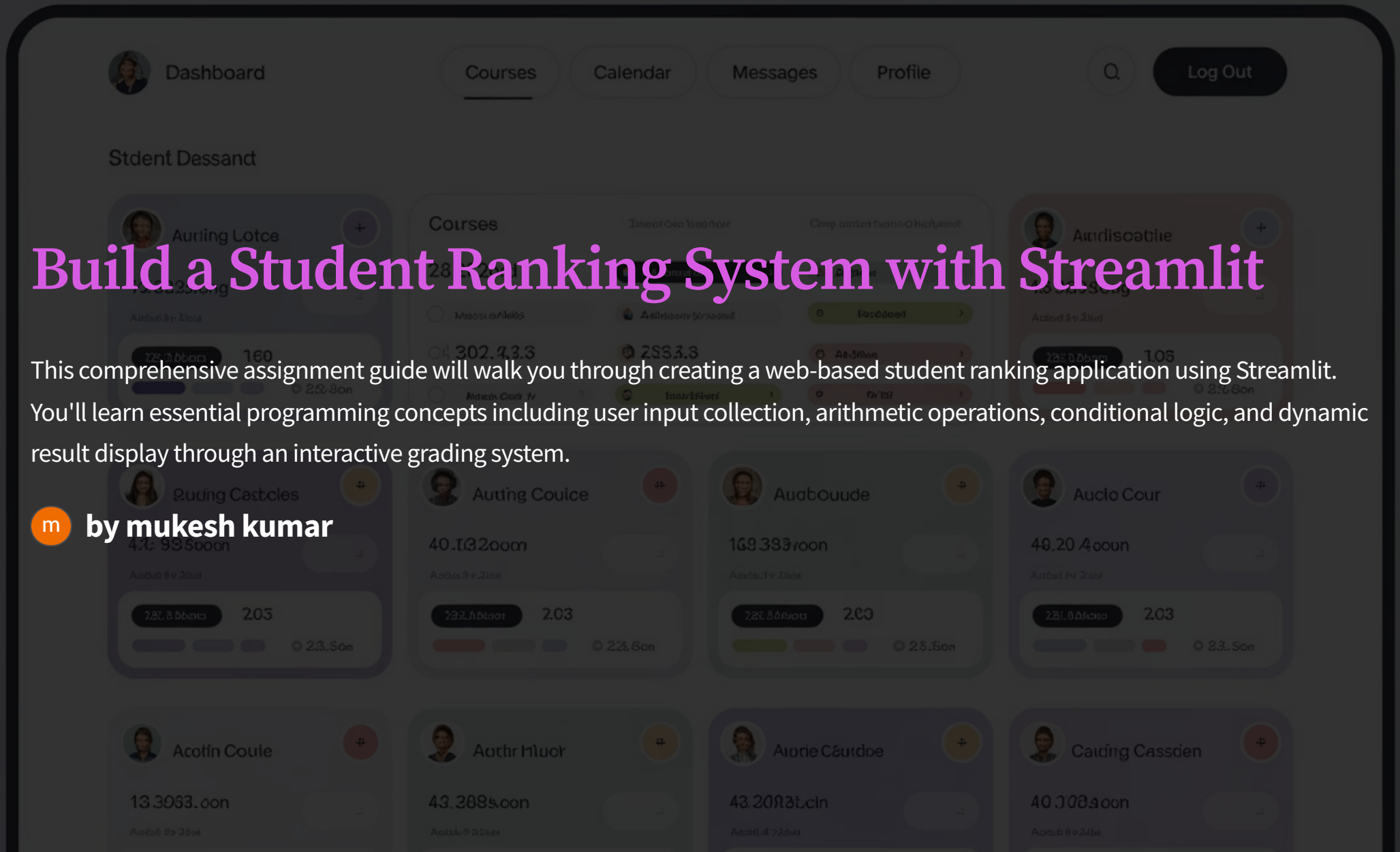


Build a Student Ranking System with Streamlit

This comprehensive assignment guide will walk you through creating a web-based student ranking application using Streamlit. You'll learn essential programming concepts including user input collection, arithmetic operations, conditional logic, and dynamic result display through an interactive grading system.

by mukesh kumar

Task Description

You are tasked with creating a **Student Ranking System** using **Python and Streamlit**. This project will demonstrate your ability to build interactive web applications with real-world functionality.

