



# Sistema Integral de Manejo de Información COE 2020

## Sinopsis:

El sistema COE2020, accesible desde la pagina [coe.jujuy.gob.ar](http://coe.jujuy.gob.ar) es de código abierto y público, su versión más actualizada es accesible desde: <https://gitlab.com/mmandrille>

El mismo cuenta con **11 módulos**, se comenzó su desarrollo el 13/03/2020 pero solo hablaremos en este documento resumidamente de los más importantes a los cuales dividiremos en 2 conjuntos:

- **Privados** (Solo accesible con usuarios autorizados y que tienen permisos específicos)
  - **Información**
  - Operadores
  - Georeferenciar
  - Inventario
  - Tareas
  - Documentos Privados
  - Gráficos Privados
  - Web Service
- **Públicos** (Aquellos que están pensados para presentar información a la gente)
  - Noticias
  - Consultas
  - Inscripciones
  - Documentos Públicos
  - Gráficos Públicos

# Tecnologías:

**Backend:** Python 3.6 + Django 2.1

**Servidor:** Nginx (*Ver configuración en: `coe.nginx`*), este servicio es levantado automáticamente y mantiene el contacto con el **gunicorn.sh** ejecutado via supervisord (*Ver configuración en: `coe.conf` el cual lanza gunicorn.sh*)

**Base Datos:** Oracle (PRD) / Sqlite (DEV)

La diferenciación se hace vía archivo `coe/coe/credenciales.py` (El cual no está incluido dentro del repositorio) donde definimos el motor de base de datos a utilizar.

A su vez el sistema de backgrounds depende del lanzamiento desde supervisord: `coe_bg.conf` el cual lanza `backgrounds.sh`

# Estructura:

Los archivos iniciales del proyecto son:

- **coe/settings.py** (Definimos inicialmente todo lo que nuestro proyecto utilizará, tiene vital importancia la lista **INSTALLED\_APPS** donde no solo están nuestras apps si no las dependencias que utilizamos)
- **coe/urls.py** (Definimos según las url de acceso que app de nuestro proyecto la procesara)

Cada app cuenta con un grupo de archivos obligatorios:

- **urls.py** > Todo pedido que llega al servidor, es interpretado por una línea específica y enviado a una VISTA, la cual realiza el procesamiento y devuelve la respuesta.
- **views.py** > Las vistas son simplemente funciones de python a las que se le ingresan ciertos parámetros como la REQUEST y otros opcionales, realiza cierto procesamiento y devuelve un HTML al usuario.
- **models.py** > Cada clase definida dentro de este archivo conforma una tabla en la base de datos, además de cada uno de los atributos iniciales que definen los campos de la tabla, puede contar con funciones que pasan a formar parte de métodos específicos que pueden requerirse.

Como dato extra: Al final del documento hay dos tipos de llamados que no conforman tablas:

- **Auditlog** (Registra el modelo para ser monitorizado y cada cambio que se le realice es almacenado en un historial que nos indica que se le cambio y quien fue el que lo hizo).
- **Signals** (Son un conjunto de instrucciones definidas en el archivo `signals.py` de la app encargados de realizar procesos cuando se guarda o elimina un archivo, por ejemplo en `información/signals.py`, línea 96, antes de guardar un domicilio

que NO es de aislamiento, si el individuo estaba aislado en una ubicación, recuperamos la plaza disponible del mismo).

- **apps.py** > Este archivo presenta la app al proyecto, a su vez en caso de ser necesario utiliza una función custom que agrega al menú de la app automáticamente la misma.

