

RAPPORT DE PROJET

Analyseur de Performance Etudiante

MEMBRES DU GROUPE :

SONTSA CHRISTIAN
NZATI STEPHANE
SAMA CAMELIA
MBOULA MONICA

Date : 15 February 2026

TABLE DES MATIERES

1. Contexte et Objectifs
2. Description du Projet
3. Architecture et Conception
4. Fonctionnalites Principales
5. Indicateurs Cles de Performance (KPI)
6. Implementation Technique
7. Tests et Validation
8. Resultats et Livrables
9. Conclusion

1. CONTEXTE ET OBJECTIFS

Ce projet a été réalisé dans le cadre d'une évaluation de groupe portant sur le développement d'une application web interactive. L'objectif principal était de créer une application permettant d'analyser les données de performance étudiante à travers une interface conviviale et intuitive. Le projet évalue les compétences en collaboration Git, manipulation de données avec DuckDB, développement d'interfaces interactives avec Streamlit, et création de visualisations de données pertinentes.

2. DESCRIPTION DU PROJET

L'application "Student Performance Analyzer" est une solution web interactive développée avec Streamlit qui permet aux éducateurs et administrateurs de :

- Téléverser et gérer des fichiers CSV contenant des données de performance étudiante
- Stocker et interroger les données de manière efficace avec DuckDB
- Appliquer des filtres dynamiques pour affiner l'analyse (genre, âge, niveau d'éducation parentale)
- Visualiser quatre indicateurs clés de performance (KPI) distincts
- Explorer les relations entre les habitudes d'étude et la performance académique

3. ARCHITECTURE ET CONCEPTION

L'application suit une architecture modulaire avec une séparation claire entre les couches de présentation, logique métier et accès aux données :

- **Couche Presentation (Streamlit)** : Interface utilisateur conviviale avec upload de fichiers, filtres dynamiques et visualisations interactives
- **Couche Logique Metier** : Traitement des données, calcul des KPI et application des filtres
- **Couche Accès aux Données** : Requêtes SQL et validation des données
- **Couche Stockage (DuckDB)** : Base de données en mémoire pour le stockage efficace des données

Modules Principaux :

- **file_manager.py** : Gestion du téléchargement et validation des fichiers CSV
- **database_manager.py** : Gestion des connexions DuckDB et opérations de base de données
- **filter_engine.py** : Construction et application des filtres dynamiques
- **kpi_calculator.py** : Calcul des quatre indicateurs clés de performance
- **visualization_engine.py** : Création des visualisations interactives avec Plotly
- **app.py** : Application principale Streamlit orchestrant tous les composants

4. FONCTIONNALITES PRINCIPALES

4.1 Televerement de Donnees

L'application accepte deux fichiers CSV : student_habits_performance.csv (1000 etudiants) et StudentPerformanceFactors.csv (6607 etudiants). Les fichiers sont valides, dédupliqués et importes dans DuckDB pour un traitement efficace.

4.2 Filtrage Dynamique

Les utilisateurs peuvent appliquer plusieurs filtres simultanement : genre, plage d'age (17-24 ans), niveau d'education parentale. Les filtres utilisent la logique ET pour affiner les resultats.

4.3 Visualisations Interactives

Quatre visualisations Plotly interactives permettent l'exploration des donnees avec survol pour afficher les details, legende cliquable pour basculer la visibilite des series, et axes etiquetes appropries.

5. INDICATEURS CLES DE PERFORMANCE (KPI)

KPI 1 : Scores Moyens par Groupe Demographique

Affiche les scores d'examen moyens par genre et niveau d'education parentale sous forme de graphique en barres.

KPI 2 : Correlation Etude-Performance

Visualise la relation entre les heures d'étude et les scores d'examen via un graphique de dispersion.

KPI 3 : Impact de l'Assiduite

Montre comment l'assiduité affecte les scores d'examen avec un graphique en ligne par plages d'assiduité.

KPI 4 : Relation Sommeil-Performance

Explore la relation entre les heures de sommeil et la performance académique via un graphique de dispersion.

6. IMPLEMENTATION TECHNIQUE

6.1 Technologies Utilisees

- **Streamlit** : Framework web pour l'interface utilisateur interactive
- **DuckDB** : Moteur de base de donnees SQL en memoire pour requetes rapides
- **Pandas** : Manipulation et transformation des donnees
- **Plotly** : Visualisations interactives et reactives
- **Pytest** : Framework de test unitaire
- **Hypothesis** : Tests bases sur les proprietes

6.2 Defis Techniques Resolus

Un defi majeur a ete la gestion des differences de noms de colonnes entre les deux fichiers CSV. Le fichier StudentPerformanceFactors.csv utilise des noms avec majuscules et underscores (ex: Exam_Score, Sleep_Hours) tandis que student_habits_performance.csv utilise des noms en minuscules. La solution implementee effectue une correspondance insensible a la casse et utilise des guillemets SQL pour gerer les noms mixtes.

7. TESTS ET VALIDATION

L'application a ete validee avec une suite complete de 66 tests unitaires couvrant tous les modules :

- **Tests du Gestionnaire de Fichiers** : Validation CSV, gestion des encodages, detection des colonnes manquantes
- **Tests du Gestionnaire de Base de Donnees** : Creation de tables, import de donnees, execution de requetes
- **Tests du Moteur de Filtrage** : Construction de clauses WHERE, application de filtres, gestion des cas limites
- **Tests du Calculateur KPI** : Calculs mathematiques corrects, gestion des donnees filtrées
- **Tests du Moteur de Visualisation** : Creation de graphiques, integrite des donnees

Réultats des Tests : Tous les 66 tests passent avec succes. Les tests de bout en bout confirment que l'application fonctionne correctement avec les deux fichiers CSV, avec et sans filtres appliques.

8. RESULTATS ET LIVRABLES

8.1 Livrables Complètes

- Application Streamlit entièrement fonctionnelle avec interface conviviale
- Intégration complète de DuckDB pour le stockage et l'interrogation des données
- Quatre visualisations KPI interactives et réactives
- Système de filtrage dynamique avec logique ET
- Suite de tests complète (66 tests unitaires)
- Documentation complète (README.md, docstrings, commentaires)
- Gestion robuste des erreurs et validation des données

8.2 Performances

- Import de données : < 1 seconde pour 7607 enregistrements
- Exécution des requêtes : < 2 secondes pour les ensembles de données filtrés
- Mise à jour des visualisations : < 1 seconde lors de l'application des filtres
- Utilisation mémoire : Minimale avec DuckDB en mémoire

9. CONCLUSION

Le projet "Student Performance Analyzer" a ete developpe avec succes en tant qu'application web interactive complete. L'équipe a demonstre une maitrise des technologies cles (Streamlit, DuckDB, Plotly) et des pratiques de developpement professionnel (architecture modulaire, tests complets, gestion des erreurs).

L'application repond a tous les criteres d'évaluation specifies : fonctionnalite complete, code de qualite bien structure, interface utilisateur intuitive, visualisations pertinentes et gestion efficace des donnees. Les defis techniques, notamment la gestion des differences de noms de colonnes entre les sources de donnees, ont ete resolus de maniere robuste et elegante.

Cette application peut servir de base pour des analyses plus avancees de performance etudiante et pourrait etre etendue avec des fonctionnalites supplementaires telles que l'export de rapports, l'analyse predictive ou l'integration avec d'autres sources de donnees.

Equipe de Developpement : SONTSA CHRISTIAN, NZATI STEPHANE, SAMA CAMELIA,
MBOULA MONICA