Procédure complète - Création d'un exécutable PyInstaller pour PromptoDYS

📋 PRÉREQUIS

Avant de commencer, vous devez avoir :

- ✅ Python 3.11+ installé
- ✅ Votre projet PromptoDYS fonctionnel
- ✅ Le dossier `build_web/` (pas `build/`) avec votre interface React
- ✅ Les fichiers `askGeminiPrompto.py`, `prompt.md` prêts
- ✅ Un environnement virtuel (.venv) activé

🎯 ÉTAPE 1 : PRÉPARATION DES FICHIERS

1.1 Vérification de l'arborescence

Ouvrez PowerShell dans votre dossier projet et vérifiez :

```powershell
Get-ChildItem
```

Vous devez voir :

- `askGeminiPrompto.py` (script principal)
- `build_web/` (interface React)
- `prompt.md` (template IA)
- `.venv/` (environnement virtuel)
- `reports/` (peut être vide)

1.2 Vérification du script Python

Votre script doit contenir cette fonction modifiée pour PyInstaller :

```python
```

```
def find_web_folder():
    """Trouve le dossier contenant index.html (compatible PyInstaller)"""
    import sys

    # Si on est dans un exécutable PyInstaller
    if hasattr(sys, '_MEIPASS'):
        base_path = sys._MEIPASS
        possible_folders = ['build_web', 'build']
    else:
        # Mode développement normal
        base_path = '.'
        possible_folders = ['build_web', 'build', 'dist', 'public', '.']

    for folder in possible_folders:
        index_path = os.path.join(base_path, folder, 'index.html')
        if os.path.exists(index_path):
            print(f"✅ Dossier web trouvé: {folder}/")
            return os.path.join(base_path, folder)

    print("⌐ Aucun index.html trouvé dans:", possible_folders)
    return None
```

1.3 Création du fichier de clé API vierge

```
powershell

echo "COLLEZ_VOTRE_CLE_API_GEMINI_ICI" > GeminiKey.txt
```

🔧 ÉTAPE 2 : INSTALLATION DE PYINSTALLER

2.1 Activation de l'environnement virtuel

```
powershell

.venv\Scripts\Activate.ps1
```

Vous devez voir `(.venv)` au début de votre invite de commande.

2.2 Installation de PyInstaller

```
powershell

pip install pyinstaller
```

2.3 Vérification de l'installation

```powershell
pyinstaller --version
```

Vous devriez voir quelque chose comme `6.15.0`.

📄 ÉTAPE 3 : CRÉATION DU FICHIER DE SPÉCIFICATION

3.1 Génération du fichier .spec de base

```powershell
pyinstaller --name=PromptoDYS --onefile --console askGeminiPrompto.py
```

Important : Utilisez `--console` pour voir les erreurs pendant les tests.

3.2 Modification du fichier .spec

Ouvrez le fichier `PromptoDYS.spec` créé :

```powershell
notepad PromptoDYS.spec
```

Remplacez TOUT le contenu par :

```python
```

```
# -*- mode: python ; coding: utf-8 -*-

a = Analysis(
    ['askGeminiPrompto.py'],
    pathex=[],
    binaries=[],
    datas=[
        ('build_web/index.html', 'build_web'),
        ('build_web/static', 'build_web/static'),
        ('build_web/fonts', 'build_web/fonts'),
        ('build_web/asset-manifest.json', 'build_web'),
        ('build_web/favicon.png', 'build_web'),
        ('prompt.md', '.'),
        ('GeminiKey.txt', '.'),
    ],
    hiddenimports=[
        'eel',
        'google.genai',
        'reportlab',
        'markdown',
        'gevent',
        'gevent.socket',
        'gevent.ssl',
    ],
    hookspath=[],
    hooksconfig={},
    runtime_hooks=[],
    excludes=[],
    noarchive=False,
    optimize=0,
)

pyz = PYZ(a.pure)

exe = EXE(
    pyz,
    a.scripts,
    a.binaries,
    a.datas,
    [],
    name='PromptoDYS',
    debug=False,
    bootloader_ignore_signals=False,
    strip=False,
    upx=True,
    upx_exclude=[],
```

```
    runtime_tmpdir=None,
    console=True,   # True pour tests, False pour version finale
    disable_windowed_traceback=False,
    argv_emulation=False,
    target_arch=None,
    codesign_identity=None,
    entitlements_file=None,
)
```

Sauvegardez et fermez le fichier.

⚙️ ÉTAPE 4 : COMPILATION DE L'EXÉCUTABLE

4.1 Nettoyage des fichiers temporaires

```powershell
Remove-Item -Recurse -Force build -ErrorAction SilentlyContinue
Remove-Item -Recurse -Force dist -ErrorAction SilentlyContinue
Remove-Item -Recurse -Force __pycache__ -ErrorAction SilentlyContinue
```

4.2 Compilation avec le fichier .spec

```powershell
pyinstaller PromptoDYS.spec
```

La compilation prend 2-5 minutes. Attendez le message "Build complete!"

