MyGest

Guida Operativa per Deployment e Sincronizzazione

Versione 1.0 - Febbraio 2026

MyGest - Guida Operativa per Deployment e Sincronizzazione

Versione: 1.0
Data: Febbraio 2026
Autore: MyGest Development Team

Indice

1. Introduzione

Questa guida fornisce istruzioni complete per:

- **Deployment** dell'applicazione MyGest sul server di produzione (VPS)
- **Sincronizzazione** dei dati tra ambiente di sviluppo e produzione
- **Gestione** dell'archivio documentale su NAS locale

La guida è pensata per utenti con conoscenze di base di Linux e Git.

1.1 Architettura del Sistema

1.2 Flussi Operativi

Sviluppo → Produzione:

1. Sviluppo in locale con dati di test

2. Commit e push su GitHub (branch `main`)
3. Deploy automatico via GitHub Actions
4. Sincronizzazione database dev → prod (se necessario)

Produzione → Sviluppo:
1. Dati reali inseriti in produzione
2. Sync database prod → dev per testing locale
3. Sviluppo nuove feature con dati reali

---

2. Prerequisiti

2.1 Software Necessario

Su PC Locale (Ambiente Sviluppo)

Sistema Operativo: Linux (Ubuntu/Debian consigliato)

Software richiesto:

```
# Python 3.10+
python3 --version  # Deve essere ≥ 3.10

# PostgreSQL
sudo apt install postgresql postgresql-contrib

# Git
sudo apt install git

# Node.js (per frontend)
curl -fsSL https://deb.nodesource.com/setup_20.x | sudo -E bash -
sudo apt install nodejs

# rclone (per backup NAS)
sudo apt install rclone

# WeasyPrint (per generazione PDF, opzionale)
sudo apt install python3-weasyprint
```

Installazione MyGest:

```
# Clone repository
git clone https://github.com/sandro6917/mygest.git
cd mygest

# Virtual environment
python3 -m venv venv
source venv/bin/activate

# Dipendenze Python
pip install -r requirements.txt

# Frontend
cd frontend
npm install
cd ..

# Database setup
createdb mygest_dev
python manage.py migrate

# Superuser
python manage.py createsuperuser
```

Su Server Produzione (VPS)

Accesso: VPS Hostinger `mygest@72.62.34.249`

Software già configurato:
- Python 3.10
- PostgreSQL 14
- Nginx
- Gunicorn
- systemd (per servizi)

2.2 Configurazione SSH

Generare chiave SSH (se non presente):

```
# Su PC locale
ssh-keygen -t ed25519 -C "tuo@email.com" -f ~/.ssh/mygest_deploy

# Copiare chiave pubblica sul VPS
ssh-copy-id -i ~/.ssh/mygest_deploy.pub mygest@72.62.34.249
```

Configurare ~/.ssh/config:

```
# Creare/editare file
nano ~/.ssh/config

# Aggiungere:
Host mygest-vps
    HostName 72.62.34.249
    User mygest
    IdentityFile ~/.ssh/mygest_deploy
    ServerAliveInterval 60
```

Test connessione:

```
ssh mygest-vps
# Dovrebbe connettersi senza password

# Verificare path applicazione
ls /srv/mygest/app
# Output: manage.py, mygest/, frontend/, ...

exit
```

2.3 Configurazione Git

Credenziali GitHub:

```
# Configurare nome e email
git config --global user.name "Tuo Nome"
git config --global user.email "tuo@email.com"

# Personal Access Token (PAT) per push
# 1. Vai su GitHub > Settings > Developer settings > Personal access tokens
# 2. Generate new token (classic)
# 3. Seleziona scopes: repo, workflow
# 4. Copia il token generato

# Salvare credenziali
git config --global credential.helper store

# Al primo push ti chiederà:
# Username: tuo_username_github
# Password: [incolla il PAT generato]
```

2.4 Variabili d'Ambiente

File `.env` locale (già presente in repo, verificare):

```
# /home/sandro/mygest/.env
DEBUG=True
SECRET_KEY=<chiave-segreta-locale>
DATABASE_URL=postgresql://user:pass@localhost/mygest_dev
ALLOWED_HOSTS=localhost,127.0.0.1
ARCHIVIO_BASE_PATH=/mnt/nas_mygest
```

File `.env` produzione (su VPS):

```
# /srv/mygest/app/.env
DEBUG=False
SECRET_KEY=<chiave-segreta-produzione>
DATABASE_URL=postgresql://mygest:password@localhost/mygest
ALLOWED_HOSTS=mygest.sandro.cloud,72.62.34.249
ARCHIVIO_BASE_PATH=/mnt/nas_mygest
```

⚠ **IMPORTANTE**: Non committare mai file `.env` su Git! È già in `.gitignore`.

