‘Bridging Pedagogy and AI: A Dual Grading Framework with Image Recognition for Academic Evaluation’

(GPA Calculator Project)

Course Information: Inferential Statistic and Applied Probability: DS-221
Instructor's Name: Sir Sajid Ali
Teacher Assistant: Sir Aamir Khan Maroofi

Project By:
Muhammad Haris 2023428

ABSTRACT:

This project introduces a Full Stack Python-based web application designed to simplify GPA calculation and analysis for educational institutions. Supporting both Absolute and Relative Grading techniques, the application provides accurate, swift results while ensuring fairness and ease of use through an intuitive graphical user interface (GUI). Leveraging modern data manipulation, statistical techniques, and Convolutional Neural Networks (CNNs), it allows users to upload images for instant GPA extraction, eliminating the need to navigate complex CSV files manually. The application also features advanced visualization tools for effective performance comparison and analysis. Accessible on any internet-connected device, this solution addresses the limitations of traditional GPA calculation methods, offering a user-friendly, efficient, and innovative approach to academic performance evaluation.

PROJECT CODE OF CONDUCT:

It is certified that the work presented in this report is be performed by Muhammad Haris alone, the student of FCSE, Computer Science major batch 33. The work is performed with the best of knowledge attained in the course DS-221, while satisfying all the adequate requirements and is submitted in due course of allotted time.

INTRODUCTION:

Every educational institute over the course of each academic year or half year semester evaluates the enrolled students based on their performance in each enrolled course. Many educational institutes grade the student first and then the total grade is summed to form the Grade Point Average, which not only provides analysis of the student's overall performance but also assists in analyzing their performance with respect to their own class. The task of GPA calculation and analysis requires well formulated techniques to maintain fairness in the overall calculation among students. The traditional ways of calculating the GPA can be very hectic and more often very difficult for instructors, faculty members etc. who do not have a major in mathematics or statistics, thus learning each technique from scratch can be very difficult.

Educational institutions may provide online apps, and tools that do not do the job well given their primitive interface and a high level of understanding the key functioning provided in analyzing the grades.

This paper introduces a solution to assist instructors, examiners, faculty members of any educational department to use the power of the modern data manipulation and statistical techniques uniting with Machine Learning to get swift and timely information about student GPAs based on either of the two major grading techniques that are:

1. Absolute Grading Technique
2. Relative Grading Technique

This paper introduces a Full Stack python-based minimalistic web application that can be accessed on any computational device at any time while being connected to the internet to calculate student GPAs in an instant while also providing statistical comparison and different outcomes based on the two above mentioned grading techniques. This web-based application is made with an attractive and easy to use GUI thus helping individuals of any educational domain to understand and use it well. Moreover, our model uses the large and extensive python-based libraries for visualization of results, easing in analysis and comparison. The results generated are showcased live over the web application while also giving the option to download them by just one click.

The most unique contribution to through our work is introducing the CNN based machine learning models for instructors/students to upload pictures to get instant results while not having to search through long and overly detailed csv files, thus easing the overall task of calculating and retrieving specificized GPA.

METHODOLOGY:

This study implements a comprehensive academic grading system that incorporates both relative and absolute grading methodologies, designed to provide flexible and fair student assessment as per the instructors/faculty need and requirements. The system is implemented using Python programming language with the front-end web application designed using Streamlit library of Python programming language, also utilizing data analysis and visualization libraries to process student performance data and generate detailed analytical insights. The dataset used for building and testing this python-based code program was taken from an Kaggle with the link attached as: https://www.kaggle.com/datasets/mexwell/student-scores.

Moreover, the complete code-based implementation of this paper can found on GitHub repository with the link attached as: _____.

System Architecture

The grading system is structured around two primary components the choice of option to select any one is given to the user:

1. Relative Grading System (RGS)

The relative grading system implements a standard deviation-based grading approach. This approach Offers both HEC (Higher Education Commission) Pakistan based pre-defined approach the details of which are given below:

HEC utilizes relative grading, a curvilinear process; most commonly, normal, in accordance with what it calls a bell-shaped standard. This ensures a normal distribution of grades among the students according to their relation to fellow peers. Such a normal distribution is characterized by the nature of certain percentages falling within standard deviations from the mean: 68 percent for one standard deviation, 95 for two, and very nearly all, 99.7, when enlarged to three. It can be used, however, to create a grade system defining ranges such as A, B, or C through means and standard deviations from maximum total marks. For example, A* is given to students scoring above two standard deviations from mean scores, while lower grades such as Cs and Ds are provided to those further below the mean. This method also contains a histogram of total marks as grade ranges and shows minimum achievements for mastery of course contents.

Fractionalized grading policy was aligned to the GPA/CGPA and was calculated in its proportional score in a semester and an annual academic system. "A" is equivalent to a GPA of 3.67-4.00 (85% and above); "A-" is equal to a GPA of 3.34-3.66 (80-84%); and so forth, down to "F," which is 0.00 GPA and below 50%. These criteria in this manner apply mechanisms that universalize grading-rendition of students and from clear criteria as to the academic performance through the different systems.

