|  |  |  |  |
| --- | --- | --- | --- |
| Paso | Descripción | Actividad | Ruta |
|  | Se crea el entorno virtual | Python –m venv empleadosenv | cmd |
|  | Se activa el entorno virtual | Cd empleadosenv/Scrips/activate | cmd |
|  | Se instala django | Pip install django | cmd |
|  | Se verifica si está instalado django | Pip freeze --local | cmd |
|  | Se crea el proyecto | Django-admin startproject empleadosproj | cmd |
|  | Se crea la carpeta settings |  | Empleadosproj |
|  | Se crea el archivo base.py | """  """  import os  # Build paths inside the project like this: os.path.join(BASE\_DIR, ...)  from unipath import Path  BASE\_DIR = Path(\_\_file\_\_).ancestor(3)  # Quick-start development settings - unsuitable for production  # See https://docs.djangoproject.com/en/3.0/howto/deployment/checklist/  # SECURITY WARNING: keep the secret key used in production secret!  SECRET\_KEY = '8fg^0qnij&5o!q2^pqr)2%i$g+$k\_82-44mb\_aiji85t#%bj#g'  # Application definition  INSTALLED\_APPS = [  'django.contrib.admin',  'django.contrib.auth',  'django.contrib.contenttypes',  'django.contrib.sessions',  'django.contrib.messages',  'django.contrib.staticfiles',  # \*\*\*\*\*\*\*\*\*\*\*\*\*\* Aplicaciones creadas por mi \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  # Se instala la aplicación departamento  'applications.departamento',  # Se instala la aplicación personas  'applications.personas',  # Se instala la aplicación home  'applications.home',  # \*\*\*\*\*\*\*\*\*\*\*\*\*\* Aplicaciones creadas por mi \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  # Se instala la aplicación ckeditor  'ckeditor',  ]  MIDDLEWARE = [  'django.middleware.security.SecurityMiddleware',  'django.contrib.sessions.middleware.SessionMiddleware',  'django.middleware.common.CommonMiddleware',  'django.middleware.csrf.CsrfViewMiddleware',  'django.contrib.auth.middleware.AuthenticationMiddleware',  'django.contrib.messages.middleware.MessageMiddleware',  'django.middleware.clickjacking.XFrameOptionsMiddleware',  ]  ROOT\_URLCONF = 'empleadosproj.urls'  TEMPLATES = [  {  'BACKEND': 'django.template.backends.django.DjangoTemplates',  'DIRS': [BASE\_DIR.child('templates')],  'APP\_DIRS': True,  'OPTIONS': {  'context\_processors': [  'django.template.context\_processors.debug',  'django.template.context\_processors.request',  'django.contrib.auth.context\_processors.auth',  'django.contrib.messages.context\_processors.messages',  ],  },  },  ]  WSGI\_APPLICATION = 'empleadosproj.wsgi.application'  # Password validation  # https://docs.djangoproject.com/en/3.0/ref/settings/#auth-password-validators  AUTH\_PASSWORD\_VALIDATORS = [  {  'NAME': 'django.contrib.auth.password\_validation.UserAttributeSimilarityValidator',  },  {  'NAME': 'django.contrib.auth.password\_validation.MinimumLengthValidator',  },  {  'NAME': 'django.contrib.auth.password\_validation.CommonPasswordValidator',  },  {  'NAME': 'django.contrib.auth.password\_validation.NumericPasswordValidator',  },  ]  # Internationalization  # https://docs.djangoproject.com/en/3.0/topics/i18n/  LANGUAGE\_CODE = 'en-us'  TIME\_ZONE = 'UTC'  USE\_I18N = True  USE\_L10N = True  USE\_TZ = True | Empleadosproj/settings |
|  | Se crea el archivo local.py | from .base import \*  # SECURITY WARNING: don't run with debug turned on in production!  DEBUG = True  ALLOWED\_HOSTS = []  # Database  # https://docs.djangoproject.com/en/3.0/ref/settings/#databases  DATABASES = {  'default': {  'ENGINE': 'django.db.backends.postgresql\_psycopg2',  'NAME': 'db',  'USER': 'andres',  'PASSWORD': 'DC4ndr3s',  'HOST': 'localhost',  'PORT': '5432',  }  }  # Password validation  # https://docs.djangoproject.com/en/3.0/ref/settings/#auth-password-validators  # Static files (CSS, JavaScript, Images)  # https://docs.djangoproject.com/en/3.0/howto/static-files/  STATIC\_URL = '/static/'  STATICFILES\_DIRS = [BASE\_DIR.child('static')] | Empleadosproj/settings |
|  | Se crea el archivo prod.py |  | Empleadosproj/settings |
|  | Se crea el archivo \_\_init\_\_.py |  | Empleadosproj/settings |
|  | Se crea la carpeta applications y en ella un archivo \_\_init\_\_.py | INSTALLED\_APPS = [  'django.contrib.admin',  'django.contrib.auth',  'django.contrib.contenttypes',  'django.contrib.sessions',  'django.contrib.messages',  'django.contrib.staticfiles',  # \*\*\*\*\*\*\*\*\*\*\*\*\*\* Aplicaciones creadas por mi \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  # Se instala la aplicación departamento  'applications.departamento',  # Se instala la aplicación personas  'applications.personas',  # Se instala la aplicación home  'applications.home',  # \*\*\*\*\*\*\*\*\*\*\*\*\*\* Aplicaciones creadas por mi \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  # Se instala la aplicación ckeditor  'ckeditor',  ] | Empleadosproj/settings |
|  | Se crea la app home | Django-admin startapp home | Empleadosproj/settings |
|  | Se edita el archivo manage | Se cambia el os.environ.setdefault por empleadosproj.settings.local | ./ |
|  | Se añanden las urls de las apps al archivo de urls principal importando include y re\_path | from django.contrib import admin  from django.urls import path, re\_path, include    urlpatterns = [  path('admin/', admin.site.urls),  # Se incluyen las urls de la app departamento  path('', include('applications.departamento.urls')),  # Se incluyen las urls de la app persona  path('', include('applications.personas.urls')),  # Se incluyen las urls de la app home  path('', include('applications.home.urls')),  ] | Empleadosproj/urls.py |
|  | Se crea la app departamento | Django-admin startapp departamento | cmd |
|  | Se crea la app personas | Django-admin startapp personas | cmd |
|  | Se crea la carpeta templates |  | ./ |
|  | Se instala unipath | Pip install unipath | cmd |
