
PERSONALIZANDO ADMINISTRADOR

CONTINUACIÓN



GRUPOS Y PERMISOS DE USUARIO PARA ACCESO AL MÓDULO DE ADMINISTRACIÓN

- Crearemos un grupo para gestionar los permisos de los usuarios que podrán acceder al módulo de administración.
- Accederemos al módulo de grupos:

| AUTHENTICATION AND AUTHORIZATION | |
|----------------------------------|--|
| Groups | + Add ✎ Change |
| Users | + Add ✎ Change |

GRUPOS Y PERMISOS DE USUARIO PARA ACCESO AL MÓDULO DE ADMINISTRACIÓN

- Crearemos el grupo Usuarios.
- Observarás una lista de permisos, para cada modelo podemos elegir 3 permisos (alta, baja y eliminación).
- Selecciona los permisos que las personas del grupo podrán realizar.
- En nuestro caso agregaremos los permisos de registro y edición para alumnos y comentarios contacto.

Name:

Permissions:

Available permissions ?

Content Types | content type | Can view content type

Módulos | Alumno | Can delete alumnos

Módulos | Alumno | Can view alumnos

Módulos | Comentario | Can add Comentario

Módulos | Comentario | Can change Comentario

Módulos | Comentario | Can delete Comentario

Módulos | Comentario | Can view Comentario

Módulos | Comentario Contacto | Can delete Comentario C

Módulos | Comentario Contacto | Can view Comentario Co

Sessions | session | Can add session

Sessions | session | Can change session

Sessions | session | Can delete session

Sessions | session | Can view session

Choose all ?

Hold down "Control", or "Command" on a Mac, to select more than one.

Chosen permissions ?

Módulos | Alumno | Can add alumnos

Módulos | Alumno | Can change alumnos

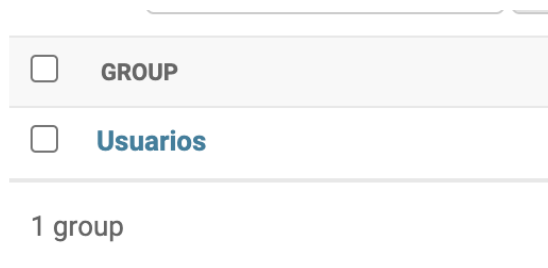
Módulos | Comentario Contacto | Can add Comentario Cor

Módulos | Comentario Contacto | Can change Comentario

Remove all

GRUPOS Y PERMISOS DE USUARIO PARA ACCESO AL MÓDULO DE ADMINISTRACIÓN

- Ahora con nuestro grupo creado, procederemos a asignar un usuario al grupo.

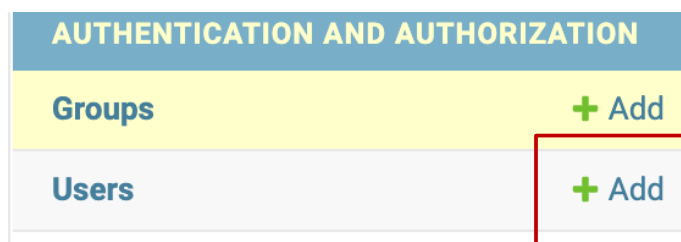


A screenshot of a user assignment interface. It features a list with two items: 'GROUP' and 'Usuarios'. Each item has an unchecked checkbox to its left. Below the list, the text '1 group' is displayed.

| | |
|--------------------------|----------|
| <input type="checkbox"/> | GROUP |
| <input type="checkbox"/> | Usuarios |

1 group

- Ingresa al módulo de usuario y podremos a agregar:



A screenshot of the 'AUTHENTICATION AND AUTHORIZATION' module. It shows a table with two rows: 'Groups' and 'Users'. The 'Groups' row has a green '+ Add' button. The 'Users' row has a green '+ Add' button, which is highlighted with a red rectangular box.

| AUTHENTICATION AND AUTHORIZATION | |
|----------------------------------|-------|
| Groups | + Add |
| Users | + Add |

GRUPOS Y PERMISOS DE USUARIO PARA ACCESO AL MÓDULO DE ADMINISTRACIÓN

- Crea un usuario
de prueba y
guarda.

Add user

First, enter a username and password. Then, you'll be able to edit more user options.

Username:

Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only.

Password:

Your password can't be too similar to your other personal information.

Your password must contain at least 8 characters.

Your password can't be a commonly used password.

Your password can't be entirely numeric.