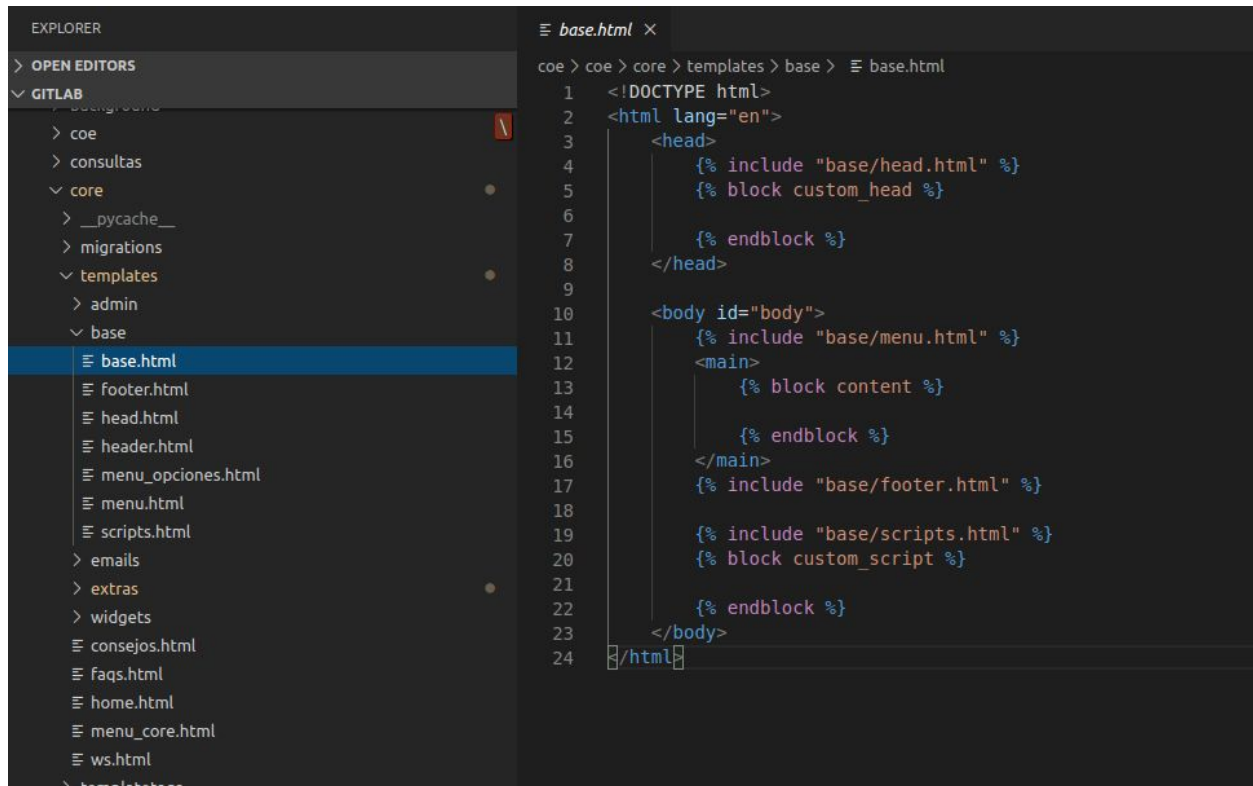
También existen archivos opcionales:

- **forms.py** > Definición de formularios generados dinámicamente, estos permiten a partir de un modelo de datos (tabla), generar de manera automática todos los campos y controles para el template.
- **signals.py** > Procesos automáticos que realiza el sistema cuando se afecta la db.
- **functions.py** > funciones reutilizables que pueden ser llamadas por varias vistas.
- **autocomplete.py** > Funciones relacionadas a select2.js para ofrecer el servicio de autocompletado en los forms con valores de nuestras tablas (*requieren ser declaradas en el urls.py*).
- **templatetags/<foo>.py** > En esta carpeta definimos funcionalidades que serán utilizadas dentro del template mientras el mismo se va creando para simplificar u optimizar el renderizado de las mismas, el ejemplo más claro puede observarse en **core/templatetags/menu\_tags.py** donde automáticamente generamos los módulos a los que el usuario puede acceder según sus permisos  
**core/templates/base/menu\_opciones.html**

## Templates

Normalmente una vista retornara una página web al usuario, la misma se compone de dos elementos: El **Template** y las **{{Variables}}** que utiliza.

La estructura de Templates que utilizamos está fragmentada para facilitar el mantenimiento y la reutilizabilidad.



```
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     {% include "base/head.html" %}
5     {% block custom_head %}
6
7     {% endblock %}
8   </head>
9
10  <body id="body">
11    {% include "base/menu.html" %}
12    <main>
13      {% block content %}
14
15      {% endblock %}
16    </main>
17    {% include "base/footer.html" %}
18
19    {% include "base/scripts.html" %}
20    {% block custom_script %}
21
22    {% endblock %}
23  </body>
24</html>
```

Tenemos un archivo en **core/templates/base/base.html**:

Él mismo llama a un conjunto de sub templates que definen el head, menu, footer y scripts construyendo el HTML completo que recibe el usuario como se puede ver en la imagen.