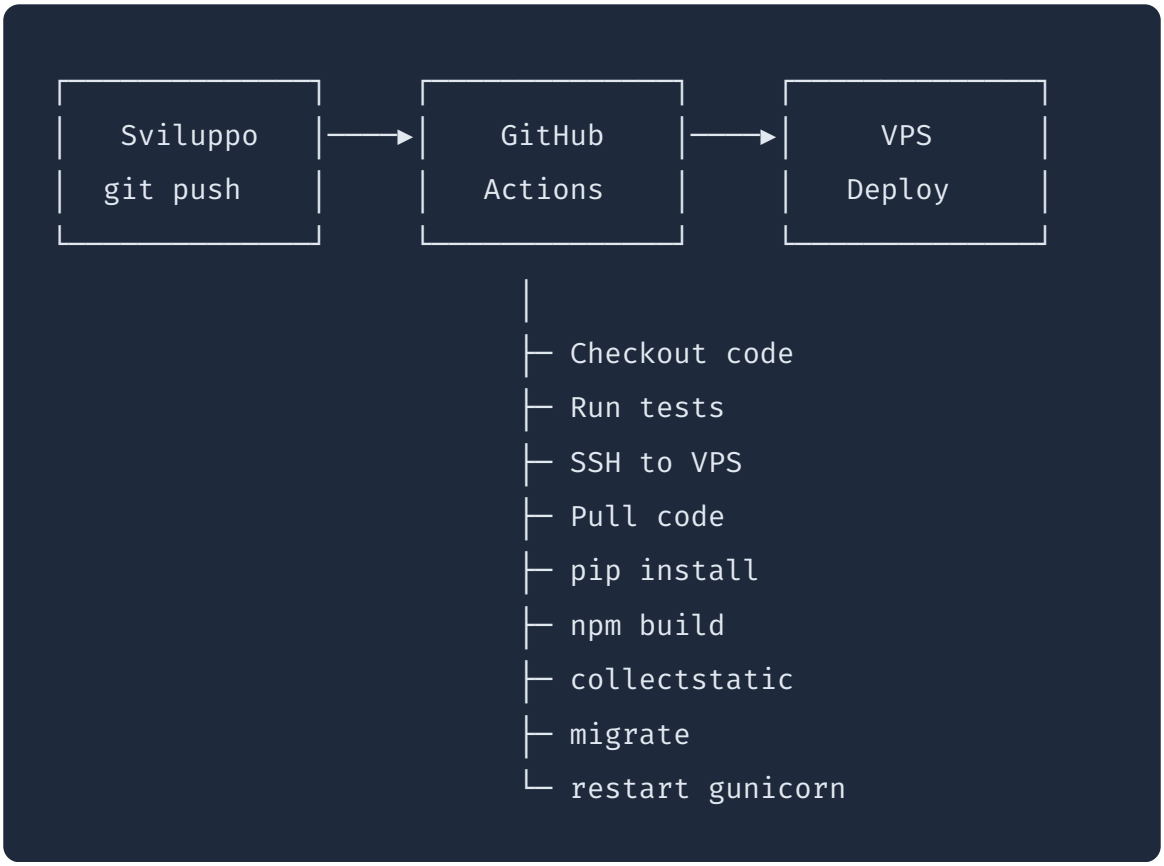
---

3. Deployment Applicazione

3.1 Deployment Automatico (GitHub Actions)

Il metodo **consigliato** per deployment è GitHub Actions.

3.1.1 Come Funziona



3.1.2 Procedura Step-by-Step

Passo 1: Sviluppare in locale

```
cd /home/sandro/mygest

# Attivare virtualenv
source venv/bin/activate

# Creare/modificare codice
#  ...  modifiche ai file  ...

# Test locale
python manage.py test
pytest

# Test server dev
python manage.py runserver
# Verificare http://localhost:8000
```

Passo 2: Commit e Push

```
# Verificare modifiche
git status
git diff

# Aggiungere file modificati
git add .

# Commit con messaggio descrittivo
git commit -m "feat: aggiunta nuova funzionalità X"

# Push su GitHub (branch main)
git push origin main
```

Passo 3: Monitorare GitHub Actions

```
# Aprire browser su:
# https://github.com/sandro6917/mygest/actions

# Verrai reindirizzato automaticamente all'ultimo workflow
# Osservare i vari step:
# - Checkout code ✓
# - Run tests ✓
# - Deploy to VPS ✓

# Tempo medio: 3-5 minuti
```

Passo 4: Verificare Deploy

```
# Aprire browser su:
https://mygest.sandro.cloud

# Verificare:
# ✓ Applicazione risponde
# ✓ Login funzionante
# ✓ Nuove modifiche visibili

# Log server (se necessario)
ssh mygest-vps
sudo journalctl -u mygest -f
```

3.1.3 Cosa Fa GitHub Actions

Il workflow definito in `.github/workflows/deploy.yml`:

1. **Checkout**: Scarica ultimo codice da GitHub
2. **Tests**: Esegue test suite (`pytest`)

3. **SSH Deploy**: Si connette al VPS e esegue:
   `bash cd /srv/mygest/app git pull origin main source /srv/mygest/venv/bin/activate pip install -r requirements.txt cd frontend && npm install && npm run build && cd .. python manage.py collectstatic --noinput python manage.py migrate --noinput sudo systemctl restart mygest`

3.1.4 Gestione Errori Deploy Automatico

Se GitHub Actions fallisce:

```
# 1. Controllare log su GitHub
# https://github.com/sandro6917/mygest/actions
# Clicca sul workflow fallito → Espandi step con errore

# 2. Errori comuni:

# a) Test failure
# Soluzione: Fixare test in locale, riprovare
pytest
git add -A
git commit -m "fix: correggi test"
git push origin main

# b) SSH connection failed
# Soluzione: Verificare chiave SSH su GitHub Secrets
# Settings > Secrets > Actions > SSH_PRIVATE_KEY

# c) Migration error
# Soluzione: Deploy manuale (vedi sezione successiva)

# 3. Forzare re-run workflow
# Su GitHub Actions > Workflow fallito > Re-run failed jobs
```

3.2 Deployment Manuale (SSH)

Quando GitHub Actions non è disponibile o serve controllo completo.

3.2.1 Script di Deploy Automatico

Uso dello script `scripts/deploy.sh` :

```
# Su PC locale, nella directory mygest
cd /home/sandro/mygest

# Eseguire script (richiede password sudo VPS)
./scripts/deploy.sh

# Output:
#   MyGest Deployment Script
# ═══════════════════════════════
#
#   Connecting to VPS...
# ✓ Connected to mygest@72.62.34.249
#
#   Pulling latest code...
# ✓ Code updated
#
#   Installing dependencies...
# ✓ Python packages installed
# ✓ NPM packages installed
#
#    Building frontend...
# ✓ Frontend built
#
#   Collecting static files...
# ✓ Static files collected
#
#    Running migrations...
# ✓ Migrations applied
#
#   Restarting services...
# ✓ Gunicorn restarted
#
#   Deployment completed successfully!
```

Cosa fa lo script:
- Si connette al VPS via SSH
- Esegue `git pull` per aggiornare codice
- Installa dipendenze Python e NPM
- Builda frontend React
- Raccoglie static files
- Applica migrations database
- Riavvia Gunicorn

3.2.2 Deployment Manuale Passo-Passo

Se lo script automatico fallisce o serve debugging:

```
# 1. Connessione SSH al VPS
ssh mygest-vps

# 2. Navigare nella directory applicazione
cd /srv/mygest/app

# 3. Verificare branch Git
git branch
# Output: * main

git status
# Deve essere clean, altrimenti:
git stash  # Salva modifiche locali

# 4. Pull codice aggiornato
git pull origin main
# Output: Updating abc1234..def5678

# 5. Attivare virtualenv
source /srv/mygest/venv/bin/activate

# 6. Aggiornare dipendenze Python
pip install -r requirements.txt --upgrade

# 7. Frontend: installare dipendenze e build
cd frontend
npm install
npm run build
cd ..

# 8. Collect static files
python manage.py collectstatic --noinput
# Output: X static files copied
```

```
# 9. Verificare migrations
python manage.py showmigrations
# Verificare che tutte abbiano [X]

# 10. Applicare migrations (se necessario)
python manage.py migrate
# Output: Operations to perform:  ...

# 11. Riavviare Gunicorn
sudo systemctl restart mygest

# 12. Verificare stato servizio
sudo systemctl status mygest
# Output: ● mygest.service - MyGest Gunicorn daemon
#         Active: active (running) since  ...

# 13. Verificare log real-time
sudo journalctl -u mygest -f
# Premi Ctrl+C per uscire

# 14. Test applicazione
curl http://localhost:8000/api/v1/health/
# Output: {"status":"ok", ... }

# 15. Exit SSH
exit
```

