UNIVERSITÉ
PARIS DESCARTES
UFR DE MATHÉMATIQUES ET INFORMATIQUE

# Manuel d'installation

## *Podcast Generator*

# 1. Introduction (ou préambule)

Ce document servira de guide afin d'effectuer une installation de l'application Podcast_Generator_1.0.zip". Il expliquera à l'utilisateur final les bonnes pratiques permettant de l'installer sans encombre.

## 1.1. Objectifs et méthodes

Les objectifs de ce document sont d'aider l'utilisateur de l'application finale afin d'en profiter dans les meilleures conditions.

## 1.2. Documents de référence

Pour des informations sur l'utilisation de l'application, vous pouvez vous référer au manuel d'utilisation.

# 2. Guide de lecture

Ce document est accessible à un large public allant du néophyte au spécialiste. Aucune connaissance supplémentaire n'est nécessaire pour suivre correctement le guide fourni par ce document. Cependant il nécessite une configuration Windows ou MacOS.
Une configuration Linux pourrait marcher mais n'est pas garantie.

# 3.Concepts de base

**Podcast Generator :**
Un logiciel destiné à automatiser la création de podcasts audio à partir de dialogues textuels générés par intelligence artificielle. Le Podcast Generator utilise des modèles de langage pré-entraînés pour produire du contenu cohérent et pertinent.

**Modèles de langage (LLM - Large Language Models) :**
Des modèles d'intelligence artificielle entraînés sur de grandes quantités de textes permettant de générer du langage naturel réaliste. Ce projet utilise principalement les formats de modèles "GGUF".

**Format GGUF :**
Un format optimisé pour stocker des modèles de langage utilisés par `llama-cpp-python`. Il permet une performance optimale pour l'inférence locale (utilisation sur votre machine sans nécessiter de connexion internet).

**llama-cpp-python :**
Une bibliothèque Python servant à exécuter localement des modèles de langage au format GGUF. Elle nécessite une compilation spécifique en fonction de votre système d'exploitation pour garantir des performances optimales.

**Environnement virtuel Python (venv) :**
Un mécanisme permettant d'isoler les dépendances Python du projet afin d'éviter les conflits avec les autres applications Python installées sur votre système.

**Launcher (script de lancement) :**
Des fichiers exécutables (scripts batch pour Windows, shell pour macOS ou Linux) facilitant le démarrage rapide de l'application en configurant automatiquement l'environnement nécessaire.

# 4.Installation du matériel

Aucune installation matérielle spécifique n'est nécessaire.

# 5. Paramétrage du système

Avant de lancer le logiciel Podcast Generator, certaines opérations de configuration sont nécessaires pour garantir un fonctionnement correct.

**Système d'exploitation compatible** :

- Windows (testé et fonctionnel)
- macOS (compatible en théorie, non testé directement)
- Linux (potentiellement compatible, mais non garanti)

**Prérequis logiciels** :

- Python 3.11 doit être installé sur le système.
  Téléchargement disponible ici :
  https://www.python.org/downloads/release/python-3110/

**Étapes recommandées** :

Vérifier l'installation de Python.
Pour cela, ouvrir une invite de commande ou un terminal, puis exécuter :

python --version

La version affichée doit être Python 3.11.x.

# 6.Installation du logiciel

Cette section décrit les étapes nécessaires pour installer le projet Podcast Generator sur votre machine.

## Étapes à suivre :

### 1. Télécharger l'archive du projet

Récupérer le fichier nommé :

***Podcast_Generator_1.0.zip***

### 2. Extraire le contenu de l'archive

Décompresser l'archive dans un dossier de travail. Après extraction, la structure attendue est la suivante :

```
Podcast_Generator_1.0/
├── Models/
└── Podcast_Generator/
```

## 3. Installer la dépendance llama-cpp-python

Cette bibliothèque permet d'exécuter localement les modèles de langage au format GGUF utilisés par le projet.

Le dépôt officiel est disponible ici :

https://github.com/abetlen/llama-cpp-python

Compilez la librairie en suivant les instructions fournies sur le dépôt GitHub :

**Sous Windows**

**Prérequis** :

- Visual Studio installé avec les outils de développement C++ (Build Tools for Visual Studio)

**Étapes** :

-Ouvrir une invite de commandes développeur depuis Visual Studio, puis exécuter :

```
git clone https://github.com/abetlen/llama-cpp-python.git
cd llama-cpp-python
python -m pip install --upgrade pip
pip install scikit-build cmake ninja
pip install -e .
```

**Sous macOS**

**Prérequis** :

- Command Line Tools de Xcode (à installer si nécessaire)

**Commande d'installation** :

xcode-select --install

**Étapes** :

-Ouvrir le Terminal, puis exécuter :

git clone https://github.com/abetlen/llama-cpp-python.git
cd llama-cpp-python
python -m pip install --upgrade pip
pip install scikit-build cmake ninja
pip install -e .