Password confirmation:

Enter the same password as before, for verification.

GRUPOS Y PERMISOS DE USUARIO PARA ACCESO AL MÓDULO DE ADMINISTRACIÓN

- Se aperturará una nueva página donde se nos solicitará diversa informacion.
- Agrega los datos de identificación del usuario:

Personal info

First name:

Usuario

Last name:

Prueba

Email address:

prueba@gmail.com

GRUPOS Y PERMISOS DE USUARIO PARA ACCESO AL MÓDULO DE ADMINISTRACIÓN

- Marca las Casillas de Activo y Staff status, este último indica que el usuario puede acceder al modulo de administración.
- Asignamos el grupo creado a nuestro nuevo usuario.

Permissions

☒ Active
Designates whether this user should be treated as active. Unselect this instead of deleting accounts.

☒ Staff status
Designates whether the user can log into this admin site.

☐ Superuser status
Designates that this user has all permissions without explicitly assigning them.

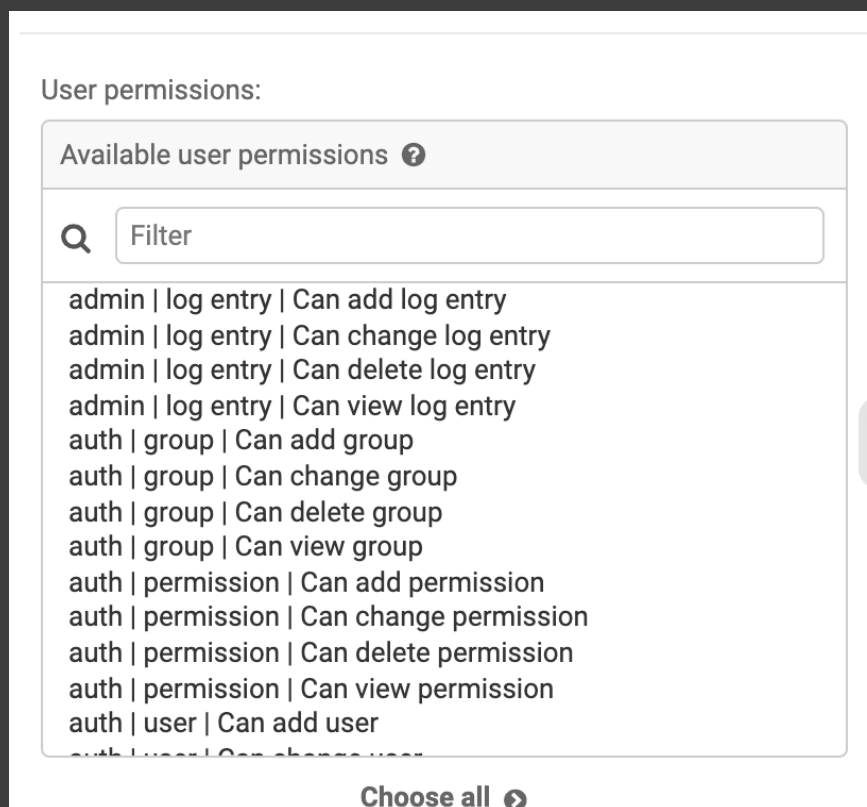
Groups:

Available groups ?
Filter

Chosen groups ?
Usuarios

GRUPOS Y PERMISOS DE USUARIO PARA ACCESO AL MÓDULO DE ADMINISTRACIÓN


- Es posible agregar permisos individuales al usuario si es necesario. En nuestro caso no asignaremos ninguno ya que los permisos están asociados al grupo.




Important dates

Last login:

Date:

Today | 

Time:


Now | 

Note: You are 5 hours behind server time.

Date joined:


Date:

2021-07-08

Today | 

Time:

17:34:26

Now | 

Note: You are 5 hours behind server time.

Delete

Save and add another

Save and continue editing

SAVE

GRUPOS Y PERMISOS DE USUARIO PARA ACCESO AL MÓDULO DE ADMINISTRACIÓN

- Procedemos a guardar los datos registrados.

| <input type="checkbox"/> | USERNAME | EMAIL ADDRESS | FIRST NAME | LAST NAME | STAFF STATUS |
|--------------------------|----------|------------------|------------|-----------|--------------|
| <input type="checkbox"/> | elena | elena@utm.com | | | ✓ |
| <input type="checkbox"/> | prueba | prueba@gmail.com | Usuario | Prueba | ✓ |

2 users

GRUPOS Y PERMISOS DE USUARIO PARA ACCESO AL MÓDULO DE ADMINISTRACIÓN

- Debemos visualizar en la tabla de usuarios a nuestro usuario de prueba.