3.2.3 Verifiche Post-Deploy

Checklist di verifica:

```
# ✓ 1. Applicazione risponde
curl -I https://mygest.sandro.cloud
# Output: HTTP/2 200 OK

# ✓ 2. Login funziona
# Aprire browser: https://mygest.sandro.cloud/login
# Inserire credenziali admin

# ✓ 3. API rispondono
curl https://mygest.sandro.cloud/api/v1/health/
# Output: {"status":"ok","database":"ok","redis":"ok"}

# ✓ 4. Static files caricati
curl -I https://mygest.sandro.cloud/static/admin/css/base.css
# Output: HTTP/2 200 OK

# ✓ 5. Frontend React funziona
# Aprire browser: https://mygest.sandro.cloud
# Verificare che React app si carichi (non errori console)

# ✓ 6. Database migrations applicate
ssh mygest-vps
cd /srv/mygest/app
source /srv/mygest/venv/bin/activate
python manage.py showmigrations | grep '\[ \]'
# Output vuoto = tutte applicate

# ✓ 7. Log senza errori
sudo journalctl -u mygest --since "5 minutes ago" | grep ERROR
# Output vuoto = nessun errore recente
```

3.3 Rollback del Deploy

Se il deploy causa problemi, tornare alla versione precedente.

3.3.1 Rollback Git

```
# 1. Connessione SSH
ssh mygest-vps
cd /srv/mygest/app

# 2. Vedere commit recenti
git log --oneline -10
# Output:
# abc1234 (HEAD → main, origin/main) feat: nuova funzione
# def5678 fix: correzione bug
# ghi9012 feat: altra modifica

# 3. Identificare commit precedente (es: def5678)
# Vedere dettagli:
git show def5678

# 4. Rollback a commit precedente
git reset --hard def5678
# ATTENZIONE: Questo elimina modifiche non committate!

# 5. Reinstallare dipendenze (potrebbero essere cambiate)
source /srv/mygest/venv/bin/activate
pip install -r requirements.txt

# 6. Rebuild frontend
cd frontend
npm install
npm run build
cd ..

# 7. Collectstatic
python manage.py collectstatic --noinput

# 8. Rollback migrations (SE NECESSARIO)
# Solo se il commit aveva migrations che causano problemi
```

```
python manage.py showmigrations app_name
# Identificare ultima migration da mantenere (es: 0008)
python manage.py migrate app_name 0008

# 9. Riavviare servizio
sudo systemctl restart mygest

# 10. Verificare
curl http://localhost:8000/api/v1/health/
sudo systemctl status mygest
```

3.3.2 Rollback Database

Se le migrations causano problemi critici:

```
# 1. BACKUP IMMEDIATO (prima di rollback)
ssh mygest-vps
cd /srv/mygest/backups
pg_dump -U mygest -F c mygest > pre_rollback_$(date +%Y%m%d_%H%M%S).dump

# 2. Elencare backup disponibili
ls -lh /srv/mygest/backups/*.dump
# Scegliere backup prima del deploy problematico

# 3. ATTENZIONE: Questo sovrascrive il database!
# Confermare che è l'azione corretta

# 4. Restore database
sudo systemctl stop mygest
pg_restore -U mygest -d mygest --clean --if-exists /srv/mygest/backups/backup

# 5. Riavviare
sudo systemctl start mygest

# 6. Verificare dati
psql -U mygest -d mygest -c "SELECT COUNT(*) FROM anagrafiche_anagrafica;"
```

3.3.3 Procedure di Emergenza

Sistema completamente non funzionante:

```
# PIANO A: Rollback completo
ssh mygest-vps
cd /srv/mygest/app

# 1. Git rollback (vedi sopra)
git reset --hard <commit_precedente_funzionante>

# 2. Database restore (vedi sopra)
pg_restore -U mygest -d mygest --clean /srv/mygest/backups/ultimo_backup.dump

# 3. Rebuild tutto
source /srv/mygest/venv/bin/activate
pip install -r requirements.txt
cd frontend && npm install && npm run build && cd ..
python manage.py collectstatic --noinput
python manage.py migrate

# 4. Restart
sudo systemctl restart mygest nginx

# PIANO B: Ripristino da snapshot VPS
# Se hai snapshot VPS su Hostinger:
# 1. Login Hostinger panel
# 2. VPS > Snapshots
# 3. Restore snapshot pre-deploy
# Tempo: 10-30 minuti
```

---

4. Sincronizzazione Database

4.1 Overview e Casi d'Uso

Quando sincronizzare:

Scenario	Direzione	Comando
Testare con dati reali	prod → dev	`./scripts/sync_prod_to_dev.sh`
Deploy dati di setup	dev → prod	`./scripts/sync_dev_to_prod.sh`
Backup pre-modifica	prod → backup	`pg_dump`
Ripristino emergenza	backup → prod	`pg_restore`

⚠ **ATTENZIONE**: La sincronizzazione sovrascrive dati! Backup sempre prima.

4.2 Sincronizzazione Dev → Prod

Quando usare:
- Setup iniziale produzione con dati di configurazione
- Importare anagrafiche/clienti da sviluppo
- Aggiornare nomenclatori/tabelle di sistema

4.2.1 Sync Merge (Consigliato)

Merge = Inserisce nuovi record + aggiorna esistenti (match su PK)

```
# Su PC locale
cd /home/sandro/mygest

# Dry-run (simulazione, non modifica)
./scripts/sync_dev_to_prod.sh --dry-run

# Output:
#   MyGest Database Sync: DEV → PROD
# ═══════════════════════════════════════
#
#   Exporting from DEV...
# ✓ Exported 1886 records to fixtures/sync/dev_export_20260221_143022.json
#
#   Uploading to VPS...
# ✓ Uploaded (245 KB)
#
#   DRY RUN - Simulating import...
#
# Model                  | Create | Update | Delete | Skip  | Errors
# ──────────────────────────────────────────────────────────────────
# anagrafiche.Anagrafica |   15   |   230  |   0    |   0   |   0
# anagrafiche.Cliente    |    5   |    96  |   0    |   0   |   0
# documenti.Documento    |   87   |   413  |   0    |   0   |   0
# fascicoli.Fascicolo    |   12   |   349  |   0    |   0   |   0
# pratiche.Pratica       |    3   |    94  |   0    |   0   |   0
# ──────────────────────────────────────────────────────────────────
# TOTALS                 |  122   |  1182  |   0    |   0   |   0
#
# ⚠  DRY RUN: No changes applied to database
#
# To apply changes, run without --dry-run

# Se tutto OK, eseguire sync reale
./scripts/sync_dev_to_prod.sh
```

```
# Confermare quando richiesto:
# ⚠  This will modify production database!
# Continue? (yes/no): yes

# Output:
#   Sync completed successfully!
# ✓ 122 records created
# ✓ 1182 records updated
# ✓ Backup saved: /srv/mygest/backups/sync/pre_sync_20260221_143045.sql
```

