



亞洲大學
ASIA UNIVERSITY

Final Project Report
Advanced Computer Programming

Student Result Dashboard

Group : 1

Instructor : DINH-TRUNG VU

2025-06

Chapter 1 Introduction

1.1 Group Introduction

- 1) **Group Project Repository:** <https://github.com/113710077/Group-1>
- 2) **Group members:**
 1. Bhoomika S Horapeti - 113710077
 2. Navya M – 113710078
 3. Marlen Rakymov - 111021139

1.2 Overview

The Student Result Dashboard is an interactive and automated system designed to facilitate the management and analysis of student academic performance data. It allows users, such as teachers and academic administrators, to upload student marks through CSV files or input them manually via a form. Once the data is submitted, the application utilizes Python's powerful data processing and visualization libraries—primarily Pandas and Matplotlib—to compute essential academic metrics. These include total and average marks, pass/fail classification based on defined thresholds, and identification of top-performing students. The results are then presented visually using dynamic charts and tables, making it easier to interpret student performance data at both individual and class levels.

What sets this project apart is its incorporation of machine learning components through the Scikit-learn library. Specifically, the dashboard includes features for grade prediction using classification models and student clustering using the KMeans algorithm. These features enable predictive analytics and segmentation of students into performance-based categories such as high, average, and low performers. This capability introduces a forward-thinking approach to educational analytics, helping educators make informed decisions and provide targeted academic support.

1.3 Project Objectives

The primary goal of this project was to create a user-friendly, web-based system that automates the analysis and visualization of student results. It combines several key programming concepts, including file I/O, data manipulation, chart generation, web development with Flask, and introductory machine learning. Beyond fulfilling academic requirements, the project was designed with scalability and real-world applicability in mind, showcasing our ability to integrate diverse programming skills into a cohesive and functional application.

Chapter 2 System Design and Implementation

2.1 Input Handling

The dashboard provides two main options for inputting student data. The first option allows users to upload a CSV file containing student information, with each record comprising fields such as ID, Name, and marks in subjects like Math, Science, and English. Alternatively, users can manually input data through an HTML form. This flexibility ensures that both small-scale and large-scale data can be easily accommodated by the system.

2.2 Data Analysis Using Pandas

Once the data is collected, it is processed using the Pandas library. Key operations include computing the total and average marks for each student, determining their pass or fail status based on a 35% threshold, and calculating class-level statistics such as subject-wise averages. The application also identifies the top five students by average score and pinpoints the single highest scorer. These features allow teachers to gain immediate insights into academic performance without having to analyze raw data manually.

2.3 Visualization Using Matplotlib

To make the analysis more intuitive and visually appealing, the application employs Matplotlib for chart generation. Three types of visualizations are created dynamically:

- A **bar chart** displaying individual scores across subjects,
- A **pie chart** showing the proportion of students who passed versus those who failed, and
- A **line graph** illustrating the class-wide average for each subject.

These visual tools help users grasp key trends and distributions at a glance, enhancing the overall utility of the dashboard.

2.4 Web Interface with Flask

The entire application is built on the Flask framework, which serves as the backbone for the web interface. Users interact with the system through a browser, where they can upload CSV files or enter data manually, and receive immediate feedback in the form of tables and charts. Flash messages guide users throughout their interaction, offering real-time notifications for errors or successful operations. The interface is styled using Bootstrap to ensure clarity, responsiveness, and ease of navigation.

Chapter 3: Advanced Features with Machine Learning

3.1 Grade Prediction

The system includes a module for predicting student grades using a Decision Tree Classifier. By training the model on historical data, the system learns to map numerical scores to categorical grades (e.g., A, B, C, D, F). When new data is input, the model can predict grades based on learned patterns, giving educators an additional layer of insight into academic trends and future performance.

3.2 Student Clustering with KMeans

Another innovative feature is the use of the KMeans clustering algorithm to group students based on their academic performance. This unsupervised learning method segments students into categories such as high performers, average performers, and low performers. These clusters can help educators identify which students need more attention and which are excelling, enabling personalized academic strategies.

