

Laboratorio DIA 3: CREAR PROYECTO CON MODELO Y BD

Pasos:

1. Abrir CMD
2. Instalar driver de conexión para mariaDB o Mysql si no lo has hecho.
pip install PyMySQL
3. Crear la base de datos **rrhh** en MariaDB usando el cliente Heidy. Se adjunta la documentación en el portal de aprendizaje. Una vez creada, dejar la conexión Abierta.
4. Ubicarse dentro del espacio de trabajo **workspace-django-project2**
5. Crear el proyecto **AdmEmpleadosDjango** ejecutando la siguiente instrucción:

django-admin startproject AdmEmpleadosDjango

```
D:\Workspace-django-project2>django-admin startproject AdmEmpleadosDjango
D:\Workspace-django-project2>
```

6. Entrar al proyecto **AdmEmpleadosDjango** con cd

```
D:\Workspace-django-project2>cd AdmEmpleadosDjango
D:\Workspace-django-project2\AdmEmpleadosDjango>
```

7. Crear la aplicación **moduloEmpleados** ejecutando la siguiente instrucción:

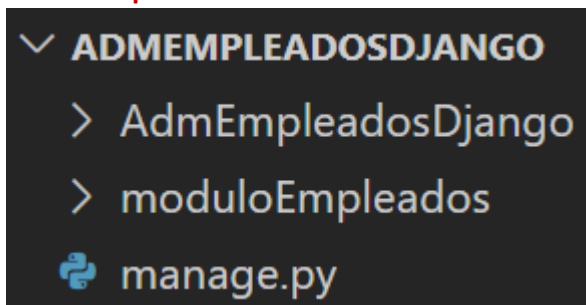
python manage.py startapp moduloEmpleados

```
D:\Workspace-django-project2\AdmEmpleadosDjango>python manage.py startapp moduloEmpleados
D:\Workspace-django-project2\AdmEmpleadosDjango>
```

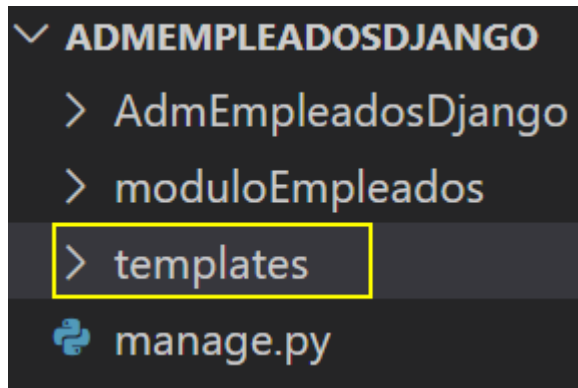
8. Abrir visual code

```
D:\Workspace-django-project2\AdmEmpleadosDjango>code .
D:\Workspace-django-project2\AdmEmpleadosDjango>
```

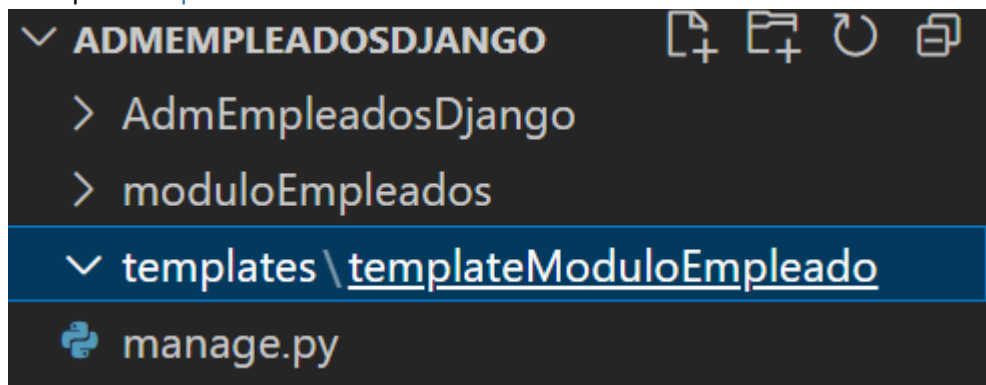
En la siguiente imagen, se puede visualizar el proyecto **AdmEmpleadosDjango** y la aplicación **moduloEmpleados**