4.2.2 Sync Full Replace (Uso Avanzato)

Full Replace = Elimina tutti i dati esistenti e ricrea da dev

⚠ **MOLTO PERICOLOSO** - Usare solo se:
- Database produzione vuoto o corrotto
- Reset completo necessario
- Hai backup verificato

```
# SEMPRE dry-run prima!
./scripts/sync_dev_to_prod.sh --full-replace --dry-run

# Leggere attentamente output:
# Model                  | Create | Update | Delete | Skip  | Errors
# ──────────────────────────────────────────────────────────────────
# anagrafiche.Anagrafica |  245   |    0   |  230   |   0   |   0
# documenti.Documento    |  500   |    0   |  489   |   0   |   0
#                           ^^^^             ^^^^
#                       Crea tutti      Elimina tutti esistenti!

# Se sicuro, eseguire:
./scripts/sync_dev_to_prod.sh --full-replace --force

# Richiede doppia conferma:
# ⚠  FULL REPLACE MODE - Will delete existing data!
# Type 'DELETE ALL DATA' to confirm: DELETE ALL DATA
# Are you absolutely sure? (yes/no): yes
```

4.2.3 Sync Selettivo per Model

Sincronizzare solo alcuni modelli:

```
# Solo anagrafiche
./scripts/sync_dev_to_prod.sh --models anagrafiche

# Output:
#   Exporting only: anagrafiche.Anagrafica, anagrafiche.Cliente, ...
# ✓ Exported 346 records

# Più modelli
./scripts/sync_dev_to_prod.sh --models anagrafiche,fascicoli

# Vedere modelli disponibili:
python manage.py export_data --help | grep "Available models"
```

4.3 Sincronizzazione Prod → Dev

Quando usare:
- Testare nuove feature con dati reali
- Debugging problemi segnalati in produzione
- Sviluppo con dataset aggiornato

4.3.1 Procedura Standard

```
# Su PC locale
cd /home/sandro/mygest

# Dry-run
./scripts/sync_prod_to_dev.sh --dry-run

# Output simile a dev→prod ma in direzione inversa
# ⚠  Mostrerà quanti record locali saranno modificati

# Eseguire sync
./scripts/sync_prod_to_dev.sh

# Confermare:
# ⚠  This will modify local development database!
# Continue? (yes/no): yes

# Verifica locale dopo sync
source venv/bin/activate
python manage.py shell

>>> from anagrafiche.models import Cliente
>>> Cliente.objects.count()
101  # Numero da produzione

>>> from documenti.models import Documento
>>> Documento.objects.count()
500  # Allineato con prod
```

4.3.2 Sync con Anonimizzazione (Privacy)

Per testare senza dati sensibili:

```
# TODO: Feature da implementare
# Permetterebbe di:
# - Anonimizzare nomi/CF/P.IVA
# - Mantenere struttura dati
# - Offuscare documenti sensibili

# Workaround attuale:
# 1. Sync normale
./scripts/sync_prod_to_dev.sh

# 2. Script Python per anonimizzare
python manage.py shell
>>> from anagrafiche.models import Anagrafica
>>> for a in Anagrafica.objects.all():
...     if a.tipo == 'PF':
...         a.nome = f"Cliente {a.pk}"
...         a.cognome = "Test"
...         a.codice_fiscale = "TSTCLI80A01H501X"
...     a.save()
```

4.4 Rollback Database

Se la sincronizzazione causa problemi.

4.4.1 Rollback Automatico (Script)

```
# Su PC locale (per rollback DEV)
cd /home/sandro/mygest
./scripts/rollback_last_sync.sh

# Output:
#   MyGest Database Sync Rollback
# ═══════════════════════════════════
#
#   Available backups:
# 1. pre_sync_20260221_143045.sql  (245 MB) - 10 minutes ago
# 2. pre_sync_20260220_162222.sql  (242 MB) - 1 day ago
# 3. pre_sync_20260215_091530.sql  (238 MB) - 6 days ago
#
# Select backup number (or 'q' to quit): 1
#
# ⚠  This will restore database to state before last sync!
# Current data will be LOST!
# Continue? (yes/no): yes
#
#   Restoring backup...
# ✓ Database restored successfully!

# Su VPS (per rollback PROD)
ssh mygest-vps
cd /srv/mygest/app
./scripts/rollback_last_sync.sh
# Procedura identica
```

4.4.2 Rollback Manuale

```
# Esempio su DEV (stesso procedimento su PROD)

# 1. Vedere backup disponibili
ls -lh /home/sandro/mygest/backups/sync/
# -rw-r--r-- 1 sandro sandro 245M Feb 21 14:30 pre_sync_20260221_143045.sql

# 2. Stopare applicazione (se in uso)
# Su DEV non necessario, su PROD:
ssh mygest-vps
sudo systemctl stop mygest

# 3. Drop database attuale
dropdb mygest_dev  # Su DEV
# dropdb mygest     # Su PROD

# 4. Ricreare database vuoto
createdb mygest_dev  # Su DEV
# createdb mygest     # Su PROD

# 5. Restore da backup
pg_restore -U sandro -d mygest_dev /home/sandro/mygest/backups/sync/pre_sync_

# Su PROD:
# pg_restore -U mygest -d mygest /srv/mygest/backups/sync/pre_sync_20260221_1

# 6. Verificare
psql mygest_dev -c "SELECT COUNT(*) FROM anagrafiche_anagrafica;"

# 7. Riavviare (su PROD)
# sudo systemctl start mygest
```

4.5 Best Practices Sincronizzazione

4.5.1 Checklist Pre-Sync

```
☐ Backup database destinazione eseguito
☐ Dry-run verificato e output confermato
☐ Nessun utente sta lavorando (per prod→dev)
☐ Applicazione in manutenzione (per sync prod)
☐ Spazio disco sufficiente (backup + export)
☐ Connessione internet stabile
```

4.5.2 Quando NON Sincronizzare

MAI sincronizzare se:
- Utenti stanno lavorando in produzione (sync dev→prod)
- Ci sono migrations non applicate su destinazione
- Database destinazione ha schema diverso
- Non hai backup verificato
- Durante orario di punta (8-18)

Orari consigliati:
- Sync dev→prod: Sera (19-22) o weekend
- Sync prod→dev: Qualsiasi (solo tu usi DEV)

4.5.3 Retention Policy Backup

```
# I backup sync sono in:
# DEV: /home/sandro/mygest/backups/sync/
# PROD: /srv/mygest/backups/sync/

# Retention automatica (ultimi 10 backup):
# - Script mantiene automaticamente 10 backup più recenti
# - Backup più vecchi eliminati automaticamente

# Backup manuali importanti:
# Creare in /backups/manual/ per conservazione permanente
pg_dump -U mygest -F c mygest > /srv/mygest/backups/manual/pre_upgrade_v2.0.0
```

---

5. Gestione Archivio NAS

5.1 Struttura Directory

L'archivio documenti è su NAS locale con backup cloud.

