Evidencia de Modelo Base Datos

**Aplicación Accounts**

*from django.contrib.auth.base\_user import BaseUserManager*

*from django.contrib.auth.models import AbstractUser*

*from django.db import models*

*from django.utils.translation import gettext\_lazy as \_*

class PerfilesManager(BaseUserManager):

def create\_user(

self, rut,

correo\_electronico,

password= str,

\*\*extra\_fields: dict[str, str | bool],

) -> "Perfiles":

if not rut:

raise ValueError("The RUT must be set")

if not correo\_electronico:

raise ValueError("The Email must be set")

correo\_electronico = self.normalize\_email(correo\_electronico)

user: Perfiles = self.model(rut=rut, correo\_electronico=correo\_electronico, \*\*extra\_fields)

user.set\_password(password)

user.save(using=self.\_db)

return user

def create\_superuser(

self,

rut,

correo\_electronico,

password=str,

\*\*extra\_fields: dict[str, str | bool],

) -> "Perfiles":

extra\_fields.setdefault('is\_staff', True)

extra\_fields.setdefault('is\_superuser', True)

extra\_fields.setdefault('is\_active', True)

return self.create\_user(rut, correo\_electronico, password, \*\*extra\_fields)

class EstadoPerfil(models.Model):

id\_estadoperfil = models.AutoField(unique=True, primary\_key=True)

descripcion = models.CharField(max\_length=30)

def \_\_str\_\_(self):

return f"{self.id\_estadoperfil} - {self.descripcion}"

class Region(models.Model):

id\_region = models.AutoField(unique=True, primary\_key=True)

nombre\_region = models.CharField(max\_length=80)

def \_\_str\_\_(self):

return f"{self.nombre\_region}"

class Comuna(models.Model):

id\_comuna = models.AutoField(unique=True, primary\_key=True)

nombre\_comuna = models.CharField(unique=True, max\_length=80)

id\_region = models.ForeignKey(Region, on\_delete=models.CASCADE)

def \_\_str\_\_(self):

return f"{self.nombre\_comuna}"

class Roles(models.Model):

id\_rol = models.AutoField(unique=True, primary\_key=True)

descripcion = models.CharField(max\_length=50)

def \_\_str\_\_(self):

return f"{self.id\_rol} - {self.descripcion}"

class Sexo(models.Model):

id\_sexo = models.AutoField(unique=True, primary\_key=True)

descripcion = models.CharField(max\_length=10)

def \_\_str\_\_(self):

return f"{self.descripcion}"

class Parentesco(models.Model):

id\_parentesco = models.AutoField(unique=True, primary\_key=True)

descripcion = models.CharField(max\_length=50)

def \_\_str\_\_(self):

return f"{self.id\_parentesco} - {self.descripcion}"

class Perfiles(AbstractUser):

username = None

first\_name = None

last\_name = None

email = None

rut = models.IntegerField(unique=True, primary\_key=True)

dv = models.CharField(max\_length=1)

nombre = models.CharField(max\_length=60)

apellido = models.CharField(max\_length=60)

fecha\_nacimiento = models.DateField(null=True, blank=True)

id\_sexo = models.ForeignKey(Sexo, on\_delete=models.CASCADE, null=True, blank=True)

id\_parentesco = models.ForeignKey(Parentesco, on\_delete=models.CASCADE, null=True, blank=True)

direccion = models.CharField(max\_length=100, blank=True)

numero\_contacto = models.CharField(max\_length=11)

correo\_electronico = models.CharField(max\_length=80, blank=True)

fecha\_incorporacion = models.DateField(null=True, blank=True)

fecha\_termino = models.DateField(null=True, blank=True)

id\_rol = models.ForeignKey(Roles, on\_delete=models.CASCADE, null=True, blank=True)

id\_comuna = models.ForeignKey(Comuna, on\_delete=models.CASCADE, null=True, blank=True)

id\_estadoperfil = models.ForeignKey(EstadoPerfil, on\_delete=models.CASCADE, null=True, blank=True, default=1)

familia = models.ForeignKey(

'Familia',

on\_delete=models.SET\_NULL,

null=True,

blank=True,

related\_name='miembros',

help\_text="Familia a la que pertenece este perfil"

)

USERNAME\_FIELD = 'rut'

REQUIRED\_FIELDS = ['correo\_electronico', 'nombre', 'apellido']

objects = PerfilesManager()

def \_\_str\_\_(self):

return f"{self.rut} - {self.nombre} {self.apellido}"

class Familia(models.Model):

id\_familia = models.AutoField(primary\_key=True)

titular = models.OneToOneField(

Perfiles,

on\_delete=models.CASCADE,

related\_name='familia\_titular',

help\_text="El titular de la familia"

)

nombre = models.CharField(

max\_length=100,

help\_text="Nombre de la familia o grupo (puedes usar el apellido del titular, por ejemplo)"

)

fecha\_creacion = models.DateTimeField(auto\_now\_add=True)

def \_\_str\_\_(self):

return f"Familia {self.nombre} (Titular: {self.titular.nombre})"

@property

def miembros(self):

"""

Retorna todos los miembros de la familia, incluyendo al titular.

