Guide Complet pour Déployer Votre Application Flask avec VSCode

Prérequis

1. **Python 3.x** installé sur votre Mac.
2. **Visual Studio Code (VSCode)** installé.
3. **Git** installé (optionnel, mais recommandé pour la gestion du code).
4. **Navigateur Web** (comme Chrome, Firefox, ou Safari).

Étape 1 : Configurer le Projet dans VSCode

1. **Ouvrir VSCode** :
   - Lancez VSCode depuis le Finder ou via le Terminal en tapant `code`.
2. **Créer un Nouveau Dossier de Projet** :
   - Dans VSCode, allez dans **File > Open** et créez un nouveau dossier, par exemple `App-IA`.
3. **Organiser la Structure des Fichiers** :
   - À l'intérieur de `App-IA`, créez la structure suivante :

```
arduino
Copier le code
App-IA/
├── app.py
├── models/
│   ├── Model-20241102_105616.keras
│   └── class_labels_20241102_105616.npy
├── templates/
│   └── index.html
├── static/
│   └── script.js
└── requirements.txt
```

4. **Ajouter Vos Fichiers Existants** :
   - **app.py** : Collez le contenu de votre fichier `app.py`.
   - **templates/index.html** : Collez le contenu de votre fichier `index.html`.
   - **static/script.js** : Collez le contenu de votre fichier `script.js`.
   - **models/** : Assurez-vous que vos fichiers modèle `.keras` et `.npy` sont placés ici.

Étape 2 : Configurer un Environnement Virtuel

1. **Ouvrir le Terminal Intégré de VSCode** :
   - Dans VSCode, allez dans **View > Terminal** ou utilisez le raccourci `Ctrl+`.
2. **Naviguer vers le Dossier du Projet** :

```
bash
Copier le code
cd /Users/jeremygay/Downloads/App-IA/
```

3. **Créer un Environnement Virtuel** :

```
bash
Copier le code
python3 -m venv venv
```

4. **Activer l'Environnement Virtuel** :

```
bash
Copier le code
source venv/bin/activate
```

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

Étape 3 : Installer les Dépendances

1. **Créer un Fichier `requirements.txt`** :
   - Dans VSCode, créez un fichier nommé `requirements.txt` avec le contenu suivant :

```
plaintext
Copier le code
Flask
tensorflow
opencv-python
numpy
matplotlib
pandas
seaborn
chart.js
```

2. **Installer les Packages** :

```
bash
Copier le code
pip install -r requirements.txt
```

   - Cela installera Flask, TensorFlow, OpenCV, et autres dépendances nécessaires.

Étape 4 : Vérifier les Chemins des Fichiers

Assurez-vous que dans `app.py`, les chemins vers le modèle et les labels sont corrects.
Votre `app.py` devrait contenir :

```
python
Copier le code
# Obtenir le chemin absolu du répertoire courant
BASE_DIR = os.path.dirname(os.path.abspath(__file__))

# Chemins absolus vers le modèle et les labels
MODEL_PATH = os.path.join(BASE_DIR, 'models', 'Model-
20241102_105616.keras')
CLASS_LABELS_PATH = os.path.join(BASE_DIR, 'models',
'class_labels_20241102_105616.npy')
```

Étape 5 : Lancer l'Application Flask

1. **Assurez-vous que l'Environnement Virtuel est Actif** :
   - Vous devriez voir `(venv)` dans votre terminal.
2. **Exécuter `app.py`** :

   ```
   bash
   Copier le code
   python app.py
   ```

   - Vous devriez voir un message indiquant que Flask est en cours d'exécution, par exemple :

     ```
     csharp
     Copier le code
     * Serving Flask app 'app'
     * Debug mode: on
     * Running on http://0.0.0.0:5001/ (Press CTRL+C to quit)
     ```

Étape 6 : Accéder à l'Application via le Navigateur

1. **Sur Votre Mac** :
   - Ouvrez votre navigateur et allez à http://127.0.0.1:5001/.
2. **Depuis un Téléphone (Optionnel)** :
   - Assurez-vous que votre téléphone est connecté au même réseau Wi-Fi que votre Mac.
   - Trouvez l'adresse IP de votre Mac :

     ```
     bash
     Copier le code
     ifconfig | grep inet
     ```

     - Recherchez une adresse locale, par exemple `192.168.1.5`.
   - Accédez à l'application depuis le navigateur de votre téléphone à `http://192.168.1.5:5001/`.