## 4. Intégration au projet :

Une fois la compilation terminée, copier le dossier llama-cpp-python à la racine du projet, aux côtés des dossiers Models/ et Podcast_Generator/

Structure finale attendue :

```
Podcast_Generator_1.0/
├── Models/
├── Podcast_Generator/
└── llama-cpp-python/
```

# 7. Paramétrage du logiciel

Une fois l'installation terminée, des scripts de lancement sont fournis pour faciliter l'exécution du programme sans avoir à utiliser manuellement la ligne de commande.

**Sous Windows** :

Le fichier launcher_windows.bat permet à la fois d'installer automatiquement les dépendances Python (si nécessaire) et de lancer directement l'application.

Il suffit de double-cliquer sur ce fichier pour démarrer l'interface principale du programme.

**Sous macOS** :

Deux scripts sont fournis :

- install_mac.command : à exécuter une seule fois. Ce script attribue les permissions nécessaires au fichier de lancement principal. Il ne lance pas directement l'application.

- launcher_mac.command : une fois rendu exécutable grâce au script précédent, ce fichier permet de démarrer automatiquement l'application, de la même manière que sous Windows.

Pour rendre le fichier de lancement exécutable, ouvrir un terminal dans le dossier du projet et exécuter la commande suivante :

chmod +x install_mac.command

Une fois cette étape réalisée, il sera possible de lancer le programme à l'aide de :

./launcher_mac.command

**Remarque** :

Ces scripts de lancement automatisent la configuration de l'environnement virtuel Python et démarrent l'interface utilisateur. Ils sont recommandés pour une utilisation standard. En cas de besoin, il est toujours possible de lancer l'application manuellement avec la commande suivante :

python Podcast_Generator/mainTerminalUI.py

# 8.Installation des données

Le logiciel Podcast Generator repose sur des modèles de langage pré-entraînés au format GGUF. Ces modèles doivent être téléchargés séparément puis placés dans le dossier prévu à cet effet.

**Téléchargement des modèles** :

Il est recommandé d'utiliser le site Hugging Face pour récupérer les modèles au format GGUF :

https://huggingface.co/docs/hub/gguf

**Modèles conseillés pour de meilleurs résultats** :

- **Pour le <u>français et l'anglais</u>** :

***Mistral-Nemo-Instruct-2407-Q8_0***
***Lien :*** ***https://huggingface.co/Niansuh/Mistral-Nemo-Instruct-2407-Q8_0-GGUF/tree/main***

- **Pour le <u>japonais et le chinois</u> (simplifié et traditionnel) :**

***Qwen 1.5 - 7B - Chat Q8_0***
***Lien :*** ***https://huggingface.co/Qwen/Qwen1.5-7B-Chat-GGUF/tree/main***

D'autres modèles peuvent être utilisés, mais ceux indiqués ci-dessus sont testés et validés pour ce projet.

**Emplacement des fichiers** :

Une fois les modèles téléchargés, placez-les dans le dossier suivant :

Podcast_Generator_1.0/Models/

Remarques :

- Il est possible de placer plusieurs modèles dans le dossier Models/

# 9.Autres informations

Le projet peut fonctionner en version CPU seule, mais les performances sont significativement réduites avec les grands modèles comme Qwen ou Mistral Q8.

Certains ordinateurs portables anciens ou peu puissants peuvent rencontrer des ralentissements ou des erreurs de mémoire.

Configuration minimale recommandée :

Processeur (CPU) :

- Sous Windows : Intel Core i5 (10e génération ou plus) ou AMD Ryzen 5 (série 3000 ou plus)
- Sous macOS : Apple M1 ou supérieur, ou Intel i7 (8e génération minimum)

Mémoire RAM :

- 16 Go minimum requis
- 32 Go recommandés pour un fonctionnement fluide si plusieurs modèles sont utilisés

Stockage :

- SSD requis
- Environ 10 Go d'espace libre (modèles, dépendances, fichiers générés)

Carte graphique (facultatif) :

- Windows : GPU NVIDIA compatible CUDA (GTX 1660 Ti, RTX 2060 ou supérieur recommandé)
- macOS : le projet fonctionne en CPU uniquement, même sur Apple Silicon (GPU non utilisé directement par llama-cpp-python à ce jour)

Système d'exploitation :

- Windows 10 ou 11 (64 bits)
- macOS Monterey (12) ou version ultérieure

# 10. Annexes

Cette section regroupe des éléments complémentaires pouvant faciliter l'installation ou l'exploitation du projet Podcast Generator.

