

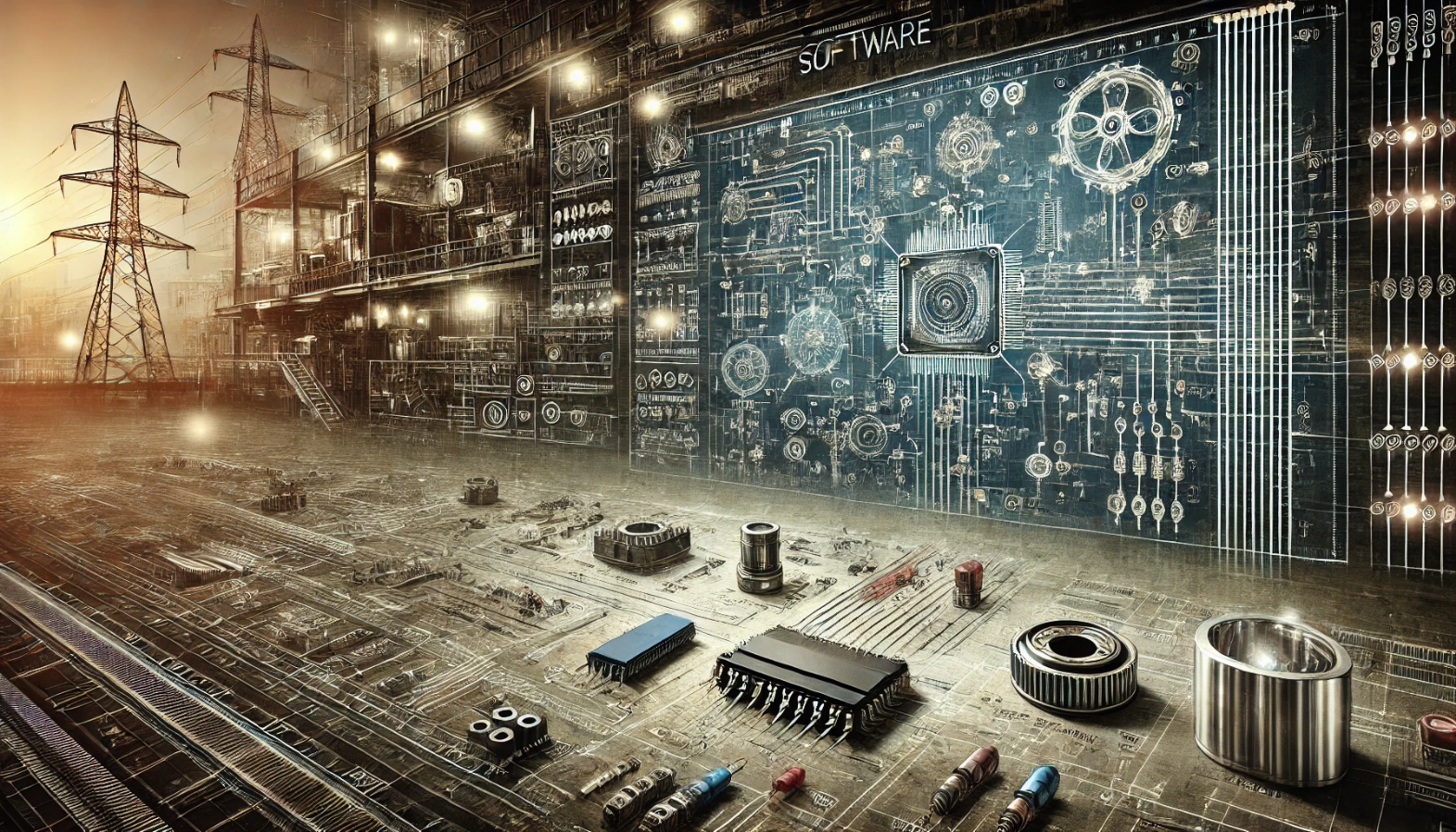
2025

Morgan Voltz

Clepsydre

17/04/2025

Projet Clepsydre V0.0



Somaire :

**Aucune entrée de table des matières n'a été trouvée.**

# I-Présentation du projet :

Créer un logiciel pouvant lire des schémas électrique ou électronique en format PDF, qui ont été scanner ou bien générer par un logiciel tier (WScad, Eplan ou autre).

* Il devra être capable de reconnaitre les caractères.
* Il devra être capable de reconnaitre les symboles normalisés.
* Il devra être capable de reconnaitre les connections.
* Exporter des données extraites des PDF en différent format :
  + Excel : afin de pouvoir garantir une compatibilité avec d’autre logiciel comme de l’impression et de marquage ; comme Pheonix Concact…
  + Json ou autre base de données
  + Autre (à déterminer au besoin)

# II-Normalisation du projet :

Pour le bien de tous et pour travailler dans de meilleure condition, nous allons Normaliser le maximum de choses : les appellations, les variables, les noms donner au diffèrent document, utilisateur et fonction.

Les variables :

Toute variable du code devra suivre impérativement cette norme, afin que nous puissions les retrouver facilement et de manière naturelle.

La méthode est composée de 3 points qui se suive :

1. Un préfix qui se constitue d’une suite de 3 lettres minuscule ou d’une lettre Majuscule : (Le préfixe permet de savoir de quel type de variable on a en un regard)
2. Enfin le nom de la variable en question en suivant la norme du *UpperCamelCase*.

