# CSV FileManager

## Overview

This project provides a minimal C++ FileManager class to export and import numeric CSV files with a header row and optional "comment" lines (prefixed with #).

#### Features

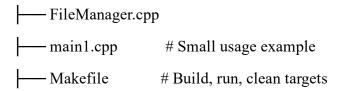
- Write CSV with:
  - o optional leading comment lines (# ...)
  - o a single header line (column names)
  - o numeric data rows (2D vector)
  - o collect data from comment lines
- Read CSV back into memory (comments, columns, and data separated).

## **CSV Format & Conventions**

- Comments: Any line beginning with # is treated as a comment.
- Header: The first non-comment line is treated as the header (comma-separated column names).
- Data Rows: Subsequent lines are numeric values separated by commas and loaded into a two-dimensional vector

## **Project Structure**

FileManager.h



## Use cases

• Name: UC1 – Export data to CSV

• Actor: Developer

• Goal: Save a 2D array with headers to a CSV file

• Preconditions: App has data in memory; target folder is writable

• Success outcome: CSV written with comments, header, and rows

## Scenario

## Happy path

- Given a dataset, column names, and a valid file path
- When the user calls export func("trial.csv", data, columns, comments)
- Then a file trial.csv is created
- And the first lines are comments starting with #
- And the next line is the header
- And following lines are numeric rows

## Alternate: invalid path

- Given a dataset and an unwritable path
- When export func is called
- Then the function logs an error, no file is created, and the program exits
- Name: UC2 Import CSV into memory
- Actor: Developer /Data Analyst

- Goal: Read comments, columns, and numeric rows from a CSV file
- Preconditions: File exists and is readable
- Success outcome: comments, columns, and data are populated; function returns true

## Scenarios

## Happy path

- Given trial.csv with # comment lines, a header line, and numeric rows
- When the user calls import\_func("trial.csv", data, cols, comments)
- Then comments are stored
- And cols contains header tokens
- And data contains all rows as doubles
- And the function returns true

# Requirements

• C++ compiler (e.g., MinGW g++ on Windows, g++/clang++ on Linux/macOS)

## Build

Using the provided Makefile (recommended):

mingw32-make.exe all

This compiles FileManager.cpp and main1.cpp and links them into a.exe (Windows)

## Run

mingw32-make.exe run

# Usage Example

```
#include "FileManager.h"
#include <vector>
#include <string>
int main() {
  std::vector<std::vector<double>> data = {
     \{1.00, 0.81, 0.59\},\
     {2.00, 0.31, 0.95}
  };
  std::vector<std::string> columns = {"it", "is", "done"};
  std::vector<std::string> comments = {"# sample header", "# another note"};
  FileManager fm;
  fm.exportFunc("trial.csv", data, columns, comments);
  std::vector<std::vector<double>> data2;
  std::vector<std::string> columns2, comments2;
  bool done = fm.importFunc("trial.csv", data2, columns2, comments2);
}
  unordered map<string, std::variant<int, double, std::string>> extracted;
  bool extraction = filem.comment extraction(comments2, extracted);
```

A working example with the same pattern is included in main1.cpp.

Writes comments, then header, then data rows into fileName. exportFunction

void FileManager::exportFunction(std::string fileName, vector<vector<double>> &data, vector<std::string> columns, vector<std::string> comments)

Parses fileName into comments, columns, and data. Returns true on success.

importFunction

bool FileManager::importFunction(std::string fileName, vector<vector<double>> &data, vector<std::string> &column, vector<std::string> &comments)

Extract data from the comments section already read from csv file

importFunction

bool FileManager::importFunction(std::string fileName, vector<vector<double>> &data, vector<std::string> &column, vector<std::string> &comments)

# Future additions

- Add current functions to a full project to treat the data collected from the CSV
- Accustom the CSV file structure to different required formats
- Add a function to append data to a CSV file while preserving existing data.

## Contributions

Nadeen Elgharably

Joshua Makar

Supervisor

Eng Khaled Mohamed