Istanbul Health and Technology University

Faculty of Engineering and Natural Sciences

Software Engineering Department

COURSE PROJECT – SWE208

Computing Systems

Project Title: Terminal-Based C++ Quiz Application

Team Members:

- Berat Kahraman (220610038)

Project Description:

I developed a terminal-based quiz application in C++ to demonstrate my understanding of key computing systems concepts. The program reads questions from a text file, allows users to take timed quizzes with colored feedback, supports secure question addition via username and password, and provides a summary of past quiz attempts. Intended users are students or instructors who need a simple, interactive quiz tool in a terminal environment.

Key Features:

- **Quiz Mode:** Presents shuffled questions with a 30-second countdown and colored correct/wrong feedback.
- **Add Question Mode:** Requires authentication against credentials stored in a file to securely append new questions.
- **View Stats Mode:** Displays total attempts, average score, and highest score from past quizzes.
- **Live Countdown Timer:** Uses non-blocking input to update time remaining on a single console line.
- **Authentication:** Reads credentials.txt and validates username:password pairs before allowing question addition.

Technologies:

- **Language:** C++
- **Development Environment:** Visual Studio Code with MSYS2 MinGW
- **Libraries:** Standard Library (<fstream>, <thread>, <chrono>, <vector>, <atomic>, <conio.h>)
- **Version Control:** Git for local code tracking

Implementation:

- **Data Structures & Algorithms:**
    - std::vector<Question> holds questions; each Question struct contains a std::string text, vector of options, and a char correct answer.
    - Shuffling implemented with std::shuffle and Mersenne Twister engine for random order.
- **Input/Output Handling:**
    - loadQuestions() parses questions.txt.
    - saveResults() appends scores to results.txt with percentage formatted via std::fixed and std::setprecision.
    - addQuestion() writes new entries to questions.txt.
    - viewStats() reads and computes statistics by parsing stored percentages.
- **Control Flow & Timer:**
    - Quiz loop uses _kbhit()/_getch() in a tight loop updating remaining time every second on one line.
    - Exits loop on keypress or when countdown reaches zero.
- **Security & Authentication:**
    - authenticate() reads credentials.txt and splits lines on : to match username and password before enabling protected operations.
- **Memory Safety:**
    - All dynamic data stored in STL containers (no manual memory management) ensures no leaks.
- **Challenges & Solutions:**
    - Resolved missing header includes by configuring MSYS2 toolchain and IntelliSense.

- Fixed timer-input conflict by switching from std::async to _kbhit() loop.
- Corrected linker errors by aligning function signatures in headers and source files.

Conclusion:

This project demonstrates practical use of file I/O, control flow, data representation, and memory safety in C++. By following an iterative development approach, using targeted research, and applying lessons learned, the final application meets the course requirements and provides a robust, interactive quiz experience.