The other approach to the relative grading system is the ‘Custom Grading Schemes’ which basically allows the user to select the standard deviation thresholds to each grade thus defining the normal distribution on their own. This approach includes automatic setting and correctness of boundaries to avoid errors in results which would eventually lead to unfair marking. Thus, the overall approach automatically keeps track of the minute detail making it easier for the user to get result and draw understanding and intuition

2. Absolute Grading System (AGS)

This method of grading system uses predefined grade thresholds, in our application the predefined threshold is also divided into two options for the user to select, one option lies in utilizing the standard HEC (Higher Education Criteria) Pakistan’s grading criteria details of which are given below:

Guidelines provided by the Higher Education Commission (HEC) specify that an absolute grading method would be used to award letter grades based on students' performance concerning the level of content mastery and GPA/percentile thresholds. The distribution of grades has been prescribed as follows:

GRADES	GRADE POINT	OBTAINED MARKS
A	3.67 - 4.00	85 and above
A-	3.34 - 3.66	80-84
B+	3.01 - 3.33	75-79
B	2.67 - 3.00	71-74
B-	2.34 - 2.66	68-70
C+	2.01 - 2.33	64-67
C	1.67 - 2.00	61-63
C-	1.31 - 1.66	58-60
D+	1.01 - 1.30	54-57
D	0.10 - 1.00	50-53
F	0.00	Below 50

This absolute grading system is ideal for clarity and consistency in assessing learners with respect to specific levels of content achievement.

The other option available to the end user easy-to-use the end use to have customizable thresholds being set my themselves through easy-to-use horizontal toggle bars, thus making the overall approach quite easy and manageable for the end user, the customizable threshold bars are made to maintains consistent grade boundaries across different student cohorts/subjects, thus this approach also allow a fair evaluation and distribution of grade.

The above-mentioned details for absolute grading can be summarized as follows:

- Multiple grading methodologies (Absolute and Relative)
- Custom grade threshold definition
- Comprehensive statistical analysis
- GPA calculation with credit hour weighting
- Batch processing of student data

DATA PROCESSING PIPELINE

This Streamlit built application follows a modular structure, having unique processing paths. Following are the details of the complete workflow:

Initial step up and configuration of the application begins with importing all the necessary libraries and setting up the stream lit configuration. Following is the list of the libraries utilized in the code program:

1. streamlit: User for creating web applications and interactive dashboards.
2. pandas: Used for data manipulation and analysis.
3. numpy: Used for numerical computations and array processing.
4. os: Used for interacting with the operating system.
5. cv2 (OpenCV): Used for image processing and computer vision tasks.
6. torch: A PyTorch framework for deep learning.
7. torch.nn: Used for building neural networks in PyTorch.
8. torch.nn.functional: Provided functional forms of PyTorch modules (e.g., activation functions).
9. torch.optim: Used for optimization algorithms like SGD and Adam in PyTorch.
10. torch.utils.data: Used for data loading and manipulation in PyTorch (Dataset, DataLoader, random_split).
11. torchvision: Tools for computer vision tasks, including transforms and datasets.
12. PIL (Pillow): Used for image processing and manipulation.
13. matplotlib.pyplot: Used for data visualization through 2D plots.
14. seaborn: Used for statistical data visualization.
15. plotly.express: An high-level interface for interactive visualizations.
16. plotly.graph_objects: Low-level API for creating detailed Plotly visualizations.
17. plotly.subplots: Used for creating subplots for Plotly visualizations.
18. sklearn.preprocessing (LabelEncoder): Used for encoding categorical labels.
19. scipy.stats: Used for statistical functions and probability distributions.
20. logging: Used for application logging and debugging.
21. sys: System-specific parameters and functions.
22. datetime: Manipulating date and time information.
23. io: Handling streams and file operations.

24. base64: Encoding and decoding data in Base64.
25. typing (Dict, List, Tuple): For type hints in Python.

The Streamlit along with these libraries initializes the session state variables and also store the results and statistics across interactions. After this the main navigation structure of the application is set up which is a sidebar for navigating through various options/functionalities provided across the web application, these applications include:

1. Home
2. Automated Student Detection
3. Absolute Grading
4. Relative Grading

The data processing flow continues once the user selects either of the three options stated above (except the home page, which just showcases main details of the overall program). If the user select the automated student detection then he will be redirected to the page containing further options, where first he must train the face detection model, this training process would require the user to put is dataset by the name ‘data’ containing two sub-folders ‘test’ and ‘train’ which can further contain as many folders as the user want, these folders are to be named after the student and must contain images of that student. Once the directory is set, the user can return to the main web application and click on the train model to train his model on the data set provided, now after that the user shall be be provided with the option to input a new picture, which will predict the accurate student and thus show his result in either absolute or relative grading based on the user preference. The above-mentioned description is summarized below as:

- Handles image upload and processing
- Performs face detection and recognition
- Integrates with the grading system
- Returns student information and academic results

The process for the selection of absolute or relative grading from the navigation bar are more relative to each other and the details of it are given below:

1. Inserting the CSV datafile to read the data/marks obtained in various subjects
2. Allowing the user to define the credit hours for each subject read from the CSV file
3. Select the method of defining the grade points of each grade (HEC based/Custom)
4. Calculate grades of students in each subject
5. Computing overall GPA. It uses the following formula to calculate the GPA after all the grade points of each grade and credit hours of each given subject is defined:
   $GPA = \Sigma(\text{Grade Points} \times \text{Credit Hours}) / \Sigma(\text{Credit Hours})$
6. Generating statistical analysis (i.e. analyzing mean, std deviation, median skewness etc. across each individual subject and overall GPA)

7. Displaying graphical information like grade distribution bar charts, score distribution histograms for each subject, normal distribution graphs, Q-Q plots.
8. Providing the user to download results in CSV files after grading for further analysis and data storage.