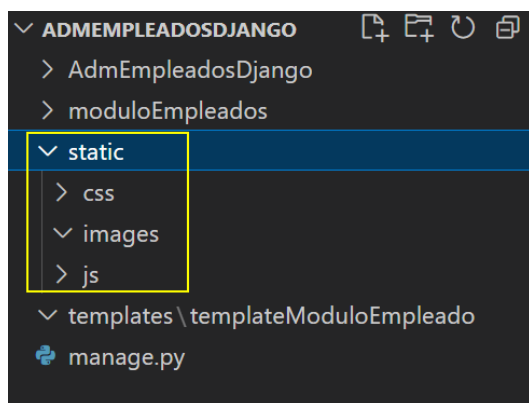
9. En el proyecto **AdmEmpleadosDjango**, crear la carpeta **templates** que almacenará los templates de las aplicaciones que posee el proyecto. Revisar la creación en la imagen siguiente.



10. Crear la carpeta **templateModuloEmpleado** almacenará los templates de la aplicación dentro de la carpeta **templates**.



11. Crear la carpeta **static** y las subcarpetas **images**, **js**, **css** para almacenar contenido estático crear la siguiente estructura de carpetas dentro del proyecto



Nota: la carpeta static está a la misma altura de la carpeta templates. Se usan los mismos pasos de creación de carpetas del punto anterior.

12. Modificar el archivo de configuración `settings.py`

- a) Incluir la aplicación `moduloEmpleados` al Proyecto `AdmEmpleadosDjango`

```
INSTALLED_APPS = [  
    'django.contrib.admin',  
    'django.contrib.auth',  
    'django.contrib.contenttypes',  
    'django.contrib.sessions',  
    'django.contrib.messages',  
    'django.contrib.staticfiles',  
    'moduloEmpleados'  
]
```

- b) Configurar la variable `TEMPLATE_DIR`

- i. Incluir import `os`

```
12 import os  
13 from pathlib import Path
```

- ii. Agregar la variable `TEMPLATE_DIR`

```
15 # Build paths inside the project like this: BASE_DIR / 'subdir'.  
16 BASE_DIR = Path(__file__).resolve().parent.parent  
17 TEMPLATE_DIR = os.path.join(BASE_DIR, 'templates')
```

- iii. Especificar la variable `TEMPLATE_DIR` en el código que se indica en el recuadro amarillo

```
55 TEMPLATES = [  
56     {  
57         'BACKEND': 'django.template.backends.django.DjangoTemplates',  
58         'DIRS': [],  
59         'APP_DIRS': True,  
60         'OPTIONS': {  
61             'context_processors': [  
62                 'django.template.context_processors.debug',  
63                 'django.template.context_processors.request',  
64                 'django.contrib.auth.context_processors.auth',  
65                 'django.contrib.messages.context_processors.messages',  
66             ],  
67         },  
68     },  
69 ]
```

El resultado es:

```
55 TEMPLATES = [  
56     {  
57         'BACKEND': 'django.template.backends.django.DjangoTemplates',  
58         'DIRS': [TEMPLATE_DIR],  
59         'APP_DIRS': True,  
60         'OPTIONS': {  
61             'context_processors': [  
62                 'django.template.context_processors.debug',  
63                 'django.template.context_processors.request',  
64                 'django.contrib.auth.context_processors.auth',  
65                 'django.contrib.messages.context_processors.messages',  
66             ],  
67         },  
68     },  
69 ]
```

- c) Revisar y configurar contenido estático
- Revisar la existencia de la variable **STATIC_URL** con la asignación de la carpeta **static** del Proyecto

```
116 # Static files (CSS, JavaScript, Images)
117 # https://docs.djangoproject.com/en/4.1/howto/static-files/
118
119 STATIC_URL = 'static/'
```

- Especificar la variable **STATICFILES_DIRS** en el código como se indica en el recuadro amarillo.

```
116 # Static files (CSS, JavaScript, Images)
117 # https://docs.djangoproject.com/en/4.1/howto/static-files/
118
119 STATIC_URL = 'static/'
120 STATICFILES_DIRS=[os.path.join(BASE_DIR, 'static')]
```

- d) Configurar la consola de administración a español

```
LANGUAGE_CODE = 'es-Es' # 'en-us' en ingles
```

En **urls.py** existe una url por defecto que es **"/admin"** al cargarla se visualiza una interfaz de acceso que posteriormente conoceremos.



