con información básica como nombre, correo electrónico y contraseña para registrarse.

[Inicio](#) [Acerca de](#) [Contáctanos](#) [Página de la Unidad](#) [manual de usuario](#)

### Registro de usuario

Nombre de usuario:

Dirección de correo electrónico:

Password:

Confirm password:

Pregunta recuperacion:

Color favorito ▾

Respuesta de recuperación:

registrar

### Login

- **Descripción:** Permite a los usuarios autenticarse en la aplicación.
- **Acción:** Los usuarios ingresan sus credenciales (usuario y contraseña) para acceder a su cuenta.



**Inicio de sesión**

Usuario:

Contraseña:

Iniciar Sesión

[registrarse](#) | [olvide clave](#)

## Recuperación de Contraseña

- **Descripción:** Permite a los usuarios restablecer su contraseña en caso de olvido.
- **Acción:** Los usuarios pueden solicitar con un su pregunta de verificación para restablecer su contraseña.

## Recuperar Contraseña

correo electronico:

Pregunta de recuperación:

Color favorito

Respuesta de recuperación:

Recuperar Contraseña

## Acerca de

- **Descripción:** Información sobre la aplicación y su propósito.
- **Acción:** Página estática que describe lo que se hace dentro de la unidad (uusmb).

## ¿Qué es el Sistema de Inventario?

El Sistema de Inventario es una herramienta fundamental para la gestión eficiente de los recursos de una unidad, garantizando la calidad y la eficiencia en la provisión de servicios y productos.

Beneficios:

Cumplimiento normativo

Optimización de recursos

Gestión eficiente de caducidades

Seguimiento de inventario en tiempo real

Generación de informes y análisis de datos



## 5.2 Gestión de Perfil del Usuario

### Inicio

- **Descripción:** Página principal después del login que puede mostrar un resumen o dashboard personalizado.
- **Acción:** Muestra información relevante para el usuario, barra de navegación y una bienvenida al usuario.

# BIENVENID@ JOSHUA

## sistema de Inventario



Este es un sistema de Inventarios para la unidad universitaria de secuenciación masiva y bioinformática donde sirve para tener un control sobre todo los reactivos que entran a la unidad y así llevar un mejor control sobre todos estos reactivos



### Perfil del Usuario

- **Descripción:** Permite a los usuarios ver y actualizar su información personal.
- **Acción:** Los usuarios pueden editar detalles como nombre, pregunta, información, etc.

PerfilInicioInventarioRecetasItemsMarcaProveedoresTiposEquiposBitácoraPermisosReporteSalir

BIENVENIDO JOSHUA

Nombre de usuario

joshua

Dirección de correo electrónico

Pregunta recuperacion

Color favorito

Respuesta de recuperación

Guardar cambiosCambiar contraseña

### Cambio de Contraseña

- **Descripción:** Permite a los usuarios cambiar su contraseña actual.
- **Acción:** Los usuarios deben proporcionar su contraseña actual y una nueva contraseña para actualizarla.

PerfilInicioInventarioRecetasItemsMarcaProveedoresTiposEquiposBitácoraPermisosReporteSalir

Cambiar Contraseña

Contraseña antigua

Contraseña nueva












Contraseña nueva (confirmación)

Cambiar contraseña

### 5.3 Gestión de Inventario

#### Items

- **Descripción:** Administración de los artículos en el inventario.
- **Acción:** Incluye funciones para agregar, editar y eliminar artículos, junto con la visualización de detalles de cada artículo.

Perfil	Inicio	Inventario	Recetas	Items	Marca	Proveedores	Tipos	Equipos	Bitácora	Permisos	Reporte	Salir
Items												
												
Id		Nombre		Contenido		Unidad de Medida		Stock		Stock Mínimo		
 	1	Miseq Reagent kit v3 box 2 of 2		1		unidad		36		100		
 	2	Oxford Nanopore		1		unidad		168		100		
 	4	Reactivo		500		Rxn		319		400		
 	5	tubo		10		ml		285		100		
 	6	flow cell		1		unidad		10		100		

#### Marcas, Proveedores, Ubicaciones y Tipos

- **Descripción:** Gestión de entidades relacionadas con el inventario (marcas, proveedores, ubicaciones, tipos).
- **Acción:** Cada entidad tiene operaciones CRUD (Crear, Leer, Actualizar, Eliminar) para gestionar registros relacionados.