4.3 Vérification de l'exécutable créé

```powershell
Get-ChildItem dist\PromptoDYS.exe | Select-Object Name, Length
```

L'exécutable doit faire environ 40-50 MB.

🧪 ÉTAPE 5 : TEST DE L'EXÉCUTABLE

5.1 Test initial (avec console visible)

```powershell
.\dist\PromptoDYS.exe
```

Vérifiez que :

- ✅ L'interface web s'ouvre dans Chrome
- ✅ Le menu console apparaît dans PowerShell
- ✅ Pas de messages d'erreur critiques

5.2 Test de fonctionnalité basique

Dans l'interface web :

1. Tapez du texte dans l'éditeur
2. Dans la console PowerShell, tapez `1` et validez
3. Vérifiez que le texte s'affiche

5.3 Si tout fonctionne : Version finale sans console

Éditez `PromptoDYS.spec` et changez :

```
python

console=False,  # Masque la console pour l'utilisateur final
```

Recompilez :

```
powershell

pyinstaller PromptoDYS.spec
```

📦 ÉTAPE 6 : CRÉATION DU PACKAGE POUR DISTRIBUTION

6.1 Création de la structure du package

```
powershell

mkdir PromptoDYS_Portable
```

6.2 Copie des fichiers nécessaires

```
powershell

Copy-Item dist\PromptoDYS.exe PromptoDYS_Portable\
Copy-Item GeminiKey.txt PromptoDYS_Portable\
```

6.3 Création du README pour l'utilisateur

Créez `PromptoDYS_Portable\README.txt` avec :

🚀 PromptoDYS - Éditeur Markdown + IA Gemini

⚡ DÉMARRAGE RAPIDE (2 minutes) :

1. 🔑 Obtenez votre clé API Gemini :
   → Allez sur https://aistudio.google.com/
   → Créez une clé API gratuite

2. 🖍️ Configurez votre clé :
   → Ouvrez le fichier "GeminiKey.txt"
   → Remplacez le texte par votre clé API
   → Sauvegardez le fichier

3. 🚀 Lancez l'application :
   → Double-cliquez sur "PromptoDYS.exe"
   → Attendez l'ouverture de la fenêtre
   → ✅ C'est parti !

💡 UTILISATION :
- Tapez votre texte dans l'éditeur web
- Utilisez le menu console pour le traitement IA (option 3)
- Vos rapports sont sauvegardés automatiquement

⚠️ IMPORTANT : Gardez votre clé API secrète !

📧 Support : [votre email]

6.4 Compression du package final

```powershell
Compress-Archive -Path PromptoDYS_Portable\* -DestinationPath PromptoDYS_v1.0.zip
```

✅ VÉRIFICATIONS FINALES

Liste de contrôle avant distribution :

- [ ] L'exécutable se lance sans erreur
- [ ] L'interface web s'affiche correctement
- [ ] Le fichier GeminiKey.txt contient le placeholder
- [ ] Le README.txt est clair et complet
- [ ] Le package ZIP contient tous les fichiers
- [ ] Taille du ZIP raisonnable (< 50 MB)

🔧 DÉPANNAGE COURANT

Problème : "Aucun index.html trouvé"

➡️ Vérifiez que `build_web/` existe et contient `index.html` ➡️ Recompilez avec PyInstaller

Problème : L'exe ne démarre pas

➡️ Changez `console=True` dans le .spec ➡️ Recompilez et regardez les messages d'erreur

Problème : Interface web noire

➡️ Vérifiez que tous les fichiers `static/` sont inclus ➡️ Vérifiez la fonction `find_web_folder()` dans le script

Problème : Erreur de compilation PyInstaller

➡️ Nettoyez les dossiers temporaires (étape 4.1)
➡️ Vérifiez que l'environnement virtuel est activé

📝 NOTES IMPORTANTES

1. **Environnement virtuel** : Toujours compiler depuis l'environnement virtuel où les dépendances sont installées.
2. **Dossier build_web** : Ne jamais le renommer en `build` car PyInstaller crée son propre dossier `build/` temporaire.
3. **Console mode** : Utilisez `console=True` pour les tests, `console=False` pour la distribution finale.
4. **Taille de l'exécutable** : 40-50 MB est normal pour une application avec toutes ces dépendances.
5. **Antivirus** : Les exécutables PyInstaller peuvent déclencher des faux positifs d'antivirus. C'est normal.

🚀 PROCHAINE FOIS : COMMANDES RAPIDES

Quand tout est configuré, pour recompiler rapidement :

```powershell
# Nettoyage
Remove-Item -Recurse -Force build, dist, __pycache__ -ErrorAction SilentlyContinue

# Compilation
pyinstaller PromptoDYS.spec

# Test
.\dist\PromptoDYS.exe
```

Procédure créée pour PromptoDYS - Mise à jour : September 2025