- e) Crear y configurar la Base de Datos
- Configurar la base de datos en el archivo **settings.py**. En otras palabras, **reemplazar** el siguiente código que se ilustra en la siguiente imagen

```
77 DATABASES = {
78     'default': {
79         'ENGINE': 'django.db.backends.sqlite3',
80         'NAME': BASE_DIR / 'db.sqlite3',
81     }
82 }
```

Se reemplaza por:

```

77 import pymysql
78 pymysql.install_as_MySQLdb()
79
80 DATABASES = {
81     'default': {
82         'ENGINE': 'django.db.backends.mysql',
83         'NAME': 'rrhh',
84         'USER': 'root',
85         'PASSWORD': 'root1502'
86     }
87 }
88

```

Base de datos creada

Usuario BD

Modificarla a la Password BD MariaDB. En laboratorio es admin

13. Iniciar el servidor

```

PS D:\Workspace-django-project2\AdmEmpleadosDjango> python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

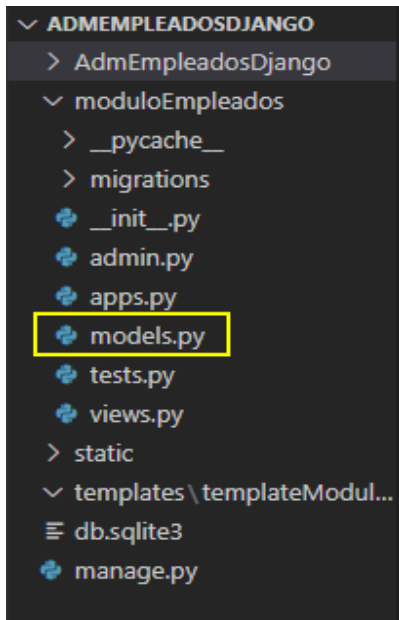
System check identified no issues (0 silenced).

You have 18 unapplied migration(s). Your project may not work properly until you apply
the migrations for app(s): admin, auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
September 28, 2022 - 23:05:57
Django version 4.1, using settings 'AdmEmpleadosDjango.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.

```



14. Crear el modelo. Editar el archivo `models.py` de la aplicación `moduloEmpleados` e ingresar el código que se ilustra en la imagen.



```
moduloEmpleados > models.py > ...
1  from django.db import models
2
3
4  # Create your models here.
5  class Employee(models.Model):
6      nombre=models.CharField(max_length=50)
7      email=models.CharField(max_length=50)
8      fono=models.CharField(max_length=15)
9
```

`models.CharField` permite definir un campo de tipo texto. Permite usar el atributo `max_length` para definir un largo permitido de caracteres.

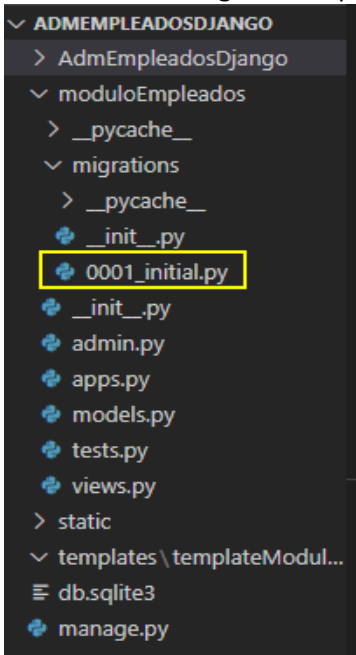
Nota: Ahora con el modelo se pueden realizar las migraciones.

15. Realizar las migraciones ejecutando la siguiente instrucción:

`python manage.py makemigrations`

```
PS D:\Workspace-django-project2\AdmEmpleadosDjango> python manage.py makemigrations
Migrations for 'moduloEmpleados':
  moduloEmpleados\migrations\0001_initial.py
    - Create model Employee
PS D:\Workspace-django-project2\AdmEmpleadosDjango>
```

16. Revisar el archivo generado por Django `moduloEmpleados\migrations\001_initial.py`



Este archivo le dice al python lo que debe de crear. Dice como construir la tabla en una base de datos. Al revisar el archivo se agrega el campo ID que no fue indicado en el modelo. Indica que es clave primaria.

Cada clase del modelo lo lleva a tabla. Una vez creada todas las clases va a generar todos los archivos de migraciones automáticamente.