5.1.1 Path e Mount Point

```
# Mount point NAS
/mnt/nas_mygest/

# Struttura directory:
/mnt/nas_mygest/
├── clienti/                    # Documenti per cliente
│   ├── CLI001_ROSSI_MARIO/
│   │   ├── 01_CONTABILITA/
│   │   │   ├── 2024/
│   │   │   │   ├── FAT-2024-001.pdf
│   │   │   │   └── FAT-2024-002.pdf
│   │   │   └── 2025/
│   │   ├── 02_FISCALE/
│   │   ├── 03_LAVORO/
│   │   └── ...
│   └── CLI002_BIANCHI_SRL/
├── archivio_fisico/            # Ubicazioni fisiche
│   ├── UF001_UFFICIO_PRINCIPALE/
│   ├── UF002_ARCHIVIO_SEMINTERRATO/
│   └── ...
└── temp/                       # Upload temporanei
    └── pending_uploads/
```

5.1.2 Convenzioni Naming

Clienti: `CLI{codice}_{cognome}_{nome}` oppure `CLI{codice}_{ragione_sociale}`

```
CLI001_ROSSI_MARIO
CLI042_ACME_SRL
```

Titolario: `{codice}_{descrizione}`

```
01_CONTABILITA
02_FISCALE
03_LAVORO
04_SOCIETARIO
```

Documenti: `{tipo}-{anno}-{sequenza}-{descrizione}.{ext}`

```
FAT-2024-001-Fattura_fornitore_X.pdf
CEDU-2024-12-Dicembre_2024.pdf
UNILAV-2025-003-Assunzione_Rossi.pdf
```

5.2 Configurazione NAS

5.2.1 Mount NAS su Sistemi Linux

Verifica mount attuale:

```
mount | grep nas_mygest
# Output: //192.168.1.100/mygest on /mnt/nas_mygest type cifs (rw, ... )

df -h | grep nas_mygest
# Output: //192.168.1.100/mygest  2.0T  1.2T  800G  61% /mnt/nas_mygest
```

Se non montato, mount manuale:

```
# Creare mount point (se non esiste)
sudo mkdir -p /mnt/nas_mygest

# Mount (CIFS/SMB)
sudo mount -t cifs //192.168.1.100/mygest /mnt/nas_mygest \
  -o username=mygest,password=password,uid=1000,gid=1000,file_mode=0664,dir_m

# Verificare accesso
ls -la /mnt/nas_mygest
touch /mnt/nas_mygest/test.txt
rm /mnt/nas_mygest/test.txt
```

Mount automatico al boot:

```
# Creare file credenziali
sudo nano /root/.nascreds

# Contenuto:
username=mygest
password=password_nas
domain=WORKGROUP

# Proteggere file
sudo chmod 600 /root/.nascreds

# Editare /etc/fstab
sudo nano /etc/fstab

# Aggiungere riga:
//192.168.1.100/mygest /mnt/nas_mygest cifs credentials=/root/.nascreds,uid=1

# Testare mount senza reboot
sudo mount -a

# Verificare
df -h | grep nas_mygest
```

5.2.2 Permessi e Ownership

Permessi corretti per MyGest:

```
# User/group che esegue Django (su VPS)
id mygest
# uid=1001(mygest) gid=1001(mygest)

# Verificare permessi directory
ls -ld /mnt/nas_mygest
# drwxrwxr-x 5 mygest mygest 4096 Feb 21 14:00 /mnt/nas_mygest

# Correggere se necessario
sudo chown -R mygest:mygest /mnt/nas_mygest
sudo chmod -R u+rwX,g+rwX,o+rX /mnt/nas_mygest

# Test scrittura
sudo -u mygest touch /mnt/nas_mygest/test_permissions.txt
# Deve avere successo
sudo -u mygest rm /mnt/nas_mygest/test_permissions.txt
```

Verifica permessi da Django:

```
# Su shell Django
python manage.py shell

>>> import os
>>> nas_path = '/mnt/nas_mygest'
>>>
>>> # Test lettura
>>> os.listdir(nas_path)
['clienti', 'archivio_fisico', 'temp']
>>>
>>> # Test scrittura
>>> test_file = os.path.join(nas_path, 'test_django.txt')
>>> with open(test_file, 'w') as f:
...     f.write('Test Django write access')
>>>
>>> # Test cancellazione
>>> os.remove(test_file)
>>>
>>> # Success = permessi OK
```

5.3 Backup Automatici con rclone

rclone sincronizza NAS → Google Drive per backup cloud.

5.3.1 Configurazione rclone

Verifica configurazione esistente:

```
# Su VPS
ssh mygest-vps

# Verificare config rclone
cat ~/.config/rclone/rclone.conf

# Output:
# [gdrive_mygest]
# type = drive
# client_id = ...
# client_secret = ...
# token = {"access_token":"...","token_type":"Bearer",...}
# root_folder_id = 1A2B3C4D5E6F7G8H9I0J

# Testare accesso Google Drive
rclone lsd gdrive_mygest:
# Output:
# 0 2026-02-21 10:30:00        -1 MyGest_Backup
```

Se non configurato, setup iniziale:

```
# Installare rclone (se non presente)
sudo apt install rclone

# Configurazione interattiva
rclone config

# Seguire wizard:
# n) New remote
# name> gdrive_mygest
# Storage> drive (Google Drive)
# client_id> (premere Enter per default o inserire)
# client_secret> (premere Enter per default o inserire)
# scope> drive (accesso completo)
# root_folder_id> (lasciare vuoto o inserire folder ID)
# service_account_file> (lasciare vuoto)
# Edit advanced config? n
# Use auto config? n (su server senza GUI)

# Seguire URL per autorizzazione:
# 1. Copiare URL mostrato
# 2. Aprire in browser
# 3. Autorizzare accesso Google Drive
# 4. Copiare codice autorizzazione
# 5. Incollare nel terminale

# Confermare e salvare config
```