FACE RECOGNITION MODEL

Key Components

1. Grading System

This system consists of two classes ('GradingSystem' which handles the Relative grading functions, and 'GPACalculator' which handle the absolute grading methods) and functions that will process as well as calculate the grade of students. The grading logic is based on absolute as well as relative grading, such as in HEC grading, and calculates the GPA with credit hours defined.

- **Data Loading and Preprocessing:**

'load_data()' methods are responsible for importing data from CSV files using dynamic identification for columns with scores (e.g., math_score, history scores etc). Errors for missing or badly formatted data are very descriptive so that they can be traced.

- **Relative Grading:**

Relative HEC grading is implemented through z-scores. This method of grading in statistics is proper in normalizing scores as grades are based on distances/standard deviations from the mean.

- **GPA Calculation:**

GPA computation is achieved by converting grades into grade points and referring to subject-specific credit hours. For subjects with no defined credit hours, defaults may be applied.

2. Face Recognition

That implement the predict_student() function for facial recognition, using PyTorch. A trained deep-learning model (FaceDetector) identifies the students by from input images.

- **Usage of Model Architecture:**

It loads a checkpoint with the classes as labels and weights already pre-trained. The preprocessing is done through resizing, normalization, and tensor conversion of an input image.

- **Inference:**

Afterward, the image flows through the model, at the end generating a probability for each class. The highest probability leads to a predicted student with his confidence scores.

Integration of Grading and Face Recognition

The process_student_image() function acts as a river between facial recognition and grading:

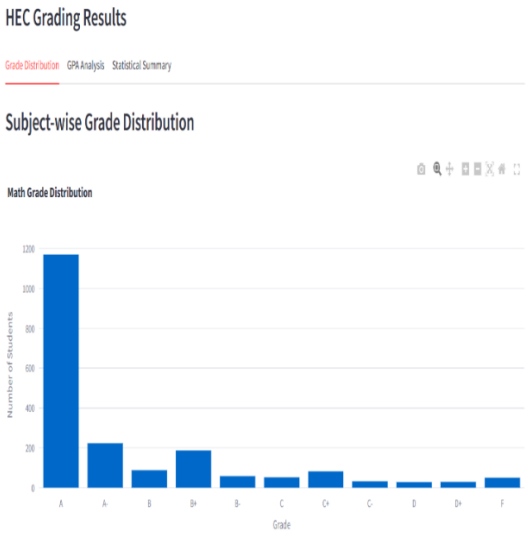
1. The prediction of the student is done through image recognition by the face recognition model.
2. The detailed student record is fetched through the predicted name from the CSV database.
3. The grades are processed using either of absolute or relative grading techniques

RESULTS

Thus, the web bases application gave very robust and accurate results that not only provided great insights through visualization but also assisted the user in getting timely and accurate results. Following screenshots highlight the main working of the program along with clear statistical insights regarding the results acquired:

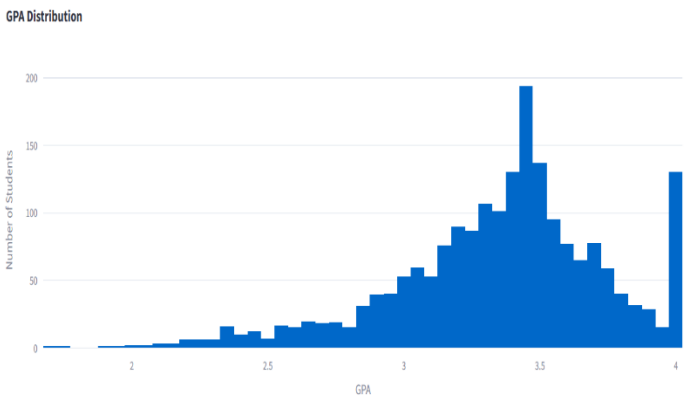
1. <u>Absolute Grading (HEC Defined thresholds)</u>:

The program not only provides subject-wise grade distributions displayed graphically but also the statistical analysis the number of certain grades obtained by each student and also its value in percentage. In the second tab of analysis labelled as 'GPA Distribution' our program uses the python's efficient libraries to highlight the graphical representations of the GPAS obtained by student over all the courses. Lastly the third tab of highlight the statistical summary of the GPAs obtained by the class with mean, std deviation and the percentage of repeatedly occurring GPAs. Thus, making it easier for the end user to understand the distribution of grades and GPAs of the students.