Étape 7 : Tester les Fonctionnalités

1. **Classification via Webcam** :
   - Autorisez l'accès à la webcam lorsque le navigateur le demande.
   - Cliquez sur **"Capturer et Classifier"** pour prendre une photo et voir le résultat avec le graphique de confiance et l'historique des classifications.
2. **Téléchargement d'Images** :
   - Cliquez sur **"Télécharger une Image"**, sélectionnez une image depuis votre ordinateur.
   - Cliquez sur **"Télécharger et Classifier"** pour voir le résultat avec la prévisualisation de l'image, le graphique de confiance et l'historique.

Étape 8 : Personnaliser et Améliorer l'Interface

1. **Modifier le Style avec Bootstrap** :
   - Vous avez déjà intégré Bootstrap via CDN dans `index.html`. Vous pouvez personnaliser davantage en ajustant les classes Bootstrap ou en ajoutant votre propre CSS dans la section `<style>`.

2. **Ajouter des Graphiques avec Chart.js** :
    - Les graphiques de confiance sont déjà intégrés. Vous pouvez personnaliser les couleurs, les types de graphiques (par exemple, camembert), ou ajouter des animations en modifiant les options dans `script.js`.
3. **Ajouter des Options Supplémentaires** :
    - **Historique des Classifications** : Vous pouvez ajouter des fonctionnalités pour sauvegarder cet historique ou le filtrer.
    - **Téléchargement d'Images en Masse** : Permettre le téléchargement multiple d'images pour des classifications en lot.
    - **Heatmaps ou Grad-CAM** : Intégrer des visualisations supplémentaires pour mieux comprendre les prédictions du modèle.

Étape 9 : Débogage et Optimisation

1. **Déboguer dans VSCode** :
    - Utilisez les fonctionnalités de débogage de VSCode pour placer des points d'arrêt et inspecter les variables.
    - Installez l'extension **Python** pour une meilleure intégration.
2. **Optimiser les Performances** :
    - **Modèle TensorFlow Lite** : Envisagez de convertir votre modèle en TensorFlow Lite pour des inférences plus rapides, surtout si vous ciblez des appareils mobiles.
    - **Asynchrone** : Implémentez des requêtes asynchrones pour améliorer la réactivité de l'application.
3. **Sécurité** :
    - **Validation des Entrées** : Assurez-vous que les images téléchargées sont valides et sécurisées.
    - **Limiter l'Accès** : Si l'application est accessible sur un réseau public, envisagez d'ajouter une authentification.

Étape 10 : Maintenir et Mettre à Jour le Projet

1. **Gérer les Dépendances** :
    - Pour ajouter de nouvelles dépendances, mettez à jour `requirements.txt` et réinstallez-les :

      ```
      bash
      Copier le code
      pip install <nom_du_package>
      pip freeze > requirements.txt
      ```

2. **Versionner le Code avec Git** (Optionnel) :
    - Initialisez un dépôt Git pour suivre les modifications :

      ```
      bash
      Copier le code
      git init
      git add .
      git commit -m "Initial commit"
      ```

3. **Sauvegarder le Projet** :

- Sauvegardez régulièrement votre travail et envisagez d'utiliser des services comme GitHub pour héberger votre code.

Résumé

En suivant ce guide, vous avez configuré un environnement de développement Flask dans VSCode, organisé vos fichiers, installé les dépendances nécessaires, et lancé une application web interactive pour la classification des déchets. L'interface utilisateur a été améliorée avec Bootstrap pour un design moderne et Chart.js pour visualiser les probabilités de classification. Vous avez également ajouté des fonctionnalités supplémentaires telles que le téléchargement d'images et un historique des classifications.

N'hésitez pas à continuer à personnaliser votre application en fonction de vos besoins spécifiques. Si vous rencontrez des problèmes ou avez besoin de fonctionnalités supplémentaires, je suis là pour vous aider !