

# Python et Tesseract pour l'extraction d'un menu de restaurant

## Contents

Chapitre 1 : Pourquoi Extraire le Prix d'un Menu et l'Adresse Web .....	4
1- Simplification de la recherche d'informations.....	4
2- Automatisation des tâches .....	4
Chapitre 2 : Pourquoi Avoir Utilisé Python et Tesseract .....	4
1- Python comme langage polyvalent.....	4
2- Tesseract pour la reconnaissance de texte .....	4
Chapitre 3 : Perspectives d'évolution :.....	4
1- Création de bases de données .....	4
2- Accessibilité pour les Personnes Malvoyantes.....	5
Chapitre 3 : Dépendances Nécessaires pour Exécuter ocr-projet.py et ocr-projet-gui.py .....	5
1- Python.....	5
2- Tesseract OCR 64bit.....	5
3- Bibliothèques Python .....	5
4- Systèmes d'Exploitation Compatibles .....	5
Chapitres4 : L'exécutable .....	5
1- Les éléments constitutifs de l'exécutable .....	5
2- Exécution de l'exécutable.....	6
Chapitres4 : Les fichiers python .py .....	7

# Introduction

Bienvenue dans le manuel utilisateur de l'application d'extraction d'information sur un image de menu de restaurant ! Ce manuel a pour but de vous guider à travers la compréhension et l'utilisation du programmes, `ocr-projet.py` et `ocr-projet-gui.py`, en expliquant leur utilité, leurs dépendances et en fournissant des instructions détaillées sur leur exécution.

# Chapitre 1 : Pourquoi Extraire le Prix d'un Menu et l'Adresse Web

J'ai choisi ce projet puisque l'automatisation des tâches dans notre vie quotidienne m'intéresse. C'est une occasion pour moi de mettre en œuvre une idée. Ce projet n'est qu'à son début. Après l'extraction des informations d'un menu de restaurant on pourrait exploiter les à des fins utiles.

## 1- Simplification de la recherche d'informations

L'extraction de données à partir d'une image de menu offre de nombreux avantages, notamment la simplification de la recherche d'informations. Les menus de restaurants peuvent contenir une grande quantité de données, notamment des prix, des descriptions de plats et d'autres informations essentielles. Sans une solution automatisée, la recherche de ces informations peut être fastidieuse et chronophage. Grâce à notre programme, on peut extraire rapidement et précisément les prix et faire une comparaison de prix avec un seuil de valeur qu'on aura choisi.

## 2- Automatisation des tâches

L'automatisation des tâches est un élément clé de l'extraction de données à partir d'images de menus. Pour une personne qui cherche à choisir un restaurant qui lui convient, il n'a qu'à télécharger le menu et des menus de plusieurs restaurants et importer l'image pour catégoriser les prix. Une page web affichera le résultat de l'extraction avec des valeurs colorés pour faciliter la lecture et la recherche d'information.

# Chapitre 2 : Pourquoi Avoir Utilisé Python et Tesseract

Dans ce chapitre, nous examinerons les raisons qui nous ont amenés à choisir Python comme langage de programmation principal et Tesseract comme outil de reconnaissance de texte pour notre projet.

## 1- Python comme langage polyvalent

Python est un langage de programmation polyvalent qui est devenu un choix privilégié pour un large éventail de projets de développement. Les raisons de notre choix de Python résident dans sa simplicité, sa lisibilité et sa flexibilité. Python offre une syntaxe claire et intuitive, ce qui le rend idéal pour le développement rapide d'applications.

## 2- Tesseract pour la reconnaissance de texte

Tesseract OCR est un outil puissant pour la reconnaissance de texte à partir d'images. Il est open source, évolutif et compatible avec de nombreux formats d'image. Son utilisation dans notre projet a été motivée par sa précision et sa capacité à traiter des images de menus de différentes tailles, qualités et polices de caractères.