GRUPOS Y PERMISOS DE USUARIO PARA ACCESO AL MÓDULO DE ADMINISTRACIÓN

WELCOME, **ELENA**. [VIEW SITE](#) / [CHANGE PASSWORD](#) / [LOG OUT](#)

- Cerramos sesión e ingresamos nuevamente con nuestro usuario de prueba con permisos limitados.

Django administration

Username:

Password:

Log in

GRUPOS Y PERMISOS DE USUARIO PARA ACCESO AL MÓDULO DE ADMINISTRACIÓN

Módulos administration

| MÓDULOS | |
|-----------------------|--|
| Alumnos | + Add ✎ Change |
| Comentarios Contactos | + Add ✎ Change |

- Observa que unicamente se habilitan los módulos a los cuales tenemos acceso y las funciones de inserción y modificación permitidos.

GRUPOS Y PERMISOS DE USUARIO PARA ACCESO AL MÓDULO DE ADMINISTRACIÓN

- Con nuestro nuevo usuario, témenos acceso a realizar modificación de alumnos.
- Imagina que te encuentras en SIGO en el módulo para administrar tus datos personales. **¿Qué datos del formulario no podrías modificar?**

Change Alumno

[HISTORY](#)

Juan

Matricula:

UTM1234

Nombre:

Juan

Carrera:

TI

Turno:

Matutino

Fotografía:

Currently: [fotos/seUTM.png](#)

Change: [Seleccionar archivo](#) No se eligió archivo

Created:

July 15, 2021, 6:19 a.m.

Updated:

July 15, 2021, 6:19 a.m.

GRUPOS Y PERMISOS DE USUARIO PARA ACCESO AL MÓDULO DE ADMINISTRACIÓN

- Actualmente el modelo muestra como solo lectura los campos de fecha de creación y modificación.
- Podemos cambiar en tiempo de ejecución los campos que aparecerán bloqueados acorde al grupo de permisos asociados al usuario que inicia sesión.

Change Alumno

HISTORY

Juan

Matricula:

UTM1234

Nombre:

Juan

Carrera:

TI

Turno:

Matutino

Fotografía:

Currently: [fotos/seUTM.png](#)

Change: [Seleccionar archivo](#) No se eligió archivo

Created:

July 15, 2021, 6:19 a.m.

Updated:

July 15, 2021, 6:19 a.m.

GRUPOS Y PERMISOS DE USUARIO PARA ACCESO AL MÓDULO DE ADMINISTRACIÓN

- Ubica la clase empleada para administrar el modelo de alumno

```
class AdministrarModelo(admin.ModelAdmin):  
    readonly_fields = ('created', 'updated')  
    list_display = ('matricula', 'nombre', 'carrera', 'turno')  
    search_fields = ('matricula', 'nombre', 'carrera', 'turno')  
    date_hierarchy = 'created'  
    list_filter = ('carrera', 'turno')
```

registros/admin.py

GRUPOS Y PERMISOS DE USUARIO PARA ACCESO AL MÓDULO DE ADMINISTRACIÓN

```
class AdministrarModelo(admin.ModelAdmin):
    readonly_fields = ('created', 'updated')
    list_display = ('matricula', 'nombre', 'carrera', 'turno')
    search_fields = ('matricula', 'nombre', 'carrera', 'turno')
    date_hierarchy = 'created'
    list_filter = ('carrera', 'turno')

    def get_readonly_fields(self, request, obj=None):
        #si el usuario pertenece al grupo de permisos "Usuario"
        if request.user.groups.filter(name="Usuarios").exists():
            #Bloquea los campos
            return ('created', 'updated', 'matricula', 'carrera', 'turno')
            #Cualquier otro usuario que no pertenece al grupo "Usuario"
        else:
            #Bloquea los campos
            return ('created', 'updated')
```

registros/admin.py

GRUPOS Y PERMISOS DE USUARIO PARA ACCESO AL MÓDULO DE ADMINISTRACIÓN

- Actualizas el navegador.
- Observa que los campos ya no son editables.

Change Alumno

HISTORY

Juan

Nombre:

Juan

Fotografía:

Currently: fotos/seUTM.png

Change: No se eligió archivo

Created: July 15, 2021, 6:19 a.m.

Updated: July 15, 2021, 6:19 a.m.