Perfil			Inicio	Inventario	Recetas	Items	Marca	Proveedores	Tipos	Equipos	Bitácora	Permisos	Reporte	Salir
--------	--	--	--------	------------	---------	-------	-------	-------------	-------	---------	----------	----------	---------	-------

## Marcas

Id	Nombre	Descripción
1	illumina	illumina, Inc. es una empresa de biotecnología estadounidense con sede en San Diego, California. Constituida el 1 de abril de 1998, illumina desarrolla, fabrica y comercializa sistemas integrados para el análisis de la variación genética y la función biológica.

## 5.4 Gestión de Recetas

### Registro y Edición de Recetas

- **Descripción:** Definición y modificación de recetas de productos o procesos.
- **Acción:** Los usuarios pueden crear nuevas recetas, editar recetas existentes, especificar cantidades necesarias de ingredientes u otros pasos.

Perfil	Inicio	Inventario	Recetas	Items	Marca	Proveedores	Tipos	Equipos	Bitácora	Permisos	Reporte	Salir
--------	--------	------------	---------	-------	-------	-------------	-------	---------	----------	----------	---------	-------

### Nueva receta

Nombre:

Descripción:

Ingredientes:

Miseq Reagent kit v3 box 2 of 2 - Registro ID: 1 (12903812309)

Cantidad

unidad

Agregar ingrediente

Sub Recetas:

pruebas para el edit

Cantidad

Agregar sub receta

Guardar

Uso de Receta y Resumen

- **Descripción:** Aplicación práctica de una receta en operaciones diarias o en producción.
- **Acción:** Los usuarios pueden aplicar una receta para realizar procesos según las especificaciones definidas.

Perfil	Inicio	Inventario	Recetas	Items	Marca	Proveedores	Tipos	Equipos	Bitácora	Permisos	Reporte	Salir
16	pruebass	pruebas para una subreceta	•	ID: 2: Oxford Nanopore: cantidad: 1	•	prueba con receta: 1						
17	pruebas 22	ukjkhk				nueva receta: 1 pruebas: 1						
18	pruebas con cantidad	lkjaksldjaksjdl										
19	pruebas con subreceta cantidad	asjdiasjdkaj askjdklajs				pruebas con cantidad: 1						
20	receta para gel	receta para un gel				pruebas: 1 pruebas con subreceta cantidad: 2 prueba con receta: 3						
21	tubo receta	receta con tubos	•	ID: 9: tubo: cantidad: 3 ID: 6: Reactivo: cantidad: 5								

Perfil

Inicio

Inventario

Recetas

Items

Marca

Proveedores

Tipos

Equipos

Bitácora

Permisos

Reporte

Salir

Buscar por nombre de receta

Id	Nombre	Descripción	Sub Recetas	Acciones
11	pruebas para el edit	pruebas edit		<div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div></div>

## 5.5 Bitácora de Actividades

### Registro de Actividades

- **Descripción:** Registro de todas las operaciones realizadas en el sistema relacionadas con el inventario y otras entidades.
- **Acción:** Cada vez que se realiza una operación de creación, edición o eliminación en el inventario, marcas, proveedores, ubicaciones, tipos o artículos, se registra en la bitácora.

Perfil	Inicio	Inventario	Recetas	Items	Marca	Proveedores	Tipos	Equipos	Bitácora	Permisos	Reporte	Salir
Bitácora de Actividades												
Acción	Usuario	Fecha	Hora	Tabla	ID Instancia	Descripción						
Filtrar por Acción	Filtrar por Usuario	Filtrar por Fecha	Filtrar por Hora	Filtrar por Tabla	Filtrar por ID							
Crear	joshua	19/06/2024	18:11	Registro	11	Se crear el Registro con ID 11 y con codigo de barras 12308912						
Crear	joshua	19/06/2024	05:23	Registro	10	Se crear el Registro con ID 10 y con codigo de barras 812371						
Actualizar	joshua	18/06/2024	19:56	Registro	6	Se actualizar el registro 6 y con codigo de barras 1232193819028						
Actualizar	joshua	18/06/2024	19:44	Registro	6	Se actualizar el registro 6 y con codigo de barras 1232193819028						
Actualizar	joshua	18/06/2024	19:44	Registro	9	Se actualizar el registro 9 y con codigo de barras 19091431						
Actualizar	joshua	18/06/2024	19:37	Item	5							
Actualizar	joshua	18/06/2024	19:35	Item	4							
Actualizar	joshua	18/06/2024	19:35	Item	1							
Actualizar	joshua	18/06/2024	19:34	Item	2							
Actualizar	joshua	18/06/2024	19:27	Item	2							
Actualizar	joshua	18/06/2024	19:27	Item	1							
Actualizar	joshua	18/06/2024	19:26	Item	1							
Actualizar	joshua	18/06/2024	19:22	Item	2							
Actualizar	joshua	18/06/2024	19:21	Item	1							
Crear	joshua	18/06/2024	19:10	Item	6	Se crear el item con ID 6 y nombre flow cell						

