



Logo Applier - Applicatore Logo su Immagini

Python 3.8+

License MIT

Platform Windows | macOS | Linux

GB English version

Applicazione desktop professionale per applicare automaticamente logo e watermark su batch di immagini, con interfaccia grafica intuitiva e supporto per sfondi personalizzati.

version 1.0.0

✦ Caratteristiche Principali

- 🎯 **Posizionamento Interattivo** - Visualizza in tempo reale il logo mentre muovi il mouse
- 🤖 **Modalità Automatica** - Applica il logo in posizioni fisse predefinite (4 angoli)
- 🎨 **Sfondi Personalizzati** - Aggiungi sfondi colorati al logo (6 colori, 3 forme)
- 📐 **Ridimensionamento Intelligente** - Il logo si adatta all'orientamento dell'immagine
- 💾 **Salvataggio Impostazioni** - Le tue preferenze vengono ricordate
- 📊 **Barra di Progresso** - Monitoraggio in tempo reale dell'elaborazione
- 🔄 **Elaborazione Batch** - Processa centinaia di immagini in pochi minuti
- 🏗️ **Architettura Modulare** - Codice organizzato secondo principi di ingegneria del software

🚀 Installazione Rapida

Requisiti

- Python 3.8 o superiore
- Pillow (installato automaticamente)

Procedura

```
# Clona il repository
git clone https://github.com/tuusername/logo-applier.git
cd logo-applier

# Installa le dipendenze
pip install -r requirements.txt

# Avvia l'applicazione
python main.py
```

📖 Come Usare

1. Configurazione Base

1. Seleziona la **cartella con le immagini** da elaborare
2. Seleziona il **file del logo** (preferibilmente PNG con trasparenza)
3. Seleziona la **cartella di destinazione** per le immagini elaborate

2. Impostazioni Logo

- **Dimensione:** Scegli tra 5%, 10%, 15%, 20% o 25% (si adatta automaticamente all'orientamento)
- **Margine:** Imposta la distanza dai bordi (1%-15%)

3. Sfondo Logo (Opzionale)

Personalizza il tuo logo con uno sfondo:

- **Colori:** Nessuno, Bianco, Giallo, Arancione, Rosso, Grigio chiaro, Rosa pallido, Giallo chiaro, Arancione chiaro, Azzurro pastello, Celeste, Verde menta
- **Forme:** Circolare, Rettangolare, Ovale

4. Modalità di Posizionamento

Modalità Manuale

Massima precisione per ogni immagine:

- Si apre una finestra di anteprima per ogni foto
- Il logo segue il cursore del mouse in tempo reale
- Clicca per confermare la posizione
- Perfetto per fotografie artistiche o composizioni specifiche

Modalità Automatica

Veloce ed efficiente per batch uniformi:

- Scegli una delle 4 posizioni fisse:
 - Alto a sinistra
 - Alto a destra
 - Basso a sinistra
 - Basso a destra
- Ideale per grandi quantità di immagini simili

Esempi d'Uso

Fotografo Professionista

```
Modalità: Manuale
Dimensione logo: 10%
Sfondo: Bianco Rettangolare
Posizione: Personalizzata per ogni foto
```

Perfetto per portfolio dove ogni immagine richiede un posizionamento specifico.

E-commerce

Modalità: Automatica
Dimensione logo: 15%
Sfondo: Nessuno
Posizione: Basso a destra

Ideale per applicare rapidamente il marchio su centinaia di foto prodotto.

Social Media

Modalità: Automatica
Dimensione logo: 20%
Sfondo: Celeste Circolare
Posizione: Alto a destra

Logo ben visibile e accattivante per massimizzare il branding.

Architettura del Progetto

```
logo-applier/  
├── main.py           # Entry point  
├── config.py        # Configurazioni  
├── gui/             # Interfaccia grafica  
│   ├── main_window.py # Finestra principale  
│   └── preview_window.py # Finestra anteprima  
└── utils/           # Utilità  
    ├── image_processor.py # Elaborazione immagini  
    └── settings_manager.py # Gestione impostazioni
```

Il progetto segue i principi **SOLID** e utilizza un'architettura modulare per facilitare manutenzione ed estensioni future.

Funzionalità Avanzate

Ridimensionamento Intelligente

Il logo viene ridimensionato in modo adattivo:

- **Immagini Orizzontali:** dimensione calcolata sulla larghezza
- **Immagini Verticali:** dimensione calcolata sull'altezza

Protezione Bordi

Nella modalità manuale, il logo non può eccedere i bordi dell'immagine - viene automaticamente limitato all'area visibile.

Salvataggio Parziale

Se interrompi l'elaborazione, tutte le immagini già processate vengono salvate automaticamente.

Formati Supportati

- **Input:** JPG, JPEG, PNG, BMP, GIF
- **Output:** JPEG alta qualità (100%)

Personalizzazione

Aggiungere Nuovi Colori

Modifica `config.py`:

```
BACKGROUND_COLORS = {  
    # ... colori esistenti ...  
    'Verde': '#00FF00', # Nuovo colore  
}
```

Modificare Qualità Output

In `config.py`:

```
OUTPUT_QUALITY = 100 # Massima qualità
```

Performance

- **10 immagini Full HD:** ~20-30 secondi
- **100 immagini 4K:** ~10-15 minuti
- **Memoria RAM:** 100-500MB (dipende dalla risoluzione)

Risoluzione Problemi

"Module not found"

```
pip install -r requirements.txt
```

"No module named 'gui'"

Verifica che esistano i file `__init__.py` in `gui/` e `utils/`:

```
touch gui/__init__.py utils/__init__.py
```

Logo non visibile

- Usa PNG con sfondo trasparente
- Aumenta la percentuale dimensione (15-20%)
- Prova ad aggiungere uno sfondo colorato

Contribuire

I contributi sono benvenuti!

1. Fork il progetto
2. Crea un branch (`git checkout -b feature/NuovaFunzionalità`)
3. Commit le modifiche (`git commit -m 'Aggiunta: NuovaFunzionalità'`)
4. Push al branch (`git push origin feature/NuovaFunzionalità`)
5. Apri una Pull Request

Idee per Contributi




- ☐ Supporto per video (watermark su frame)
- ☐ Rotazione del logo
- ☐ Effetti (ombra, trasparenza)
- ☐ Supporto per testo come watermark
- ☐ Anteprima batch thumbnail

Licenza

Questo progetto è rilasciato sotto licenza MIT. Vedi [LICENSE](#) per dettagli.

Autore

Michele Barbella

-  Email: m.barbella5@gmail.com
-  LinkedIn: [michele-barbella](#)
-  GitHub: [@michelebarbella](#)