Estándares

Se exige usar todos los estándar para una claridad y un único estilo al código, incluyendo el uso del inglés en todo el apartado de código.

Estándares a utilizar:
PEP 8
Documentación de Django
Conventional Commits.

Implementaciones adicionales

Investigar e implementar el uso de:
```from django.db import transaction```

Esto evita la atomicidad y problemas concurrentes
en el registro de usuarios.

Ejemplo de su uso:
transacciones

Investigar cómo evitar la herencia de diamante y como configurar el modelo al archivo setting (consideración específica 1.)

Consideraciones específica

1. usuario → user Crítico la palabra user es exclusivo en django se recomienda usar YogaCenterUser (nombre predeterminado opcional se usará de referencia)

2. padre → rol Lógica en la base de datos y los modelos. (consultar diagrama de objeto)
3. Uso de snake_case para atributos del modelo y Cap_works para nombres del modelos (consultar documentación oficial de django)

Implementación técnica

El modelo UserCenterYoga debe implementar de AbstractUser
por lo que no es necesario especificar esto atributos:
fisrt_name, second_name,email,password
```

Considerar incluir el atributo con el modelo
```teléfono → **PhoneNumberField** `django-phonenumber-field` ```

UserCenterYoga →Yogi relacion de 1 a 1

url_profile_yogi → ImageField

Configurar bien imagefield

Prueba de testeo

abrir:
```python manage.py shell ```
primero importar las librería

```from tu_app.models import
#se debe importar los modelos del usuario, yogui e instructor
from django.db import IntegrityError
#esta para validacion ```

prueba número 1
```
# Definimos el usuario en una variable 'u'
u = YogaCenterUser.objects.create_user(
    username="prueba_manual",
    password="123",
    email="test@yoga.com",
    first_name="Juan",
    phone="+584121234567"
)

# Validamos que se creó:
print(u.id)
# Debería salir un número (ej: 1)

print(u.phone)
# Debería salir: +584121234567 ```

prueba número 2

```
# Creamos el Yogi vinculándolo a la variable 'u' que creamos arriba
y = Yogi.objects.create(
    user=u,
```

```
    urlprofile="foto_yogi.jpg"
)

# Validamos que existe:
print(y.id)

print(y.user.first_name)
# Debería responder: "Juan" ```
```

prueba número 3

```
i = Instructor.objects.create(
    user=u,
    rating=4.5,
    profile_photo="foto_profe.jpg"
)


# Validamos:
print(i.rating)
# Debería salir: 4.5 ```
```

prueba número 4

```
# Intentamos asignar el rol al usuario ID 9999 (que no existe)
try:
    Yogi.objects.create(user_id=9999, urlprofile="fantasma.jpg")
except IntegrityError:
    print("¡PRUEBA EXITOSA! La base de datos impidió crear un rol huérfano.")
except Exception as e:
    print(f"Ocurrió otro error: {e}") ```
```

—----------

Referencia:

diagrama de objeto y clases UML
https://app.diagrams.net/#G1YAHawV_4b5U76BS-9gmL8sMlVYVHPauZ#%7B%22pageId%22%3A%22U-iC_3jA6ggDQsA9Twkj%22%7D
Casos de usos
https://docs.google.com/document/d/1zZqVG-jTQpbP9ZQByzLtTL7_y9ByX7rc5bEmPENYOZQ/edit?tab=t.0

transacciones
ttps://docs.djangoproject.com/en/6.0/topics/db/transactions/

imagefield
https://www.geeksforgeeks.org/python/imagefield-django-models/

PEP 8
https://peps.python.org/pep-0008/
Documentación oficial de Django
https://docs.djangoproject.com/es/6.0/

convención de commit
https://www.conventionalcommits.org/en/v1.0.0/