|  | Se edita el BASE\_DIR | from unipath import Path  BASE\_DIR = Path(\_\_file\_\_).ancestor(3)  TEMPLATES = [  {  'BACKEND': 'django.template.backends.django.DjangoTemplates',  'DIRS': [BASE\_DIR.child('templates')],  'APP\_DIRS': True,  'OPTIONS': {  'context\_processors': [  'django.template.context\_processors.debug',  'django.template.context\_processors.request',  'django.contrib.auth.context\_processors.auth',  'django.contrib.messages.context\_processors.messages',  ],  },  },  ] | Empleadosproj/settings |
|  | Se instalan las apps en el archivo base.py | INSTALLED\_APPS = [  'django.contrib.admin',  'django.contrib.auth',  'django.contrib.contenttypes',  'django.contrib.sessions',  'django.contrib.messages',  'django.contrib.staticfiles',  # \*\*\*\*\*\*\*\*\*\*\*\*\*\* Aplicaciones creadas por mi \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  # Se instala la aplicación departamento  'applications.departamento',  # Se instala la aplicación personas  'applications.personas',  # Se instala la aplicación home  'applications.home',  # \*\*\*\*\*\*\*\*\*\*\*\*\*\* Aplicaciones creadas por mi \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  # Se instala la aplicación ckeditor  'ckeditor',  ] | Empleadosproj/settings/base.py |
| 1 | Se crea la clase Departamento | cf CharField | aplications/departamento/models.py |
| mb booleano |
| fk llave Foranea |
| fimg para imagen |
| m2m muchos a muchos |
| 2 | Se hace un python manage.py makemigrations | python manage.py makemigrations | cmd |
| 3 | Se hace un python manage.py migrate | python manage.py migrate | cmd |
| 4 | Registro el modelo | from .models import Departamento | aplications/departamento/admin.py |
| admin.site.register(Departamento) |
| Con el editable=False no lo deja cambiar |
| 5 | Se crea la clase Empleado | Se debe crear la lista primero JOB\_CHOICES y luego se usa choices se usa para crear opciones de selección | aplications/personas/models.py |
|
| 6 | Se importa la clase Departamento para hacer la relación | from applications.departamento.models import Departamento | aplications/personas/models.py |
| 7 | Registro el modelo | from .models import Empleado | aplications/personas/admin.py |
| admin.site.register(Empleado) |
|  | Se instala PostgreSLQ | 10.13 | https://www.postgresql.org/download/windows/ |
| 8 | Se abre la shell de PSQL y de deja por default lo que pide (se da enter y se deja vacio) | SQL Shell (psql) | Inicio/PostgeSQL 10/SQL Shell |
|  | Se crea la DB | CREATE DATABASE db; | Consola PSQL |
|  | Se crea un usuario | CREATE USER andres; | Consola PSQL |
|  | Se accede a la DB | \c db; | Consola PSQL |
|  | Se da permisos al usuario | ALTER ROLE andres WITH PASSWORD 'DC4ndr3s'; | Consola PSQL |
|  | Instalo psycopg2  psycopg2 Transforma de Python a PSQL | pip install psycopg2 | CMD |
|  | Se indica la configuración de BD con la que se va a trabajar | 'ENGINE': 'django.db.backends.postgresql\_psycopg2', | empleadosproj/settings/local.py en DATABASES |
| 'NAME': 'db', |
| 'USER': 'andres', |
| 'PASSWORD': 'DC4ndr3s', |
| 'HOST': 'localhost', |
| 'PORT': '5432', |
|  |
|  | Se crean las migraciones | python manage.py makemigrations | CMD |
|  | Se crean las migraciones | python manage.py migrate | CMD |
|  | Se crea un super usuario debido a que la DB es nueva | python manage.py createsuperuser | CMD |
|
|  | Con el class Meta se dice como quiere que algo se muestre en el espacio del administrador | class Meta: | aplications/departamento/models.py |
| verbose\_name = 'Mi Departamento' |
| verbose\_name\_plural = 'Mis Departamentos' |
| orderin = ['name'] \*\*\*Ordena |
| unique\_together = () \*\*\*Para que algo sea único |
|  | Con el class Meta se dice como quiere que algo se muestre en el espacio del administrador | class Meta: | aplications/personas/models.py |
| verbose\_name = 'Persona' |
| verbose\_name\_plural = 'Personas' |
| orderin = ['last\_name'] |
|  |
|  | Se agrega una nueva clase Habilidades | class Habilidades(models.Model): | aplications/personas/models.py |
|  | Se crean las migraciones | python manage.py makemigrations | CMD |
|  | Se crean las migraciones | python manage.py migrate | CMD |
|  | Se registra la clase habilidades | admin.site.register(Habilidades) | aplications/personas/admin.py |
|  | Se crea un decorador para que muestre en el admin las columnas que se le indican y se pone como segundo parametro | class EmpleadoAdmin(admin.ModelAdmin): | aplications/personas/admin.py |
| list\_display=( |
| 'first\_name', |
| 'last\_name', |
| 'departamento', |
| 'job', |
| ) |
| admin.site.register(Empleado, EmpleadoAdmin) |
|  | Se crea un buscador | search\_fields = ('firts\_name',) | aplications/personas/admin.py |
| list\_filter = ('job',) |
| filter\_horizontal = ('habilidades',) |
|  | Se busca que muestre el nombre completo | 'full\_name', | aplications/personas/admin.py |
| def full\_name(self, obj): |
| return obj.firts\_name+' '+obj.last\_name (26) |
|  |
|  | Se instala un editor | pip install django-ckeditor | CMD |
|  | Se añade ckeditor | # Se instala la aplicación ckeditor  'ckeditor', | Empleado/settings/base.py  INSTALLED\_APPS |
|  | Se añade un campo hoja de vida al Empleado | from ckeditor.fields import RichTextField | aplications/personas/models.py |
|  | Se crean las migraciones | python manage.py makemigrations | CMD |
|  | Se crean las migraciones | python manage.py migrate | CMD |
|  | Se crea una clase para Listar a todos los empleados | from .models import Empleado  class ListAllEmpleados(ListView):  template\_name = 'personas/list\_all.html'  model = Empleado | aplications/personas/view.py |
|  | Se añaden las urls de la vista | path('listar-todo/',views.ListAllEmpleados.as\_view()), | aplications/personas/urls.py |
|  | Se crea el template list\_all.html | <h1>Lista de todos los empleados</h1>  <ul>  {% for lista in object\_list %}  <li>{{lista}}</li>  {% endfor %}  </ul> | Templates/personas/list\_all.html |