## 5.6 Gestión de Permisos y Grupos de Usuarios

### Permisos y Creación de Grupos

- **Descripción:** Configuración de permisos específicos para diferentes roles de usuarios.
- **Acción:** Los administradores pueden crear grupos de usuarios y asignar permisos granulares a cada grupo, controlando qué acciones pueden realizar los usuarios.

PerfilInicioInventarioRecetasItemsMarcaProveedoresTiposEquiposBitácoraPermisosReporteSalir

Asignar Permisos a Usuarios

Seleccione un usuario:

leslie

Permisos para leslie

Seleccione un grupo:

teclab

☐ Puede agregar ítem

☒ Puede ver ítem

☐ Puede eliminar ubicación

☐ Puede cambiar marca

☒ Puede agregar proveedor

☐ Puede ver proveedor

☒ Puede eliminar registro

☒ Puede cambiar ítem

☐ Puede agregar ubicación

☒ Puede ver ubicación

☐ Puede eliminar marca

☒ Puede cambiar proveedor

☐ Puede agregar registro

☒ Puede ver registro

☒ Puede eliminar ítem

☒ Puede cambiar ubicación

☒ Puede agregar marca

☒ Puede ver marca

☒ Puede eliminar proveedor

☒ Puede cambiar registro

☐ Activo

Guardar

Uso de bitácora y reportes de recetas

- Descripción:** modulo para ver las recetas usadas y generar reportes por fechas de las recetas
- Acción:** los usuarios pueden ver las recetas usadas y pueden generar reportes para ver el costo total de las recetas usadas.

PerfilInicioInventarioRecetasItemsMarcaProveedoresTiposEquiposBitácoraPermisosReporteSalir

Usos de Recetas

Generar Reporte

Receta	Cantidad	Cotización Total	Fecha de Uso
nueva receta	1	\$34,70	18 Jun 2024 13:56
tubo receta	5	\$912,50	18 Jun 2024 13:44
nueva receta	2	\$69,40	18 Jun 2024 10:21
nueva receta	1	\$34,70	17 Jun 2024 12:35
nueva receta	2	\$69,40	17 Jun 2024 12:21
pruebas	2	\$347,00	17 Jun 2024 00:45
nueva receta	3	\$520,50	17 Jun 2024 00:45
pruebas con cantidad	2	\$3047,80	17 Jun 2024 00:40



Receta	Cantidad	Cotización Total
nueva receta	9	\$728.70
tubo receta	5	\$912.50
pruebas	2	\$347.00
pruebas con cantidad	2	\$3047.80
Total Cantidad: 18		Total Costo: \$5036.00

Dentro del views cada uno tiene su función para poder funcionar de manera correcta dentro de la aplicación.

```
@login_required
def index(request):
    return render(request, 'index.html')
```

Visualizar los diferentes templates o HTML dentro de la pagina.

```

@login_required
@permission_required('app.view_bitacora')
def bitacora(request):
    # Diccionario de traducción
    traduccion_modelos = {
        'Item': 'Producto',
        'Location': 'Ubicación',
        'Type': 'Tipo'
    }

    registros = Bitacora.objects.all().order_by('-fecha_hora')

    # Traducir los nombres de los modelos en los registros
    for registro in registros:
        if registro.modelo in traduccion_modelos:
            registro.modelo = traduccion_modelos[registro.modelo]

    return render(request, 'bitacora.html', {'registros': registros})

```

Mostrar bitácora y agregando traducciones para las tablas las cuales son item, location y type para mejor lectura dentro de la bitácora.