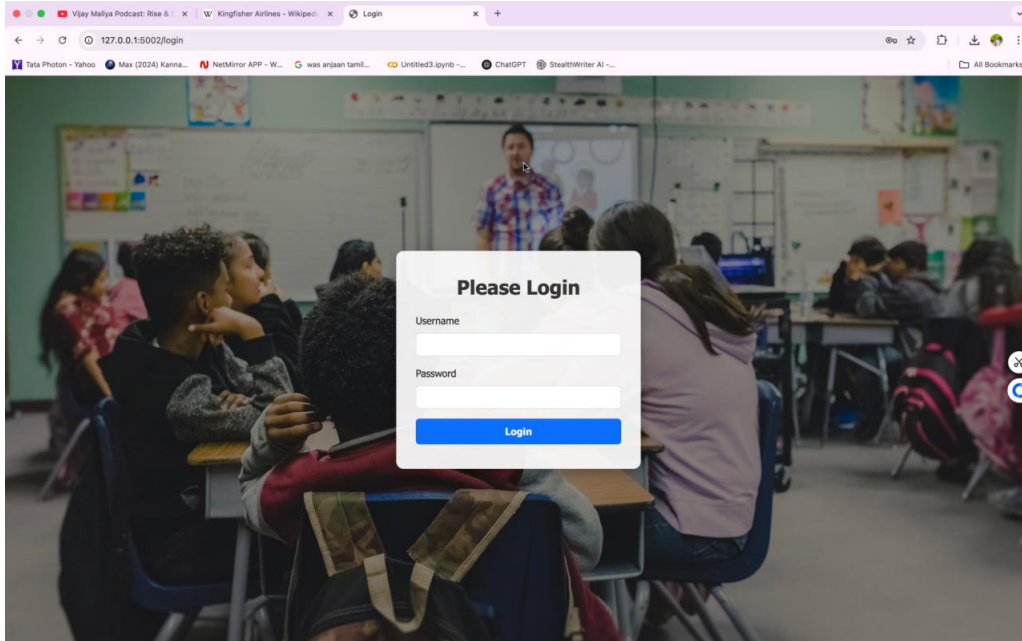
During development, several challenges were encountered, including handling missing or incorrectly formatted data, dealing with dynamic chart rendering in Flask, and integrating machine learning into a real-time web interface. Debugging template errors and managing stateful interactions between the server and client also required a solid understanding of Flask's request-response cycle.

From a learning perspective, the project significantly deepened our understanding of Python's core and advanced features. It strengthened our skills in data analysis, web development, and machine learning. More importantly, it demonstrated how different technologies and libraries can be brought together to solve a practical, real-world problem in a user-centric way.

Chapter 4: Results

A. Login/Logout Page – User Access Control

The login/logout functionality ensures that only authorized users such as administrators or instructors can access and manage student data. This adds a layer of security and personalization to the application, aligning it with real-world academic portals.



B. Homepage – Central Access to Features

The homepage serves as the central hub of the Student Result Dashboard, offering users an intuitive interface to interact with the system. From this page, users can choose to upload a CSV file, manually enter student marks, view analysis results, and access graphical visualizations. It also integrates login/logout functionality to manage user sessions securely.

C. CSV Upload – Bulk Student Data Input

The dashboard allows users to upload student records via CSV files, making it easy to handle large datasets efficiently. Upon upload, the system validates the data and extracts fields like student names and subject-wise marks for further analysis.

The screenshot shows the 'Student Result Dashboard' with a 'Welcome' message. Below it, there's a section for 'Upload CSV or Enter Student Data Manually'. The 'Manual Student Entry' form is active, showing a table with columns for Name, Math, Science, and English. The form is titled 'Manual Student Entry (At least 5 students)' and has a 'Submit Manual Data' button at the bottom.

D. Manual Data Entry – Flexibility in Input

In addition to uploading CSV files, the dashboard provides a manual data entry form that allows users to input student information directly into the system. This feature is especially useful in scenarios where data is available in non-digital formats or needs to be entered for a small number of students. The form captures essential fields such as student name and subject-wise marks, which are then processed similarly to CSV-uploaded data for analysis and visualization.

The screenshot shows the 'Student Result Dashboard' with the 'Manual Student Entry' form filled with sample data. The form is titled 'Manual Student Entry (At least 5 students)' and has a 'Submit Manual Data' button at the bottom. Below the form, there is a 'Results Table' section.