|  | Se crea una clase para Listar a todos los empleados de un área.  Para que devuelva algo en particular de usa queryset | class ListAllEmpleadosArea(ListView):  template\_name = 'personas/list\_all\_area.html'  queryset = Empleado.objects.filter(  departamento\_\_name = 'contabilidad'  ) | aplications/personas/view.py |
|  | Se amplía el filtro | list\_filter = ('job','habilidades','departamento') | aplications/personas/admin.py |
|  | Se añaden las urls de la vista | path('listar-todo-area/',views.ListAllEmpleadosArea.as\_view()), | aplications/personas/urls.py |
|  | Se crea el template list\_all\_area.html | <h1>Lista de todos los empleados</h1>  <ul>  {% for lista in object\_list %}  <li>{{lista}}</li>  {% endfor %}  </ul> | Templates/personas/list\_all\_area.html |
|  | Se crea una segunda clase para Listar a todos los empleados de un área con un parámetro en la URL.  Para que devuelva algo en particular de usa queryset.  Con area.self.kwargs['shorname'] se recoje lo de la url | class ListAllEmpleadosArea2(ListView):  template\_name = 'personas/list\_all\_area2.html'    def get\_queryset(self):  area = self.kwargs['shorname']  queryset = Empleado.objects.filter(  departamento\_\_shor\_name = area  )  return queryset | aplications/personas/view.py |
|  | Se añaden las urls de la vista con <name> que es lo que se va a agregar en la url | path('listar-todo-area2/<shorname>',views.ListAllEmpleadosArea2.as\_view()), | aplications/personas/urls.py |
|  | Se crea el template list\_all\_area.html | <h1>Lista de todos los empleados</h1>  <ul>  {% for lista in object\_list %}  <li>{{lista}}</li>  {% endfor %}  </ul> | Templates/personas/list\_all\_area2.html |
|  | Se crea una clase para Listar a todos los empleados por una palabra clave con una caja de texto. | class ListAllEmpleadosPalabra(ListView):  template\_name = 'personas/list\_all\_palabra.html'  context\_object\_name = 'empleados'  def get\_queryset(self):  palabra = self.request.GET.get("palabraClave",'')  queryset = Empleado.objects.filter(  firts\_name = palabra  )  return queryset | aplications/personas/view.py |
|  | Se añaden las urls de la vista | path('buscar-empleado',views.ListAllEmpleadosPalabra.as\_view()), | aplications/personas/urls.py |
|  | Se crea el template list\_all\_palabra.html  Se usa para dar seguridad {% csrf\_token %} | <h1>Lista de empleados por palabra</h1>  <form action="" method="get">{% csrf\_token %}  <input type="text" id="palabraClave" name="palabraClave" placeholder="Ingrese nombre aquí">  <button type="submit">Buscar</button>  </form>  <h2>Lista de resultados</h2>  <ul>  {% for lista in object\_list %}  <li>{{lista}}</li>  {% endfor %}  </ul> | Templates/personas/list\_all\_palabra.html |
|  | Se crea una clase para Listar a todos los empleados paginados.  Funciona 127.0.0.1:8000/'listar-paginado/?page=# | class ListAllEmpleadosPaginados(ListView):  template\_name = 'personas/list\_paginados.html'  paginate\_by = 2  model = Empleado | aplications/personas/view.py |
|  | Se añaden las urls de la vista | path('listar-paginado/',views.ListAllEmpleadosPaginados.as\_view()), | aplications/personas/urls.py |
|  | Se crea el template list\_paginados.html | <h1>Lista de todos los empleados</h1>  <ul>  {% for lista in object\_list %}  <li>{{lista}}</li>  {% endfor %}  </ul> | Templates/personas/list\_paginados.html |
|  | Se crea una clase para listar las habilidades de un empleado  Lo primero es seleccionar un empleado | class ListAllHabilidades(ListView):  template\_name = 'personas/list\_habilidades.html'  context\_object\_name = 'habilidades'  def get\_queryset(self):  identificador = self.request.GET.get("identificado",'')  empleado = Empleado.objects.get(id=identificador)  print(empleado.firts\_name)  queryset = Empleado.objects.filter(  id = identificador  )  return empleado.habilidades.all | aplications/personas/view.py |
|  | Se añaden las urls de la vista | path('habilidades',views.ListAllHabilidades.as\_view()), | aplications/personas/urls.py |
|  | Se crea el template list\_habilidades.html | <h1>Lista de Habilidades</h1>  <form action="" method="get">{% csrf\_token %}  <input type="text" id="identificado" name="identificado" placeholder="Ingrese id aquí">  <button type="submit">Buscar</button>  </form>  <ul>  {% for lista in habilidades %}  <li>{{lista}}</li>  {% endfor %}  </ul> | Templates/personas/ list\_habilidades.html |
|  | Se importa el DetailView | from django.views.generic import ListView, DetailView | aplications/personas/view.py |
|  | Muestra los detalles de un objeto, artículo u otro | class EmpleadoDetail(DetailView):  model = Empleado  template\_name='personas/detail\_empleado.html' | aplications/personas/view.py |
|  | Se añaden las urls de la vista  Para que el DetailView funcione su parámetro debe ser un pk | path('detalles/<pk>',views.EmpleadoDetail.as\_view()), | aplications/personas/urls.py |
|  | Se crea el template list\_detalles.html | <h1>Detalle empleado</h1>  {{empleado}}  <h3>Todos los datos</h3>  <p>{{empleado.firts\_name}} </p>  <p>{{empleado.last\_name}} </p>  <p>{{empleado.job}}</p>  <p>{{empleado.departamento}}</p>  <p>{{empleado.habilidades}}</p> | Templates/personas/ list\_detalles.html |
|  | Se importa el CreateView y el reverse\_lazy para enrutar | from django.views.generic import ListView, DetailView, CreateView, TemplateView  from django.urls import reverse\_lazy | aplications/personas/view.py |
|  | Se crea la clase empleado  fields = ('\_\_all\_\_') muestra todos los campos  success\_url = '/crear' Crea la ruta después del guardado | class EmpleadoCreateView(CreateView):  template\_name = "personas/crear\_empleado.html"  model = Empleado  fields = ('\_\_all\_\_')  success\_url = reverse\_lazy('persona\_app:correcto') | aplications/personas/view.py |
|  | Se añaden las urls de la vista | path('crear/',views.EmpleadoCreateView.as\_view(), name='add'), | aplications/personas/urls.py |
