

# 🛡 Seguridad, Moderación y Cuentas de Menores – Proyecto DeRevés

---

“Una red social deportiva segura, moderada y responsable.”

---

## ⌚ Objetivo

Garantizar que **DeRevés** sea una plataforma:

- Segura, responsable y sin contenido ofensivo.
  - Respetuosa de las leyes sobre menores de edad.
  - Con mecanismos de moderación efectivos y transparentes.
- 

## 🧩 Principios Generales

### 1. Nada de anonimato total

Todo usuario debe tener una cuenta verificada (correo y alias visible).

### 2. Moderación activa y preventiva

Filtros automáticos + reportes + revisión manual.

### 3. Control de edad y supervisión adulta

Menores de edad solo pueden participar con autorización de un tutor.

### 4. Tolerancia cero a lenguaje ofensivo, acoso o discriminación.

### 5. Transparencia en las reglas de convivencia, visibles antes de publicar.

---

## 👤 Clasificación de Cuentas

PROF

Tipo de cuenta	Descripción	Restricciones
<b>Adulto (Jugador, Dueño, Organizador)</b>	Usuario mayor de 18 años.	Acceso total a red social, reservas y torneos.
<b>Menor supervisado</b>	Jugador menor de 18 años, asociado a un tutor.	Puede ver contenido, crear partidos, pero requiere aprobación del tutor para reservas.
<b>Tutor responsable</b>	Adulto que autoriza y supervisa la cuenta del menor.	Aprueba reservas, controla visibilidad y permisos.

---

## 👤 Modelo de Datos

Campos sugeridos:

- `fecha_nacimiento`
  - `es_menor` (boolean calculado)
  - `tipo_cuenta (ADULTO, MENOR_SUPERVISADO, TUTOR)`
  - `tutor` (FK a Usuario adulto)
  - `tutor_validado` (boolean)
  - `fecha_consentimiento_tutor`
- 

## ↳ Flujo de Registro con Control de Edad

1. **Usuario completa el formulario** con fecha de nacimiento.
2. Si es menor, se activa el flujo de **cuenta supervisada**.
3. Se solicitan los datos del tutor:
  - Nombre, email y teléfono.
4. Se envía un correo al tutor con un enlace de **confirmación de consentimiento**.
5. Hasta que el tutor acepte, la cuenta queda en **modo lectura**.

## ✳️ Estados de la cuenta

Estado	Descripción
<code>PENDIENTE_TUTOR</code>	El tutor aún no validó la cuenta.
<code>SUPERVISADA</code>	Cuenta activa con control adulto.
<code>ADULTO_VALIDADO</code>	Usuario sin restricciones.

---

## Monkey Modo Menor Supervisado

- Las reservas creadas por menores quedan en estado “**PENDIENTE\_TUTOR**”.
- El tutor recibe una notificación (correo o app).
- Solo tras la aceptación, el turno pasa a “**CONFIRMADO**”.

PROF

## ⌚ Reglas UX

- En las pantallas de menores, se indica:

“Tu cuenta está supervisada. Algunas acciones requieren la aprobación de tu tutor.”

- Los botones de reserva muestran un ícono de candado  mientras están pendientes.
  - No pueden enviar mensajes privados a usuarios no autorizados.
- 

## 👨‍👩‍👧 Rol del Tutor Responsable

El **tutor** tiene un panel de control con:

### ◇ Sección “Mis jugadores menores”

- Lista de menores asociados.
- Estado de cuenta (supervisada / pendiente).
- Permisos configurables:
  - Aprobar manualmente cada reserva.
  - Monto máximo por reserva.
  - Horarios habilitados.

## Reservas pendientes

- Lista de reservas “pendientes de aprobación”.
- Botones **Aceptar / Rechazar**.
- Historial de actividad del menor.

## Moderación del Contenido

### Capas de Defensa

Capa	Descripción	Acción
<b>1. Filtro automático</b>	Lista de palabras prohibidas (insultos, contenido sexual, acoso).	Bloquea o reemplaza palabras.
<b>2. Sistema de reportes</b>	Los usuarios pueden reportar publicaciones ofensivas.	Marca como <b>REPORTADO</b> y alerta a los moderadores.
<b>3. Revisión manual</b>	Panel para moderadores / Mantis.	Revisión, sanción o eliminación.
<b>4. Suspensión / Baneo</b>	Reincidentes son bloqueados.	Se guarda registro del motivo.