Grade Distribution for Math:

	Grade	Number of Students	Percentage
0	A	1,170	58.53
1	A-	223	11.16
2	B	88	4.4
3	B+	187	9.35
4	B-	58	2.9
5	C	52	2.6
6	C+	82	4.1
7	C-	32	1.6
8	D	28	1.4
9	D+	29	1.45



GPA Statistics

	Statistic	Value
0	count	1,999
1	mean	3.35
2	std	0.39
3	min	1.71
4	25%	3.14
5	50%	3.38
6	75%	3.62
7	max	4

Download hec_results.csv

Description of Attached Pictures:

The data showcases the performance of students in Math and their overall GPA distribution.

1. Math Grade Distribution:

   Over half the students (56.53%) secured an 'A' grade, indicating strong performance in Math.

   Lower grades like 'D+' and 'D' are rare (around 1.4% each), showing only a small number of students struggling.

2. GPA Distribution:

   The GPA histogram shows a peak around 3.0-3.5, with a mean GPA of 3.35, reflecting above-average academic performance.

   The GPA ranges from 1.71 to 4, with 75% of students scoring 3.62 or higher.

Overall, the data reflects strong academic outcomes, with most students excelling in Math and maintaining high GPAs.

2. Absolute Grading (Custom Thresholds)

This implementation of absolute grading give the power to the end user to define the threshold of each grade also manually inputting the grade point, he want to assign to each of the grades. The program takes in the user inputs well and thus displays the results based on the user defined input but also along side with HEC based absolute grading which allows the user to evaluate his inputs in comparison to the official standards. This comparison is performed in all three major descriptions showed by our python-based implementation.

1. Subject Wise grade analysis
2. GPA analysis
3. Statistical summary

Description of Attached Pictures:

Overall Distribution:

The scores exhibit a bimodal distribution, with peaks at low and high ends, indicating two distinct performance groups.

HEC Grading:

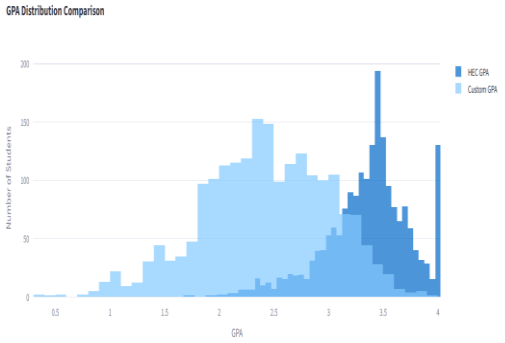
Grades are tightly clustered, with a high mean (3.89) and most scores at the upper end, possibly inflating performance.

Many students receive low grades in math (mostly "F" and "D").

Custom Grading:

Scores are more evenly distributed, with a lower mean (2.39) and wider range, providing better differentiation.

Math grades are balanced, with fewer "F"s and more students in middle grades ("A", "B", "C").



HEC Grading Statistics

	Statistic	Value
0	count	1,999
1	mean	3.35
2	std	0.39
3	min	1.71
4	25%	3.14
5	50%	3.38
6	75%	3.62
7	max	4

Download hec_results.csv

Custom Grading Statistics

	Statistic	Value
0	count	1,999
1	mean	2.42
2	std	0.59
3	min	0.34
4	25%	2.01
5	50%	2.43
6	75%	2.84
7	max	3.95

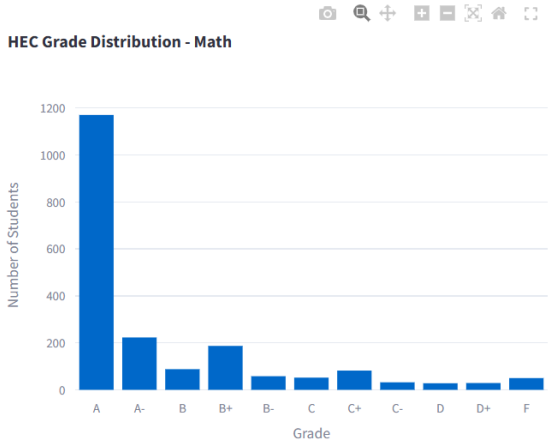
Download custom_results.csv

Download hec_results.csv Download custom_results.csv

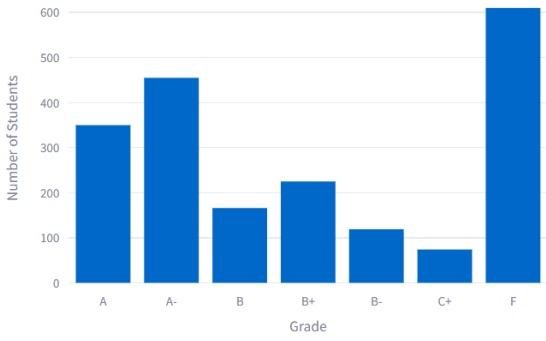
Subject-wise Grade Distribution

Math

HEC Grading



Custom Grading



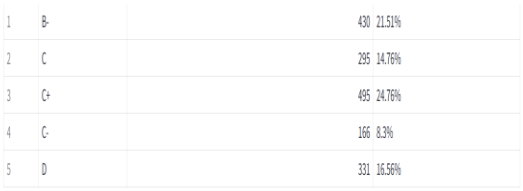
3. <u>Relative Grading (HEC-Defined Thresholds)</u>

