ShazaPiano — Install & Permissions Playbook (Puzzle Ready)

But: tout installer/configurer vite, sans oubli. Donne ce PDF à Cursor et suis les étapes.

Cible: Android + Backend local (FastAPI) + Firebase + IAP (1$).

0) Vue d'ensemble — ordre logique du puzzle

1 Pré-requis outils (Flutter, Android, Python 3.11, FFmpeg, Git).

2 Bootstrap du mono-repo & dépendances.

3 Backend local: FastAPI + CORS + FFmpeg + BasicPitch.

4 App Android: permissions micro & réseau, thème sombre, Home/Previews.

5 Firebase: Auth anonyme + Firestore + google-services.json.

6 IAP Android: produit non-consommable (1$) + restauration.

7 Tests rapides & checklist QA.

1) Pré-requis — installation outils (OS)

Outil	Windows	macOS	Ubuntu/Debian
Git	winget install Git.Git	brew install git	sudo apt-get install -y git
Python 3.11	winget install Python.Python.3.11	brew install python@3.11	sudo apt-get install -y python3.11 python3.11-venv
Flutter SDK	winget install Flutter.Flutter	brew install flutter	sudo snap install flutter --classic
Android Studio + SDK	Installer depuis site, puis SDK/AVD	brew install --cask android-studio	Télécharger .tar.gz Android Studio
FFmpeg	winget install Gyan.FFmpeg	brew install ffmpeg	sudo apt-get install -y ffmpeg
JDK (si besoin)	winget install EclipseAdoptium.Temurin.17.JDK	brew install --cask temurin	sudo apt-get install -y openjdk-17-jdk

Commande de vérification

```
flutter doctor -v
```

2) Bootstrap du projet (mono-repo & dépendances)

```
# Dossiers & Git mkdir -p shazapiano/app shazapiano/backend shazapiano/docs && cd
shazapiano && git init # Backend (Python 3.11 + venv + deps) cd backend python -m
venv .venv && source .venv/bin/activate pip install fastapi uvicorn[standard]
numpy scipy soundfile librosa pretty_midi moviepy python-multipart torch
basic-pitch # Vérifier ffmpeg -version basic-pitch --help # App Flutter cd ../app
flutter create --org com.ludo --project-name shazapiano . flutter pub add dio
permission_handler record video_player go_router flutter_riverpod
shared_preferences firebase_core firebase_auth cloud_firestore in_app_purchase
path_provider
```

3) Backend local — exécution + CORS + run

```
# backend/app.py — middleware CORS (ajouter après app = FastAPI()) from
fastapi.middleware.cors import CORSMiddleware app.add_middleware( CORSMiddleware,
allow_origins=["*"], # dev allow_credentials=True, allow_methods=["*"],
allow_headers=["*"], )
```

```
# Lancer le backend en dev uvicorn app:app --reload --host 0.0.0.0 --port 8000 #
Test rapide curl http://localhost:8000/health
```

4) App Android — permissions & réseau (Manifest)

```
...>
```

```
# Chemin d'enregistrement — utiliser path_provider (évite /sdcard/...) final dir =
await getTemporaryDirectory(); // package:path_provider final audioPath =
p.join(dir.path, 'shaza_tmp.m4a');
```

5) Lancer l'app (Android émulateur)

```
# Émulateur Android ouvert + backend en cours flutter run
--dart-define=BACKEND_BASE=http://10.0.2.2:8000
```

6) Firebase — Auth & Firestore (MVP)

```
1) Créer projet Firebase (console) → activer Authentication (Anonymous). 2) Créer
Firestore (mode production). 3) Ajouter l'app Android (package:
com.ludo.shazapiano) et placer google-services.json dans android/app/. 4) Règles
Firestore (MVP sécurisées) : rules_version = '2'; service cloud.firestore { match
/databases/{database}/documents { function isSignedIn() { return request.auth !=
null; } match /users/{uid} { allow read, write: if isSignedIn() && uid ==
request.auth.uid; match /entitlements/{doc} { allow read, write: if isSignedIn()
&& uid == request.auth.uid; } match /history/{doc} { allow read, write: if
isSignedIn() && uid == request.auth.uid; } } match /{document=**} { allow read,
write: if false; } } }
```

7) IAP Android — produit 1$ (plus tard dans le puzzle)

```
Play Console : - Créer 1 produit non-consommable: piano_all_levels_1usd (prix 1
$). - Comptes test (licensing). Flutter (quand on branche) : -
queryProductDetails(['piano_all_levels_1usd']) - buyNonConsumable(productDetails)
- on purchased -> acknowledge + set entitlements (SharedPreferences + Firestore) -
restore purchases au démarrage
```

8) QA rapide (à cocher)

- Backend /health OK; /process répond (même avec previews noires POC).
- App: tap bouton → capture 8 s → obtention de 4 PREVIEWS 16 s lisibles.
- Pas de crash permissions micro; HTTP local accepté (cleartext).
- Firebase projet créé; (IAP branché plus tard).

9) Accélérateurs (facultatif, pratique)

- Makefile: cibles make dev-api / dev-app.
- Docker backend (ensuite) pour garder FFmpeg identique partout.
- Jeu de WAV de test + script POST automatisé pour /process.

Fin du playbook. Quand tout est vert, on branche l’IAP puis le Practice.