🐾 Cat Identifier

SYSTÈME DE RECONNAISSANCE FÉLINE EN TEMPS RÉEL – VERSION WEB FLASK

📅 Dernière version : mai 2025
📍 Auteur : Yann Bucaille

📝 Introduction

Cat Identifier est une application web développée en **Flask** permettant d'identifier automatiquement des chats via webcam.
Elle fonctionne en deux étapes principales :

- **Détection** des chats dans l'image à l'aide d'un modèle **YOLOv11**.
- **Classification** fine pour déterminer lequel de vos chats est présent grâce à un **ResNet18 fine-tuné**.

L'interface peut être utilisée **via la webcam du PC ou du smartphone** (connexion réseau locale).

🌐 Lancement et usage

- Lancement du serveur Flask :

```
python app.py
```

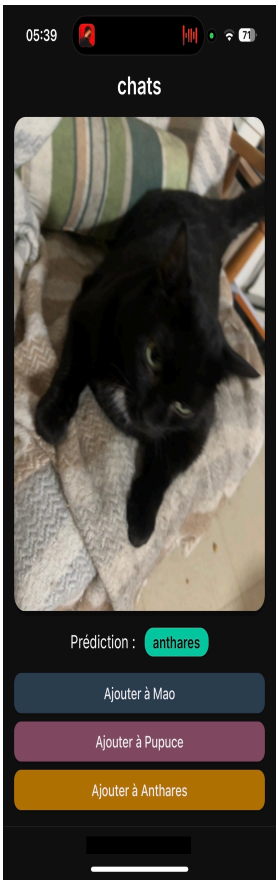
- L'application est disponible en local à l'adresse :

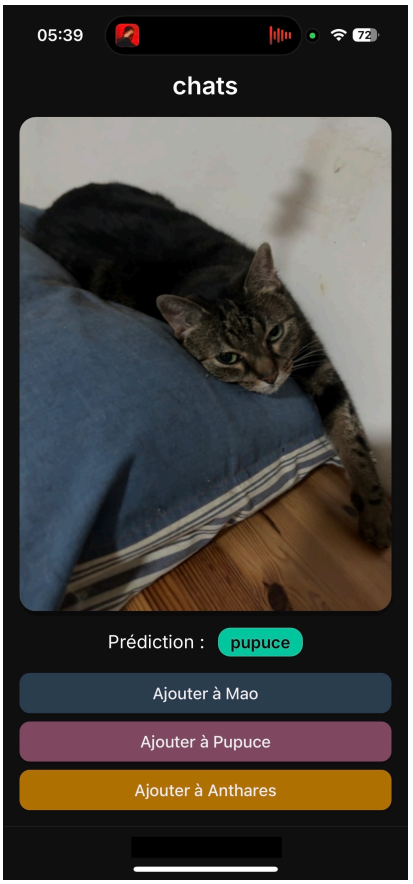
```
http://localhost:5050
```

- Une fois lancé, elle accède à la **caméra** (navigateur), traite les frames en direct et affiche :
  - une **boîte de détection**,
  - le **nom du chat identifié**,
  - et enregistre les images si la confiance est suffisante pour entraînement futur.

Interface et Reconnaissance

Voici quelques captures d'écran de l'interface web et du système de reconnaissance en action (via webcam).





🔧 Pipeline du système

Étape	Description
Détection	YOLO11n repère les chats dans la webcam (`models/yolo11n.pt`)
Recadrage	Extraction de la zone détectée pour classification
Classification	Le modèle ResNet18 (`cats_identifier_last.pth`) prédit l'identité du chat
Collecte	Les images de qualité sont stockées dans `dataset/<nom_du_chat>/`
Réentraînement	Dès qu'un seuil d'images est atteint, le script relance un entraînement

🧱 Arborescence du projet

```
predict_live_v1.2_stable/
├── app.py                          # Serveur web Flask principal
```

```
├── train_classifier_auto.py      # Entraînement automatique du classifieur
├── validate_model.py             # Validation du modèle avant de le remplacer
pour la détection
├── templates/                    # Page html
├── static/                       # css/Js
├── models/
│   ├── yolo11n.pt               # Modèle YOLOv11 pour la détection
│   └── cats_identifier_last.pth # Classificateur individuel (ResNet18)
├── dataset/
│   ├── Pupuce/                  # Dossier auto-rempli pendant la capture
│   ├── Anthares/
│   └── Pupuce/
├── logs/
│   └── last_train.json          # Historique des entraînements
├── utils/                       # Scripts de détection, préprocessing, etc.
└── requirements.txt             # Libs nécessaires au lancement.
```

📈 Performances observées

- **Précision** : ~92 % en condition d'éclairage normal.
- **Latence** : ~60 ms/image sur GPU.
- **Collecte** : automatisée, permet une amélioration continue.

🧭 Prochaines améliorations envisagées

- Ajout du **tracking inter-frame** (DeepSort, supervision ID).
- **Validation manuelle des images** avant entraînement.
- Mode multi-utilisateur avec base de chats distincte par profil.
- Déploiement sur **Jetson Nano / Raspberry Pi + Coral TPU**.

📎 Notes

- Le projet est pensé comme un **système vivant** : plus vous l'utilisez, plus il devient précis.
- Les entraînements sont relancés **dès qu'un nombre minimum de nouvelles images** est atteint, sans intervention manuelle.

✅ Pour tester

- Lance `app.py`
- Accède à l’interface via navigateur sur téléphone
- Assure-toi que la caméra est autorisée
- Observe l'identification, et regarde le dossier `dataset/` se remplir