This implementation of the grading system was based upon the normal distribution of grades around a fitting normal curve as specified by the HEC polices, given above. Here after uploading the data, the user was provided:

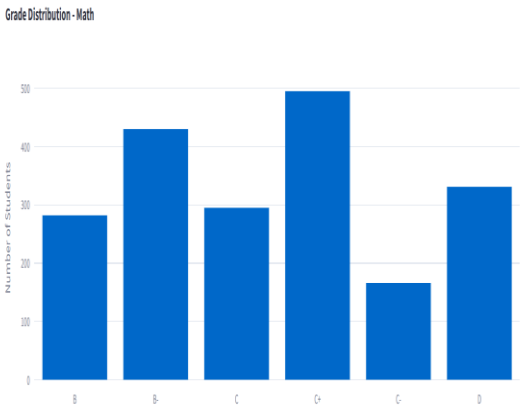
1. Grade distribution of students in each subject with statistical summary of the number of grades assigned to each student.

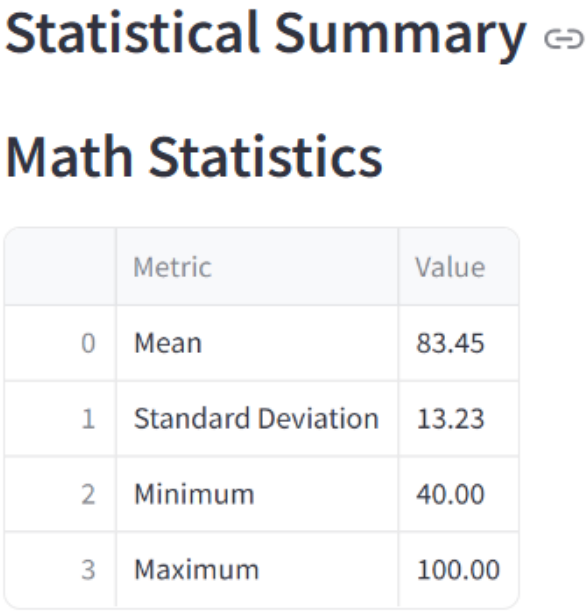
2. GPA Distribution of student.

3. Normal Distribution graphs and Q-Q Plots for better understanding,



1	B-	430	21.51%
2	C	295	14.76%
3	C+	495	24.76%
4	C-	166	8.3%
5	D	331	16.56%

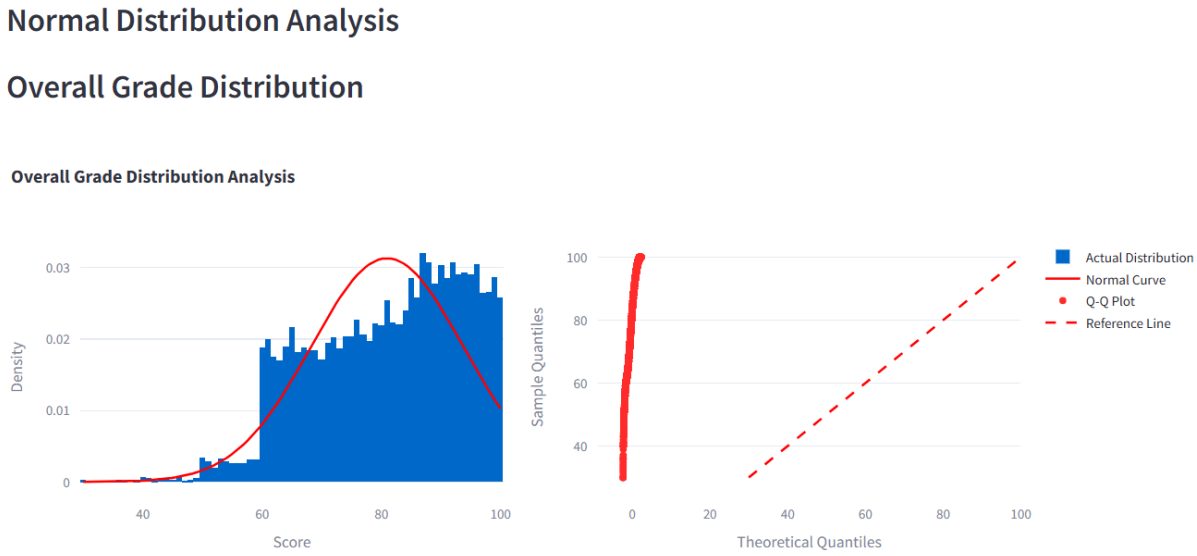




Statistical Summary

Math Statistics

	Metric	Value
0	Mean	83.45
1	Standard Deviation	13.23
2	Minimum	40.00
3	Maximum	100.00



Description of Attached Pictures:

Statistical Summary (Math):

Mean: 83.45, indicating overall high performance in math.