Matricula: UTM1234

Carrera: TI

Turno: Matutino

Save and add another

Save and continue editing

SAVE

GRUPOS Y PERMISOS DE USUARIO PARA ACCESO AL MÓDULO DE ADMINISTRACIÓN

```
def get_readonly_fields(self, request, obj=None):
    #si el usuario pertenece al grupo de permisos "Usuario"
    if request.user.groups.filter(name="Usuarios").exists():
        #Bloquea los campos
        return ('matricula', 'carrera', 'turno')
    #Cualquier otro usuario que no pertenece al grupo
    "Usuario"
    else:
        #Bloquea los campos
        return ('created', 'updated')
```

- Elimina los campos created y updated

registros/admin.py

GRUPOS Y PERMISOS DE USUARIO PARA ACCESO AL MÓDULO DE ADMINISTRACIÓN

- Actualizas el navegador.
- Como los campos de fecha son automáticos, podemos ocultarlos al eliminarlos de la lista.

Change Alumno

HISTORY

Juan

Nombre:

Juan

Fotografía:

Currently: [fotos/seUTM.png](#)

Change: No se eligió archivo

Matricula: UTM1234

Carrera: TI

Turno: Matutino

Save and add another

Save and continue editing

SAVE

GRUPOS Y PERMISOS DE USUARIO PARA ACCESO AL MÓDULO DE ADMINISTRACIÓN

- Cierra sesión e ingresa con tu usuario administrador.

- Deberás poder editar los campos que no están disponibles para nuestro grupo “Usuarios”

Juan

HISTORY

Matricula:

UTM1234

Nombre:

Juan

Carrera:

TI

Turno:

Matutino

Fotografía:

Currently: [fotos/seUTM.png](#)

Change: [Seleccionar archivo](#) No se eligió archivo

Created:

July 15, 2021, 6:19 a.m.

Updated:

July 15, 2021, 6:19 a.m.

Delete

Save and add another

Save and continue editing

SAVE

CONSULTAS

- Realiza una copia de principal.html
- Este archivo tiene la consulta de alumnos que emplearemos para probar diferentes funciones de consulta.
- Nombra el archivo como **consultas.html**

```
✓ templates / registros
  <> confirmarEliminacion.html
  <> consultaContacto.html
  <> consultas.html
  <> contacto.html
  <> formEditarComentario.html
  <> principal.html
```

-
- Ingresa al módulo de administración y registra los siguientes alumnos:

| Foto | Matricula | Nombre | Carrera | Turno |
|---|------------|--------|---------|------------|
|  | UTM7785TI | Monica | TI | Vespertino |
|  | UTM1234 | Juan | TI | Matutino |
|  | utm2 | Ana | BIO | Vespertino |
|  | UTM1234TIC | Juan | TI | Matutino |

CONSULTAR CON CONDICIÓN

Función FILTER

filter nos retornará los registros que coinciden con los parámetros de #búsqueda dados.

```
def consultar1(request):  
    #con una sola condición  
    alumnos=Alumnos.objects.filter(carrera="TI")  
    return render(request,"registros/consultas.html",{ 'alumnos':alumnos})
```

views.py

CONSULTAS CON CONDICIÓN

- Agregamos una nueva url:

```
path('consultas1', views_registros.consultar1, name="Consultas"),
```

urls.py

CONSULTAS CON CONDICIÓN

- Ejecutamos servidor y accedemos a la ruta:
- `http://127.0.0.1:8000/consultas1`

| Foto | Matricula | Nombre | Carrera | Turno |
|---|------------|--------|---------|------------|
|  Universidad Tecnológica de Morelia | UTM7785TI | Monica | TI | Vespertino |
|  ¡Únete a la Universidad Tecnológica de Morelia! | UTM1234 | Juan | TI | Matutino |
|  | UTM1234TIC | Juan | TI | Matutino |

```
alumnos=Alumnos.objects.filter(carrera="TI")
```

CONSULTAR CON MULTIPLES CONDICIONES

```
def consultar2(request):  
    #multiples condiciones adicionando .filter() se analiza #como AND  
    alumnos=Alumnos.objects.filter(carrera="TI").filter(turno="Matutino")  
    return render(request,"registros/consultas.html",{ 'alumnos':alumnos})
```

views.py

CONSULTAR CON MULTIPLES CONDICIONES

- Agregamos una nueva url:

```
path('consultas2', views_registros  
.consultar2, name="Consultas2"),
```

urls.py

CONSULTAS

- Ejecutamos servidor y accedemos a la ruta:
 - `http://127.0.0.1:8000/consultas2`

| Foto | Matricula | Nombre | Carrera | Turno |
|---|------------|--------|---------|----------|
|  | UTM1234 | Juan | TI | Matutino |
|  | UTM1234TIC | Juan | TI | Matutino |

```
alumnos=Alumnos.objects.filter(carrera="TI").filter(turno="Matutino")
```