"""

return Perfiles.objects.filter(familia=self)

**Aplicación dashboarda**

from django.db import models

from accounts.models import Comuna, Perfiles

from .utils import obtener\_coordenadas

class EstadoJuntaVecinos(models.Model):

id\_estado = models.AutoField(unique=True, primary\_key=True)

nombre\_estado = models.CharField(max\_length=100)

def \_\_str\_\_(self):

return f"{self.id\_estado} - {self.nombre\_estado}"

class JuntaVecinos(models.Model):

id\_juntavecino = models.AutoField(unique=True, primary\_key=True)

nombre\_organizacion = models.CharField(max\_length=100)

id\_comuna = models.ForeignKey(Comuna, on\_delete=models.CASCADE)

fecha\_integracion = models.DateField()

fecha\_termino = models.DateField(blank=True, null=True)

direccion = models.CharField(max\_length=100)

id\_estado = models.ForeignKey(EstadoJuntaVecinos, on\_delete=models.CASCADE)

perfiles = models.ManyToManyField(Perfiles, blank=True)

# Campos para las coordenadas

latitud = models.DecimalField(max\_digits=9, decimal\_places=6, null=True, blank=True)

longitud = models.DecimalField(max\_digits=9, decimal\_places=6, null=True, blank=True)

def save(self, \*args, \*\*kwargs):

# Si no hay coordenadas guardadas, intenta obtenerlas automáticamente

if not self.latitud or not self.longitud:

comuna\_nombre = self.id\_comuna.nombre\_comuna

region\_nombre = self.id\_comuna.id\_region.nombre\_region

direccion = self.direccion

# Llamar a la función con dirección, comuna y región

lat, lon = obtener\_coordenadas(direccion, comuna\_nombre, region\_nombre)

print(f"Coordenadas obtenidas: lat={lat}, lon={lon}") # <-- Depuración

if lat and lon:

self.latitud = lat

self.longitud = lon

super().save(\*args, \*\*kwargs)

def \_\_str\_\_(self):

return f"{self.nombre\_organizacion} - {self.id\_comuna}"e

**Aplicación dashboardjv**

from django.db import models

from accounts.models import Perfiles

from dashboarda.models import JuntaVecinos

class EstadoPostulacion(models.Model):

id\_estadopostulacion = models.AutoField(unique=True, primary\_key=True)

descripcion = models.CharField(max\_length=20)

class EstadoCertificado(models.Model):

id\_estadocertificado = models.AutoField(unique=True, primary\_key=True)

descripcion = models.CharField(max\_length=30)

class EstadoNoticia(models.Model):

id\_estadonoticia = models.AutoField(unique=True, primary\_key=True)

descripcion = models.CharField(max\_length=30)

class EstadoActividad(models.Model):

id\_estadoactividad = models.AutoField(unique=True, primary\_key=True)

descripcion = models.CharField(max\_length=30)

class Actividad(models.Model):

id\_actividad = models.AutoField(unique=True, primary\_key=True)

nombre = models.CharField(max\_length=80)

descripcion = models.TextField()

fecha\_inicio = models.DateField()

fecha\_termino = models.DateField()

horario\_inicio = models.TimeField()

horario\_termino = models.TimeField()

id\_estadoactividad = models.ForeignKey(EstadoNoticia, on\_delete=models.CASCADE)

class PostulacionProyectos(models.Model):

id\_proyecto = models.AutoField(unique=True, primary\_key=True)

titulo = models.CharField(max\_length=150)

descripcion = models.TextField()

objetivos = models.TextField()

presupuesto = models.DecimalField(max\_digits=15, decimal\_places=2)

id\_juntavecino = models.ForeignKey(JuntaVecinos, on\_delete=models.CASCADE)

rut\_postulante = models.ForeignKey(Perfiles, on\_delete=models.CASCADE)

id\_estadopostulacion = models.ForeignKey(EstadoPostulacion, on\_delete=models.CASCADE)

class CertificadosResi(models.Model):

id\_certificado = models.AutoField(unique=True, primary\_key=True)

nombre\_pstulante = models.CharField(max\_length=120)

direccion\_postulante = models.CharField(max\_length=120)

fecha\_emision = models.DateField()

id\_juntavecino = models.ForeignKey(JuntaVecinos, on\_delete=models.CASCADE)

rut\_postulante = models.ForeignKey(Perfiles, on\_delete=models.CASCADE)

id\_estadocertificado = models.ForeignKey(EstadoCertificado, on\_delete=models.CASCADE)

class Noticias(models.Model):

id\_noticia = models.AutoField(unique=True, primary\_key=True)

nombre = models.CharField(max\_length=80)

descripcion = models.TextField()

fecha\_inicio = models.DateField()

fecha\_termino = models.DateField(null=True, blank=True)

id\_estadonoticia = models.ForeignKey(EstadoNoticia, on\_delete=models.CASCADE)

id\_juntavecinos = models.ForeignKey(JuntaVecinos, on\_delete=models.CASCADE)

def \_\_str\_\_(self):

return f"{self.nombre} ({self.fecha\_inicio} - {self.fecha\_termino})"