|  | Se crea el template crear\_empleado.html  {{form.as\_p}} muestra el formulario con todos los datos solicitados | <h1>Crear empleado</h1>  <form method="post">  {% csrf\_token %} {{form.as\_p}}  <button type="submit">Guardar</button>  </form> | Templates/personas/ crear\_empleado.html |
|  | Se importa el TemplateView | from django.views.generic import ListView, DetailView, CreateView, TemplateView | aplications/personas/view.py |
|  | Se crea un namespace | app\_name = "persona\_app" | aplications/personas/urls.py |
|  | Se crea la clase para mostrar que se guardó | class SuccessView(TemplateView):  template\_name = "personas/success.html" | aplications/personas/view.py |
|  | Se añaden las urls de la vista | path('guardado/',views.SuccessView.as\_view(), name='correcto'), | aplications/personas/urls.py |
|  | Se crea un nuevo campo en el modelo persona | class Empleado(models.Model):  JOB\_CHOICES =(  ('0','Contador'),  ('1','Administrador'),  ('2','Desarrollador'),  ('3','Analista'),  )  firts\_name = models.CharField('Nombres',max\_length=60)  last\_name = models.CharField('Apellidos',max\_length=60)  full\_name = models.CharField('Apellidos',max\_length=200, blank=True, null=True)  job = models.CharField('Trabajo',max\_length=50, choices=JOB\_CHOICES)  departamento = models.ForeignKey(Departamento, on\_delete=models.CASCADE)  habilidades = models.ManyToManyField(Habilidades)  hoja\_vida = RichTextField()  # image = models.ImageField(\_(""), upload\_to=None, height\_field=None, width\_field=None, max\_length=None) | aplications/personas/modelos.py |
|  | Se actualiza la DB | python manage.py makemigrations  python manage.py migrate |  |
|  | Se requiere recuperar el nombre y apellido con lo que es necesario recuperar los datos guardados en la DB con form.save()  Se crea la variable para concatenarlos empleado.full\_name = empleado.firts\_name+' '+empleado.last\_name  y se guarda empleado.save() | class EmpleadoCreateView(CreateView):  template\_name = "personas/crear\_empleado.html"  model = Empleado  fields = ('\_\_all\_\_')  success\_url = reverse\_lazy('persona\_app:empleados\_admin')  def form\_valid(self, form):  empleado = form.save(commit=False)  empleado.full\_name = empleado.firts\_name+' '+empleado.last\_name  empleado.save()  return super(EmpleadoCreateView, self).form\_valid(form) | aplications/personas/view.py |
|  | Se importa la librería UpdateView | from django.views.generic import ListView, DetailView, CreateView, TemplateView, UpdateView | aplications/personas/view.py |
|  | Se crea la clase para actualizar EmpleadoUpdateView | class EmpleadoUpdateView(UpdateView):  model = Empleado  template\_name = "personas/update.html"  fields = ['firts\_name','last\_name','job','departamento','habilidades',]  success\_url = reverse\_lazy('persona\_app:correcto') | aplications/personas/view.py |
|  | Se activa la url con un parámetro para que identifique a que persona va a actualizar | path('actualizar/<pk>/',views.EmpleadoUpdateView.as\_view(), name='modificar\_empleado'), | aplications/personas/urls.py |
|  | Se crea el template update.html | <h1>Actualizar todos los empleados</h1>  <form method="post">{% csrf\_token %} {{form.as\_p}}  <button type="submit">Actualizar</button>  </form> | Templates/ personas/update.html |
|  | Se importa la librería DeleteView | from django.views.generic import ListView, DetailView, CreateView, TemplateView, UpdateView, DeleteView | aplications/personas/view.py |
|  | Se crea la clase para actualizar EmpleadoDeleteView | class EmpleadoDeleteView(DeleteView):  model = Empleado  template\_name = "personas/delete.html"  success\_url = reverse\_lazy('persona\_app:correcto') | aplications/personas/view.py |
|  | Se activa la url con un parámetro para que identifique a que persona va a actualizar | path('borrar/<pk>/',views.EmpleadoDeleteView.as\_view(), name='modificar\_empleado'), | aplications/personas/urls.py |
|  | Se crea el template delete.html | <h1>Eliminar</h1>  <form method="post">  {% csrf\_token %} {{form.as\_p}}  <p>Confirme que desea eliminar</p>  <button type="submit">Confirme</button>  </form> | Templates/ personas/delete.html |
|  | Se crea el archivo forms.py  Este se hace debido a que se quiere usar información de dos modelos diferentes | from django import forms  class NewDepartamentoForm(forms.Form):  nombre = forms.CharField( max\_length=50)  apellido = forms.CharField( max\_length=50)  departamento = forms.CharField( max\_length=50)  shorname = forms.CharField( max\_length=20) | aplications/departamento/forms.py |
|  | Se importan los modelos departamento y personas | from applications.personas.models import Empleado  from applications.departamento.models import Departamento | aplications/departamento/views.py |
|  | Se crea una vista que no está relacionada con ningún modelo para poder trabajar solo con los formularios.  Se llama el formulario creado form\_class = NewDepartamentoForm  Se crea el objeto departamento departament y se guarda en la BD departament.save()  Se crea un empleado y este se guarda de forma automática debido a que se uso Empleado.objects.create | class NewDepartamentoView(FormView):  template\_name = 'departamento/new\_departamento.html'  form\_class = NewDepartamentoForm  success\_url = '/'  def form\_valid(self, form):  departament = Departamento(  name = form.cleaned\_data['departamento'],  shor\_name = form.cleaned\_data['shorname'],  )  departament.save()  nombre = form.cleaned\_data['nombre']  apellido = form.cleaned\_data['apellido']  Empleado.objects.create(  firts\_name=nombre,  last\_name=apellido,  job='1',  departamento=departament,  )  return super(NewDepartamentoView, self).form\_valid(form) | aplications/departamento/views.py |
|  | Se crea la ruta | path('new-departamento/',views.NewDepartamentoView.as\_view(), name='nuevo\_departamento'), | aplications/departamento/urls.py |