CONSULTAR SOLO CIERTOS CAMPOS

```
def consultar3(request):  
    #Si solo deseamos recuperar ciertos datos agregamos la #función only,  
    #listando los campos que queremos obtener de #la consulta emplear  
    #filter() o #en el ejemplo all()  
    alumnos=Alumnos.objects.all().only("matricula", "nombre", "carrera",  
    "turno", "imagen")  
    return render(request,"registros/consultas.html",{ 'alumnos':alumnos})
```

views.py

CONSULTAR SOLO CIERTOS CAMPOS

- Agregamos una nueva url:

```
path('consultas3', views_registros.  
consultar3, name="Consultas3"),
```

urls.py

CONSULTAR SOLO CIERTOS CAMPOS

- Ejecutamos servidor y accedemos a la ruta:
 - <http://127.0.0.1:8000/consultas3>

| Foto | Matricula | Nombre | Carrera | Turno |
|---|------------|--------|---------|------------|
|  | UTM7785TI | Monica | TI | Vespertino |
|  | UTM1234 | Juan | TI | Matutino |
|  | utm2 | Ana | BIO | Vespertino |
|  | UTM1234TIC | Juan | TI | Matutino |

```
alumnos=Alumnos.objects.all().only("matricula", "nombre",  
"carrera", "turno", "imagen")
```

CONSULTAR CON EXPRESIONES

- Podemos agregar expresiones de consulta colocando el nombre del campo dos guiones bajos y la expresión:

campo__expresion

- Ejemplos de expresiones:
 - __contains: **LIKE**
 - __exact: **IGUAL**
 - __iexact: **NO DISTINGUE ENTRE MAYUSCULAS Y MINUSCULAS**

CONSULTAR CON EXPRESIONES

■ Ejemplos:

- **__lt: MENOR QUE**
- **__lte: MENOR O IGUAL QUE**
- **__gt: MAYOR QUE**
- **__gte: MAYOR O IGUAL QUE**
- **__in: VERIFICA UN INTERVALO DE VALORES [1, 3, 4]**
- **__startswith: INICIA CON**
- **__endswith: TERMINA CON**
- **__range : BUSQUEDA POR RANGO DE FECHAS**

DOCUMENTACIÓN OFICIAL:

- <https://docs.djangoproject.com/en/3.2/ref/models/expressions/>
- <https://docs.djangoproject.com/en/3.2/ref/models/querysets/#field-lookups>

CONSULTAR CON EXPRESIONES

__contains: LIKE

```
def consultar4(request):  
    alumnos=Alumnos.objects.filter(turno__contains="Vesp")  
    return  
    render(request,"registros/consultas.html",{ 'alumnos':alumnos  
    })
```

views.py

CONSULTAR CON EXPRESIONES



- Agregamos una nueva url:

```
path('consultas4', views_registros.consultar4, name="Consulta4"),
```

urls.py

CONSULTAR CON EXPRESIONES

- Ejecutamos servidor y accedemos a la ruta:
 - `http://127.0.0.1:8000/consultas4`

| Foto | Matricula | Nombre | Carrera | Turno |
|--|-----------|--------|---------|------------|
|  | UTM7785TI | Monica | TI | Vespertino |
|  | utm2 | Ana | BIO | Vespertino |

```
alumnos=Alumnos.objects.filter(turno__contains="Vesp")
```

CONSULTAR CON EXPRESIONES

`__in`: VERIFICA UN INTERVALO DE VALORES

```
def consultar5(request):  
    alumnos=Alumnos.objects.filter(nombre__in=["Juan", "Ana"])  
    return  
    render(request,"registros/consultas.html",{ 'alumnos':alumnos})
```

views.py

CONSULTAR CON EXPRESIONES

- Agregamos una nueva url:

```
path('consultas5', views_registros.consultar5,  
     name="Consulta5"),
```

urls.py

CONSULTAR CON EXPRESIONES

- Ejecutamos servidor y accedemos a la ruta:
 - <http://127.0.0.1:8000/consultas5>

| Foto | Matricula | Nombre | Carrera | Turno |
|---|------------|--------|---------|------------|
|  | UTM1234 | Juan | TI | Matutino |
|  | utm2 | Ana | BIO | Vespertino |
|  | UTM1234TIC | Juan | TI | Matutino |

```
alumnos=Alumnos.objects.filter(nombre__in=["Juan", "Ana"])
```

