

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. Fonctionnalités Principales
5. Indicateurs Clés de Performance (KPI)
6. Implémentation Technique
7. Tests et Validation
8. Résultats 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 separation claire entre les couches de presentation, logique metier et acces aux donnees :

- **Couche Presentation (Streamlit)** : Interface utilisateur conviviale avec upload de fichiers, filtres dynamiques et visualisations interactives
- **Couche Logique Metier** : Traitement des donnees, calcul des KPI et application des filtres
- **Couche Acces aux Donnees** : Requetes SQL et validation des donnees
- **Couche Stockage (DuckDB)** : Base de donnees en memoire pour le stockage efficace des donnees

Modules Principaux :

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

4. FONCTIONNALITES PRINCIPALES

4.1 Televersement de Donnees

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

4.2 Filtrage Dynamique

Les utilisateurs peuvent appliquer plusieurs filtres simultanément : 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 appropriés.

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'etude et les scores d'examen via un graphique de dispersion.

KPI 3 : Impact de l'Assiduite

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

KPI 4 : Relation Sommeil-Performance

Explore la relation entre les heures de sommeil et la performance academique 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 été validée avec une suite complète de 66 tests unitaires couvrant tous les modules :

- **Tests du Gestionnaire de Fichiers** : Validation CSV, gestion des encodages, détection des colonnes manquantes
- **Tests du Gestionnaire de Base de Données** : Création de tables, import de données, exécution de requêtes
- **Tests du Moteur de Filtrage** : Construction de clauses WHERE, application de filtres, gestion des cas limites
- **Tests du Calculateur KPI** : Calculs mathématiques corrects, gestion des données filtrées
- **Tests du Moteur de Visualisation** : Création de graphiques, intégrité des données

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

8. RESULTATS ET LIVRABLES

8.1 Livrables Completes

- Application Streamlit entierement fonctionnelle avec interface conviviale
- Integration complete de DuckDB pour le stockage et l'interrogation des donnees
- Quatre visualisations KPI interactives et reactives
- Systeme de filtrage dynamique avec logique ET
- Suite de tests complete (66 tests unitaires)
- Documentation complete (README.md, docstrings, commentaires)
- Gestion robuste des erreurs et validation des donnees

8.2 Performances

- Import de donnees : < 1 seconde pour 7607 enregistrements
- Execution des requetes : < 2 secondes pour les ensembles de donnees filtres
- Mise a jour des visualisations : < 1 seconde lors de l'application des filtres
- Utilisation memoire : Minimale avec DuckDB en memoire

9. CONCLUSION

Le projet "Student Performance Analyzer" a été développé avec succès en tant qu'application web interactive complète. L'équipe a démontré une maîtrise des technologies clés (Streamlit, DuckDB, Plotly) et des pratiques de développement professionnel (architecture modulaire, tests complets, gestion des erreurs).

L'application répond à tous les critères d'évaluation spécifiés : fonctionnalité complète, code de qualité bien structuré, interface utilisateur intuitive, visualisations pertinentes et gestion efficace des données. Les défis techniques, notamment la gestion des différences de noms de colonnes entre les sources de données, ont été résolus de manière robuste et élégante.

Cette application peut servir de base pour des analyses plus avancées de performance étudiante et pourrait être étendue avec des fonctionnalités supplémentaires telles que l'export de rapports, l'analyse prédictive ou l'intégration avec d'autres sources de données.

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