**PubliScript - Rifattorizzazione del Codice**

**Panoramica**

Questa rifattorizzazione del codice organizza la codebase precedente di PubliScript in una struttura modulare. Il progetto ora è organizzato in sottodirectory logiche che separano diversi aspetti funzionali del sistema, rendendo il codice più gestibile e manutenibile.

**Struttura del Progetto**

crisp/

│

├── \_\_init\_\_.py # Rende importabile l'intero progetto

│

├── ai\_interfaces/ # Gestione connessione e invio messaggi all'AI

│ ├── \_\_init\_\_.py # Esporta le funzioni principali

│ ├── genspark\_driver.py # Gestione comunicazione con Genspark

│ ├── browser\_manager.py # Configurazione e gestione del browser

│ ├── file\_text\_utils.py # Utilità per file e testo

│ └── interaction\_utils.py # Funzioni di interazione con l'interfaccia

│

├── framework/ # Framework CRISP per la gestione dei libri

│ ├── \_\_init\_\_.py # Esporta le classi e funzioni principali

│ ├── crisp\_framework.py # Framework principale CRISP

│ ├── crisp\_extractors.py # Estrazione dati strutturati

│ ├── crisp\_utils.py # Utilità generiche per il framework

│ └── analysis/ # Moduli specializzati per analisi

│ ├── \_\_init\_\_.py

│ ├── market\_analysis.py # Analisi di mercato

│ ├── buyer\_persona.py # Analisi della buyer persona

│ └── gap\_analysis.py # Analisi delle criticità e gap

│

├── ui/ # Interfaccia utente Gradio

│ ├── \_\_init\_\_.py # Esporta le classi UI

│ ├── app\_launcher.py # Avvio dell'interfaccia

│ ├── ai\_connection.py # Gestione connessione al servizio AI

│ ├── chat\_manager.py # Gestione conversazioni e log

│ ├── cooldown\_manager.py # Limitazione rate delle richieste

│ ├── book\_builder.py # Classe principale per la creazione di libri

│ ├── log\_utils.py # Utilità di logging

│ └── components.py # Componenti UI riutilizzabili

│

└── main.py # Entry point principale

**Principali Componenti**

**ai\_interfaces/**

Questo modulo gestisce tutta l'interazione con il servizio AI Genspark:

* **genspark\_driver.py**: Coordina l'interazione di alto livello con Genspark, gestendo login, sessioni e invio prompt
* **browser\_manager.py**: Gestisce l'istanza del browser Selenium con configurazioni ottimizzate
* **file\_text\_utils.py**: Contiene utility per la gestione di file e la manipolazione del testo
* **interaction\_utils.py**: Funzioni specializzate per interagire con l'interfaccia di Genspark

**framework/**

Implementa il framework CRISP per la generazione e gestione dei libri:

* **crisp\_framework.py**: La classe principale CRISPFramework che gestisce l'esecuzione dei prompt e l'estrazione dei dati
* **crisp\_extractors.py**: Funzioni per estrarre dati strutturati dalle risposte dell'AI
* **crisp\_utils.py**: Utilities di supporto per il framework CRISP
* **analysis/**: Moduli specializzati per diversi tipi di analisi

**ui/**

Gestisce l'interfaccia utente basata su Gradio:

* **app\_launcher.py**: Avvia l'interfaccia Gradio
* **ai\_connection.py**: Gestisce la connessione iniziale all'AI
* **chat\_manager.py**: Gestisce le conversazioni e i file di log/contesto
* **cooldown\_manager.py**: Regola la frequenza delle richieste per evitare timeout
* **book\_builder.py**: Classe principale AIBookBuilder per la generazione di libri
* **components.py**: Componenti UI riutilizzabili di Gradio

**Entry Point**

* **main.py**: Punto di ingresso principale dell'applicazione che può avviare la versione GUI o CLI

**Flusso delle Operazioni**

[ main.py ]

↓

[ app\_launcher.py (UI Launcher) ]

↓

[ book\_builder.py (AIBookBuilder) ]

↓

[ crisp\_framework.py (CRISP Core) ]

↓

[ analysis modules (market\_analysis.py, buyer\_persona.py, gap\_analysis.py) ]

↓

[ ai\_interfaces (Genspark driver, browser manager, interaction utils) ]

↓

[ Browser Chrome + Genspark ]

**Provenienza dal Software Originale**

| **Nuovo file** | **Deriva da** |
| --- | --- |
| genspark\_driver.py | genschat.py |
| browser\_manager.py | genschat.py |
| file\_text\_utils.py | genschat.py |
| interaction\_utils.py | genschat.py |
| crisp\_framework.py | crisp\_framework.py |
| crisp\_extractors.py | crisp\_extractors.py |
| crisp\_utils.py | crisp\_utils.py |
| market\_analysis.py | crisp\_extractors.py |
| buyer\_persona.py | crisp\_extractors.py |
| gap\_analysis.py | crisp\_extractors.py |
| app\_launcher.py | genschat\_gui.py |
| ai\_connection.py | genschat\_gui.py |
| chat\_manager.py | genschat\_gui.py |
| cooldown\_manager.py | genschat\_gui.py |
| book\_builder.py | genschat\_gui.py |
| log\_utils.py | genschat\_gui.py |
| components.py | genschat\_gui.py |
| main.py | nuovo |

**Note di Implementazione**

1. **Import Circolari**: Sono stati risolti spostando alcuni import all'interno delle funzioni dove necessario.
2. **Compatibilità**: La rifattorizzazione mantiene tutte le funzionalità originali, modificando solo la struttura organizzativa.
3. **Gestione delle Istanze**: L'istanza del browser viene gestita centralmente per evitare duplicazioni.
4. **Conservazione del Codice**: La rifattorizzazione è stata volutamente "leggera", mantenendo intatto il codice originale e modificando solo gli import e l'organizzazione dei file.

**Utilizzo**

Per avviare l'applicazione, eseguire:

python -m crisp.main

**Miglioramenti Futuri**

* Implementazione di test automatici
* Ulteriore modularizzazione delle funzionalità più complesse
* Miglioramento della documentazione in-line
* Aggiornamento dell'interfaccia utente con componenti più modulari