Programming Project / C++
- Hangman Game -

Student 1: Sandor Paul-Filip

Student 2: Popelca Andrei Ricardo

I. Task Description

Student 1 is responsible for managing shared resources:
- Adding, deleting, listing words in the dictionary (words.txt);
- Resetting or viewing the leaderboard and match history.

Student 2 is responsible for interacting with the game:
- Running Hangman rounds and calculating scores;
- Recording scores/ history and updating the leaderboard.

Bidirectional flow:

→ App 1 modifies words.txt → App 2 immediately plays with updated words;
→ App 2 records scores in leaderboard.txt & history.txt → App 1 can view them.

II. Data Structures Used by the Team

- Word: char word[100];
- Player: char name[100]; int score;
- History entry:
  1. char date[17];
  2. char time[8];
  3. char name[100];
  4. int gained;
  5. int total.

These structs are defined identically in both executables to allow symmetric parsing.

III. File Structure

words.txt
Each line: <word> (lowercase, no spaces)
leaderboard.txt
Each line: <player_name> <total_score>

history.txt
Each line: <YYYY-MM-DD> <HH:MM> <player_name> <points_gained> <total_score_after_game>
All files are plain text so both apps can append / read without binary dependencies.

IV. Interacting with Executables

Application 1: hangman_admin.exe

```
hangman_admin.exe list_words
hangman_admin.exe add_word <word>
hangman_admin.exe delete_word <word>
hangman_admin.exe reset_leaderboard
hangman_admin.exe view_leaderboard
hangman_admin.exe view_history
```

Application 2: hangman_game.exe

```
hangman_game.exe play    # starts a game session
hangman_game.exe view_leaderboard  # read-only
hangman_game.exe view_history
```