**REPORT FOR STUDENT MANAGEMENT SYSTEM**

**& PERFORMANCE ANALYTICS DASHBOARD**

**FODS**

**A logo with blue and red text

AI-generated content may be incorrect.**

**Submitted by**

Name: Aayushma Mehta

Student ID: 24071176

Cyber Security and Digital Forensics

Kathmandu, Nepal

1. **Project Overview**

This Python-based project is a comprehensive system designed to manage student profiles and analyze their academic performance through visual dashboards. It consists of two main modules. The first module, **Student Profile Management System**, allows for secure login functionality, where users are classified as either "Admin" or "Student". Depending on their role, different features become available. Admins can manage student records—adding new students, updating information, or deleting records—while students are allowed to view their profiles and academic results.

The second module is the **Performance Analytics Dashboard**, which uses data visualization techniques to analyze student grades and the impact of extracurricular activities (ECA) on performance. It also helps flag underperforming students based on predefined grade thresholds. These two modules work together to provide a structured, role-based, and data-driven approach to managing and evaluating student performance.

1. **Code Explanation**
   1. Classes and Inheritance

The code is structured using Object-Oriented Programming (OOP) principles. At the core of the system is the User base class, which stores common attributes such as the username, role, and user\_id. This class is inherited by two subclasses: Student and Admin. The Student class contains functions specific to student users, including the ability to view their own profile and grades. The Admin class enables administrative operations such as adding new students, updating student information, deleting users, and generating performance insights using the dashboard module.

* 1. Methods and Functions

File handling is a central part of this system. The application reads from and writes to several .txt files (users.txt, passwords.txt, grades.txt, eca.txt) to persist user data. The program uses standard file I/O methods such as open(), readlines(), and write() to handle user credentials, grades, and extracurricular data.

The login mechanism authenticates users by checking input credentials against the users.txt and passwords.txt files. Once a user logs in successfully, the system checks their role and presents an appropriate menu. Students can only view their data, while Admins can manage student records and generate analytics reports.

Admin-specific functions include add\_student(), which allows the administrator to input a new student's data and store it across the relevant files. The update\_student() function modifies existing student details, and delete\_student() removes a student’s data from all files. There's also a view\_students() function that displays all stored student profiles in a tabular format for quick reference.

In the second module (Task\_2.py), three analytical functions are defined. grade\_trends() uses matplotlib to generate a line chart that visualizes grades across different subjects for each student. eca\_impact() creates a scatter plot to show the correlation between ECA involvement and academic performance. Finally, performance\_alert() analyzes students' average grades and alerts if they fall below a set threshold (e.g., 60), helping identify those at risk academically.

1. **Flowcharts**

To visualize the logic flow of the system, two flowcharts were created. The first represents the **Login Process**. Upon starting the program, the user is prompted to enter login credentials. These are checked for validity against stored data. If valid, the system identifies whether the user is an Admin or a Student. Based on this role, different operations are made available. Invalid login attempts exit the system or display an error.

The second flowchart depicts **Admin Operations and Dashboard Integration**. Once logged in, an Admin is presented with a menu to manage students or view analytics. Management operations include adding, updating, deleting, and viewing student data. The dashboard component provides insights such as grade trends, ECA involvement, and performance alerts. These are generated in real-time using the data stored in the text files.

Logic Flowchart Dashboard Flowchart

A diagram of a program

AI-generated content may be incorrect. A diagram of performance analysis

AI-generated content may be incorrect.

1. **Test Outputs and Results**

The program was tested using sample input and output scenarios. When valid user credentials are entered, the role-based menus appear correctly. Admins are able to successfully add and edit student records, while students can only view their information.

The **Grade Trends** chart clearly shows how student grades vary across different subjects, helping to identify strengths and weaknesses. The **ECA Impact** chart demonstrates the relationship between the number of extracurricular activities and average academic performance, supporting the idea that active involvement may correlate with better results.

Sample of Input/Output

Screenshots of both input and output are captured and attached for reference. These show the login flow, admin functionalities, student view, and dashboard graphs.

A white background with colorful text

AI-generated content may be incorrect.

A screen shot of a computer

AI-generated content may be incorrect.

A computer screen shot of a computer code

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

A graph of colored lines

AI-generated content may be incorrect.

A white screen with black text

AI-generated content may be incorrect.