Standard Deviation: 13.23, showing moderate variability in scores.

Range: Scores range from 40 (minimum) to 100 (maximum), suggesting no extreme outliers but a significant spread.

Normal Distribution Analysis:

The bar chart and fitted curve show a fairly symmetric distribution of scores, close to a normal distribution but slightly skewed to the left.

The peak lies near the higher score range, indicating a concentration of students performing well.

Overall Grade Distribution:

The histogram confirms a peak around high scores, with a smooth normal curve fitting most data points.

The QQ Plot (Quantile-Quantile) indicates good alignment with the theoretical normal distribution, except for slight deviations at the extremes.

4. Relative Grading (Custom Threshold):

As described before this implementation of the code described the outcomes based on the grade threshold set by the user after tweaking the standard deviation limits of each grade. This implementation thus provided not the results based on the custom thresholds but also showed in comparison how the HEC based implementation the relative grading would had been. It provided the following details:

1. Custom Grade's distribution of each subject compared along sider with HEC defined grades. Also provided statistically summary of the number of grades assigned to students.

2. Provided a GPA Distribution graphs that was in comparison with the HEC based/defined grade outcomes.

3. Provided with statistical summary of each subject's grade distribution i.e. mean, std. deviation, min, max etc.

4. Normal distribution graphs along with Q-Q plots describing for more insights.

Description of Attached Pictures:

HEC vs. Custom Grade Distribution (Math):

HEC Grading:

- Grades are more evenly spread, with notable peaks at mid-to-low grades (e.g., "B", "C", and "D").
- Shows a broader distribution of student performance.

Custom Grading:

A larger concentration in higher grades like "A" and "B", with fewer failing grades ("F").

Indicates a more lenient or rewarding grading curve for mid-to-high performance.

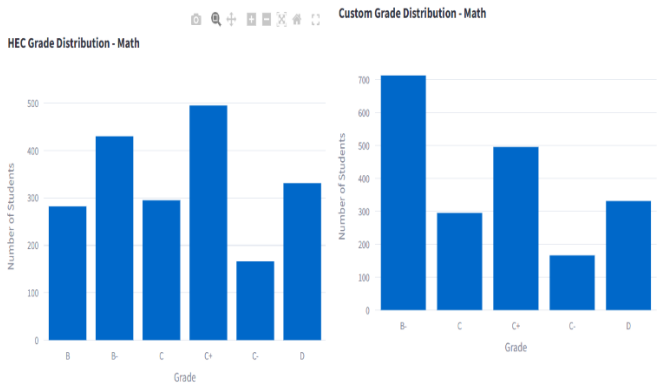
GPA Distribution Comparison:

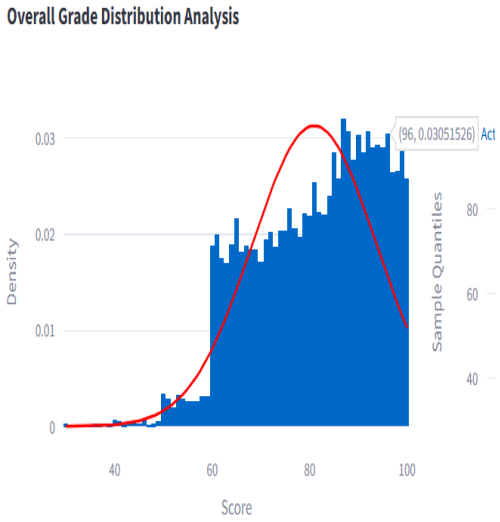
HEC GPA: Shows a broader, more uniform spread across GPA values, with a peak near the lower mid-range (around 2.2).

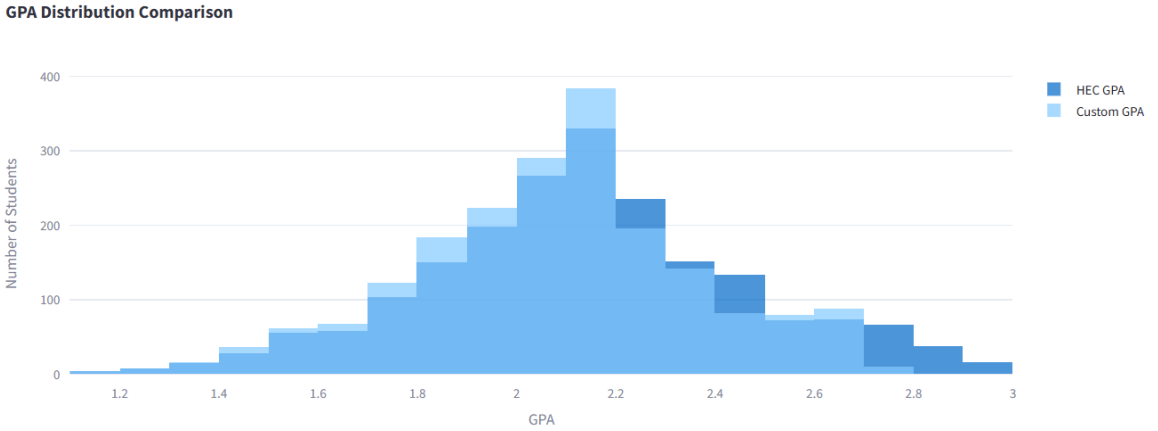
Custom GPA: Narrower and more concentrated around the middle, indicating less variability and tighter clustering.