|  | Se crea el template 'departamento/new\_departamento.html' | <h1>Crear Departamento</h1>  <form method="post">{% csrf\_token %}  <h3>Datos emplado</h3>  <p>{{form.nombre}}</p>  <p>{{form.apellido}}</p>  <h3>Datos Departamento</h3>  <p>{{form.departamento}}</p>  <p>{{form.shorname}}</p>  <button type="submit">Guardar</button>  </form> | aplications/departamento/new\_departamento.html |
|  | Se crea la carpeta static en la raíz del proyecto y en ella se crean  css  js  img | static | static |
|  | Se le indica a django donde estarán los archivos estaticos | STATICFILES\_DIRS = [BASE\_DIR.child('static')] | empleadosproj/settings/local.py |
|  | Se descarga foundations |  | https://download.get.foundation/sites/download/ |
|  | Se ponen sus archivos en static |  | empleadosproj/static |
|  | Se crea una carpeta llamada includes |  | Templates/includes |
|  | Se crea el archivo base.html | {% load static %}  <!DOCTYPE html>  <html lang="en">  <head>  <meta charset="UTF-8">  <meta name="viewport" content="width=device-width, initial-scale=1.0">  <link rel="stylesheet" href="{% static 'css/foundation.min.css' %}">    <title>  {% block title %}{% endblock title %}  </title>  </head>  <body>  {% include 'includes/header.html' %}  {% block content %}{% endblock content %}  <script src="{% static 'js/vendor/jquery.js' %}"></script>  <script src="{% static 'js/vendor/what-input.js' %}"></script>  <script src="{% static 'js/vendor/foundation.min.js' %}"></script>  <script>  $(document).foundation();  </script>  </body>  </html> | Templates/base.html |
|  | Se crea la clase InicioView | class InicioView(TemplateView):  template\_name = 'inicio.html' | aplications/personas/views.py |
|  | Se crea el template inicio.html |  |  |
|  | Se crea la url | path('',views.InicioView.as\_view()), | aplications/personas/urls.py |
|  | Se crea el archivo header | <div class="title-bar" data-responsive-toggle="responsive-menu" data-hide-for="medium">  <button class="menu-icon" type="button" data-toggle="responsive-menu"></button>  <div class="title-bar-title">Menu</div>  </div>    <div class="top-bar" id="responsive-menu">  <div class="top-bar-left">  <ul class="dropdown menu" data-dropdown-menu>  <li class="menu-text">Empleados</li>  <li class="has-submenu"><a href="#0">Listar</a></li>  <li><a href="#0">Departamentos</a></li>  <li><a href="#0">Administrar</a></li>  </ul>  </div>  <div class="top-bar-right">  <ul class="menu">  <li><input type="search" placeholder="Buscar empleado"></li>  <li><button type="button" class="button">Registrar Nuevo</button></li>  </ul>  </div>  </div> | Templates/includes/header.html |
|  | Se crea el archivo inicio.html | {% extends 'base.html' %}  {% load static %}  {% block title %}Inicio{% endblock title %}  {% block content %}  <!--Inicio de la imágen-->  <div class="orbit" role="region" aria-label="Favorite Space Pictures" data-orbit>  <div class="orbit-wrapper">  <div class="orbit-controls">  <button class="orbit-previous"><span class="show-for-sr">Previous Slide</span>&#9664;&#xFE0E;</button>  <button class="orbit-next"><span class="show-for-sr">Next Slide</span>&#9654;&#xFE0E;</button>  </div>  <ul class="orbit-container">  <li class="is-active orbit-slide">  <figure class="orbit-figure">  <img class="orbit-image" src="{% static 'img/1.jpg' %}" alt="Space">  <figcaption class="orbit-caption">Space, the final frontier.</figcaption>  </figure>  </li>  <li class="orbit-slide">  <figure class="orbit-figure">  <img class="orbit-image" src="{% static 'img/2.jpg' %}" alt="Space">  <figcaption class="orbit-caption">Lets Rocket!</figcaption>  </figure>  </li>  </ul>  </div>  </div>  <!--Fin de la imágen-->  <!--Inicio bloque de bienvenida-->  <div class="grid-x align-center">  <h1 class="cell large-8" style="font-size: 25px; text-align: center;">Bienvenido al Sistema de Registro de Empelados</h1>  <div class="cell large-7">  <div class="callout success">  <h5>Registra y controla empelados</h5>  <p>It has an easy to override visual style, and is appropriately subdued. It has an easy to override visual style, and is appropriately subdued.</p>  <a href="#">Ver empleados</a>  </div>  </div>  </div>  <!--Fin bloque de bienvenida-->  {% endblock content %} | Tamplates/inicio.html |
|  | Se modifica la vista ListAllEmpleados para que muestre los empleados acorde a una letra o lo que se escriba en el buscador usando \_\_icontains | class ListAllEmpleadosPaginados(ListView):  template\_name = 'personas/list\_paginados.html'  paginate\_by = 5  #model = Empleado  def get\_queryset(self):  palabra = self.request.GET.get("palabraClave",'')  queryset = Empleado.objects.filter(  firts\_name\_\_icontains=palabra  )  return queryset | Applications/personas/views.py |
|  | Se modifica el list\_paginados con el fin de que la insterfaz sea clara.  Para que el buscador funcione debe estar en un form.  Con el fin de ver en detalle a una persona se debe enviar el id de esta persona en el url | {% extends 'base.html' %}  {% block titulo %}Lista{% endblock titulo %}  {% block content %}  <div class="grid-container">  <div class="grid-x">  <h1 class="cell">Lista de empleados</h1>  <form class="cell grid-x grid-margin-x" method="GET">{%csrf\_token%}  <div class="cell large-7">  <input type="text" name="palabraClave" id="palabraClave" placeholder="Buscar empleado">  </div>  <div class="cell large-3">  <button type="submit" class="success button">Buscar</button>  </div>  </form>  <div class="cell">  <table>  <thead>  <tr>  <th width="200">ID</th>  <th>Nombres</th>  <th width="150">Apellidos</th>  <th width="150">Departamento</th>  <th width="150">Acción</th>  </tr>  </thead>  <tbody>  {% for lista in object\_list %}  <tr>  <td>{{lista.id}}</td>  <td>{{lista.firts\_name}}</td>  <td>{{lista.last\_name}}</td>  <td>{{lista.departamento}}</td>  <td><a class="button warning" href="{% url 'persona\_app:empleado\_detalle' lista.id %}">Ver</a></td>  </tr>  {% endfor %}  </tbody>  </table>  </div>  </div>  </div>  {% endblock content %} | Templates/personas/list\_paginados.html |