### Tipos de reporte

PROF

- Lenguaje ofensivo / insulto.
- Acoso o discriminación.
- Contenido sexual o violento.
- Spam / fraude.
- Otro (con descripción libre).

### Flujo de Reporte

1. Usuario pulsa “ Reportar” en una publicación.
2. Se abre modal con motivos.
3. Se guarda el reporte con usuario, motivo y texto.
4. Si el contenido supera un umbral de reportes (ej. 3), se **oculta automáticamente** hasta revisión.

## Ejemplo HTML: Registro con Tutor

```
<form id="registro-jugador">
  <h2>Crear cuenta DeRevés</h2>

  <label>Nombre completo</label>
  <input type="text" name="nombre" required>

  <label>Alias</label>
  <input type="text" name="alias" required>

  <label>Email</label>
  <input type="email" name="email" required>

  <label>Fecha de nacimiento</label>
  <input type="date" name="fecha_nacimiento" required>

  <div id="seccion-tutor" style="display:none;">
    <h3>Datos del tutor responsable</h3>
    <p>Sos menor de edad. Necesitás un adulto que autorice tu cuenta y tus reservas.</p>

    <label>Nombre del tutor</label>
    <input type="text" name="tutor_nombre">

    <label>Email del tutor</label>
    <input type="email" name="tutor_email">

    <label>Teléfono del tutor</label>
    <input type="tel" name="tutor_telefono">
  </div>

  <label>
    <input type="checkbox" required>
    Acepto las normas de convivencia y política de uso.
  </label>

  <button type="submit" class="btn-primary">Crear cuenta</button>
</form>

<script>
const inputFecha =
  document.querySelector('input[name="fecha_nacimiento"]');
const seccionTutor = document.getElementById('seccion-tutor');

inputFecha.addEventListener('change', () => {
  const hoy = new Date();
  const fechaNac = new Date(inputFecha.value);
  let edad = hoy.getFullYear() - fechaNac.getFullYear();
  const m = hoy.getMonth() - fechaNac.getMonth();
  if (m < 0 || (m === 0 && hoy.getDate() < fechaNac.getDate())) edad--;
})
```

—  
PROF

```
    seccionTutor.style.display = edad < 18 ? 'block' : 'none';
  });
</script>
```

▷ Ejemplo HTML: Publicación con botón de reporte

```
<article class="card-publicacion">
  <header>
    
    <div><strong>Juan Pérez</strong><br><small>hace 2h · 7ma</small>
  </div>
  </header>

  <p class="contenido-publicacion">
    Tremendo partido hoy en DeRevés Padel Club 🔥
  </p>

  <footer>
    <button class="btn-ghost">❤️ Me gusta</button>
    <button class="btn-ghost">💬 Comentar</button>
    <button class="btn-link btn-reportar">⚠️ Reportar</button>
  </footer>
</article>

<div id="modal-reporto" style="display:none;">
  <div class="modal-contenido">
    <h3>Reportar publicación</h3>
    <select id="motivo-reporto">
      <option value="">Motivo...</option>
      <option value="insulto">Insulto / lenguaje ofensivo</option>
      <option value="acoso">Acoso / amenaza</option>
      <option value="sexual">Contenido sexual</option>
      <option value="spam">Spam / engaño</option>
      <option value="otro">Otro</option>
    </select>
    <textarea placeholder="Comentario opcional"></textarea>
    <button class="btn-primary">Enviar reporte</button>
    <button class="btn-secundario">Cancelar</button>
  </div>
</div>
```

⌚ Panel del Tutor (HTML conceptual)

```
<section>
  <h2>Mis jugadores menores</h2>

  <article class="card-menor">
    <header>
      <strong>Valentín Pérez</strong> · 13 años · 8va categoría
      <span class="badge-supervisado">Cuenta supervisada</span>
    </header>

    <p>Permisos:</p>
```

```

<label>
    <input type="checkbox" checked>
        Aprobar manualmente cada reserva
</label>
<label>
    Monto máximo por reserva:
    <input type="number" value="5000"> ARS
</label>

<h3>Reservas pendientes</h3>
<ul>
    <li>
        Sábado 18:00 - Cancha Padel 1 - Complejo X
        <button class="btn-primary">Aceptar</button>
        <button class="btn-secundario">Rechazar</button>
    </li>
</ul>
</article>
</section>

```

## ❸ Recomendaciones técnicas (backend)

Middleware de verificación de edad y permisos.