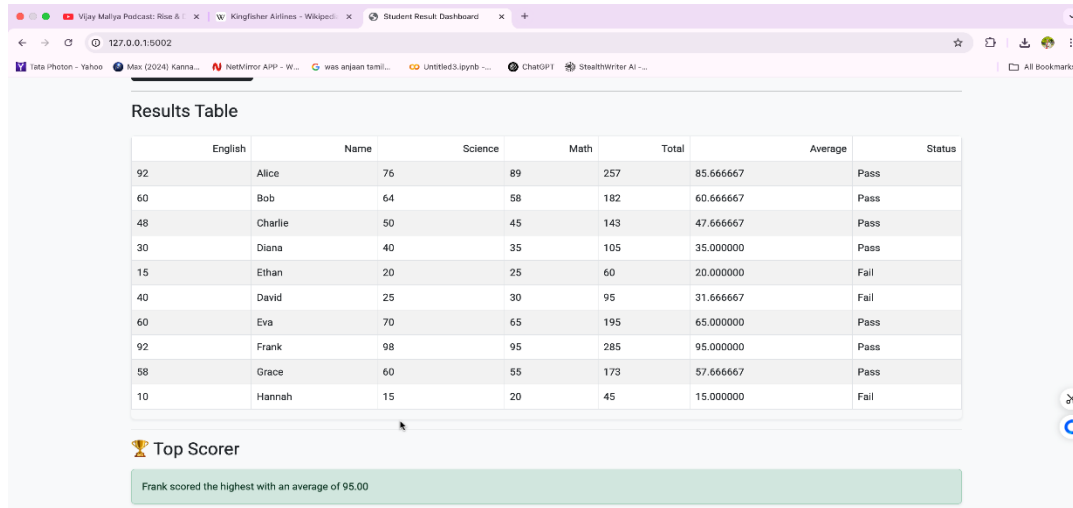
Name	Math	Science	English
Bhoomika	99	100	98
Olivia	45	19	11
Noah	56	78	79
Isabella	34	66	87
Eva	99	90	100

E. HTML Table – Student Result Summary

This table displays detailed analytics per student, including total marks, average, and pass/fail status. It offers a structured view of the processed data, supporting both individual assessment and overall academic overview.

F. Top Scorer Highlight

The top scorer is automatically identified and displayed with their average score. This feature acknowledges academic excellence and can be used to motivate peers.



The screenshot shows a web browser window with the title 'Student Result: Dashboard'. The address bar shows '127.0.0.1:8002'. The page contains a 'Results Table' with the following data:

English	Name	Science	Math	Total	Average	Status
92	Alice	76	89	257	85.666667	Pass
60	Bob	64	58	182	60.666667	Pass
48	Charlie	50	45	143	47.666667	Pass
30	Diana	40	35	105	35.000000	Pass
15	Ethan	20	25	60	20.000000	Fail
40	David	25	30	95	31.666667	Fail
60	Eva	70	65	195	65.000000	Pass
92	Frank	98	95	285	95.000000	Pass
58	Grace	60	55	173	57.666667	Pass
10	Hannah	15	20	45	15.000000	Fail

Below the table, there is a 'Top Scorer' section with a trophy icon and a green box stating: 'Frank scored the highest with an average of 95.00'.

G. Top 5 Students List

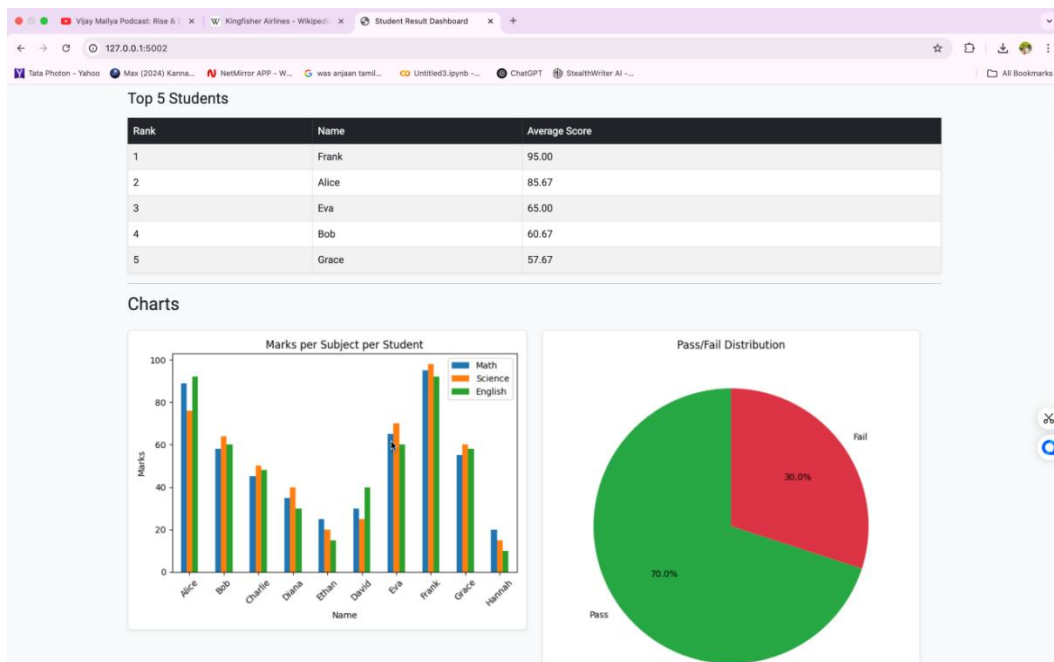
A list of the top five performing students is shown, helping instructors quickly identify high achievers. This can assist in merit-based recognition and planning for advanced learning opportunities.