Fichiers inclus dans l'archive principale :

- **Podcast_Generator/** : dossier principal contenant le code source de l'application.
- **Models/** : dossier prévu pour stocker les modèles de langage téléchargés.
- **launcher_windows.bat** : script de lancement pour Windows.
- **install_mac.command** : script permettant d'autoriser le lancement sous macOS.
- **launcher_mac.command** : script de lancement principal pour macOS.
- **requirements.txt** : liste des dépendances Python nécessaires à l'exécution.

Recommandations générales :

- Conservez toujours une copie des modèles téléchargés en dehors du dossier d'installation pour éviter toute perte en cas de mise à jour ou de remplacement du projet.
- Si plusieurs utilisateurs travaillent sur le même poste, chacun peut créer son propre environnement virtuel Python.
- Les modèles utilisés peuvent être mis à jour manuellement si de nouvelles versions sont publiées, mais il est conseillé de tester leur compatibilité avec le projet.

Documentation complémentaire :

Un **manuel d'utilisation** séparé est fourni pour expliquer le fonctionnement détaillé de l'application, après installation.

Les scripts sont commentés pour faciliter leur compréhension et leur modification en cas de besoin.

# 11. Glossaire

Ce glossaire définit les principaux termes techniques utilisés dans ce manuel.

**Podcast Generator**
Logiciel permettant de générer automatiquement des podcasts à partir de dialogues textuels, en utilisant des modèles de langage et une synthèse vocale.

**Modèle de langage (LLM)**
Modèle d'intelligence artificielle capable de générer ou de compléter du texte en langage naturel. LLM signifie "Large Language Model".

**GGUF**
Format optimisé pour stocker des modèles de langage. Il est utilisé pour les modèles compatibles avec `llama-cpp-python`, permettant une exécution locale efficace.

**llama-cpp-python**
Bibliothèque Python permettant d'exécuter localement des modèles de langage au format GGUF, sans connexion internet. Nécessite une compilation spécifique par système.

**Environnement virtuel (venv)**
Fonctionnalité de Python qui permet d'isoler les dépendances d'un projet dans un répertoire dédié, afin d'éviter les conflits avec d'autres programmes.

**Launcher**
Script d'automatisation permettant de lancer rapidement une application. `launcher_windows.bat` et `launcher_mac.command` sont les launchers utilisés dans ce projet.

**Terminal / Invite de commande**
Interface en ligne de commande permettant d'exécuter des instructions sur le système (macOS/Linux : Terminal, Windows : Invite de commandes).

**Compilation**
Processus de transformation du code source (souvent en C/C++) en un fichier exécutable adapté au système d'exploitation de l'utilisateur.

**Hugging Face**
Plateforme de partage de modèles d'intelligence artificielle, utilisée ici pour télécharger les modèles de langage compatibles avec le projet.

# 12. Références

Liste des sources et documents complémentaires mentionnés dans ce manuel :

Dépôt officiel de `llama-cpp-python`
https://github.com/abetlen/llama-cpp-python

Téléchargement officiel de Python 3.11
https://www.python.org/downloads/release/python-3110/

Documentation Hugging Face sur les modèles GGUF
https://huggingface.co/docs/hub/gguf

Modèle recommandé pour le français et l'anglais
Mistral-Nemo-Instruct-2407-Q8_0
https://huggingface.co/Niansuh/Mistral-Nemo-Instruct-2407-Q8_0-GGUF/tree/main

Modèle recommandé pour le japonais et le chinois
Qwen 1.5 - 7B - Chat Q8_0
https://huggingface.co/Qwen/Qwen1.5-7B-Chat-GGUF/tree/main

Manuel d'utilisation (fichier séparé, fourni avec le projet)

# 13. Index

Ce tableau indexe les termes importants utilisés dans le présent manuel, ainsi que les sections où ils apparaissent pour la première fois ou de manière significative.

**G**



**H**



**I**



**L**



**M**

**P**



**Q**



**R**



**S**



**T**



**U**



**W**