# Chapitre 3 : Perspectives d'évolution :

## 1- Création de bases de données

En stockant les données extraites dans une base de données, vous pouvez les organiser de manière structurée et y accéder facilement à tout moment. Cela ouvre la porte à de nombreuses possibilités, telles que la génération de rapports, l'analyse des tendances de prix, la comparaison des menus entre différents établissements, et bien plus encore.

## 2- Accessibilité pour les Personnes Malvoyantes

Notre programme pourrait servir à rendre les informations accessibles aux personnes malvoyantes en ajoutant un synthétiseur vocal. Il pourrait potentiellement extraire le texte à partir d'images de menus et de le lire à voix haute à l'utilisateur. Cela rend l'information contenue dans le menu, y compris les prix, les descriptions de plats et d'autres détails, accessibles aux personnes malvoyantes.

## Chapitre 3 : Dépendances Nécessaires pour Exécuter ocr-projet.py et ocr-projet-gui.py

### 1- Python

Ce programme a été développé avec python 3.11.5. <https://www.python.org/downloads/>

### 2- Tesseract OCR 64bit

Tesseract OCR est un composant clé de nos applications pour la reconnaissance de texte à partir d'images.

<https://github.com/tesseract-ocr/tesseract>

### 3- Bibliothèques Python

- Pillow : Cette bibliothèque permet la manipulation d'images et est utilisée pour charger et traiter les images dans nos programmes.
- tkinter : Tkinter est la bibliothèque standard pour créer des interfaces utilisateur graphiques (GUI) en Python. Elle est utilisée dans ocr-projet-gui.py pour créer l'interface graphique de l'application.
- pytesseract : Pytesseract est une interface Python pour Tesseract OCR. Il est utilisé pour effectuer la reconnaissance de texte à partir d'images.
- re : Le module re de Python est utilisé pour effectuer des opérations de traitement de texte, y compris la recherche de motifs dans le texte.
- json : La bibliothèque standard de Python pour travailler avec le format JSON. Elle est utilisée pour stocker les données extraites dans un format JSON.
- webbrowser : Cette bibliothèque est utilisée pour ouvrir des pages web dans un navigateur, ce qui est utile pour consulter des liens extraits.

### 4- Systèmes d'Exploitation Compatibles

L'application a été développée et testée sur windows11. Mais devrait normalement être exécutable sur les autres systèmes d'exploitation