0001_initial.py

```
operations = [
    migrations.CreateModel(
        name='Employee',
        fields=[
            ('id', models.BigAutoField(auto_created=True, primary_key=True, serialize=False, verbose_name='ID')),
            ('nombre', models.CharField(max_length=50)),
            ('email', models.CharField(max_length=50)),
            ('fono', models.CharField(max_length=15)),
        ],
    ),
]
```

17. Ejecutar las migraciones ejecutando la siguiente instrucción:

`python manage.py migrate`

```
PS D:\Workspace-django-project2\AdmEmpleadosDjango> python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, moduloEmpleados, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying auth.0010_alter_group_name_max_length... OK
  Applying auth.0011_update_proxy_permissions... OK
  Applying auth.0012_alter_user_first_name_max_length... OK
  Applying moduloEmpleados.0001_initial... OK
  Applying sessions.0001_initial... OK
PS D:\Workspace-django-project2\AdmEmpleadosDjango>
```

Se crean varias tablas en la base de datos. Las tablas son tablas de configuración y las asociadas a las clases del modelo.

18. Revisar las tablas creadas en la base de datos:

Para que la base de datos muestre la actualización debes hacer click sobre ella y presionar refrescar o recargar.

rrhh	368,0 KiB
auth_group	32,0 KiB
auth_group_permissions	48,0 KiB
auth_permission	32,0 KiB
auth_user	32,0 KiB
auth_user_groups	48,0 KiB
auth_user_user_permissions	48,0 KiB
django_admin_log	48,0 KiB
django_content_type	32,0 KiB
django_migrations	16,0 KiB
django_session	16,0 KiB
moduloempleados_employee	16,0 KiB

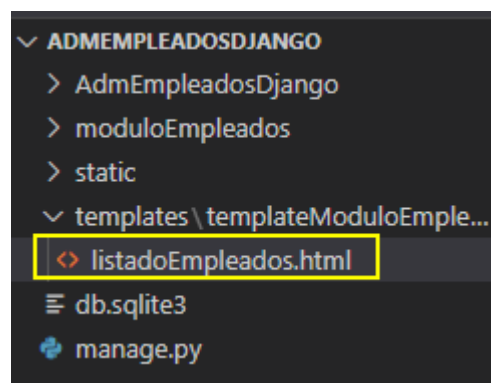
La tabla **moduloempleados_employee** corresponde a la clase creada en el modelo.

Tabla se ilustra a continuación

Columnas: ➕ Agregar ➖ Borrar ⬆ Subir ⬇ Bajar									
#	Nombre	Tipo de datos	Longitud/Conjunto	Sin signo	Permitir NULL	Relle...	Predeterminado	Comentario	
1	id	BIGINT	20	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	AUTO_INCREMENT		
2	nombre	VARCHAR	50	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Sin valor predeter...		
3	email	VARCHAR	50	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Sin valor predeter...		
4	fono	VARCHAR	50	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Sin valor predeter...		

rrhh.moduloempleado_employee: 0 filas en total (aproximadamente)				
🔑 id	nombre	email	fono	

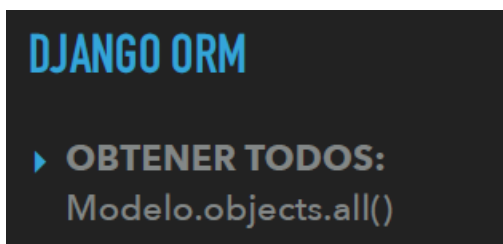
19. Crear el templates del Proyecto, específicamente en la carpeta **templates/templateModuloEmpleado** crear el archivo **listadoEmpleados.html**



20. Crear la vista para el template **listadoEmpleados.html**. Editar el archivo **views.py** de la aplicación **moduloEmpleados** y agregar la función **getListadoEmpleados** como se indica en la imagen.

```
moduloEmpleados > views.py > ...
1  from django.shortcuts import render
2
3  from moduloEmpleados.models import Employee
4
5  # Create your views here.
6  def getListadoEmpleados(request):
7      empleados=Employee.objects.all()
8      data={'empleados':empleados}
9      return render(request,'templateModuloEmpleado/listadoEmpleados.html',data)
10
```

Una de las características más poderosas de **Django** es su Mapeador Relacional de Objetos (**ORM**), que le permite interactuar con su base de datos, como lo haría con instrucciones SQL (Structured Query Language). La siguiente funcionalidad permite traer todos los registros de la tabla asociada al modelo.