CONSULTAR CON EXPRESIONES

__range : BUSQUEDA POR RANGO DE FECHAS

```
import datetime
```

```
def consultar6(request):  
    fechaInicio = datetime.date(2022, 7, 1)  
    fechaFin = datetime.date(2022, 7, 13)  
    alumnos=Alumnos.objects.filter(created__range=(fechaInicio,fechaFin))  
    return render(request,"registros/consultas.html",{ 'alumnos':alumnos})
```

views.py

CONSULTAR CON EXPRESIONES

- Agregamos una nueva url:

```
path('consultas6', views_registros.consultar6,  
     name="Consulta6"),
```

urls.py

CONSULTAR CON EXPRESIONES

Para visualizar la fecha se colocó el campo created en la lista de campos a mostrar en el archivo admin.py

```
list_display = ('matricula', 'nombre', 'carrera', 'turno', 'created')
```

| NOMBRE | CARRERA | TURNO | CREATED |
|--------|---------|------------|--------------------------|
| Monica | TI | Vespertino | July 15, 2021, 7:52 a.m. |
| Juan | TI | Matutino | July 15, 2021, 6:19 a.m. |
| Ana | BIO | Vespertino | July 1, 2021, 7:39 p.m. |
| Juan | TI | Matutino | June 25, 2021, 2:28 a.m. |

Datos del
administrador

Consulta a realizar sobre los datos

```
fechaInicio = datetime.date(2021, 7, 1)
fechaFin = datetime.date(2021, 7, 13)
alumnos=Alumnos.objects.filter(created__range=(fechaInicio,fechaFin))
```

CONSULTAR CON EXPRESIONES

- Ejecutamos servidor y accedemos a la ruta:
 - <http://127.0.0.1:8000/consultas6>

Salida

| Foto | Matricula | Nombre | Carrera | Turno |
|---|-----------|--------|---------|------------|
|  | UTM7785TI | Monica | TI | Vespertino |
|  | UTM1234 | Juan | TI | Matutino |
|  | utm2 | Ana | BIO | Vespertino |

```
fechaInicio = datetime.date(2021, 7, 1)
fechaFin = datetime.date(2021, 7, 13)
alumnos=Alumnos.objects.filter(created__range=(fechaInicio,fechaFin))
```

CONSULTAR ENTRE MODELOS - RELACIONES

Si recuerdas, nuestro modelo de Alumnos esta relacionado con un modelo comentario que administramos desde el módulo de admin:

MÓDULOS

Alumnos

+ Add

Comentarios

+ Add

Comentarios Contactos

+ Add

Q

Search

◀ 2021

June 29

Action:

▼

Go

0 of 1 selected

| <input type="checkbox"/> | CLAVE | COMENTARIO |
|--------------------------|-------|-------------|
| <input type="checkbox"/> | 1 | No Inscrito |

1 Comentario

CONSULTAR ENTRE MODELOS - RELACIONES

- Para realizar una consulta entre modelos, empleamos la función filter, pero en la condición es posible indicar el nombre del modelo y el campo que deseamos condicionar.
- En nuestro ejemplo la condición es que el alumno este en comentario y que el comentario sea No Inscrito :

```
def consultar7(request):  
    #Consultando entre modelos  
    alumnos=Alumnos.objects.filter(comentario__coment__contains='No inscrito')  
    return render(request,"registros/consultas.html',{'alumnos':alumnos})
```

views.py

CONSULTAR ENTRE MODELOS - RELACIONES

- Agregamos una nueva url:


```
path('consultas7', views_registros.consultar7,  
      name="Consulta7"),
```

urls.py

CONSULTAR ENTRE MODELOS - RELACIONES

- Ejecutamos servidor y accedemos a la ruta:
 - <http://127.0.0.1:8000/consultas7>

Salida

| Foto | Matricula | Nombre | Carrera | Turno |
|---|------------|--------|---------|----------|
|  | UTM1234TIC | Juan | TI | Matutino |

```
alumnos=Alumnos.objects.filter(comentario__coment__contains='No Inscrito')
```