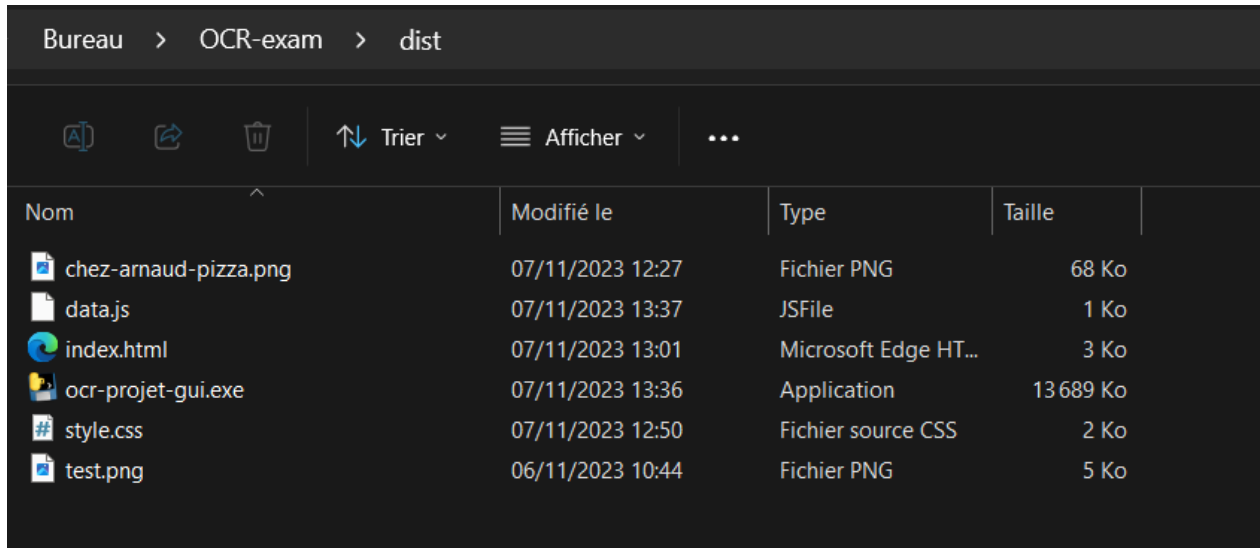
## Chapitres4 : L'exécutable







### 1- Les éléments constitutifs de l'exécutable

L'exécutable a été généré à partir du bibliothèque "pyinstaller" de python

L'exécutable : Dans le dossier « dist » contient 4 fichiers.

- Chez-arnaud-pizza.png et test.png sont des images de menus à pour tester l'application
- data.js contiendra les données extraites, et ces données seront affichée dans index.html
- index.html affichera dans un tableau données extraites
- ocr-projet-gui.exe est l'exécutable à lancer pour démarrer l'application
- style.css et un fichier qui va servir pour mettre en forme la page index.html

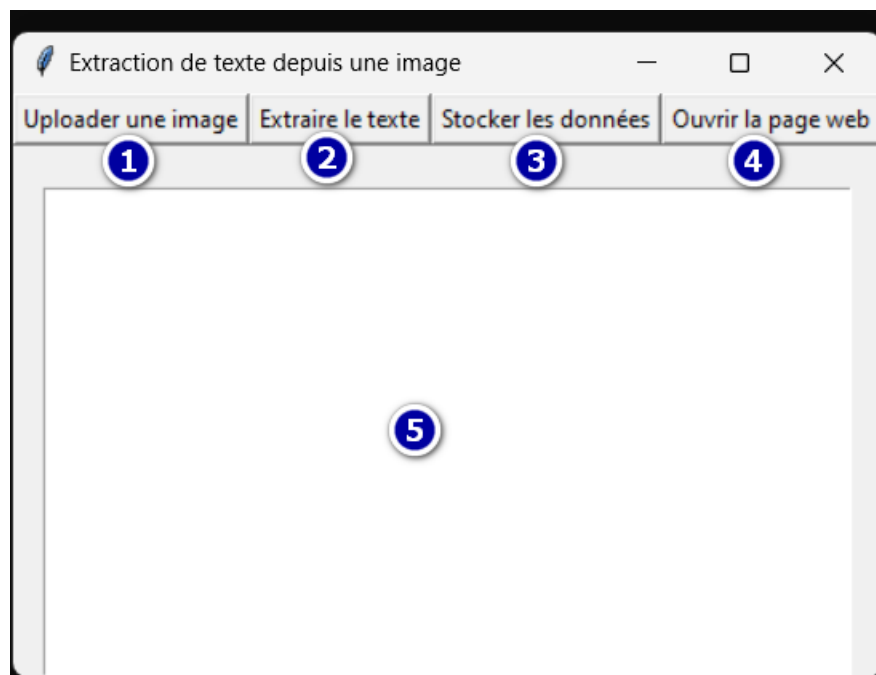


Nom	Modifié le	Type	Taille
 chez-arnaud-pizza.png	07/11/2023 12:27	Fichier PNG	68 Ko
 data.js	07/11/2023 13:37	JSFile	1 Ko
 index.html	07/11/2023 13:01	Microsoft Edge HT...	3 Ko
 ocr-projet-gui.exe	07/11/2023 13:36	Application	13 689 Ko
 style.css	07/11/2023 12:50	Fichier source CSS	2 Ko
 test.png	06/11/2023 10:44	Fichier PNG	5 Ko

## 2- Exécution de l'exécutable

Double cliquer sur ocr-projet-gui.exe

L'interface graphique suivant va s'afficher



- Cliquer sur Uploader une image pour charger une image
- Une fois l'image charger cliquer sur extraire le texte. Les textes extraits seront consultables dans l'emplacement en blanc (5).
- Cliquer sur Stocker (3) pour stocker les données dans data.js.
- Cliquer sur Ouvrir la page web (4) pour visualiser les donnees extraites dans une table de page web

