EasySave v1.0 – Manuel d’utilisation

🔷 Contexte général

EasySave est un projet de sauvegarde de fichiers développé dans le cadre du module de Génie Logiciel au sein de l’éditeur fictif **ProSoft**. L’objectif est de fournir un outil fiable de copie de données, structuré selon les bonnes pratiques professionnelles.

Ce livrable (v1.0) correspond à une **application console C#** utilisat le framework **.NET 8.0**, avec des fonctionnalités de **sauvegarde complète et différentielle**, ainsi qu’un **suivi via fichiers JSON**.

🔷 Objectifs fonctionnels

- Créer, configurer et exécuter des **travaux de sauvegarde**.
- Support de 2 types :
  - **Complète** : copie intégrale
  - **Différentielle** : copie des fichiers modifiés depuis la dernière complète
- Génération de :

- `log.json` : historique de sauvegarde
- `state.json` : état en temps réel

🔷 Architecture technique

- **Langage** : C#
- **Framework** : .NET 8.0
- **Type** : Application console
- **Structure du projet** :
  - `Models/` : classes métiers (BackupJob, FileDetails...)
  - `Services/` : logique applicative (BackupManager, LogManager, etc.)
  - `Utils/` : fonctions utilitaires
  - `Program.cs` : point d'entrée

🔷 Dépôt GitHub

Le code source est disponible publiquement :

🔗 **https://github.com/Lewarde/Genie-Logiciel**

Pour récupérer le projet :

```
git clone https://github.com/Lewarde/Genie-Logiciel.git
cd Genie-Logiciel/EasySave
```

🔷 Installation & Exécution

Prérequis :

- Windows 10 ou +
- .NET SDK 8.0
- Visual Studio 2022 (ou VS Code)

Compilation et lancement :

```
dotnet build
dotnet run
```

🔷 Utilisation

L'application propose une interface interactive dans le terminal :

- **Créer un nouveau travail** :
  - Nom, chemin source, chemin cible, type
- **Exécuter un travail** :
  - Lancement manuel d'un des travaux enregistrés
- **Suivi JSON** :
  - `log.json` : fichier horodaté des actions
  - `state.json` : état en temps réel de la progression

🔷 Bonnes pratiques

- Ne pas modifier les fichiers JSON à la main.
- Vérifier que les chemins d'accès sont corrects et accessibles.
- Lancer les sauvegardes quand peu d'autres processus sont actifs.

🔷 Limitations (v1.0)

- Interface uniquement **console**.
- Pas de planification automatique.
- Sauvegarde différentielle basée uniquement sur la **date de modification**.
- Pas de multi-threading.

🔷 Perspectives d'évolution (v2.0)

- Interface graphique WPF (pattern **MVVM**)
- Optimisation des performances
- Gestion multi-thread
- Filtres d'exclusion (extensions, tailles...)
- Sauvegarde réseau / cloud