1 Input Collection

Accept marks for five subjects (English, Hindi, Telugu, Maths, Science) through number input fields with a range of 0 to 100.

2 Processing Logic

Include a Submit button to trigger data processing and calculations.

3 Results Display

Calculate total marks, percentage, and determine student rank using conditional logic, then display results using Streamlit elements.

Ranking Logic

The ranking system uses percentage-based criteria to assign letter grades. This standardized approach ensures consistent evaluation across all students.

Percentage Range	Rank
≥ 90	A+
80 – 89.99	A
70 – 79.99	B
60 – 69.99	C
50 – 59.99	D
< 50	F (Fail)

Implementation Requirements

Follow these specific steps to build your student ranking application. Each component serves a crucial role in creating a functional and user-friendly interface.

1

Create File Structure

Create a file named student_ranking_app.py as your main application file.

2

Input Collection

Use Streamlit's number_input() to collect marks for all 5 subjects with proper validation.

3

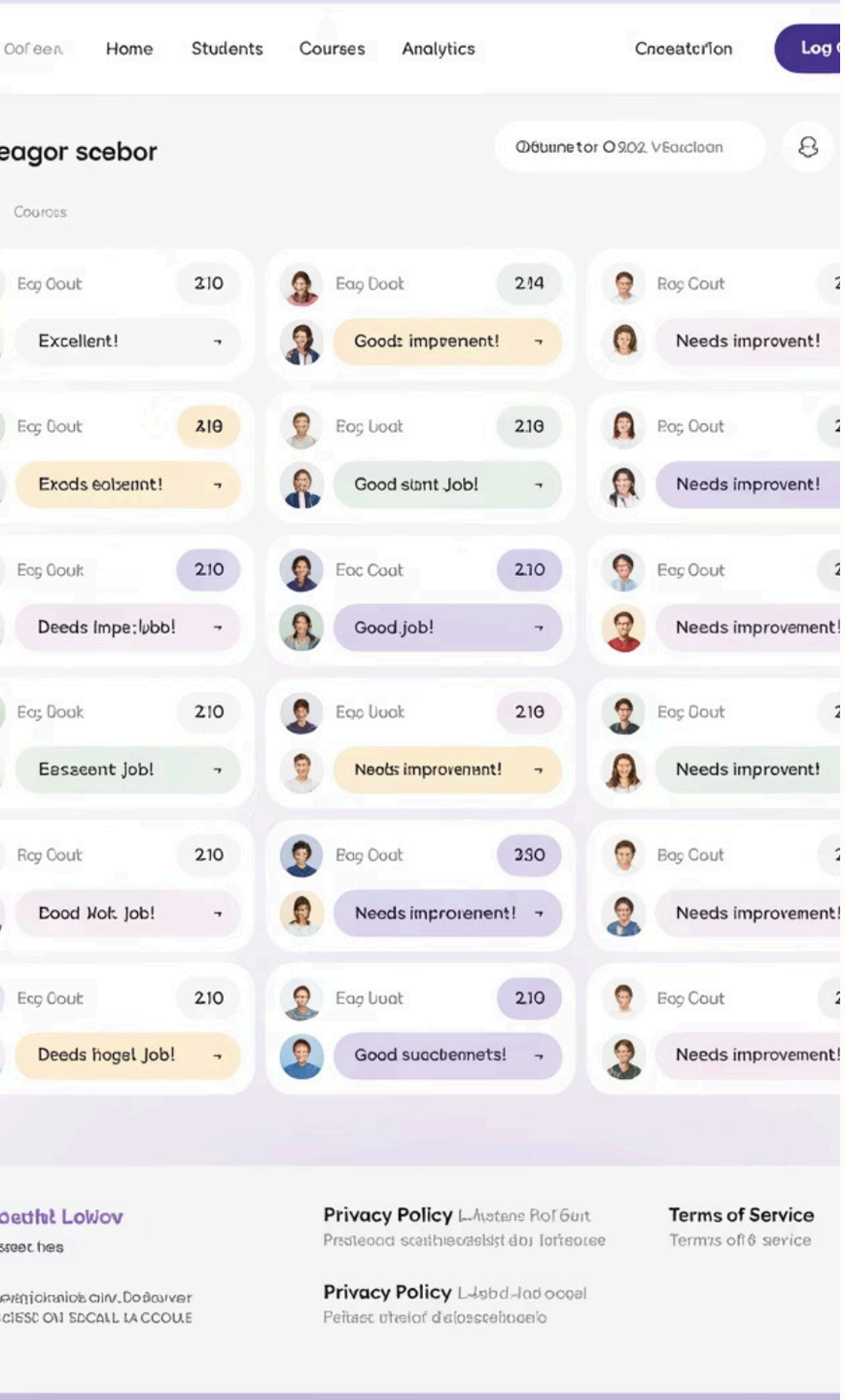
Processing Block

Implement an if st.button("Submit"): block to handle calculations and result generation.

4

Display Results

Use st.success(), st.info(), and st.warning() to present outputs in an attractive, user-friendly format.



Expected Output Example

Here's what your completed application should look like when a user enters their marks and submits the form. This example demonstrates the clean, professional interface your app should achieve.

Input Interface

🎓 Student Ranking System with input fields for each subject: English (88), Hindi (92), Telugu (75), Maths (90), Science (85), followed by a Submit button.

Calculation Results

🎉 Total Marks: 430

📊 Percentage: 86.0%

🏅 Rank: A

Visual Feedback

Results displayed using Streamlit's success, info, and warning message components for clear visual hierarchy and user experience.