Aparte de que dentro de la bitácora mostrara la hora de cuando se hizo la acción y los diferentes modelos en donde estén configurado.

```

@permission_required('app.delete_marca')
@login_required
def eliminar_marca(request, marca_id):
    marca = get_object_or_404(Marca, id=marca_id)
    if request.method == 'POST':
        descripcion_personalizada = request.POST.get('descripcion_personalizada', '')
        usuario = request.user

        # Llama a delete y pasa los argumentos adicionales
        marca.delete(usuario=usuario, descripcion_personalizada=descripcion_personalizada)

        return JsonResponse({'message': 'Marca eliminada exitosamente', 'usuario': usuario.username})
    else:
        return JsonResponse({'error': 'Método no permitido'}, status=405)

```

Función para eliminar: esta función es la misma para todos los modelos en donde por el método post obtiene el id de la marca que se eliminara y al eliminarlo obtendría el usuario que hace dicha acción y la descripción personalizada del usuario del porque la esta eliminando.



```

permission_required('app.add_marca')
@login_required
def marcaregistro(request):
    if request.method == 'POST':
        form = marcaform(request.POST)
        if form.is_valid():
            marca = form.save(commit=False)
            marca.usuario = request.user
            marca.save()
            messages.success(request, 'Se agregó la marca correctamente')
            return redirect('marca')
        else:
            form = marcaform()
    return render(request, 'insertar_marca.html', {'form': form})

```

Función para registrar, esta función es la misma para los diferentes modelos lo único que puede cambiar es el nombre del form ya que cambia sería el nombre de ese formulario. Y en caso de que no se use el formulario y se cree uno propio tienes que colocar el nombre, el método post y el nombre dentro del formulario.

```

from decimal import Decimal
@permission_required('app.add_registro')
@login_required
def registrar_item(request):
    items = Item.objects.all()

    if request.method == 'POST':
        item_id = int(request.POST.get('item'))
        item = get_object_or_404(Item, pk=item_id)

        cod_barras = request.POST.get('cod_barras')
        no_referencia_inv = request.POST.get('no_referencia_inv')
        fecha_caducidad = request.POST.get('fecha_caducidad') or None # Tratar valor
        lote = request.POST.get('lote')
        fecha_recepcion = request.POST.get('fecha_recepcion') or None # Tratar valor
        cantidad = int(request.POST.get('cantidad'))
        cod = request.POST.get('cod')
        status = request.POST.get('status')

        # Obtener el precio y convertirlo a float
        precio_str = request.POST.get('precio')
        precio = Decimal(precio_str.replace(',','.')) if precio_str else None

```

```

@login_required
@permission_required('app.change_marca')
def editar_marca(request, marca_id):
    marca = get_object_or_404(Marca, id=marca_id)

    if request.method == 'POST':
        form = marcaform(request.POST, instance=marca)
        if form.is_valid():
            marca = form.save(commit=False)
            marca.usuario = request.user
            marca.save()

            # Verificar los campos modificados en el formulario
            cambios_realizados = []
            if 'nombre' in form.changed_data:
                cambios_realizados.append(f"se editó el nombre")
            if 'descripcion' in form.changed_data:
                cambios_realizados.append(f"se editó la descripción")

            descripcion_personalizada = request.POST.get('descripcion_personalizada', '')
            if not descripcion_personalizada:

```

```

                descripcion_personalizada = request.POST.get('descripcion_personalizada', '')
            if not descripcion_personalizada:
                messages.warning(request, 'No se proporcionó ninguna descripción personalizada.')
                Bitacora.objects.create(
                    accion='Actualizar',
                    usuario=request.user,
                    modelo='Marca',
                    instancia_id=marca.id,
                    descripcion=', '.join(cambios_realizados)
                )
            else:
                # Construir el mensaje completo para la bitácora
                mensaje_bitacora = f"Se actualizó la marca con ID: {marca.id} ({', '.join(cambios_realizados)})."

                Bitacora.objects.create(
                    accion='Actualizar',
                    usuario=request.user,
                    modelo='Marca',
                    instancia_id=marca.id,
                    descripcion=mensaje_bitacora
                )

```

Dentro de esta función se optionen los campos dentro del formulario por el método post y se guardan con los nuevos datos editados con la diferencia que hay unas líneas de código mas que serán para la bitácora con los campos dentro del formulario para saber que se edito.