## Chapitres4 : Les fichiers python .py

Nom	Modifié le	Type	Taille
__pycache__	07/11/2023 13:17	Dossier de fichiers	
build	07/11/2023 13:35	Dossier de fichiers	
dist	07/11/2023 14:28	Dossier de fichiers	
chez-arnaud-pizza.docx	07/11/2023 12:26	Microsoft Word D...	16 Ko
chez-arnaud-pizza.pdf	07/11/2023 12:26	Microsoft Edge PD...	86 Ko
chez-arnaud-pizza.png	07/11/2023 12:27	Fichier PNG	68 Ko
data.js	07/11/2023 13:29	JSFile	1 Ko
index.html	07/11/2023 13:52	Microsoft Edge HT...	3 Ko
Manuel.docx	07/11/2023 14:27	Microsoft Word D...	18 Ko
ocr-projet.py	07/11/2023 12:30	Python.File	2 Ko
ocr-projet-gui.py	07/11/2023 13:29	Python.File	4 Ko
ocr-projet-gui.spec	07/11/2023 13:35	Fichier SPEC	1 Ko
style.css	07/11/2023 12:50	Fichier source CSS	2 Ko
test.png	06/11/2023 10:44	Fichier PNG	5 Ko

C'est exactement le même principe que le fichier exécutable sauf que là on a encore des fichiers py.

Pour le fichier ocr-projet.py il faut spécifier manuellement le chemin de l'image à extraite dans le code pour pouvoir l'exécuter. Puis exécuter le programme en la ligne de commande comme ceci : `python ocr-projet.py`

```
ocr-projet.py x
ocr-projet.py
1 from PIL import Image
2 import pytesseract
3 import re
4 import json
5
6 # Chemin vers l'exécutable Tesseract
7 pytesseract.pytesseract_cmd = r'C:\Program Files\Tesseract-OCR\tesseract.exe' # Assurez-vous d'ajuster le chemin selon votre installation
8
9 # Charger l'image
10 image = Image.open('chez-arnaud-pizza.png') # Assurez-vous de remplacer 'test.png' par le nom réel de votre image
11
12 # Utiliser Tesseract pour faire de la reconnaissance de texte
13 texte_reconnu = pytesseract.image_to_string(image)
14
15 # Patron de recherche pour les prix
16 pattern_prix = r'(?i)(\d{1,4}([\s,.]*)\d{3})\s*(Ar|Ariary))' # Recherche de chiffres, espaces, points, virgules et 'Ar' ou 'Ariary' (en ignorant la casse)
17 prix_tous = re.findall(pattern_prix, texte_reconnu)
18
19 # Convertir les prix en nombres
20 prix = []
21
22 for prix_str in prix_tous:
23     # Nettoyer la chaîne pour ne garder que les chiffres, les points et les virgules
24     prix_clean = re.sub(r'^\d.,|', '', prix_str[0])
25
26     # Si le prix est vide, on l'ignore
27     if prix_clean:
28         # Remplacer la virgule par un point si nécessaire
29         prix_clean = prix_clean.replace(',', '.')
30
31         # Convertir en nombre
32         prix_float = float(prix_clean)
33         prix.append(prix_float)
34
35 # Patron de recherche pour les URL
36 pattern_url = r'http[s]?://(?:[a-zA-Z]{0-9}[$-_.&+][!*\(\)\[\]:%[0-9a-fA-F][0-9a-fA-F])+'
37 urls = re.findall(pattern_url, texte_reconnu)
38
39 # Stocker les prix et l'URL dans un fichier JavaScript
40 data = {
41     'prices': prix,
42     'url': urls[0] if urls else '' # Prend la première URL trouvée, s'il y en a une
43 }
44
45 with open('data.js', 'w') as js_file:
46     js_file.write(f'var extractedData = {json.dumps(data)};')
47
48 print('Données extraites et stockées dans data.js')
49
```

Le fichier ocr-projet-gui.py : c'est le projet avec l'interface graphique. C'est ce fichier que j'ai transformé avec pyinstaller en exécutable.