Ce qui nous donne :

|  |  |
| --- | --- |
| IntUpperCamelCase | Entiers (int) |
| floUpperCamelCase | **Nombres à virgule flottante (float)** |
| comUpperCamelCase | **Nombres complexes (complex)** |
| strUpperCamelCase | **Variables textuelles (string)** |
| booUpperCamelCase | **Variables booléennes (bool)** |
| LUpperCamelCase | **Listes** |
| TUpperCamelCase | **Tuples** |

Les fonctions :

* Les noms de fonctions suivront cette méthode : texte en majuscule et en « *Snake\_Case* »

Exemple : FONCTION\_DU\_CODE ou CLASS\_FILE\_OPEN.

# III-déterminer le code utilisé pour créer ce programme :

|  |  |
| --- | --- |
| Python | C# |
|  |  |
| 1. Analyse de fichiers PDF (images scannées ou vecteurs)  👉 Langage recommandé : [Python]   * Excellente bibliothèque pour PDF : PyMuPDF, pdfminer.six, pdfplumber, etc. * OCR : Tesseract avec pytesseract * Vision par ordinateur : OpenCV * ML (si besoin d'entraîner une IA) : scikit-learn, TensorFlow, PyTorch |  |
| 2. Reconnaissance des symboles et connexions  Besoin de détecter les symboles électriques, les nœuds, les fils, etc.  👉 Langage recommandé : [Python]   * Pour la reconnaissance d’image et d’objets : OpenCV, Detectron2, ou YOLOv8 (Ultralytics) * Possibilité d’entraîner des modèles personnalisés pour reconnaître des composants |  |
| 3. Interface graphique (GUI)   * Visualiser le schéma * Permettre des interactions : zoom, sélection de composants, annotation, etc.   👉 Choix possibles :   * Python : PyQt / Tkinter / Kivy | **C#** : très bon pour applications Windows avec WinForms ou WPF |
| *4. Connectivité avec logiciels tiers (Eplan, WScad)*   * *Certains de ces logiciels exportent aussi en formats XML, DXF, DWG, ou ont des API* * *Tu pourrais tirer parti de ces formats pour des résultats plus fiables que le PDF*   *👉 Langage recommandé : Python*   * *Python : avec des parseurs XML, ou des outils comme ezdxf pour le DXF* | C# : très bon pour interagir avec API Windows (certains outils d'automatisation d'Eplan sont en C#) |

# IV-Créer les principales fonctions de l’application :

**✅ 1. Chargement du PDF**

* Extraire les pages en image (si scanné) ou en vecteurs (si possible).
* Bibliothèques :
  + PyMuPDF (aussi appelé fitz)
  + pdf2image (convertit pages en images)

**✅ 2. OCR pour les textes**

* Reconnaître les textes écrits sur le schéma
* Outils :
  + pytesseract + Tesseract OCR
  + OpenCV pour prétraitement (seuil, netteté, contours…)

**✅ 3. Détection de symboles / composants**

* Utiliser des techniques de vision par ordinateur (contours, templates, matching) ou de l’IA si on va plus loin
* Outils :
  + OpenCV (template matching, Canny, HoughLines)
  + YOLOv8 (optionnel, si tu veux entraîner un modèle plus tard)

**✅ 4. Reconnaissance des connexions (fils)**

* Analyse des traits, intersections, connexions
* Outils :
  + OpenCV (détection de lignes)
  + Analyse de graphes : networkx

**✅ 5. Affichage et interface graphique (simple au début)**

* Visualiser le schéma et les composants détectés
* Outils :
  + Tkinter (basique)
  + PyQt5 ou PySide6 (plus complet)

# V-Fonction

* + 1. Gérer les fichiers

-> Importer les PDF

-> Supprimer

-> Modifier le nom

-> Trier les fichiers

Par nom

Par date

Par type (Vectoriel ou Scan)

->Aperçu du fichier

2

# VI-Créer une interface simplifier pour importer/traiter les PDF :

01 Déterminer les outils qui sera utiliser pour créer cette interface

|  |  |  |
| --- | --- | --- |
| Fond principal | Gris foncé | #1E1E1E |
| Zones secondaires, cards | Gris moyen | #2E2E2E |
| Traits, séparateurs | Gris claire | #3C3C3C |
| Texte principal | Blanc cassé | #F2F2F2 |
| Bouton, actions | Accent bleu | #3B82F6 |
| Hover, warning | Accent orange | #F97316 |
| Succès, valide | Vert clair | #22C55E |

02 Trouver un style visuel et une identité visuelle

03 créer un premier squelette (Version simplifier qui pourras être amélioré et étoffé facilement)

04-Créer un premier béta, afin de la tester et d’en tirer une première expérience et des données afin d’avoir des pistes pour l’amélioration du produit.

# Gestion des fichiers importé et de ses informations :

* + 1. **Savoir si le fichier est un .PDF :**

-Si non afficher un message d’erreur.

-Si oui l’utilisateur peut importer le fichier.

-Pouvoir l’afficher pour informer l’utilisateur de quel type son PDF est.

* + 1. **Savoir si le document est un PDF Vectoriel ou un Scan :**
    2. **Savoir le nombre de page de ce document :**
    3. **Enregistrer son nom avec la possibilité de le modifier au besoin :**
    4. **Enregistrer la date de l’import du fichier :**
    5. **Savoir si c’est le premier, second, troisième, quatrième… :**
    6. **Savoir si le document a été modifier ou non :**

-Si oui, afficher la date et le nom de l’utilisateur qui la modifier.

-Si non ne rien afficher de particulier…

* + 1. **Si le PDF est seul ou regrouper avec d’autre document du même projet :**

-Si non ne rien afficher

-Si oui l’afficher et le nom du projet

Les fichiers importer devront par la suite stocker, trier et avoir la possibilité de les regrouper dans de fichier ou non, sous forme de « projet ». Car il se peut que plusieurs PDF soient le sou ensemble d’un plus gros projet.