# 📄 Cahier des Charges

## 1. Introduction

Le présent cahier des charges décrit l’ensemble des besoins fonctionnels et techniques pour le développement d’une application éducative destinée aux enfants autistes, dont tu seras le développeur unique. Il sert de référence pour la conception, le développement, les tests et la mise en production. 🛠️

## 2. Objectifs du Projet

• 🎯 But principal : Concevoir une application mobile/desktop ludique et pédagogique adaptée aux besoins des enfants TSA (3–10 ans).  
• 👥 Utilisateurs : Enfants diagnostiqués TSA, parents et professionnels (enseignants, thérapeutes).  
• 🌟 Bénéfices attendus : Amélioration des compétences cognitives (logos, mémoire, reconnaissance émotionnelle), apprentissage de l’alphabet, des objets quotidiens et des chiffres.

## 3. Périmètre (Portée)

• 📱 Plateformes ciblées : Android, iOS, Windows, macOS, Linux.  
• 🔄 Modules inclus : Jeux (puzzle, mémoire, émotions) et rubriques éducatives (alphabet, objets, chiffres).  
• 🚫 Exclusion : Module de suivi avancé des performances, module de communication inter-utilisateurs.

## 4. Spécifications Fonctionnelles

### 4.1. Jeux  
1. 🧩 Puzzle  
 - 10 images par thème (animaux, objets), 3 niveaux de difficulté.  
 - Drag & drop, retour visuel et sonore à chaque pièce placée.  
2. 🧠 Jeu de Mémoire  
 - 12 paires d’images/sons.  
 - Chronomètre facultatif.  
3. 😃 Jeu des Émotions  
 - 8 cartes d’expressions faciales.  
 - Reconnaissance (choix sur photo) et association (nom de l’émotion).  
  
### 4.2. Rubriques Éducatives  
1. 🔤 Alphabet  
 - 26 lettres interactives avec prononciation.  
 - Exemple d’un mot illustré par lettre.  
2. 🏠 Objets du Quotidien  
 - 5 catégories (maison, école, cuisine, nature, vêtements), 10 items chacune.  
 - Audio prononciation et image.  
3. 🔢 Chiffres  
 - Nombres 1–10 (puis 20, 50, 100).  
 - Illustration graphique du concept de quantité.

## 5. Exigences Non Fonctionnelles

• ⚡ Performance : < 1 seconde de chargement par écran.  
• ♿ Accessibilité : Contraste adapté, textes redimensionnables, navigation simplifiée.  
• 🔒 Sécurité : Stockage local des données utilisateur chiffré.  
• 🌐 Internationalisation : Support du français; prévoir structure multi-langues.

## 6. Architecture Technique

• 🐍 Langage : Python 3.8+  
• 🖼️ Framework UI : Kivy (version 2.1.0 ou supérieure)  
• 📋 Langage de layout : Fichiers .kv pour la définition des interfaces  
• 🗄️ Base de données : SQLite via SQLAlchemy ou Kivy Storage (JsonStore)  
• ☁️ API Cloud (optionnel) : Firebase REST ou AWS S3/Cognito  
• 🔄 CI/CD : Buildozer pour Android, PyInstaller/briefcase pour desktop  
dev (Windows/macOS/Linux)

## 7. Contraintes

• 👤 Développement Solo : Code clair, commenté, architecture modulaire.  
• 💰 Budget : Outils open-source.  
• 🗓️ Délai : MVP en 3 mois, lancement officiel en 6 mois.

## 8. Planning Prévisionnel

|  |  |  |
| --- | --- | --- |
| Phase | Durée estimée | Période |
| Spécifications | 2 semaines | Semaine 1–2 |
| Prototype UI | 3 semaines | Semaine 3–5 |
| Développement MVP | 6 semaines | Semaine 6–11 |
| Tests Utilisateurs | 3 semaines | Semaine 12–14 |
| Corrections & Release | 4 semaines | Semaine 15–18 |

## 9. Gestion des Risques

• 🔧 Risque technique : Limitations de Kivy sur desktop → tests multi-OS précoces.  
• ⏳ Risque planning : Retard de développement → Buffer de 2 semaines.

## 10. Livrables

• 📄 Document de spécifications validé.  
• 🔄 Prototype interactif Kivy.  
• 💻 Application MVP multi-plateforme (APK, exécutables).  
• 📊 Rapport de tests utilisateurs.

## 11. Annexes

• 📖 Glossaire des termes.  
• 📚 Références bibliographiques sur l’autisme et l’UI adaptée.

## 12. Structure du Projet (Architecture de Dossiers)

Pour un développement Kivy sous VS Code en Python, voici une organisation recommandée :  
  
mon\_app\_autisme/ # Racine du projet  
├── src/ # Code source de l'application  
│ ├── main.py # Point d'entrée (Builder.load\_file, App.run)  
│ ├── app/ # Package principal  
│ │ ├── \_\_init\_\_.py  
│ │ ├── ui/ # Fichiers .kv et widgets personnalisés  
│ │ │ ├── main.kv  
│ │ │ ├── screens/ # Screens .kv et classes Python associées  
│ │ │ └── widgets/ # Composants Kivy personnalisés  
│ │ ├── models/ # Logique métier et classes de données  
│ │ ├── controllers/ # Gestionnaires d’événements et coordination  
│ │ └── assets/ # Images, sons, polices  
│ └── tests/ # Tests unitaires et d’intégration (pytest)  
│ ├── test\_models.py  
│ └── test\_ui.py  
├── resources/ # Fichiers complémentaires (docs, cahier des charges)  
│ └── images/ # Images générales du projet  
├── docs/ # Documentation (README, guides, spécifications)  
│ └── Cahier\_des\_Charges\_Autisme.docx  
├── .vscode/ # Configurations VS Code  
│ ├── settings.json # Paramètres du workspace  
│ ├── launch.json # Configurations de debug Python/Kivy  
│ └── tasks.json # Tâches (buildozer, pytest)  
├── requirements.txt # Dépendances Python (kivy, sqlalchemy, pytest)  
├── buildozer.spec # Configuration pour Android  
├── pyproject.toml # Si utilisation de Poetry  
├── setup.py # Packaging via setuptools  
├── README.md # Présentation du projet et instructions  
└── LICENSE # Licence du projet