5.3.2 Script Backup Automatico

Backup manuale test:

```
# Sync dry-run (simulazione)
rclone sync /mnt/nas_mygest/ gdrive_mygest:MyGest_Backup/ \
  --dry-run \
  --progress \
  --exclude "temp/**" \
  --exclude ".DS_Store" \
  --log-file=/var/log/rclone_sync.log

# Output:
# 2024/02/21 15:00:00 NOTICE: MyGest_Backup/: 0 differences, 0 copies, 0 dele
# 2024/02/21 15:00:01 NOTICE: Transferred: 0 B / 0 B, -, 0 B/s, ETA -

# Sync reale
rclone sync /mnt/nas_mygest/ gdrive_mygest:MyGest_Backup/ \
  --progress \
  --exclude "temp/**" \
  --log-file=/var/log/rclone_sync.log

# Progress bar:
# Transferred:        1.234 GiB / 1.234 GiB, 100%, 5.123 MiB/s, ETA 0s
# Checks:         1234 / 1234, 100%
# Elapsed time:   4m30s
```

Cron job per backup automatico:

```
# Su VPS
ssh mygest-vps

# Creare script backup
sudo nano /usr/local/bin/mygest_backup.sh

# Contenuto:
#!/bin/bash
# MyGest NAS → Google Drive Backup Script

LOG_FILE="/var/log/mygest_backup.log"
DATE=$(date +"%Y-%m-%d %H:%M:%S")

echo "[$DATE] Starting backup..." >> $LOG_FILE

rclone sync /mnt/nas_mygest/ gdrive_mygest:MyGest_Backup/ \
  --exclude "temp/**" \
  --exclude ".DS_Store" \
  --log-file=$LOG_FILE \
  --log-level INFO \
  --stats 1m

RESULT=$?

if [ $RESULT -eq 0 ]; then
  echo "[$DATE] Backup completed successfully" >> $LOG_FILE
else
  echo "[$DATE] Backup failed with error code $RESULT" >> $LOG_FILE
fi

# Rendere eseguibile
sudo chmod +x /usr/local/bin/mygest_backup.sh

# Testare script
```

```
sudo /usr/local/bin/mygest_backup.sh

# Configurare cron job (esecuzione notturna)
sudo crontab -e

# Aggiungere:
# Backup NAS ogni notte alle 2:00 AM
0 2 * * * /usr/local/bin/mygest_backup.sh

# Verificare cron job
sudo crontab -l
```

5.3.3 Monitoraggio Backup

Verificare log backup:

```
# Ultimi backup
tail -50 /var/log/mygest_backup.log

# Errori recenti
grep ERROR /var/log/mygest_backup.log | tail -20

# Statistiche ultimo backup
grep "Transferred" /var/log/mygest_backup.log | tail -1
# Output: Transferred:        1.234 GiB / 1.234 GiB, 100%
```

Verificare spazio Google Drive:

```
rclone about gdrive_mygest:

# Output:
# Total:   15.0 GiB
# Used:     8.5 GiB
# Free:     6.5 GiB
# Trashed:  0.1 GiB
```

Alert email su errori (opzionale):

```
# Modificare script backup per inviare email su errore
sudo nano /usr/local/bin/mygest_backup.sh

# Aggiungere dopo RESULT=$?:
if [ $RESULT -ne 0 ]; then
  echo "MyGest backup failed on $(hostname) at $(date)" | \
    mail -s "ALERT: MyGest Backup Failed" admin@example.com
fi

# Installare mailutils se necessario
sudo apt install mailutils
```

5.4 Recovery da Backup

5.4.1 Restore Completo

Scenario: NAS locale danneggiato, restore da Google Drive

```
# Su VPS o PC locale

# 1. Verificare spazio disco disponibile
df -h /mnt/nas_mygest
# Deve avere almeno spazio = dimensione backup

# 2. Creare directory temporanea restore
sudo mkdir -p /mnt/nas_restore
sudo chown mygest:mygest /mnt/nas_restore

# 3. Download da Google Drive
rclone copy gdrive_mygest:MyGest_Backup/ /mnt/nas_restore/ \
  --progress \
  --log-file=/var/log/rclone_restore.log

# Progress bar:
# Transferred:        8.5 GiB / 8.5 GiB, 100%, 10 MiB/s, ETA 0s
# Elapsed time:   15m30s

# 4. Verificare integrità dati
du -sh /mnt/nas_restore
# 8.5G    /mnt/nas_restore

tree /mnt/nas_restore -L 2
# Verificare struttura directory corretta

# 5. Stopare applicazione (per evitare scritture)
sudo systemctl stop mygest

# 6. Backup NAS corrente (se parzialmente recuperabile)
sudo mv /mnt/nas_mygest /mnt/nas_mygest.old

# 7. Spostare restore in produzione
sudo mv /mnt/nas_restore /mnt/nas_mygest
```

```
# 8. Ripristinare permessi
sudo chown -R mygest:mygest /mnt/nas_mygest
sudo chmod -R u+rwX,g+rwX,o+rX /mnt/nas_mygest

# 9. Riavviare applicazione
sudo systemctl start mygest

# 10. Verificare accesso documenti
curl https://mygest.sandro.cloud/api/v1/documenti/1/
# Verificare che "file" URL sia accessibile
```

5.4.2 Restore Selettivo

Recuperare solo alcuni file/directory:

```
# Esempio: restore solo documenti cliente CLI042

# 1. Elencare contenuto backup remoto
rclone lsf gdrive_mygest:MyGest_Backup/clienti/

# Output:
# CLI001_ROSSI_MARIO/
# CLI042_ACME_SRL/
# CLI089_VERDI_GIUSEPPE/

# 2. Download singola directory
rclone copy \
  gdrive_mygest:MyGest_Backup/clienti/CLI042_ACME_SRL/ \
  /tmp/restore_cli042/ \
  --progress

# 3. Verificare file
ls -lR /tmp/restore_cli042/

# 4. Copiare in NAS produzione
sudo cp -r /tmp/restore_cli042/* /mnt/nas_mygest/clienti/CLI042_ACME_SRL/

# 5. Correggere permessi
sudo chown -R mygest:mygest /mnt/nas_mygest/clienti/CLI042_ACME_SRL/
```

5.4.3 Restore Point-in-Time

Recuperare file da versione specifica:

```
# rclone tiene versioni Google Drive (se attivato)

# 1. Vedere versioni file
rclone lsl gdrive_mygest:MyGest_Backup/clienti/CLI001_ROSSI_MARIO/01_CONTABILITA

# 2. Restore con versione specifica
# Google Drive tiene automaticamente versioni ultimi 30-100 giorni

# Metodo alternativo: snapshot NAS periodici
# Configurare snapshot giornalieri su NAS per point-in-time recovery
```