|  | Se les agrega name a las url para hacer más fácil su ubicación | from django.contrib import admin  from django.urls import path  from . import views  app\_name = "persona\_app"  urlpatterns = [  path('listar-todo/',views.ListAllEmpleados.as\_view(), name='empleados\_all'),  path('listar-paginado/',views.ListAllEmpleadosPaginados.as\_view(), name='empleados\_all\_pag'),  path('listar-todo-area/',views.ListAllEmpleadosArea.as\_view()),  path('listar-todo-area2/<shorname>',views.ListAllEmpleadosArea2.as\_view()),  path('buscar-empleado/',views.ListAllEmpleadosPalabra.as\_view()),  path('habilidades/',views.ListAllHabilidades.as\_view()),  path('detalles/<pk>',views.EmpleadoDetail.as\_view(), name='empleado\_detalle'),  path('crear/',views.EmpleadoCreateView.as\_view()),  path('',views.InicioView.as\_view(),name='inicio'),  path('guardado/',views.SuccessView.as\_view(), name='correcto'),  path('actualizar/<pk>/',views.EmpleadoUpdateView.as\_view(), name='modificar\_empleado'),  path('borrar/<pk>/',views.EmpleadoDeleteView.as\_view(), name='modificar\_empleado'),  ] | Applications/personas/urls.py |
|  | Se añade paginación a la lista de empleados | <div class="cell">  {% if is\_paginated %}  <nav aria-label="Pagination">  <ul class="pagination">  {% if page\_obj.has\_previous %}  <li class="pagination-previous"><a href="?page={{page\_obj.previous\_page\_number}}">Atras</a></li>  {% endif %}  {% for pagina in paginator.page\_range %}  {% ifequal pagina page\_obj.number %}  <li class="current"><span class="show-for-sr">You're on page</span> {{ pagina }}</li>  {% else %}  <li class=""><a href="?page={{pagina}}">{{ pagina }}</a></li>  {% endifequal %}  {% endfor %}  {% if page\_obj.has\_next %}  <li class="pagination-next"><a href="?page={{page\_obj.next\_page\_number}}">Siguiente</a></li>  {% endif %}  </ul>  </nav>  {% endif %}  </div> | Templates/personas/list\_paginados.html |
|  | Se crea la vista para listar departamentos | class ListDepartamentoView(ListView):  template\_name = 'departamento/list\_all.html'  model = Departamento  context\_object\_name = 'departamentos' | Applications/departamento/views.py |
|  | Se añade la url y le crea un name | from django.contrib import admin  from django.urls import path  from . import views  app\_name = "departamento\_app"  urlpatterns = [  path('new-departamento/',views.NewDepartamentoView.as\_view(), name='nuevo\_departamento'),  path('listar\_departamento/',views.ListDepartamentoView.as\_view(), name='departamentos'),  ] | Applications/departamento/urls.py |
|  | Se crea ek temnplate list\_all  Se llama la vista ListAllEmpleadosArea2 se personas | {% extends 'base.html' %}  {% block title %}Departamento{% endblock title %}  {% block content %}  <div class="grid-container">  <div class="grid-x">  <h1 class="cell">Lista de Departamentos</h1>  <div class="cell">  <table>  <thead>  <tr>  <th width="200">Id</th>  <th width="200">Nombre</th>  <th width="150">Código</th>  <th width="150">Estado</th>  </tr>  </thead>  <tbody>  {% for lista in departamentos %}  <tr>  <td>{{lista.id}}</td>  <td>{{lista.name}}</td>  <td>{{lista.shor\_name}}</td>  <td>{{lista.estado}}</td>  <td><a class="button warning" href="{% url 'persona\_app:empleados\_area' lista.shor\_name %}">Ver empleados</a>  </td>  </tr>  {% endfor %}  </tbody>  </table>  </div>  </div>  </div>  {% endblock content %} | Templates/departamento/list\_all.html |
|  | Se actualiza la ulr de personas | path('listar-todo-area2/<shorname>',views.ListAllEmpleadosArea2.as\_view(), name='empleados\_area'), | Applications/personas/urls.py |
|  | Se edita el archive list\_all\_area2.html | {% extends 'base.html' %}  {% block title %}Lista{% endblock title %}  {% block content %}  <div class="grid-container">  <div class="grid-x">  <h1 class="cell">Lista de empleados por area</h1>  <div class="cell">  <table>  <thead>  <tr>  <th width="200">ID</th>  <th>Nombres</th>  <th width="150">Apellidos</th>  <th width="150">Departamento</th>  <th width="150">Acción</th>  </tr>  </thead>  <tbody>  {% for lista in object\_list %}  <tr>  <td>{{lista.id}}</td>  <td>{{lista.firts\_name}}</td>  <td>{{lista.last\_name}}</td>  <td>{{lista.departamento}}</td>  <td><a class="button warning" href="{% url 'persona\_app:empleado\_detalle' lista.id %}">Ver</a>  </td>  </tr>  {% endfor %}  </tbody>  </table>  </div>  </div>  </div>  {% endblock content %} | Templates/personas/list\_all\_area2.html |
|  | Se crea la clase ListAllEmpleadosAdmin para poder editar y eliminar empelados | class ListAllEmpleadosAdmin(ListView):  template\_name = 'personas/list\_empleados.html'  paginate\_by = 6  def get\_queryset(self):  palabra = self.request.GET.get("palabraClave",'')  queryset = Empleado.objects.filter(  firts\_name\_\_icontains=palabra  )  return queryset | Applications/personas/views.py |
|  | Se crea la url | path('empleados\_admin/',views.ListAllEmpleadosAdmin.as\_view(),name='empleados\_admin'), | Applications/personas/urls.py |