El apartado:

**{% block content %}**

**{% endblock %}**

Es el área donde nuestro template llamado en la vista normalmente inserta su código generado, por ejemplo:

Cuando ingresa alguien a la url: <http://coe.jujuy.gob.ar/consultas/lista/consultas> se desencadena el siguiente proceso:

1. **coe/urls.py** en su línea 40:

```
27
28 #Path
29 urlpatterns = [
30     path('admin/', admin.site.urls),
31     #Admin-User Paths
32     url(r'^login/$', auth_views.LoginView.as_view(template_name="users/login.html"), name="login"),
33     url(r'^logout/$', auth_views.LogoutView.as_view(next_page='/'), name='logout'),
34     #Extras
35     url(r'^tinymce/', include('tinymce.urls')),
36     #Debug:
37     path('__debug__/', include(debug_toolbar.urls)),
38     #Apps:
39     path('', include('core.urls')),
40     path('consultas/', include('consultas.urls')),
41     path('georef/', include('georef.urls')),
42     path('noticias/', include('noticias.urls')),
43     path('operadores/', include('operadores.urls')),
44     path('actas/', include('actas.urls')),
45     path('tarefas/', include('tarefas.urls')),
46     path('inventario/', include('inventario.urls')),
47     path('informacion/', include('informacion.urls')),
48     path('graficos/', include('graficos.urls')),
49     path('documentos/', include('documentos.urls')),
50     path('inscripciones/', include('inscripciones.urls')),
51     path('background/', include('background.urls')),
52     #APIS:
53     path('covid19/', include('informacion.api_urls')),
54 ]
55 #Agregamos destinos de Archivos Estaticos
56 urlpatterns += static(settings.STATIC_URL, document_root=settings.STATIC_ROOT)
57 urlpatterns += static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
58
```

Sabe que debe enviar ese pedido completo a **consultas.urls**

2. Entonces la misma llega a **consultas/urls.py**:

```
⌵ faqs.html  + urls.py  x
coe > coe > consultas > + urls.py > ...
1  #Imports de Django
2  from django.conf.urls import url
3  from django.urls import path
4  #Imports de la app
5  from . import views
6  #Definimos nuestros Paths
7  app_name = 'consultas'
8  urlpatterns = [
9      #Publico
10     path('consulta', views.contacto, name='consultas'),
11     #Consultas
12     path('', views.menu, name='menu'),
13     path('lista/consultas', views.lista_consultas, name='lista_consultas'),
14     path('lista/respondidas', views.lista_respondidas, name='lista_respondidas'),
15     path('ver/consulta/<int:consulta_id>', views.ver_consulta, name='ver_consulta'),
16     path('consulta/respondida/<int:consulta_id>', views.consulta_respondida, name='consulta_respondida'),
17
```

En su línea 13, sabe que debe llamar a la función **lista\_consultas** que se encuentra en **consultas/views.py**

3. La misma recibe el pedido, realiza una búsqueda en la tabla Consultas de todas las que hayan sido validadas y no se encuentren respondidas y entrega esta información pidiendo que sea renderizada por el template

## consultas/templates/lista\_consultas.html

```
views.py
43 return render(request, 'menu_consultas.html', {})
44
45 @permission_required('operadores.consultas')
46 def lista_consultas(request):
47     consultas = Consulta.objects.filter(valida=True, respondida=False)
48     return render(request, 'lista_consultas.html', {
49         "consultas": consultas,
50         "has_table": True,
51         "refresh": True,
52     })
53
```