Decoradores en vistas:

```
@restringir_menores
def crear_reserva(request): ...
```

Tabla ReporteContenido para moderación:

```
class ReporteContenido(models.Model):
    autor = models.ForeignKey(User, on_delete=models.CASCADE)
    contenido_id = models.PositiveIntegerField()
    tipo = models.CharField(max_length=50)
    motivo = models.TextField()
    fecha = models.DateTimeField(auto_now_add=True)
    estado = models.CharField(max_length=20, default='PENDIENTE')
```

### ✓ Beneficios

Área      Beneficio

Seguridad infantil Cumple con normas legales (control parental y autorización).

Moderación Previene abusos, lenguaje inapropiado y spam.

Confiabilidad Aumenta la reputación de la plataforma.

Escalabilidad Sistema preparado para múltiples moderadores o IA de detección futura.

 Hecho por Mantis

Mantis Software Solutions

“Cuidamos a los jugadores, dentro y fuera de la cancha.”

📍 Cruz Alta, Córdoba, Argentina  
🌐 www.mantistec.com

✉️ contacto@mantistec.com

---

¿Querés que el siguiente documento que preparemos sea `docs/models\_moderacion\_y\_tutores.md`, con los modelos Django exactos (`Usuario`, `PerfilJugador`, `Tutor`, `ReporteContenido`, `Reserva`) listos para pegar en tu proyecto base?  
Así dejamos resuelto el backend de seguridad y control parental.

Tú dijiste:

si

ChatGPT Plus

Perfecto, vamos directo al código ☑  
Acá tenés el documento docs/models\_moderacion\_y\_tutores.md con modelos Django listos para pegar, organizados por apps (cuentas, reservas, social).

Podés copiarlos tal cual y luego ir ajustando detalles (nombres de apps, imports, etc.).

# 📚 Modelos de Moderación y Tutores – DeRevés

> Modelos Django para: usuarios, jugadores, tutores, reservas con menores y reportes de contenido.

---

## 📁 1. App `cuentas` – Usuario, PerfilJugador y Tutor

—  
PROF  
### 1.1. Usuario (Custom User)

Archivo: `apps/cuentas/models.py`

```
```python
from django.contrib.auth.models import AbstractUser
from django.db import models
from django.utils import timezone
from datetime import date

class Usuario(AbstractUser):
    """
        Usuario base del sistema DeRevés.
        Se extiende de AbstractUser para agregar campos propios.
    """

    # Tipos de cuenta posibles
    TIPO CUENTA_CHOICES = [
```

```

        ('ADULTO', 'Adulto'),
        ('MENOR_SUPERVISADO', 'Menor supervisado'),
        ('TUTOR', 'Tutor responsable'),
    ]

tipo_cuenta = models.CharField(
    max_length=20,
    choices=TIPO CUENTA_CHOICES,
    default='ADULTO',
)
fecha_nacimiento = models.DateField(null=True, blank=True)

# Tutor responsable (solo para MENOR_SUPERVISADO)
tutor = models.ForeignKey(
    'self',
    null=True,
    blank=True,
    on_delete=models.SET_NULL,
    related_name='menores_a_cargo',
    help_text='Usuario adulto responsable de este menor.',
)
tutor_validado = models.BooleanField(
    default=False,
    help_text='Indica si el tutor aceptó la supervisión de este menor.')
fecha Consentimiento_tutor = models.DateTimeField(
    null=True,
    blank=True,
    help_text='Fecha y hora en que el tutor aceptó las condiciones.')
)

telefono = models.CharField(
    max_length=30,
    blank=True,
    help_text='Teléfono de contacto.')
)

def calcular_edad(self) -> int:
    if not self.fecha_nacimiento:
        return 0
    hoy = date.today()
    edad = hoy.year - self.fecha_nacimiento.year
    if (hoy.month, hoy.day) < (self.fecha_nacimiento.month,
self.fecha_nacimiento.day):
        edad -= 1
    return edad