5.5 Manutenzione Archivio

5.5.1 Pulizia File Temporanei

```
# File temp più vecchi di 7 giorni
find /mnt/nas_mygest/temp/ -type f -mtime +7 -delete

# Verificare spazio liberato
du -sh /mnt/nas_mygest/temp/

# Cron job pulizia automatica
sudo crontab -e

# Aggiungere:
# Pulizia temp ogni domenica alle 3 AM
0 3 * * 0 find /mnt/nas_mygest/temp/ -type f -mtime +7 -delete
```

5.5.2 Verifica Integrità File

```
# Verificare file corrotti (PDF)
find /mnt/nas_mygest/clienti/ -name "*.pdf" -type f | while read pdf; do
  pdfinfo "$pdf" > /dev/null 2>&1 || echo "Corrupt: $pdf"
done

# Log file problematici
# Manualmente verificare e risolvere
```

5.5.3 Report Utilizzo Spazio

```
# Spazio per cliente (top 20)
du -sh /mnt/nas_mygest/clienti/*/ | sort -hr | head -20

# Output:
# 450M    /mnt/nas_mygest/clienti/CLI042_ACME_SRL/
# 320M    /mnt/nas_mygest/clienti/CLI001_ROSSI_MARIO/
# 280M    /mnt/nas_mygest/clienti/CLI089_VERDI_GIUSEPPE/
#  ...

# File più grandi (top 50)
find /mnt/nas_mygest/clienti/ -type f -exec du -h {} + | sort -hr | head -50

# Trend crescita (da monitorare)
df -h /mnt/nas_mygest | awk '{print $5}' | tail -1
# Output: 61% (uso attuale)
```

---

6. Troubleshooting

6.1 Problemi Deploy

6.1.1 GitHub Actions Fallito

Errore: SSH connection failed

```
Error: Connection timed out
Error: Could not connect to 72.62.34.249:22
```

Soluzione:

```
# 1. Verificare VPS online
ping 72.62.34.249

# 2. Verificare SSH manuale
ssh mygest@72.62.34.249

# 3. Verificare GitHub Secret SSH_PRIVATE_KEY
# Settings > Secrets > Actions > SSH_PRIVATE_KEY
# Deve contenere chiave privata completa:
# -----BEGIN OPENSSH PRIVATE KEY-----
# ...
# -----END OPENSSH PRIVATE KEY-----

# 4. Verificare authorized_keys su VPS
ssh mygest-vps
cat ~/.ssh/authorized_keys
# Deve contenere chiave pubblica corrispondente
```

Errore: Tests failed

```
FAILED tests/test_models.py::TestModel::test_validation
```

Soluzione:

```
# 1. Eseguire test in locale
pytest tests/test_models.py::TestModel::test_validation -vv

# 2. Fixare errore
#  ...  modifiche codice  ...

# 3. Re-commit e push
git add -A
git commit -m "fix: risolto test validation"
git push origin main
```

6.1.2 Deployment Manuale Fallito

Errore: pip install requirements failed

```
ERROR: Could not find a version that satisfies the requirement package==1.2.3
```

Soluzione:

```
# 1. Verificare Python version
python --version
# Deve essere ≥ 3.10

# 2. Aggiornare pip
pip install --upgrade pip setuptools wheel

# 3. Verificare package disponibile
pip search package

# 4. Installare con verbose
pip install -r requirements.txt -v
```

Errore: npm build failed

```
Error: ENOSPC: System limit for number of file watchers reached
```

Soluzione:

```
# Aumentare limite watchers Linux
echo fs.inotify.max_user_watches=524288 | sudo tee -a /etc/sysctl.conf
sudo sysctl -p
```

Errore: collectstatic failed

```
OSError: [Errno 28] No space left on device
```

Soluzione:

```
# Verificare spazio disco
df -h

# Pulire cache vecchi static
rm -rf /srv/mygest/app/staticfiles/*
python manage.py collectstatic --noinput --clear

# Pulire log vecchi
sudo journalctl --vacuum-time=7d
```

6.1.3 Servizio Non Parte

Errore: Gunicorn failed to start

```
sudo systemctl status mygest

# Output:
# ● mygest.service - MyGest Gunicorn daemon
#    Loaded: loaded
#    Active: failed (Result: exit-code)
```

Soluzione:

```
# 1. Vedere log dettagliato
sudo journalctl -u mygest -n 100 --no-pager

# Errori comuni:

# a) ModuleNotFoundError: No module named 'X'
# Soluzione: pip install mancante
source /srv/mygest/venv/bin/activate
pip install X

# b) django.db.utils.OperationalError: FATAL: database "mygest" does not exis
# Soluzione: ricreare database
createdb -U mygest mygest
python manage.py migrate

# c) PermissionError: [Errno 13] Permission denied: '/srv/mygest/ ... '
# Soluzione: correggere ownership
sudo chown -R mygest:mygest /srv/mygest/

# 2. Riavviare servizio
sudo systemctl restart mygest

# 3. Verificare OK
sudo systemctl status mygest
curl http://localhost:8000/api/v1/health/
```

6.2 Problemi Sincronizzazione DB

6.2.1 Sync Script Failed

Errore: SSH connection to VPS failed

```
ssh: connect to host 72.62.34.249 port 22: Connection refused
```

Soluzione: Vedi 6.1.1

Errore: Export failed - Database query timeout

```
django.db.utils.OperationalError: canceling statement due to statement timeou
```

Soluzione:

```
# Database troppo grande, export selettivo
./scripts/sync_dev_to_prod.sh --models anagrafiche,clienti

# Oppure aumentare timeout PostgreSQL
psql -U mygest -d mygest -c "ALTER DATABASE mygest SET statement_timeout = '6
```

Errore: Import failed - IntegrityError

```
django.db.utils.IntegrityError: duplicate key value violates unique constrair
```

Soluzione:

```
# 1. Verificare dati duplicati in sorgente
python manage.py shell
>>> from anagrafiche.models import Anagrafica
>>> Anagrafica.objects.values('codice_fiscale').annotate(count=models.Count('

# 2. Risolvere duplicati manualmente o:

# 3. Usare --full-replace per reset completo (ATTENZIONE)
./scripts/sync_dev_to_prod.sh --full-replace --dry-run
```

6.2.2 Rollback Failed

Errore: Backup file not found

```
Error: /srv/mygest/backups/sync/pre_sync_20260221_143045.sql not found
```

Soluzione:

```
# 1. Vedere backup disponibili
ls -lh /srv/mygest/backups/sync/

# 2. Usare backup più recente disponibile
pg_restore -U mygest -d mygest /srv/mygest/backups/sync/pre_sync_<altra_data>

# 3. Se nessun backup sync, usare backup giornaliero
ls -lh /srv/mygest/backups/daily/
pg_restore -U mygest -d mygest /srv/mygest/backups/daily/backup_<data>.dump
```