H. Bar Chart – Marks per Subject per Student

This chart visualizes individual student performance across core subjects like Math, Science, and English. Each group of bars represents one student, showing their scores in the respective subjects. It helps quickly compare subject-wise strengths and weaknesses at an individual level.

I. Pie Chart – Pass/Fail Distribution

The pie chart illustrates the proportion of students who passed or failed based on their average marks. This visualization makes it easy to understand the overall class performance and the effectiveness of academic instruction.



J. Line Chart – Average Performance Across Subjects

The line chart shows the class average marks for each subject. This offers insight into which subjects are generally stronger or weaker across the entire group, aiding in academic planning and targeted intervention.

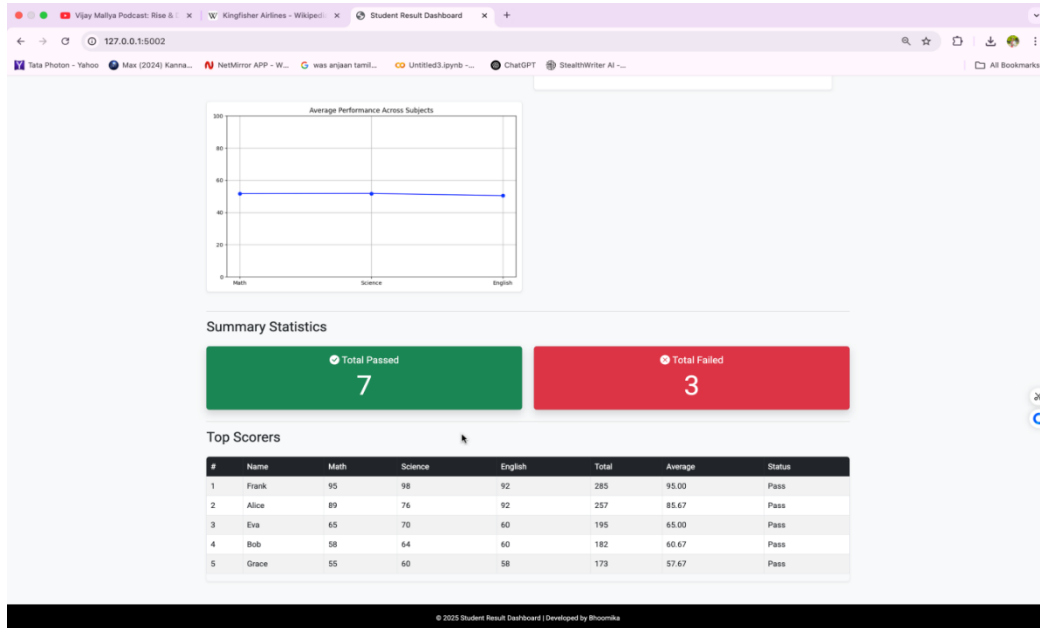
K. Summary Statistics – Pass/Fail Distribution

The summary statistics section of the dashboard presents a clear breakdown of the number of students who have passed or failed based on a predefined threshold. After data upload or manual entry, the system processes the marks and instantly displays the total count of students who passed and those who did not meet the passing criteria. This immediate overview helps instructors assess the overall performance distribution and identify any areas requiring academic intervention.

L. Top Scorers

The Top Scorers section highlights the students with the highest overall performance based on their total marks. Once the data is processed, the dashboard identifies and displays the names and scores of the top-performing students. This

feature helps in recognizing academic excellence and can be useful for awarding merit-based acknowledgments or further analysis of success patterns.



Chapter 5: Conclusion and Future Work

The Student Result Dashboard is a robust and extendable application that successfully achieves its core objectives. It provides a seamless way for educators to analyze student performance, backed by visual aids and predictive intelligence. The project showcases the integration of multiple technical skills and offers valuable insights into the power of combining data science with web development.

In the future, the system could be enhanced by adding features such as time-series tracking of academic performance, integration with student databases or LMS systems, and user authentication for secure data access. The machine learning models could also be expanded to include more advanced algorithms and broader datasets for greater accuracy and utility.