2	C	295	14.76%
3	C+	495	24.76%
4	C-	166	8.3%
5	D	331	16.56%

2	C+	495	24.76%
3	C-	166	8.3%
4	D	331	16.56%

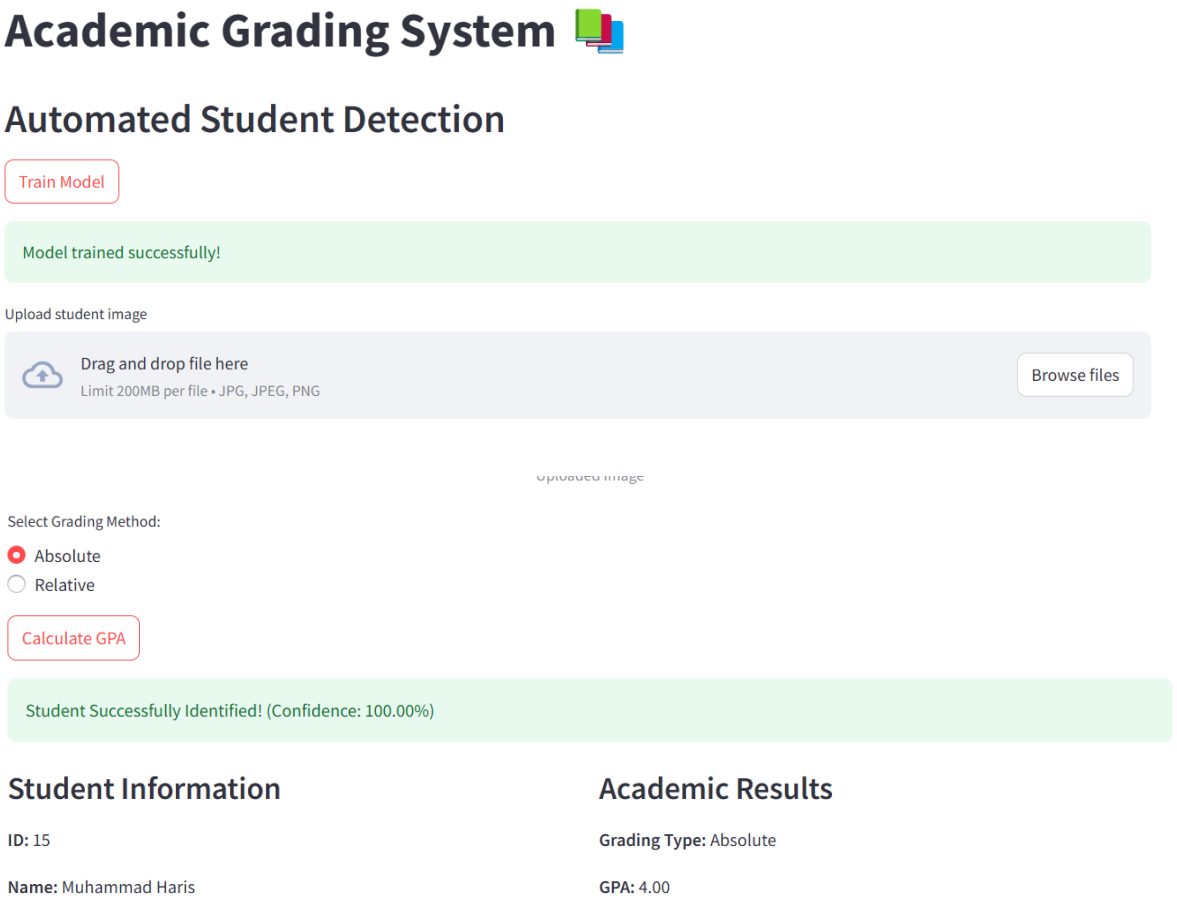




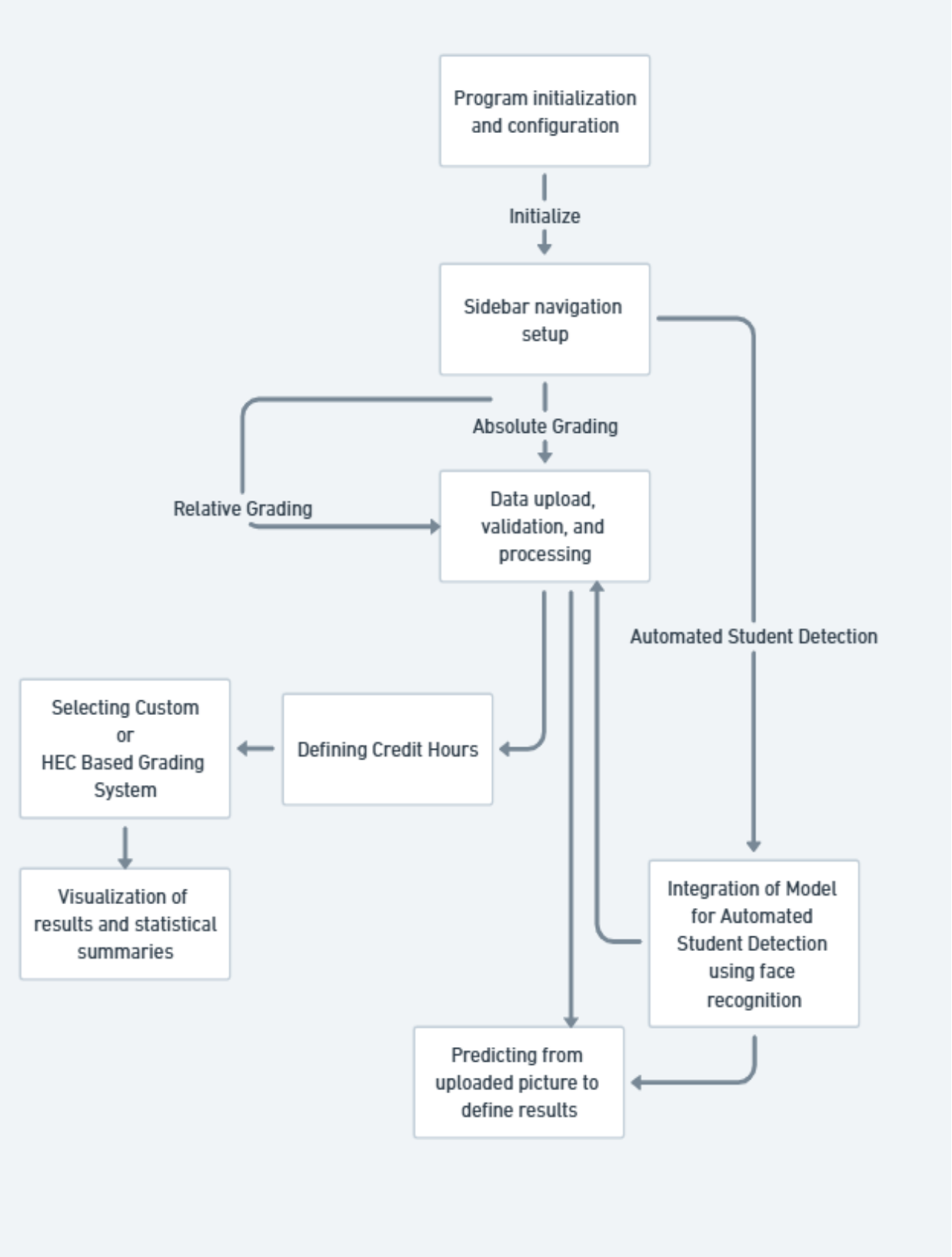


5. Automated Student Recognition:

With a healthy amount of dataset, the Face recognition model showed great success in showcasing outstanding results in predicting the right student and matching his name in CSV file to acquire the grade in either relative or absolute grading.



VISUAL SUMMARIZATION



FUTURE IMPROVEMENT

1. **Offline Functionality:**
   Introduce offline access to ensure that users in areas with limited internet connectivity can still benefit from the application.
2. **Customizable Reports and Analytics:**
   Allow users to generate detailed, customizable reports with insights such as individual subject performance, trends over semesters, and class rankings, enhancing the analytical capabilities of the application.
3. **Mobile Application Version:**
   Develop a dedicated mobile app to improve accessibility and usability for on-the-go users, with features like push notifications for updates or reminders.
4. **Multilingual Support:**
   Add support for multiple languages to cater to a broader audience and make the application accessible to non-English-speaking users.
5. **Integration with Learning Management Systems (LMS):**
   Enable integration with popular LMS platforms like Moodle, Blackboard, or Google Classroom to automate data input and GPA calculations directly from existing records.
6. **Enhanced Security Features:**
   Strengthen security measures, including data encryption, secure logins, and role-based access control, to protect sensitive academic information.
7. **AI-Powered Performance Predictions:**
   Incorporate predictive analytics to provide forecasts of students' future performance based on their current and past GPAs, helping educators intervene proactively.

CHALLENGES FACED

During the development of this project, several challenges were encountered, which provided valuable learning experiences:

1. **Designing an Intuitive GUI:**
   Crafting a minimalistic yet functional user interface that caters to both tech-savvy users and individuals with minimal technical background was challenging. Ensuring a seamless user experience required iterative feedback and rigorous testing.

2. **Implementing Grading Techniques:**
   Incorporating both absolute and relative grading systems accurately while ensuring the calculations remained fair and consistent was a complex task. Fine-tuning the algorithms to handle edge cases and diverse datasets added to the difficulty.

3. **Integration of Machine Learning Models:**
   Developing and integrating CNN-based models for image-based GPA extraction posed challenges in terms of model training, optimization, and performance. Ensuring the system's robustness for diverse image inputs was time-consuming and required extensive debugging.