

**Rapport technique n°1— Choix et justification des technologies**

**SAE 5.01** — Développement d’une application mobile de reconnaissance d’objets avec Intelligence Artificielle

**Projet :** SneakScan — Application de reconnaissance de sneakers

**Date :** 17/10/2025

**Lieu :** IUT Technologique Saulcy à Metz

**Membres du projet :**

BRAHMIA Noureddine

RAHUEL Bastien

BOUCHATROUCH Kaiss

## **1. Introduction et contexte**

Le projet **SneakScan** s’inscrit dans le cadre de la **SAE 5.01**, dont l’objectif est de développer une **application mobile** intégrant des fonctionnalités de **reconnaissance d’objets** à l’aide de l’**intelligence artificielle (IA)**.

Notre groupe a choisi d’appliquer cette thématique à la **reconnaissance automatique de sneakers** (chaussures de sport). L’idée est de concevoir une application capable d’**identifier un modèle précis** de sneaker à partir d’une simple photo prise avec la caméra du smartphone.

Les fonctionnalités principales prévues sont :

* Capturer une image à l’aide de la caméra du téléphone ;
* Identifier le modèle exact de la sneaker (et pas seulement la marque) ;
* Afficher le résultat de la reconnaissance avec un pourcentage de confiance ;
* Permettre la sauvegarde et le partage des résultats.

Ce projet combine plusieurs domaines de l’informatique :

* **La vision par ordinateur**, qui permet à la machine de comprendre et d’analyser des images ;
* **Le deep learning**, une technique d’apprentissage automatique reposant sur des réseaux de neurones ;
* Et le **développement mobile multiplateforme**, pour rendre l’application disponible sur Android et iOS.

## **2. Objectifs techniques**

L’objectif principal du projet est de concevoir une **application mobile multiplateforme** intégrant un **modèle d’intelligence artificielle** capable de reconnaître en temps réel les modèles de sneakers.

Les sous-objectifs sont :

* Créer une interface fluide et ergonomique (bonne **expérience utilisateur**, ou UX) ;
* Développer et intégrer un modèle d’IA entraîné sur un **ensemble d’images (dataset)** de sneakers ;
* Documenter l’ensemble du processus, du **recueil des données** jusqu’à l’**intégration du modèle sur mobile**.

## **3. Technologies retenues**

|  |  |  |
| --- | --- | --- |
| **Domaine** | **Technologie** | **Rôle principal** |
| Développement mobile | **React Native (Expo)** | Développement d’une application Android/iOS et de son interface utilisateur |
| Intelligence artificielle | **TensorFlow.js** | Exécution du modèle IA directement sur le téléphone |
| Modèle de base | **MobileNetV2 (transfert d’apprentissage)** | Modèle de classification d’images optimisé pour mobile |
| Entraînement du modèle | **Google Colab / Keras** | Ajustement (fine-tuning) du modèle sur le dataset de sneakers |
| Hébergement du modèle | **Firebase / GitHub Pages** | Stockage et chargement du modèle exporté |
| Gestion de la caméra | **Expo Camera API** | Capture d’images en temps réel |
| Dataset | **Kaggle + images originales** | Base d’images d’entraînement |

## **4. Constitution du dataset et entraînement**

Le modèle doit être capable d’identifier des modèles précis tels que **Nike Air Force 1**, **Adidas Yeezy 350**, ou **Air Jordan 1**.

**Sources des données :**

* Jeux d’images publics provenant de **Kaggle** ;
* Photos prises directement par l’équipe ;
* Nettoyage et **annotation manuelle** des images pour attribuer la bonne classe à chaque photo.

**Structure du dataset :**

* 4 classes principales : *AirJordan1*, *NikeAirForce1*, *AdidasYeezy350*, *ConverseChuck* ;
* Environ 150 à 300 images par classe, soit environ 1 200 à 2 000 images au total.

**Paramètres d’entraînement :**

* Modèle de base : *MobileNetV2* pré-entraîné sur *ImageNet* ;
* Optimiseur : *Adam*, taux d’apprentissage (*learning rate*) = 1e-4 ;
* Taille de lot (*batch size*) = 32 ;
* Environ 30 itérations (*epochs*) ;
* Augmentation de données (rotation, zoom, recadrage) pour éviter le surapprentissage.

Le modèle atteint une **précision de validation entre 88 % et 93 %** selon les classes.  
 Il est ensuite exporté en format **.h5**, puis converti pour **TensorFlow.js**.