Creación de reportes: esta función creara un reporte donde recibirá parámetros por el metodo POST los cuales son las fechas de inicio y la fecha de fin y en base a esas fechas calculara y agrupara todas las recetas que se usaron, la cantidad y el costo total para posteriormente generar un PDF con la biblioteca de canva.

```
@permission_required('app.view_recetas')
@login_required
def generar_reporte(request):
    if request.method == 'POST':
        fecha_inicio = request.POST.get('fecha_inicio')
        fecha_fin = request.POST.get('fecha_fin')

        if fecha_inicio and fecha_fin:
            fecha_inicio = timezone.datetime.strptime(fecha_inicio, '%Y-%m-%d').date()
            fecha_fin = timezone.datetime.strptime(fecha_fin, '%Y-%m-%d').date()

            usos_recetas = UsoReceta.objects.filter(fecha_uso__date__range=(fecha_inicio, fecha_fin)).order_by('fecha_uso')

            receta_agrupada = defaultdict(lambda: {'cantidad': 0, 'cotizacion_total': 0})
            for uso in usos_recetas:
                receta_agrupada[uso.receta.nombre]['cantidad'] += uso.cantidad
                receta_agrupada[uso.receta.nombre]['cotizacion_total'] += uso.cotizacion_total
```

```
# Crear el PDF
buffer = BytesIO()
p = canvas.Canvas(buffer, pagesize=letter)

# Función para agregar una nueva página y el encabezado de la tabla
def agregar_pagina():
    p.showPage()
    p.setFont("Helvetica-Bold", 12)
    p.drawString(0.5 * inch, 9.5 * inch, "Receta")
    p.drawString(2 * inch, 9.5 * inch, "Cantidad")
    p.drawString(3.5 * inch, 9.5 * inch, "Cotización Total")
    p.setFont("Helvetica", 10)
    return 9.25 * inch

# Añadir la imagen más abajo y más estirada
image_path = 'app/static/imagenes/uusmb.png'
p.drawImage(image_path, 0.5 * inch, 9.5 * inch, width=2 * inch, height=1 * inch)

# Título del reporte
p.setFont("Helvetica-Bold", 16)
p.drawCentredString(4.25 * inch, 10 * inch, "Reporte de Recetas")
```

```

# Subtítulo con fechas
p.setFont("Helvetica", 12)
p.drawCentredString(4.25 * inch, 9.7 * inch, f"Del {fecha_inicio.strftime('%d/%m/%Y')} al {fecha_fin.strftime('%d/%m/%Y')}")

# Líneas divisorias
p.setStrokeColor(colors.black)
p.setLineWidth(1)
p.line(0.5 * inch, 9.5 * inch, 7.5 * inch, 9.5 * inch)

# Table headers
p.setFont("Helvetica-Bold", 12)
p.drawString(0.5 * inch, 9.0 * inch, "Receta")
p.drawString(2 * inch, 9.0 * inch, "Cantidad")
p.drawString(3.5 * inch, 9.0 * inch, "Cotización Total")

# Líneas divisorias
p.setLineWidth(0.5)
p.line(0.5 * inch, 8.95 * inch, 7.5 * inch, 8.95 * inch)

y = 8.75 * inch
total_cantidad = 0

```

```

for nombre_receta, datos in receta_agrupada.items():
    p.drawString(0.5 * inch, y, nombre_receta)
    p.drawString(2 * inch, y, str(datos['cantidad']))
    p.drawString(3.5 * inch, y, f"${datos['cotizacion_total']:.2f}")

    total_cantidad += datos['cantidad']
    total_costo += datos['cotizacion_total']

    y -= 0.25 * inch

    if y < 1 * inch:
        y = agregar_pagina()

if y < 1.5 * inch:
    y = agregar_pagina() - 0.25 * inch

y -= 0.25 * inch
p.setFont("Helvetica-Bold", 12)
p.drawString(1 * inch, y, f"Total Cantidad: {total_cantidad}")
p.drawString(3.5 * inch, y, f"Total Costo: ${total_costo:.2f}")

p.showPage()

```