📄 Cahier des charges : Outil de traitement de grilles QCM + Application Android

1. Objectif général

Développer une solution complète permettant :

- Le traitement automatique de copies scannées de **grilles de réponses QCM** via un **outil Python**.
- La génération de fichiers de résultats dans plusieurs formats paramétrables.
- L'analyse statistique des performances des étudiants.

---

2. Fonctionnalités – Partie Python

2.1. Entrée des copies

- Les copies (images) seront déposées dans un dossier sans nommage particulier.
- Formats supportés : `.jpg`, `.jpeg`, `.png`, `.pdf`.
- Traitement par lot automatique.

2.2. Lecture et interprétation

- Extraction automatique des informations :
  - **Nom** (champ OCR ou grille à cocher)
  - **Prénom**
  - **Numéro étudiant** (identifiant numérique)
  - **20 réponses QCM** (cases cochées parmi A, B, C, D)
- Reconnaissance par détection de zones sur une grille **standardisée**
- Correction d'orientation, rotation, et perspective.

2.3. Corrigé et notation

- Corrigé fourni dans un fichier (`correction.json` ou `.csv`).
- Barème paramétrable :
  - Points positifs pour bonne réponse
  - Malus optionnel pour mauvaise réponse
  - Absence de réponse neutre

2.4. Formats de sortie

- **Formats supportés** :
  - CSV

    - JSON
    - Excel (`.xlsx`)
    - **Apogée** (CSV avec texte spécifique à gauche et à droite des données)
- **Paramétrage** via options CLI ou fichier de configuration.
- **Tri** configurable par `nom`, `prénom` ou `note`.

2.5. Statistiques

- Moyenne, médiane, écart-type, note max/min
- Taux de réussite par question
- Graphiques optionnels (histogrammes, courbes)
- Sauvegarde dans `statistiques.json` et/ou `.xlsx`

---

3. Interface en ligne de commande (CLI)

Exemple d'usage :

```
python grille_qcm.py --input_dir ./copies \
                     --output_dir ./resultats \
                     --correction correction.csv \
                     --format apogee \
                     --tri note
```

Options :

- `--input_dir` : dossier contenant les copies
- `--output_dir` : dossier pour les résultats
- `--correction` : fichier du corrigé
- `--format` : csv, json, excel, apogee
- `--tri` : nom | prénom | note

---

4. Application Android

4.1. Objectif

- Développer les mêmes fonctionnalités que l'application Python

4.2. Technologies

- Utilisation locale ou connectée (selon besoin d'intégration).
- Interface simple et épurée, responsive.

---

5. Template de fiche réponse (Proposition)

- **Format A4**, orientation paysage.
- 4 coins noirs pour détection automatique.
- Grille **nom**/**prénom** : zones de cases à cocher par lettre (A-Z).
- **Numéro étudiant** : 8 chiffres avec cases 0 à 9.
- **20 blocs QCM** avec 4 cases (A-D) par ligne ou colonne.
- Instructions d'utilisation en haut de la feuille.
- Utilisation de zones calibrées pour fiabiliser l'OCR/détection.

---

6. Contraintes et livrables

6.1. Techniques

- Langage : **Python 3.10+**
- Librairies : `OpenCV`, `numpy`, `pytesseract`, `pandas`, `openpyxl`
- Compatible Windows/Linux/macOS
- Android : application native Kotlin ou Flutter (selon profil)

6.2. Livrables

- Code source Python + documentation (`README.md`)
- Script CLI principal : `grille_qcm.py`
- Exemple de fichier de correction
- Grille modèle (vierge)
- Application Android (.apk + code)
- Fichier `requirements.txt`
- Fichiers d'exemple : sortie CSV, Apogée, JSON, Excel

---

7. Planning sur 1 mois

Semaine	Tâches
S1	Définition du gabarit, OCR, détection de zones, maquette appli Android
S2	Traitement QCM, export formats, statistiques
S3	Application Android : affichage résultats, intégration données
S4	Tests, documentation, livrables finaux

---

8. Points d'attention

- Aucun jeu de test fourni : des grilles types seront générées et utilisées.
- L'efficacité dépend fortement du respect du gabarit proposé.
