User Manual for Eduvos Student Grade Calculator  
1.Introduction  
Welcome to the user manual for the Eduvos Student Grade Calculator developed by Zukisa Ncokazi. This software application is designed to assist students in calculating their overall grades based on weighted assessments. This manual will guide you through the functionalities of the interface and how to use them effectively.  
  
2.System Requirements (Pre-Software Launch)  
Before using the Eduvos Student Grade Calculator, ensure your system meets the following requirements:  
Operating System: Windows, macOS, or Linux  
Python version 3.6 or higher installed  
PyQt5 library installed (pip install PyQt5)  
Installation  
Python Installation: Ensure Python is installed on your system. You can download it from [python.org](http://python.org/).  
PyQt5 Installation: Install PyQt5 using pip: pip install PyQt5.  
  
  
3.Interface Components (Running the Software)  
3.1. CSV File Handling  
CSV File Path: Enter the path to the CSV file containing student grades manually or use the "Browse" button to navigate to the file location.  
Load CSV File: Click on the "Load CSV File" button to select the CSV file for importing student grades data.  
3.2. Buttons  
Calculate Grades: This button triggers the calculation of overall grades for each student based on the provided assessment weights (Quiz, Project, Final Exam, Practical Exam).  
Display Grades Here: Clicking this button populates the tree widget with the calculated grades for visualization.  
3.3. Tree Widget  
The tree widget displays student grades organized by modules (Quiz, Project, Final Exam, Practical Exam) and includes an "Overall Grade" column.  
Each row represents a student's grades for different modules, along with the calculated overall grade.  
  
4.Using the Application (Running the Software)  
Upon launching, you will see the main window of the application with various input fields and buttons described above.  
  
4.1Import Data:  
Click Browse Button to select your CSV file using the BUTTON or manually PASTE the file path in the slot above the button.  
Ensure the CSV file format adheres to the expected structure (module name, quiz score, project score, final exam score, practical exam score).  
  
4.2Calculate Grades:  
After importing the data, click on the Calculate Button to compute overall grades.  
The program will display the calculated grades in the Tree Widget.  
  
4.3View Grades:  
Click on the Display Button to view the calculated grades for each module.  
The Tree Widget will show module names, individual scores, and overall grades in a structured format.  
  
5.Additional Features ( Working internally within the Software)  
Multi-threading: The application uses multi-threading to improve performance during grade calculations, ensuring efficient processing of student data.  
  
Here's how it works:  
When the user clicks on the "Calculate Grades" button, the application spawns multiple threads, each responsible for calculating grades for individual students concurrently. This approach ensures that even if there are a large number of students or complex calculations involved, the interface remains responsive and doesn't freeze during processing.  
  
Here’s a breakdown of the process (Multi-threading):  
Thread Creation: For each student's data imported from the CSV file, a new thread is created using Python's threading module. This allows multiple calculations to occur simultaneously, leveraging the multi-core capabilities of modern processors.  
  
Parallel Processing: Each thread executes the calculate\_grade function independently. This function computes the weighted scores (quiz, project, final exam, practical exam) and subsequently the overall grade for each student.  
  
6.Handling Errors:  
If there are any errors in the CSV file format or data, the application will attempt to handle them gracefully and provide informative feedback.  
  
In my code I have incorporated:  
6.1 CSV File Validation: Before processing begins, the application checks each row of the CSV file to ensure that all numeric values (grades) are in the expected format. This is done using a validation step (all(item.replace('.', '', 1).isdigit() for item in row[1:])) during the read\_csv\_file function.  
  
6.2 Error Handling For Grace of My Application: If the application encounters any discrepancies or non-numeric entries in the CSV file, it gracefully handles these errors. Instead of crashing or displaying cryptic error messages, it logs informative feedback to the console (print(f"We apologize, we seem to think there is an invalid data in row: {row}")).  
  
7.Troubleshooting  
7.1Ensure that the CSV file adheres to the required format (numeric values for grades, correct structure).  
7.2File Loading Issues: If the CSV file does not load correctly, re-copy the file path address , paste it and then click the .  
  
8.Calculation Errors: Check for totally missing grades or incorrect data types in the CSV file. Ensure all values are numeric, for the software can and WILL NOT COOK SCORES where there are none.  
  
You now have a comprehensive understanding of how to use the Eduvos Student Grade Calculator. If you encounter any issues not covered in this manual, please refer to the troubleshooting section or contact support for assistance WhatsApp Zukisa Ncokazi at (074 254 5904).Thank you for choosing the Eduvos Student Grade Calculator.