6.3 Problemi Archivio NAS

6.3.1 NAS Non Montato

Errore: ls /mnt/nas_mygest: Input/output error

```
ls /mnt/nas_mygest
# ls: cannot access '/mnt/nas_mygest': Input/output error
```

Soluzione:

```
# 1. Unmount forzato
sudo umount -f /mnt/nas_mygest

# 2. Verificare NAS accessibile
ping 192.168.1.100

# 3. Testare connessione SMB
smbclient -L //192.168.1.100 -U mygest

# 4. Remount
sudo mount /mnt/nas_mygest

# 5. Verificare
ls -la /mnt/nas_mygest
```

6.3.2 Permessi Negati

Errore: Django cannot write to NAS

```
PermissionError: [Errno 13] Permission denied: '/mnt/nas_mygest/clienti/CLI00
```

Soluzione:

```
# 1. Verificare ownership
ls -ld /mnt/nas_mygest
# Deve essere: drwxrwxr-x mygest mygest

# 2. Correggere
sudo chown -R mygest:mygest /mnt/nas_mygest
sudo chmod -R u+rwX,g+rwX /mnt/nas_mygest

# 3. Verificare mount options in /etc/fstab
# Deve avere: uid=1001,gid=1001 (id user mygest)

# 4. Remount con opzioni corrette
sudo umount /mnt/nas_mygest
sudo mount /mnt/nas_mygest
```

6.3.3 Backup rclone Failed

Errore: rclone sync failed - 403 Forbidden

```
ERROR : Failed to copy: googleapi: Error 403: User rate limit exceeded
```

Soluzione:

```
# 1. Google Drive quota exceeded, attendere o:

# 2. Usare --drive-chunk-size più piccolo
rclone sync /mnt/nas_mygest/ gdrive_mygest:MyGest_Backup/ \
  --drive-chunk-size 32M \
  --transfers 1

# 3. Oppure aumentare quota Google Drive (Google One)
```

Errore: Token expired

```
ERROR : Failed to create file system: token refresh failed
```

Soluzione:

```
# 1. Refresh token rclone
rclone config reconnect gdrive_mygest:

# Seguire URL per re-autorizzare

# 2. Testare
rclone lsd gdrive_mygest:
```

6.4 Errori Comuni e Quick Fix

Errore	Causa	Quick Fix
`502 Bad Gateway`	Gunicorn down	`sudo systemctl restart mygest`
`500 Internal Server Error`	Exception Python	`sudo journalctl -u mygest -n 50`
`404 Not Found` static	collectstatic mancante	`python manage.py collectstatic`
`Connection refused :8000`	Gunicorn non in ascolto	Verificare gunicorn.service
`CSRF token missing`	Cookie non impostati	Verificare ALLOWED_HOSTS
`Database locked` SQLite	Usare PostgreSQL	Migrare a PostgreSQL
`No space left on device`	Disco pieno	`df -h && sudo apt clean`

---

7. FAQ

7.1 Deploy e Git

Q: Posso fare deploy di un branch diverso da main?

A: Sì, modificando GitHub Actions:

```
# .github/workflows/deploy.yml
on:
  push:
    branches: [ main, staging ]  # Aggiungi branch
```

Q: Come annullo un commit già pushato?

A:

```
# Revert ultimo commit (crea nuovo commit)
git revert HEAD
git push origin main

# Oppure reset hard (PERICOLOSO, riscrive storia)
git reset --hard HEAD~1
git push origin main --force
```

Q: Posso fare deploy senza GitHub Actions?

A: Sì, usa `./scripts/deploy.sh` o procedura manuale (vedi 3.2)

7.2 Database e Sincronizzazione

Q: Con che frequenza sincronizzare dev→prod?

A: Dipende:
- Setup iniziale: 1 volta
- Aggiornamento nomenclatori: quando necessario
- **Mai** per dati utente (usare prod→dev invece)

Q: Posso sincronizzare solo una app Django?

A: Sì:

```
./scripts/sync_dev_to_prod.sh --models anagrafiche
```

Q: I backup sync vengono eliminati?

A: Automaticamente dopo 10 sync (retention policy). Backup importanti salvare in `/backups/manual/`

Q: Come sincronizzo anche i file NAS insieme al DB?

A: Attualmente manuale:

```
# Sync DB
./scripts/sync_dev_to_prod.sh

# Sync file (da implementare script dedicato)
rsync -avz /mnt/nas_mygest/ mygest-vps:/mnt/nas_mygest/
```

7.3 Archivio e Backup

Q: Quanto spazio serve per backup Google Drive?

A: Dipende da dimensione archivio. Verificare:

```
du -sh /mnt/nas_mygest/
rclone about gdrive_mygest:
```

Q: Google Drive gratuito basta?

A: 15 GB gratis. Se superi, considera Google One (100GB €2/mese)

Q: Posso usare altri cloud invece di Google Drive?

A: Sì, rclone supporta 40+ provider:

```
rclone config
# Scegliere: Dropbox, OneDrive, AWS S3, etc.
```

Q: Come recupero un file cancellato per errore?

A:
1. Verificare Google Drive (tiene versioni 30 gg)
2. Restore da backup sync più recente
3. Snapshot NAS (se configurato)

7.4 Troubleshooting

Q: Il sito è lento dopo deploy, perché?

A: Possibili cause:
- Cache Redis da svuotare: `redis-cli FLUSHALL`
- Query N+1: Ottimizzare con select_related/prefetch_related
- Static files non compressati: Verificare Nginx gzip

Q: Dopo sync il login non funziona più

A: Password hashate diverse. Soluzione:

```
python manage.py changepassword admin
# Reimpostare password conosciuta
```

Q: Deploy bloccato, come sbloccare?

A:

```
# Killare processo gunicorn appeso
sudo pkill -9 gunicorn

# Riavviare servizio
sudo systemctl restart mygest
```

Supporto

Per problemi non coperti da questa guida:

1. **Log applicazione**: `sudo journalctl -u mygest -f`
2. **Log Nginx**: `sudo tail -f /var/log/nginx/error.log`
3. **Log PostgreSQL**:
   `sudo tail -f /var/log/postgresql/postgresql-14-main.log`
4. **Issue GitHub**: https://github.com/sandro6917/mygest/issues

---

Riferimenti

- Django Deployment Checklist (https://docs.djangoproject.com/en/5.0/howto/deployment/checklist/)
- PostgreSQL Backup & Restore (https://www.postgresql.org/docs/current/backup.html)
- rclone Documentation (https://rclone.org/docs/)
- GitHub Actions Documentation (https://docs.github.com/actions)

Fine della Guida

MyGest Development Team - Febbraio 2026

Per supporto: GitHub Issues (https://github.com/sandro6917/mygest/issues)