Bonus Challenge

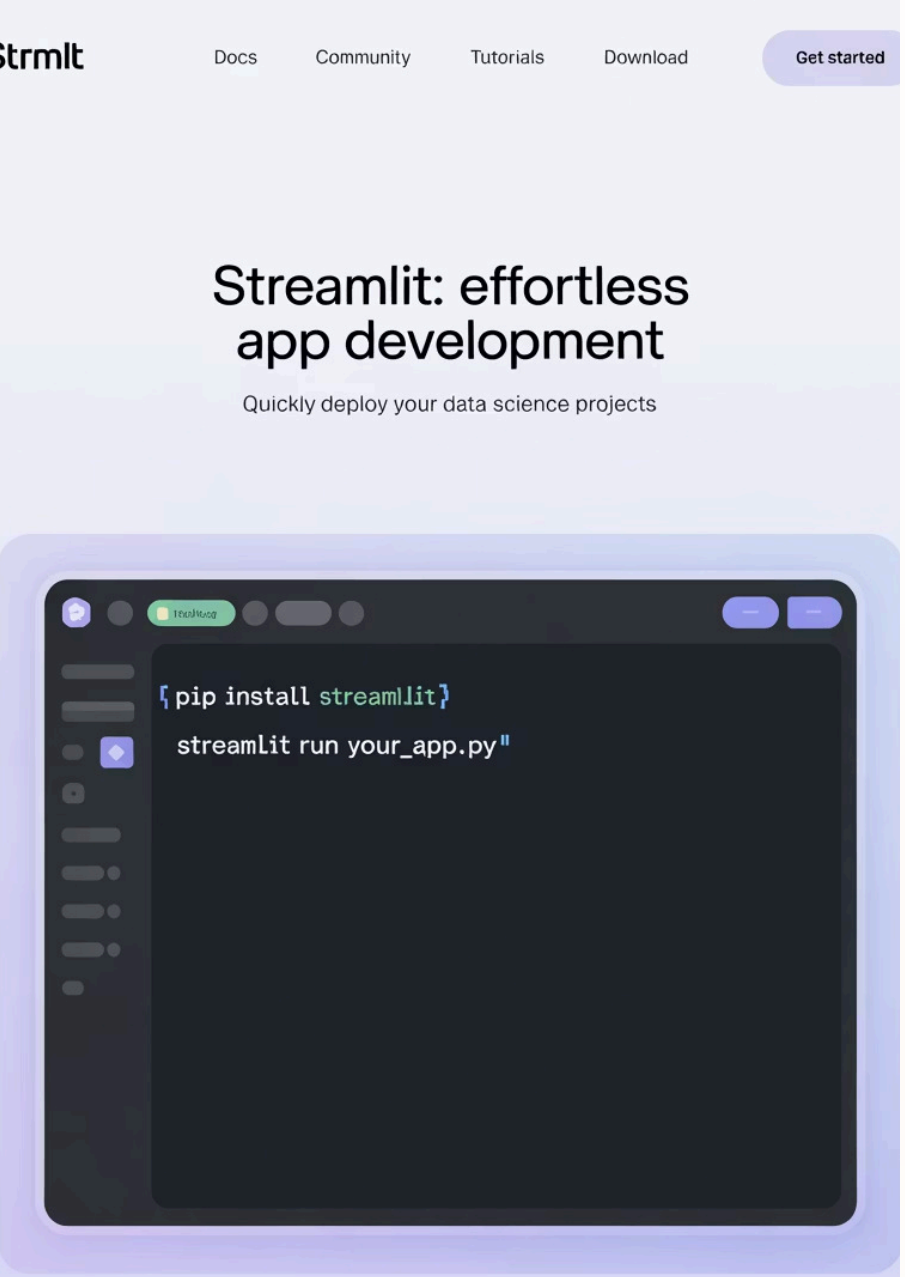
Enhance your application with additional validation features to improve user experience and data integrity. This optional challenge will demonstrate advanced programming practices.

Validation Requirements

Add a validation message if the user enters all 0s for their subjects or forgets to fill any subject field. This prevents invalid submissions and guides users toward proper data entry.

Implementation Benefits

Implementing validation improves application robustness, provides better user feedback, and demonstrates professional development practices that are essential in real-world applications.



Installation and Setup Instructions

Follow these step-by-step instructions to set up your development environment and run your student ranking application successfully.

Install Streamlit

Open your terminal or command prompt and run: **pip install streamlit** to install the required framework for your web application.

Run Your Application

Navigate to your project directory and execute: **streamlit run student_ranking_app.py** to launch your application in a web browser.

Access Your App

Streamlit will automatically open your default web browser and display your application, typically at localhost:8501.

Learning Outcomes

This assignment provides comprehensive exposure to essential programming concepts and web development skills that form the foundation of modern application development.

Decision Making

Master if, elif, else statements for implementing complex conditional logic and branching program flow.

User Interfaces

Develop skills in creating form-based user interfaces using Streamlit's interactive widgets and components.

Grading Logic

Implement basic grading algorithms and mathematical calculations in Python for real-world applications.

Result Display

Learn to present calculated results in user-friendly formats using modern web interface design principles.

Project Completion and Next Steps

Congratulations on completing this comprehensive Streamlit assignment! You've successfully built a functional web application that demonstrates key programming concepts including user input handling, conditional logic, and dynamic result display.

This project serves as an excellent foundation for more advanced web development projects. Consider expanding your application with additional features such as data persistence, multiple student records, or advanced analytics. The skills you've developed here—form handling, conditional logic, and user interface design—are transferable to many other programming projects and real-world applications.