4. El mismo se encarga del display de la información de la siguiente manera:

```
lista_consultas.html
1 {% extends 'base_informacion.html' %}
2
3 {% load static %}
4
5 {% block content %}
6     <div class="beneficiario container">
7         <div class="row">
8             <h2>Consultas Validas:</h2>
9         </div>
10        <div class="row">
11            <div class="col-md-12">
12                <table class="listado" id="table">
13                    <thead>
14                        <tr>
15                            <th>autor</th>
16                            <th>email</th>
17                            <th>telefono</th>
18                            <th>asunto</th>
19                            <th>descripcion</th>
20                            <th>fecha_consulta</th>
21                            <th>Ver Detalle</th>
22                        </tr>
23                    </thead>
24                    <tbody>
25                        {% for consulta in consultas %}
26                            <tr>
27                                <td>{{consulta.autor}}</td>
28                                <td>{{consulta.email}}</td>
29                                <td>{{consulta.telefono}}</td>
30                                <td>{{consulta.asunto}}</td>
31                                <td>{{consulta.descripcion|safe}}</td>
32                                <td>{{consulta.fecha_consulta}}</td>
33                                <td><a href="{% url 'consultas:ver_consulta' consulta_id=consulta.id %}" target="_blank">(Ver Detalle)</a></td>
34                            </tr>
35                        {% endfor %}
36                    </tbody>
37                </table>
38            </div>
39        </div>
40    {% endblock %}
```

- a. Inicialmente genera un loop de todas las consultas que obtuvimos en la vista creando la tabla y llama a **core/template/base/base.html** para rellenar el bloque content entregando el siguiente resultado:

<div> <div>  <div> <div>energía viva</div> </div> </div> <div> <div>Inicio</div> <div>Noticias</div> <div>¿Que es el COE?</div> <div>Contacto</div> <div>Servicio</div> </div> </div>						
Consultas Validas:						
<div> <div>Search</div> <div>Excl</div> </div>						
autor	email	telefono	asunto	descripcion	fecha_consulta	ver detalle
Adriana Laura Cabaya	labulauu6@gmail.com	02881 155 733331	Consulta	Buenas Tardes! Soy enfermera profesional y Tecnica Superior en Especialidades. Aun no cuento con la matricula de enfermeria. Actualmente no ejerzo la profesion, mi pregunta es si puedo inscribirme?	25 Marzo 2020 21:49	[Ver Detalle]
Adriana Noemi Tejerina	adrianatepi@gmail.com	388-4095116	Inscripción	No puedo acceder a inscribirme soy Técnica en Farmacia Hospitalaria me podria dar información gracias	25 Marzo 2020 21:49	[Ver Detalle]
Adriana Noemi Tejerina	adrianatepi@gmail.com	388-4095116	Inscripción	No puedo acceder a inscribirme soy Técnica en Farmacia Hospitalaria me podria dar información gracias	25 Marzo 2020 21:49	[Ver Detalle]
Aldo cabana	gmaiculy18@gmail.com	3885023991	Chofer	Buenas tardes sr gobernador quisiera dar una mano en lo que sea para poder subsistir en estos momentos tan dificiles. Soy chofer profesional categoria D2 espero pueda dar una mano. Espero una respuesta	24 Marzo 2020 21:49	[Ver Detalle]
Amante Carmen Elina	carmenelina2019@gmail.com	3818708754	Consulta	Quisiera formar parte del equipo, pero tengo el inconveniente de no tener el título en mano ya q x razones de publico conocimiento no puedo viajar a la prov de Tucuman a buscarlo. Podria tener otra opcion para inscribirme ?	25 Marzo 2020 21:49	[Ver Detalle]
Ana Daniela Calderon	ac2721821@gmail.com	3887471714	Soy técnico en farmacia	Ayudar en lo que necesite el coe	25 Marzo 2020 21:49	[Ver Detalle]
Ana Laura Velasquez	analauravelasquez@gmail.com	388-4468852	Consulta acerca de una solución	Buenas tardes soy Ana Laura Velasquez Cuit: 27-38973442-5, estoy interesada en trabajar en la problemática emergente como profesional de la salud, soy licenciada y profesora en educación para la salud. Actualmente cuento con el título de la licenciatura de la carrera pero no con el título de la licenciatura y profesorado (ya que se encuentran en trámite) con la matricula. Quiero saber si serian tan amables de informarme sobre alguna solución ya que en la página de inscripción no puedo subir mis datos por que no pude sube escanalar la cedula, matricula y cuit ya	25 Marzo 2020 21:49	[Ver Detalle]

Ademas, dentro de cualquier template podemos agregar los apartados:

**{% block custom\_head %}**

**{% endblock %}**

**{% block custom\_script %}**

**{% endblock %}**

Para agregar css y js extras que son requeridos por algún template específico.