**Who Wants to Be a Millionaire – C++ Project**

**Student 1**: Oprea Tudor Alex
**Student 2**: Sava Sergiu

## I. Project Tasks

**Student 1 Responsibilities**:

- Developed the core quiz logic for App 1 (millionaire_game).
- Created the Player, Game, and Question classes.
- Handled JSON file I/O for storing and retrieving questions and player scores.

**Student 2 Responsibilities**:

- Built App 2 (millionaire_util) for managing the leaderboard and adding questions.
- Implemented the Leaderboard and QuestionManager utilities.
- Integrated command-line argument parsing (no cin used).
- Coordinated command-line usage for both applications.

## II. Data Structures

**Class: Player**

string name;

float score;

**Class: Question**

string text;

vector<string> options;

int correct_index;

**Class: Leaderboard**

vector<Player> players;

void loadFromFile(string filename);

void saveToFile(string filename);

void addPlayer(Player p);

void show();

**Class: Game**

vector<Question> questions;

Player currentPlayer;

bool used5050;

void play();

void showInstructions();

**Class: QuestionManager**

vector<Question> questions;

void load(string filename);

void save(string filename);

void addQuestion(...);

**III. File Structure**

- **questions.json** – Bank of quiz questions.


```
{
  "games": [
    {
      "questions": [
        {
          "question": "What is the capital of France?",
          "content": ["Paris", "London", "Rome", "Berlin"],
          "correct": 0
        },
        ...
      ]
    }
  ]
```

}

- **leaderboard.json** – Records scores for each player.

[

{"name": "Alice", "score": 5.0},

{"name": "Vlad", "score": 7.0}

]

**IV. Interaction with the Executables**

**App 1: millionaire_game**

This app handles gameplay and instructions.

**Usage:**

./millionaire_game start <player_name> <difficulty>

./millionaire_game instructions

- start: Launches a new quiz session or continues from saved state.
- instructions: Displays how to play and explains lifelines.

**App 2: millionaire_util**

This app manages the leaderboard and question bank.

**Usage:**

./millionaire_util leaderboard

./millionaire_util player <player_name>

./millionaire_util add_question "<question>" "<A>" "<B>" "<C>" "<D>" <correct_index> <difficulty>

- leaderboard: Displays all scores.
- player <name>: Shows scores for a specific player.
- add_question: Adds a new question to the database.

**V. Commands Implemented**

**App 1: millionaire_game**

1. start <player_name> <difficulty>– Start quiz.

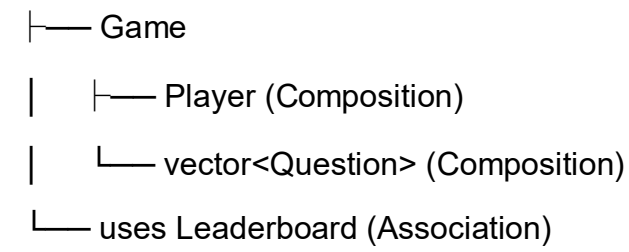
2. instructions – Show game instructions.

**App 2: millionaire_util**

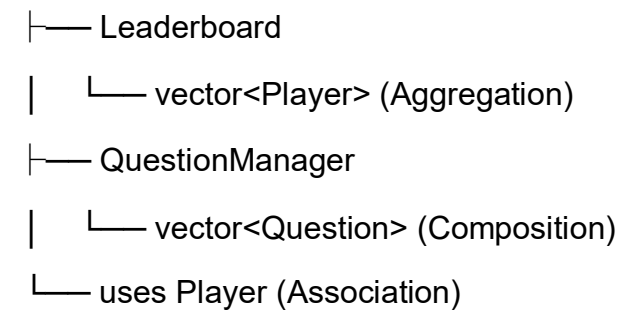
3. leaderboard – View global scores.
4. player <name> – View a player's performance.
5. add_question – Add a question to questions.json.

**VI. Class Relationships**

App 1: millionaire_game




App 2: millionaire_util




- **Composition**: Game owns Player and Question objects directly.
- **Aggregation**: Leaderboard aggregates Player records (loaded from file).
- **Association**: Loose interaction between Game and Leaderboard, as well as between Leaderboard and Player.