## **5. Comparaison des solutions techniques**

### **a. Développement mobile**

|  |  |  |  |
| --- | --- | --- | --- |
| **Critère** | **Flutter** | **React Native** | **Android (Kotlin)** |
| Langage | Dart | JavaScript/TypeScript | Kotlin |
| Multiplateforme | Oui | Oui | Non |
| Performance | Élevée | Bonne | Élevée |
| Facilité d’apprentissage | Moyenne | Facile | Moyenne |
| Intégration IA | Complexe | Simple (via TFJS) | Très bonne (via TFLite) |
| Temps de développement | Long | Court | Moyen |

**Choix final : React Native (Expo)**

Ce framework permet un développement rapide, une bonne compatibilité multiplateforme et une intégration native de TensorFlow.js.

### **b. Framework IA**

|  |  |  |  |
| --- | --- | --- | --- |
| **Critère** | **TensorFlow Lite** | **TensorFlow.js** | **PyTorch Mobile** |
| Langage | Java/C++/Swift | JavaScript | C++/Java |
| Intégration avec React Native | Complexe | Native | Faible |
| Exécution locale | Oui | Oui | Oui |
| Taille du modèle | Légère | Moyenne | Moyenne |
| Conversion du modèle | Oui (.tflite) | Oui (.json) | Non |

**Choix final : TensorFlow.js**

Ce framework permet l’exécution directe du modèle sur le téléphone, en JavaScript, sans dépendances natives.

### **c. Modèle IA**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Modèle** | **Type** | **Taille** | **Vitesse** | **Précision** | **Complexité** |
| MobileNetV2 | Classification | 14 Mo | Rapide | Bonne | Simple |
| YOLOv8 | Détection et classification | 80 Mo | Moyenne | Excellente | Élevée |
| EfficientNet | Classification | 30 Mo | Rapide | Très bonne | Moyenne |

**Choix final : MobileNetV2**

Léger, rapide et adapté aux appareils mobiles tout en offrant une bonne précision.

## **6. Architecture du projet**

+-----------------------------------------------------+

| SneakScan App |

|-----------------------------------------------------|

| React Native (Expo) |

| - Interface utilisateur |

| - Capture d’image (Expo Camera) |

| - Conversion en tenseur |

| - Chargement du modèle TFJS |

| - Affichage du résultat (+ pourcentage confiance) |

|-----------------------------------------------------|

| Modèle IA : MobileNetV2 (entraîné et converti) |

| - Dataset sneakers (4 classes principales) |

| - Hébergement sur Firebase / GitHub Pages |

+-----------------------------------------------------+

## **7. Avantages du choix technologique**

* **Performance mobile** : inférence rapide (~150–200 ms/image) ;
* **Déploiement simplifié** : aucune dépendance native à installer ;
* **Maintenance aisée** : code entièrement en JavaScript ;
* **Compatibilité multiplateforme** : Android et iOS ;
* **Évolutivité** : ajout de nouvelles classes d’objets facilement réalisable.

## **8. Limites et perspectives d’évolution**

|  |  |
| --- | --- |
| **Limite identifiée** | **Amélioration envisagée** |
| Peu de modèles de sneakers dans le dataset | Enrichir la base d’images |
| Reconnaissance d’un seul objet à la fois | Ajouter la détection multi-objets (ex : YOLO, SSD) |
| Modèle chargé uniquement en ligne | Implémenter un cache local pour l’utilisation hors ligne |
| Entraînement long et consommateur de ressources | Utiliser Google Colab pour bénéficier de GPU gratuits |

## **9. Sécurité et respect de la vie privée**

* Les images capturées restent **locales** sur le téléphone de l’utilisateur ;
* Aucun transfert externe sans consentement explicite ;
* Utilisation exclusive de modèles et de bibliothèques **open source**, sans collecte de données personnelles.

## **10. Conclusion**

Le choix du trio **React Native + TensorFlow.js + MobileNetV2** représente un excellent compromis entre performance, simplicité et portabilité.  
 Ce projet démontre notre capacité à mettre en œuvre une **chaîne complète d’intelligence artificielle** sur mobile, de l’entraînement du modèle à son intégration dans une application fonctionnelle.

SneakScan constitue une base solide pour de futures évolutions, notamment la **détection multi-objets**, la **reconnaissance de nouveaux modèles** ou encore une **authentification automatique** des sneakers.