|  | Se crea el template /list\_empleados.html | {% extends 'base.html' %}  {% block title %}Lista{% endblock title %}  {% block content %}  <div class="grid-container">  <div class="grid-x">  <h1 class="cell">Lista de empleados</h1>  <form class="cell grid-x grid-margin-x" method="GET">{%csrf\_token%}  <div class="cell large-7">  <input type="text" name="palabraClave" id="palabraClave" placeholder="Buscar empleado">  </div>  <div class="cell large-3">  <button type="submit" class="success button">Buscar</button>  </div>  </form>  <div class="cell">  <table>  <thead>  <tr>  <th width="200">ID</th>  <th>Nombres</th>  <th width="150">Apellidos</th>  <th width="150">Departamento</th>  <th width="150">Editar</th>  <th width="150">Eliminar</th>  </tr>  </thead>  <tbody>  {% for lista in object\_list %}  <tr>  <td>{{lista.id}}</td>  <td>{{lista.firts\_name}}</td>  <td>{{lista.last\_name}}</td>  <td>{{lista.departamento}}</td>  <td>  <a class="button success" href="{% url 'persona\_app:modificar\_empleado' lista.id %}">Editar</a>  </td>  <td>  <a class="button warning" href="{% url 'persona\_app:borrar\_empleado' lista.id %}">Eliminar</a>  </td>  </tr>  {% endfor %}  </tbody>  </table>  </div>  <div class="cell">  {% if is\_paginated %}  <nav aria-label="Pagination">  <ul class="pagination">  {% if page\_obj.has\_previous %}  <li class="pagination-previous"><a href="?page={{page\_obj.previous\_page\_number}}">Atras</a></li>  {% endif %}  {% for pagina in paginator.page\_range %}  {% ifequal pagina page\_obj.number %}  <li class="current"><span class="show-for-sr">You're on page</span> {{ pagina }}</li>  {% else %}  <li class=""><a href="?page={{pagina}}">{{ pagina }}</a></li>  {% endifequal %}  {% endfor %}  {% if page\_obj.has\_next %}  <li class="pagination-next"><a href="?page={{page\_obj.next\_page\_number}}">Siguiente</a></li>  {% endif %}  </ul>  </nav>  {% endif %}  </div>  </div>  </div>  {% endblock content %} | Templates/personas/list\_empleados.html |
|  | Se edita la clase para actualizar | class EmpleadoUpdateView(UpdateView):  model = Empleado  template\_name = "personas/update.html"  fields = ['firts\_name','last\_name','job','departamento','habilidades',]  success\_url = reverse\_lazy('persona\_app:empleados\_admin') | Applications/personas/views.py |
|  | Se edita el path | path('actualizar/<pk>/',views.EmpleadoUpdateView.as\_view(), name='modificar\_empleado'), | Applications/personas/urls.py |
|  | Se edita el template update.html | {% extends 'base.html' %}  {% block title %}Actualziar{% endblock title %}  {% block content %}  <div class="grid-container">  <div class="grid-x">  <h1 class="cell">Actualizar empleado</h1>  <div class="cell">  <form class="grid-x grid-margin-x" method="post">{% csrf\_token %}  <div class="medium-6 cell">  <label>Nombres  {{form.firts\_name}}  </label>  </div>  <div class="medium-6 cell">  <label>Apellidos  {{form.last\_name}}  </label>  </div>  <div class="medium-6 cell">  <label>Job  {{form.job}}  </label>  </div>  <div class="medium-6 cell">  <label>Departamento  {{form.departamento}}  </label>  </div>  <div class="medium-6 cell">  <label>Habilidades  {{form.habilidades}}  </label>  </div>  <div class="medium-12 cell">  <button type="submit" class="success button">Guardar</button>  </div>  </form>  </div>  </div>  </div>  {% endblock content %} | Templates/personas/update.html |
|  | Se edita la clase para borrar | class EmpleadoDeleteView(DeleteView):  model = Empleado  template\_name = "personas/delete.html"  success\_url = reverse\_lazy('persona\_app:empleados\_admin') | Applications/personas/views.py |
|  | Se edita el path | path('borrar/<pk>/',views.EmpleadoDeleteView.as\_view(), name='borrar\_empleado'), | Applications/personas/urls.py |
|  | Se edita el template delete.html | {% extends 'base.html' %}  {% block title %}Eliminar{% endblock title %}  {% block content %}  <div class="grid-container">  <div class="grid-x">  <h1 class="cell">Eliminar empleado: {{ object.firts\_name}} {{ object.last\_name}}</h1>  <div class="cell">  <form method="post">  <div class="callout alert">  {% csrf\_token %} {{form.as\_p}}  <h5>Confirme que desea eliminar</h5>  <button type="submit" class="alert button">Borrar</button>  </div>  </form>  </div>  </div>  </div>  </div>  {% endblock content %} | Templates/personas/delete.html |
|  | Se modifica el header para registrar un usuario | <div class="title-bar" data-responsive-toggle="responsive-menu" data-hide-for="medium">  <button class="menu-icon" type="button" data-toggle="responsive-menu"></button>  <div class="title-bar-title">Menu</div>  </div>    <div class="top-bar" id="responsive-menu">  <div class="top-bar-left">  <ul class="dropdown menu" data-dropdown-menu>  <li class="menu-text">Empleados</li>  <li class="has-submenu"><a href="{% url 'persona\_app:empleados\_all\_pag' %}">Listar</a></li>  <li><a href="{% url 'departamento\_app:departamentos' %}"">Departamentos</a></li>  <li><a href="{% url 'persona\_app:empleados\_admin' %}">Administrar</a></li>  </ul>  </div>  <div class="top-bar-right">  <ul class="menu">  <li><input type="search" placeholder="Buscar empleado"></li>  <li><a href="{% url 'persona\_app:add' %}" class="button">Registrar Nuevo</a></li>  </ul>  </div>  </div> | Templates/include/header.html |
|  | Se edita el archivo crear\_empleado | {% extends 'base.html' %}  {% block title %}Actualziar{% endblock title %}  {% block content %}  <div class="grid-container">  <div class="grid-x">  <h1 class="cell">Crear empleado</h1>  <div class="cell">  <form class="grid-x grid-margin-x" method="post">{% csrf\_token %}  <div class="medium-6 cell">  <label>Nombres {{form.firts\_name}}</label>  </div>  <div class="medium-6 cell">  <label>Apellidos {{form.last\_name}}</label>  </div>  <div class="medium-6 cell">  <label>Trabajo {{form.job}}</label>  </div>  <div class="medium-6 cell">  <label>Departamento {{form.departamento}}</label>  </div>  <div class="medium-12 cell">  <label>Habilidades {{form.habilidades}}</label>  </div>  <div class="medium-12 cell">  <button type="submit" class="success button">Guardar</button>  </div>  </form>  </div>  </div>  </div>  {% endblock content %} | Templates/personas/crear\_empleado.html |
|  | Se instala Pillow | pip install Pillow | cmd |
|  | Se crea la carpeta media |  | ./ |
|  | Se añade la carpeta al archivo local | MEDIA\_URL = '/media/'  MEDIA\_ROOT = BASE\_DIR.child('media') | Empleadosproj/settings/local.py |
