Manual: Actualizar GitHub con mis cambios locales (Windows + Python/Django)

# 1. Configuración de Git (una sola vez en tu PC)

Ejecutar en Git Bash:

git config --global core.autocrlf input

# 2. Crear .gitignore (una sola vez en tu proyecto)

En la raíz del proyecto (c:/appTurnos/.gitignore) pega esto:

# Python  
\_\_pycache\_\_/  
\*.py[cod]  
\*.pyo  
  
# Entornos virtuales  
.venv/  
venv/  
env/  
.env  
  
# Archivos temporales / logs  
\*.log  
\*.sqlite3  
\*.db  
  
# VSCode  
.vscode/  
  
# Sistema  
.DS\_Store  
Thumbs.db  
  
# Test/coverage  
.pytest\_cache/  
.mypy\_cache/  
.coverage  
htmlcov/

# 3. (Opcional) Crear .gitattributes

En la raíz del proyecto (c:/appTurnos/.gitattributes):

# Normalizar saltos de línea  
\* text=auto  
  
# Python siempre en LF  
\*.py text eol=lf  
  
# Archivos web en LF  
\*.html text eol=lf  
\*.css text eol=lf  
\*.js text eol=lf  
  
# Configuración / docs en LF  
\*.json text eol=lf  
\*.yml text eol=lf  
\*.yaml text eol=lf  
\*.md text eol=lf

# 4. Flujo cada vez que quieras subir tus cambios locales a GitHub (forzando)

⚠️ Esto sobrescribe GitHub con tu versión local. Los commits que haya en GitHub se perderán, salvo que hagas un respaldo antes.

1. Ir al proyecto  
 cd /c/appTurnos  
  
2. Verificar rama  
 git branch -vv  
  
3. (Opcional) Respaldo de la rama remota antes de sobrescribirla  
 git fetch origin  
 git push origin origin/main:backup-origin-main  
  
4. Asegurar .gitignore y .gitattributes en el repo  
 git add .gitignore .gitattributes  
 git commit -m "Agrega .gitignore y .gitattributes" || echo "Ya estaban agregados"  
  
5. Añadir tus cambios locales y commitear  
 git add -A  
 git commit -m "Actualiza proyecto con cambios locales" || echo "Sin cambios que commitear"  
  
6. Forzar push (subir tu versión local a GitHub)  
 git push origin main --force-with-lease

# 5. Para actualizaciones normales (sin sobrescribir)

Si no hay divergencias con GitHub y solo quieres subir cambios:

git add -A  
git commit -m "Mensaje de los cambios"  
git push origin main

# Resultado

- Tu repo en GitHub queda exactamente igual a tu versión local.  
- No se suben \_\_pycache\_\_, .pyc, .env, etc.  
- Todos los archivos quedan normalizados con saltos de línea LF (estándar en Python).