@property
def es_menor(self) -> bool:

```

—

PROF

```

edad = self.calcular_edad()
return edad > 0 and edad < 18

def es_menor_supervisado(self) -> bool:
    return self.tipo_cuenta == 'MENOR_SUPERVISADO'

def es_tutor(self) -> bool:
    return self.tipo_cuenta == 'TUTOR'

def __str__(self):
    return f'{self.username} ({self.get_tipo_cuenta_display()})'

```

## 1.2. PerfilJugador

```

class CategoriaNivel(models.Model):
    """
    Ej.: 8va, 7ma, 6ta, etc.
    """
    nombre = models.CharField(max_length=20, unique=True)
    orden = models.PositiveIntegerField(default=0)
    descripcion = models.TextField(blank=True)

    class Meta:
        ordering = ['orden']

    def __str__(self):
        return self.nombre

```

```

class PerfilJugador(models.Model):
    """
    Perfil extendido para jugadores.
    """
    ESTADO_JUEGO_CHOICES = [
        ('RECREATIVO', 'Recreativo'),
        ('COMPETITIVO', 'Competitivo'),
    ]

    usuario = models.OneToOneField(
        'cuentas.Usuario',
        on_delete=models.CASCADE,
        related_name='perfil_jugador'
    )

    alias = models.CharField(max_length=50)
    localidad = models.CharField(max_length=100, blank=True)
    estado_juego = models.CharField(
        max_length=20,
        choices=ESTADO_JUEGO_CHOICES,
        default='RECREATIVO'
    )
    categoria_nivel = models.ForeignKey(
        CategoriaNivel,

```

```

        null=True,
        blank=True,
        on_delete=models.SET_NULL
    )

biografia = models.TextField(blank=True)

# Estadísticas (pueden llenarse por lógica de negocio)
total_partidos = models.PositiveIntegerField(default=0)
total_torneos = models.PositiveIntegerField(default=0)
puntos_ranking = models.IntegerField(default=0)
pelotitas = models.PositiveIntegerField(default=0)

creado_en = models.DateTimeField(auto_now_add=True)
actualizado_en = models.DateTimeField(auto_now=True)

def __str__(self):
    return self.alias or self.usuario.username

```

### 1.3. Modelo simple de Tutor (opcional)

Si quisieras tener un modelo específico para datos extra de tutor:

```

class PerfilTutor(models.Model):
    """
    Datos adicionales para usuarios que actúan como tutores.
    """

    usuario = models.OneToOneField(
        'cuentas.Usuario',
        on_delete=models.CASCADE,
        related_name='perfil_tutor'
    )

    # Configuración de permisos sobre menores
    aprobar_cada_reserva = models.BooleanField(
        default=True,
        help_text='Si está activo, cada reserva del menor requiere
aprobación manual.'
    )

    monto_maximo_reserva = models.DecimalField(
        max_digits=10,
        decimal_places=2,
        default=0,
        help_text='Si es mayor que 0, limita el valor de cada reserva
del menor.'
    )

    horario_desde = models.TimeField(
        null=True,
        blank=True,
        help_text='Horario mínimo permitido para reservas del menor.'
    )

```

```
horario_hasta = models.TimeField(
    null=True,
    blank=True,
    help_text='Horario máximo permitido para reservas del menor.'
)
```

```
def __str__(self):
    return f'Tutor: {self.usuario.get_full_name()} or
self.usuario.username'
```

## 2. App reservas – Reserva con control de tutor

Archivo: apps/reservas/models.py

```
from django.db import models
from django.utils import timezone

from cuentas.models import Usuario
from complejos.models import Cancha # asumimos existencia de esta app

class Reserva(models.Model):
    """
    Reserva de una cancha. Si el usuario es menor supervisado,
    la reserva puede quedar pendiente de aprobación del tutor.
    """