|  | Se modifica el archivo url del proyecto | from django.contrib import admin  from django.urls import path, re\_path, include  from django.conf import settings  from django.conf.urls.static import static    urlpatterns = [  path('admin/', admin.site.urls),  # Se incluyen las urls de la app departamento  path('', include('applications.departamento.urls')),  # Se incluyen las urls de la app persona  path('', include('applications.personas.urls')),  # Se incluyen las urls de la app home  path('', include('applications.home.urls')),  ] + static(settings.MEDIA\_URL, document\_root=settings.MEDIA\_ROOT) | Empleadosproj/urls.py |
|  | Se actualiza el formulario de crear empleado no se puede olvidar el enctype parea que guarde las imágenes | {% extends 'base.html' %}  {% block title %}Crear{% endblock title %}  {% block content %}  <div class="grid-container">  <div class="grid-x">  <h1 class="cell">Crear empleado</h1>  <div class="cell">  <form class="grid-x grid-margin-x" method="post" enctype="multipart/form-data">{% csrf\_token %}  <div class="medium-6 cell">  <label>Nombres {{form.firts\_name}}</label>  </div>  <div class="medium-6 cell">  <label>Apellidos {{form.last\_name}}</label>  </div>  <div class="medium-6 cell">  <label>Trabajo {{form.job}}</label>  </div>  <div class="medium-6 cell">  <label>Departamento {{form.departamento}}</label>  </div>  <div class="medium-6 cell">  <label>Habilidades {{form.habilidades}}</label>  </div>  <div class="medium-6 cell">  <label>Avatar {{form.avatar}}</label>  </div>  <div class="medium-12 cell">  <button type="submit" class="success button">Guardar</button>  </div>  </form>  </div>  </div>  </div>  {% endblock content %} | Templates/personas/crear\_empleado.html |
|  | Se modifica el header para tener el nuevo link | <div class="title-bar" data-responsive-toggle="responsive-menu" data-hide-for="medium">  <button class="menu-icon" type="button" data-toggle="responsive-menu"></button>  <div class="title-bar-title">Menu</div>  </div>  <div class="top-bar" id="responsive-menu">  <div class="top-bar-left">  <ul class="dropdown menu" data-dropdown-menu>  <li class="menu-text">Empleados</li>  <li class="has-submenu"><a href="{% url 'persona\_app:empleados\_all\_pag' %}">Listar</a></li>  <li><a href="{% url 'departamento\_app:departamentos' %}"">Departamentos</a></li>  <li><a href=" {% url 'persona\_app:empleados\_admin' %}">Administrar</a></li>  <li><a href="{% url 'departamento\_app:nuevo\_departamento' %}">Nuevo Departamento</a></li>  </ul>  </div>  <div class="top-bar-right">  <ul class="menu">  <li><input type="search" placeholder="Buscar empleado"></li>  <li><a href="{% url 'persona\_app:add' %}" class="button">Registrar Empleado</a></li>  </ul>  </div>  </div> | Templates/includes/header.html |
|  | Se modifica el template que crea departamentos nuevos | {% extends 'base.html' %}  {% block title %}Crear Departamento{% endblock title %}  {% block content %}  <div class="grid-container">  <div class="grid-x">  <h1 class="cell">Crear Departamento</h1>  <div class="cell">  <form class="cell callout grid-x grid-margin-x" method="post">{% csrf\_token %}  <h3 class="medium-12 cell">Datos empleado</h3>  <div class="medium-6 cell"><label>Nombre:{{form.nombre}}</label></div>  <div class="medium-6 cell"><label>Apellido:{{form.apellido}}</label></div>  <h3 class="medium-12 cell">Datos Departamento</h3>  <div class="medium-6 cell"><label>Nombre:{{form.departamento}}</label></div>  <div class="medium-6 cell"><label>Sigla:{{form.shorname}}</label></div>  </form>  <div class="medium-12 cell">  <button type="submit" class="success button">Guardar</button>  </div>  </form>  </div>  </div>  </div>  {% endblock content %} | Tamplates/departamento/new\_departamento |
|  | Se modifica el archivo detail\_empleado para que muestre los datos y la foto de un empleado | {% extends 'base.html' %}  {% load static %}  {% block title %}{{empleado.last\_name}}{% endblock title %}  {% block content %}  <div class="grid-container">  <div class="grid-x grid-margin-x align-center callout">  <h4 class="cell large-12 ">Todos los datos de:</h4>  <div class="card large-8" style="width: 300px;">  <div class="card-divider">  {{empleado.full\_name}}  </div>  {% if empleado.avatar %}  <img src="{{empleado.avatar.url}}">  {% else %}  <img src="{% static 'img/1.jpg' %}">  {% endif %}  <div class="card-section">  <p>{{empleado.firts\_name}}</p>  <p>{{empleado.last\_name}}</p>  <p>{{empleado.get\_job\_display}}</p>  <p>{{empleado.departamento.name}}</p>  <p><span class="label">Habilidades:</span></p>  <ul class="vertical menu">  {% for habilidad in empleado.habilidades.all %}  <li>{{ habilidad.habilidad }}</li>  {% endfor %}  </ul>  </div>  </div>  </div>  {% endblock content %} | Templates/personas/detail\_empleado.html¿ |
|  | Se crea un forms para editar el formulario de crear personas | from django import forms  from .models import Empleado  class EmpleadoForm(forms.ModelForm):  """Form definition for MODELNAME."""  class Meta:  """Meta definition for MODELNAMEform."""  model = Empleado  fields = (  'firts\_name',  'last\_name',  'job',  'departamento',  'habilidades',  'avatar',  )  widgets = {  'habilidades': forms.CheckboxSelectMultiple(),  } | Applications/personas/forms.py |
|  | Se importa el form | from .forms import EmpleadoForm | Apllications/personas/views.py |
|  | Se usa el form en la vista que crea personas | class EmpleadoCreateView(CreateView):  template\_name = "personas/crear\_empleado.html"  model = Empleado  #fields = ('\_\_all\_\_')  form\_class = EmpleadoForm  success\_url = reverse\_lazy('persona\_app:empleados\_admin')  def form\_valid(self, form):  empleado = form.save(commit=False)  empleado.full\_name = empleado.firts\_name+' '+empleado.last\_name  empleado.save()  return super(EmpleadoCreateView, self).form\_valid(form) | Apllications/personas/views.py |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |