**Programme de recherche de mots dans des fichiers PDF**

Le programme est une application de recherche de mots dans des fichiers PDF. Il permet à l'utilisateur de rechercher des mots-clés spécifiques au sein de fichiers PDF individuels ou d'un ensemble de fichiers PDF dans un dossier.

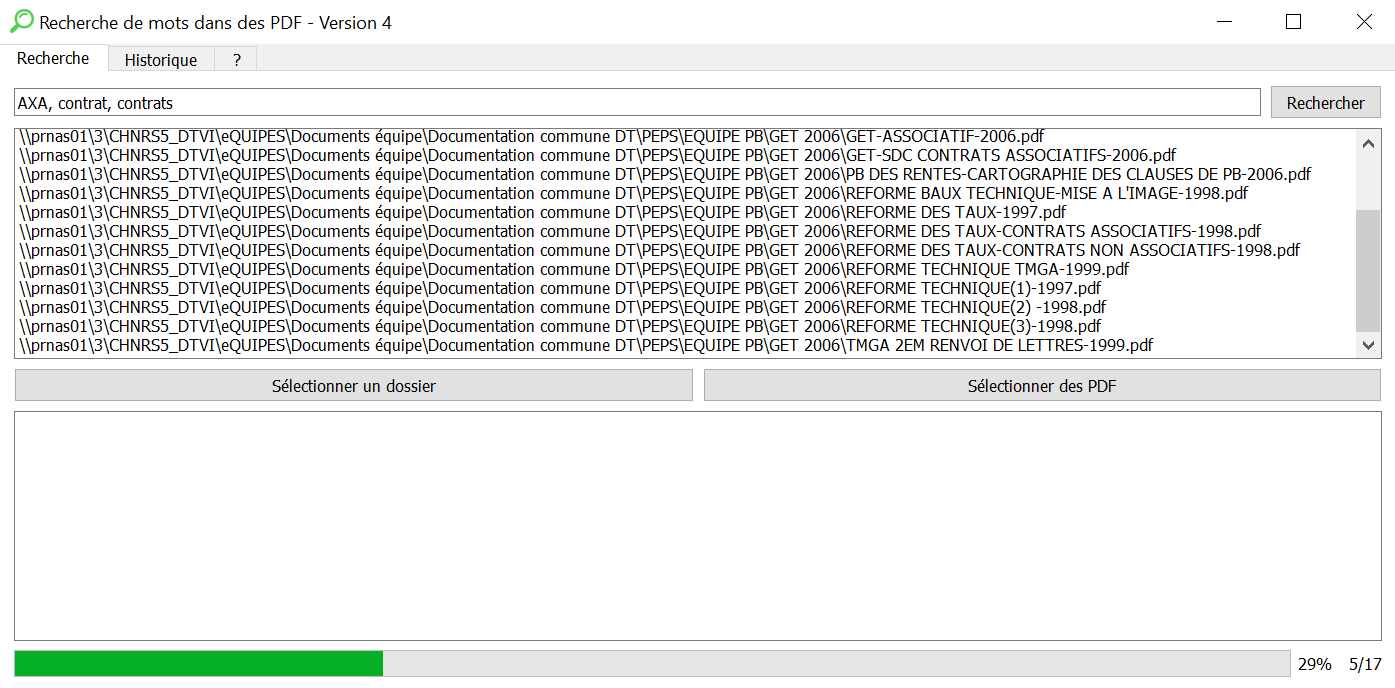
**Interface Utilisateur :**

Le programme a une interface utilisateur graphique (GUI) créée à l'aide de la bibliothèque PyQt5. Voici les composants de l'interface utilisateur et leur utilité :

1. **Onglet "Recherche" :**
2. **Méthode searchPDFs :**
   * Gère la recherche de mots-clés dans les fichiers PDF en utilisant PyMuPDF (**fitz**). Récupère les mots-clés et les fichiers PDF sélectionnés par l'utilisateur.
3. **Gestion des Résultats :**
   * Stocke les résultats dans **found\_results** organisés par mot-clé, associés aux fichiers PDF et aux numéros de page.
4. **Affichage des Résultats :**
   * Affiche les résultats à l'utilisateur dans la zone de texte dédiée, distinguant les documents trouvés, les pages associées, et les documents vides.
   * Rafraîchit l'affichage des résultats dans l'onglet "Recherche" via **updateSearchResults**.
5. **Interactivité et Rafraîchissement :**
   * Maintient l'interface réactive pendant la recherche grâce à QApplication.processEvents().

Zone de mot(s) à rechercher

Zone d’affichage des fichiers sélectionnés



Fichier en cours de traitement / Nombre total de fichiers à traiter

1. **Onglet "Historique" :**

1. **Méthode updateHistoryTab :**

* La méthode **updateHistoryTab** est responsable de la mise à jour de l'onglet "Historique" avec les résultats des recherches antérieures. Elle utilise un objet de type **defaultdict** pour regrouper les résultats par mots-clés, simplifiant ainsi le processus d'affichage.

2. **Format HTML Dynamique :**

* La méthode génère une présentation HTML dynamique en utilisant des balises **<b>** pour les mots-clés et des liens hypertextes (**<a>**) pour les documents. La mise en forme est réalisée en construisant une chaîne HTML qui est ensuite attribuée à la propriété **html** de l'élément **QTextBrowser**.

3. **Utilisation de QTextBrowser :**

* L'interface utilisateur utilise un objet **QTextBrowser** pour afficher le résultat HTML généré. La méthode **setOpenExternalLinks(True)** permet l'ouverture des liens hypertextes dans un navigateur externe, offrant une fonctionnalité interactive.

4. **Traitement des Résultats :**

* Les résultats de recherche antérieurs sont traités et organisés dans la méthode **updateHistoryTab**. Les informations sur chaque document trouvé, telles que le nom du document et les pages associées, sont extraites et formatées de manière à être facilement compréhensibles.

5. **defaultdict pour le Groupement :**

* Un objet de type **defaultdict** est utilisé pour organiser les résultats par mots-clés. Cela évite la nécessité de vérifier l'existence d'une clé avant chaque ajout.

6. **Gestion des Événements :**

* L'onglet "Historique" peut être mis à jour en réponse à des événements spécifiques, tels que l'achèvement d'une nouvelle recherche dans l'onglet "Recherche". La mise à jour de l'onglet se fait en appelant la méthode **updateHistoryTab** au bon moment dans le flux d'exécution de l'application.

1. **Onglet "Aide" :**
   * L'onglet "Aide" contient des informations explicatives sur le fonctionnement de l'application.

**Fonctions Principales :**

Voici les principales fonctions implémentées dans le programme :

1. **\_\_init\_\_(self):** Cette fonction est appelée lors de la création d'une instance de la classe PDFSearchApp. Elle initialise l'interface utilisateur, crée les onglets "Recherche", "Historique" et "Aide", et configure les widgets et les boutons nécessaires.
2. **selectPDFs(self):** Cette fonction est appelée lorsque l'utilisateur clique sur le bouton "Sélectionner des PDF". Elle ouvre une boîte de dialogue de sélection de fichiers PDF, permet à l'utilisateur de choisir un ou plusieurs fichiers PDF, et affiche les chemins des fichiers sélectionnés dans la zone de texte.
3. **selectFolder(self):** Fonctionne pareil que **selectPDFs(self)** mais permet de sélectionner un dossier. Tous les PDF dans le dossier et les sous-dossiers seront traités
4. **get\_pdf\_files\_in\_folder(self, folder\_path):** Cette fonction prend en entrée un chemin de dossier et recherche tous les fichiers PDF dans ce dossier et ses sous-dossiers. Elle retourne une liste des chemins des fichiers PDF trouvés.
5. **addSearchToHistory(self, keywords, results):** Cette fonction prend en entrée une liste de mots-clés et les résultats de la recherche. Elle ajoute ces informations à l'historique de recherche de l'application. Si les mêmes mots-clés existent déjà dans l'historique, elle fusionne les résultats.
6. **generateHTMLHistory(self):** Cette fonction génère une page HTML représentant l'historique des recherches effectuées. Elle crée une mise en page HTML avec des liens cliquables vers les fichiers PDF trouvés, puis enregistre cette page sous le nom "history.html" dans le répertoire de l'application.
7. **backupExistingHistory(self):** Cette fonction crée une copie de sauvegarde de l'ancien historique (history.html) sous un nom de fichier différent ("history\_backup.html").
8. **loadExistingHistory(self):** Cette fonction charge l'ancien historique (history.html) s'il existe, analyse son contenu HTML et le stocke dans l'attribut self.history.
9. **filterHistory(self):** Cette fonction est appelée chaque fois que l'utilisateur modifie le texte dans le champ de recherche de l'onglet "Historique". Elle filtre les résultats d'historique en fonction du texte de recherche et affiche les résultats filtrés dans la zone d'affichage de l'onglet "Historique".
10. **addToHistoryText(self, keywords\_text, results):** Cette fonction ajoute un nouvel élément à l'affichage de l'historique HTML et génère la nouvelle page HTML de l'historique.
11. **searchPDFs(self):** Cette fonction est appelée lorsque l'utilisateur clique sur le bouton "Rechercher" dans l'onglet "Recherche". Elle effectue la recherche de mots-clés dans les fichiers PDF sélectionnés, met à jour la zone de résultats et ajoute les résultats à l'historique.

**Remarques importantes :**

* Les fichiers PDF vides (sous forme d'images) ne peuvent pas être traités comme les autres documents. Il est recommandé d'utiliser un programme OCR (Reconnaissance Optique de Caractères) pour extraire le texte de ces fichiers.
* Lors de la sélection d'un dossier, tous les fichiers PDF dans le dossier et ses sous-dossiers seront recherchés.