    ESTADO_CHOICES = [
        ('PENDIENTE', 'Pendiente'),
        ('PENDIENTE_TUTOR', 'Pendiente de aprobación del tutor'),
        ('CONFIRMADA', 'Confirmada'),
        ('RECHAZADA_TUTOR', 'Rechazada por el tutor'),
        ('CANCELADA', 'Cancelada'),
        ('NO_ASISTIO', 'No asistió'),
        ('COMPLETADA', 'Completada'),
    ]
    cancha = models.ForeignKey(
        Cancha,
        on_delete=models.CASCADE,
        related_name='reservas'
    )
    jugador_principal = models.ForeignKey(
        Usuario,
        on_delete=models.CASCADE,
        related_name='reservas_realizadas'
    )
    fecha = models.DateField()
    hora_inicio = models.TimeField()
    hora_fin = models.TimeField()
```

```
    precio = models.DecimalField(max_digits=10, decimal_places=2,
default=0)

    estado = models.CharField(
        max_length=20,
        choices=ESTADO_CHOICES,
        default='PENDIENTE'
    )

    # Auditoría de tutor
    tutor = models.ForeignKey(
        Usuario,
        null=True,
        blank=True,
        on_delete=models.SET_NULL,
        related_name='reservas_como_tutor',
        help_text='Tutor que aprueba o rechaza reservas de menores.'
    )

    fecha_aprobacion_tutor = models.DateTimeField(
        null=True,
        blank=True
    )

    observaciones = models.TextField(blank=True)

    creado_en = models.DateTimeField(auto_now_add=True)
    actualizado_en = models.DateTimeField(auto_now=True)

    class Meta:
        ordering = ['-fecha', '-hora_inicio']
        verbose_name = 'Reserva'
        verbose_name_plural = 'Reservas'

    def __str__(self):
        return f'{self.cancha} - {self.fecha} {self.hora_inicio}'

    def marcar_pendiente_tutor(self):
        self.estado = 'PENDIENTE_TUTOR'
        self.save(update_fields=['estado', 'actualizado_en'])

    def aprobar_por_tutor(self, tutor: Usuario):
        self.estado = 'CONFIRMADA'
        self.tutor = tutor
        self.fecha_aprobacion_tutor = timezone.now()
        self.save(update_fields=['estado', 'tutor',
'fecha_aprobacion_tutor', 'actualizado_en'])

    def rechazar_por_tutor(self, tutor: Usuario):
        self.estado = 'RECHAZADA_TUTOR'
        self.tutor = tutor
        self.fecha_aprobacion_tutor = timezone.now()
        self.save(update_fields=['estado', 'tutor',
```

```
'fecha_aprobacion_tutor', 'actualizado_en'])
```

Regla de negocio:

Cuando el jugador\_principal sea un MENOR\_SUPERVISADO, la lógica del servicio (en services.py) debe llamar a marcar\_pendiente\_tutor() en lugar de dejar CONFIRMADA directamente.

### 3. App social – Publicaciones y Reportes de Contenido

Archivo: apps/social/models.py

#### 3.1. Publicación (base de la red social)

```
from django.db import models
from django.utils import timezone
from cuentas.models import Usuario, PerfilJugador

class Publicacion(models.Model):
    """
    Publicación dentro de la red social de DeRevés.
    Puede estar asociada a un partido, torneo, logro, etc.
    """

    TIPO_PUBLICACION_CHOICES = [
        ('GENERAL', 'General'),
        ('LOGRO', 'Logro'),
        ('TORNEO', 'Torneo'),
        ('PARTIDO', 'Partido'),
    ]

    autor = models.ForeignKey(
        PerfilJugador,
        on_delete=models.CASCADE,
        related_name='publicaciones'
    )

    tipo = models.CharField(
        max_length=20,
        choices=TIPO_PUBLICACION_CHOICES,
        default='GENERAL'
    )

    texto = models.TextField()
    imagen = models.ImageField(
        upload_to='publicaciones/',
        null=True,
        blank=True
    )

    visible_publico = models.BooleanField(default=True)

    # Moderación
    bloqueada = models.BooleanField(
        default=False,
```

—  
PROF

```

        help_text='Si está bloqueada, no se muestra a otros usuarios.'
    )

motivo_bloqueo = models.TextField(blank=True)

creado_en = models.DateTimeField(auto_now_add=True)
actualizado_en = models.DateTimeField(auto_now=True)

def __str__(self):
    return f'Publicación de {self.autor} ({self.tipo})'

