Technical Design Document #1

**Paul Campos**  
**Date Created: July 6, 2025**

**Program Description:**

This program manages student exam records by collecting student names and their exam scores, saving them into a CSV file named grades.csv, and then reading the file to display the data in a neatly formatted table. It uses Python’s csv module for file handling, applying best practices such as the with statement for safe file operations. The program is organized with a Grades class containing methods to create the CSV file and read/display the contents. This modular design promotes code clarity and reusability.

**Functions Used in the Program (listed in order of execution):**

**1. Function Name:** create\_file()  
**Description:** Collects student data via user input, writes headers and records to grades.csv.  
**Parameters:** None  
**Variables:**

* csv\_file (str): Name of the CSV file (grades.csv).
* csv\_headers (list): List of headers for the CSV file.
* first\_name, last\_name (str): User inputs for student names.
* exam\_1, exam\_2, exam\_3 (float): User inputs for exam grades.
* rows (zip object): Combines the lists of student data for writing.  
  **Logical Steps:**

1. Open grades.csv in write mode and write the header row.
2. Use a loop to prompt the instructor to enter student names and three exam grades.
3. Store the inputs in separate lists.
4. After data entry ends, combine the lists into rows and append them to the CSV file.  
   **Returns:** None

**2. Function Name:** read\_file  
**Description:** Reads the contents of grades.csv and displays the data in a formatted table.  
**Parameters:** None  
**Variables:**

* csv\_file (str): Name of the CSV file (grades.csv).
* csvreader (csv.reader): CSV reader object to parse the file.
* data (list): List of rows read from the file.  
  **Logical Steps:**

1. Open grades.csv in read mode with with statement.
2. Use csv.reader to read all rows into a list.
3. Print the headers with formatting.
4. Print each student record aligned in tabular format using string formatting.  
   **Returns:** None

**3. Function Name:** main  
**Description:** Drives the program execution by creating an instance of the Grades class and calling methods to write and read the CSV file.  
**Parameters:** None  
**Variables:**

* grades (Grades): Instance of the Grades class.  
  **Logical Steps:**

1. Instantiate the Grades class.
2. Call create\_file() to collect and save data.
3. Call read\_file() to display saved data.  
   **Returns:** None

**Logical Execution Flow:**

1. The script is executed.
2. main() function runs.
3. Grades.create\_file() prompts for and writes student data to the CSV.
4. Grades.read\_file() reads the CSV and displays the table.

**Link to Repository:**

[CampsPA/Paul\_Campos\_CSV\_Programming\_Exercise](https://github.com/CampsPA/Paul_Campos_CSV_Programming_Exercise)

**Screenshot:**

A screenshot of a computer

AI-generated content may be incorrect.