Nota: Para obtener todos los empleados de la base de datos se hace uso de las funciones de Mapeo relacional de objetos que provee Django.

21. Modificar el archivo **urls.py** del proyecto e incluir la url para la función **getListadoEmpleados**

```
from django.contrib import admin
from django.urls import path

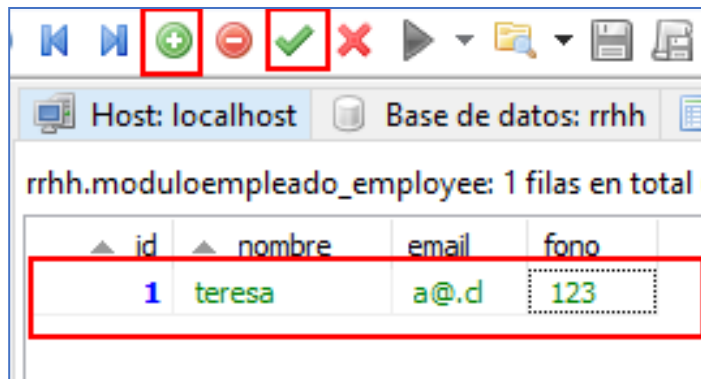
from moduloEmpleados.views import getListadoEmpleados

urlpatterns = [
    path('admin/', admin.site.urls),
    path('empleados/', getListadoEmpleados)
]
```

22. Implementar el archivo **listadoEmpleados.html** considerando cuando no hay empleados a mostrar y cuando tenga al menos un registro de empleados como se ilustra a continuación.



Para visualizar un registro se aconseja crear el registro manualmente en la base de datos como sigue:




1. Presiona el signo + de la ventana
2. Insertar un registro manualmente
3. Confirma los cambios con el victo bueno



Para usar iconos de Bootstrap ir a la siguiente página <https://icons.getbootstrap.com/#icons>
CDN
Agregar a la cabecera del archivo...

```
<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.9.1/font/bootstrap-icons.css">
```



Download


Download the SVG to use or edit.

Download SVG

Icon font

Using the web font? Copy, paste, and go.

```
<i class="bi bi-trash-fill"></i>
```



Download

Download the SVG to use or edit.

Download SVG

Icon font

Using the web font? Copy, paste, and go.

```
<i class="bi bi-pencil-fill"></i>
```

La implementación considera la cabecera como:

```

templates > templateModuloEmpleado > listadoEmpleados.html
1  <!DOCTYPE html>
2  {% load static %}
3  <html>
4      <head>
5          <meta charset="utf-8">
6          <meta name="viewport" content="width=device-width, initial-scale=1.0">
7          <meta http-equiv="X-UA-Compatible" content="ie=edge">
8          <meta name="description" content="">
9          <title>Listado Empleados</title>
10         <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-18mE4kBq78iYhF1dvKuhfTAU
11         <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-ka75k0G1n4gmtz2M1QnikT1wXgYs0g+OMhuF
12
13         <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.9.1/font/bootstrap-icons.css">
14     </head>

```

```

<body class="container mt-5">
  <div class="alert alert-info display-1 text-center"> LISTADO DE EMPLEADOS </div>
  {% if empleados %}
  <table class="table">
    <thead>
      <tr>
        <th>ID</th>
        <th>Nombre</th>
        <th>Email</th>
        <th>Fono</th>
        <th>--</th>
      </tr>
    </thead>
    <tbody>
      {% for emp in empleados %}
      <tr>
        <td>{{emp.id}}</td>
        <td>{{emp.nombre}}</td>
        <td>{{emp.email}}</td>
        <td>{{emp.fono}}</td>
        <td>
          <a class="btn btn-success btn-small" href="#">
            <i class="bi bi-pencil-fill"></i>
          </a>
          <a class="btn btn-danger btn-small" href="#">
            <i class="bi bi-trash-fill"></i>
          </a>
        </td>
      </tr>
      {% endfor %}
    </tbody>
  </table>

```

```

    {% else %}
    <div class="alert alert-warning text-center">
      No se encuentran empleados en el sistema
    </div>
    {% endif %}
    <a href=".." class="btn btn-danger">VOLVER</a>
    <a href="agregar" class="btn btn-success float-end">AGREGAR EMPLEADO</a>
  </body>
</html>

```