3.2. ReporteContenido

class ReporteContenido(models.Model):
    """
    Reporte de una publicación por parte de un usuario.
    Se utiliza para moderación: insultos, acoso, contenido inapropiado,
etc.
    """

ESTADO_CHOICES = [
    ('PENDIENTE', 'Pendiente'),
    ('REVISADO', 'Revisado'),
    ('DESCARTADO', 'Descartado'),
    ('ACCION_TOMADA', 'Acción tomada'),
]

MOTIVO_CHOICES = [
    ('INSULTO', 'Insulto / lenguaje ofensivo'),
    ('ACOSO', 'Acoso / amenaza'),
    ('DISCRIMINACION', 'Discriminación'),
    ('SEXUAL', 'Contenido sexual'),
    ('VIOLENCIA', 'Violencia'),
    ('SPAM', 'Spam / engaño'),
    ('OTRO', 'Otro'),
]

publicacion = models.ForeignKey(
    Publicacion,
    on_delete=models.CASCADE,
    related_name='reportes'
)

autor_reporte = models.ForeignKey(
    Usuario,
    on_delete=models.CASCADE,
    related_name='reportes_realizados'
)

motivo = models.CharField(
    max_length=30,
    choices=MOTIVO_CHOICES
)

```

—  
PROF

```
comentario = models.TextField(
    blank=True,
    help_text='Comentario opcional sobre el reporte.'
)

estado = models.CharField(
    max_length=20,
    choices=ESTADO_CHOICES,
    default='PENDIENTE'
)

moderador = models.ForeignKey(
    Usuario,
    null=True,
    blank=True,
    on_delete=models.SET_NULL,
    related_name='reportes_moderados',
    help_text='Usuario que revisó el reporte.'
)

accion_tomada = models.TextField(
    blank=True,
    help_text='Descripción de la acción tomada por el moderador.'
)

creado_en = models.DateTimeField(auto_now_add=True)
actualizado_en = models.DateTimeField(auto_now=True)

class Meta:
    verbose_name = 'Reporte de contenido'
    verbose_name_plural = 'Reportes de contenido'

def __str__(self):
    return f'Reporte sobre {self.publicacion_id} por {self.autor_reporte_id}'
```

—  
PROF

### 3.3. Seguidor (para completar lo social)

```
class Seguidor(models.Model):
    """
    Relación de seguimiento entre jugadores (tipo red social).
    """
    seguidor = models.ForeignKey(
        PerfilJugador,
        on_delete=models.CASCADE,
        related_name='siguiendo'
    )
    seguido = models.ForeignKey(
        PerfilJugador,
        on_delete=models.CASCADE,
        related_name='seguidores'
    )
```

```
creado_en = models.DateTimeField(auto_now_add=True)

class Meta:
    unique_together = ('seguidor', 'seguido')

def __str__(self):
    return f'{self.seguidor} sigue a {self.seguido}'
```

## 💡 Notas de uso y lógica de negocio

Al crear una reserva:

Si usuario.es\_menor\_supervisado() es True:

estado inicial → PENDIENTE\_TUTOR.

notificar al tutor.

Si es adulto:

estado inicial → PENDIENTE o CONFIRMADA según reglas del complejo.

Al aprobar desde el panel del tutor:

llamar a reserva.aprobar\_por\_tutor(tutor).

Al recibir muchos reportes sobre una publicación:

marcar publicacion.bloqueada = True y guardar motivo\_bloqueo.

## ✓ Resumen

Estos modelos definen:

—  
PROF  
Usuarios con control de edad y tutor.

Jugadores con perfil deportivo.

Reservas con estados especiales para menores supervisados.

Publicaciones moderables y reportes de contenido.

Relaciones de seguimiento social.

Son una base sólida para implementar toda la capa de seguridad, moderación y control parental en DeRevés.

✖ Hecho por Mantis

Mantis Software Solutions

“Protegemos la experiencia deportiva, desde el diseño del modelo de datos.”

 Cruz Alta, Córdoba, Argentina  
 [www.mantistec.com.ar](http://www.mantistec.com.ar)