Task 6. Final Project // Grading Details

The code for evaluation is checked for correctness against the requirements specified in the specification, as well as against the general requirements for C++ code. When giving a specific grade (on a 10-point scale), one should be guided by the following principles.

Grade	Requirements and notes
10	10 means that the work is *brilliant* (or at least *outstanding*) and it is almost nothing possible to improve. It is normal to have only 1, or 2 or 3 10s per course. This is in the spirit of the HSE. **Code**. Clear, readable, decomposed, structured. Does not contain any tricks. The code style is followed to the point with no single exception (even an opening curly bracket in the line of the function header). Decomposition is considerably logical. There is no requirement to use some specific complex C++ features. Moreover, overcomplicated solutions are unlikely to be considered ideal. Nevertheless, it is preferred to use OOP approach (as compared with the procedural one) to solve the problem, especially where it is clearly stated in the initial task. For instance, the model part of the application is represented by individual classes. The shared code is not duplicated, but separated in the base class(es).
9	9 means that the work is excellent, almost brilliant, but the solution suffers from the presence of some minor issues/bugs. **Code**. Still clear and great, however some minor bugs can be observed. They are mostly related to readability and the coding conventions. Any inefficient solutions / code duplication / unsuccessful decomposition make the grade lower than 9.
8	8 means that the work is great, but not excellent. It is a lower bound of “great-excellent-brilliant” category. All the points related to the “9” case are valid here, but the provided result is slightly lower than in the “9” case. 8 is also applied when 2 of 3 sections (code/report/discussion) are graded at “9” level, and 1/3 has some significant/major issue(s).
7	7 opens the levels for “almost great” results. It means, that at least 1 of 3 mentioned sections (code/report/discussion) suffers from presence of a number of significant/major issue(s). **Code**. The readability and decomposition is average, yet the code still implements the logic of application correctly (there could be only a few exceptional cases with incorrect results).
6	6 is a good mark which differs from 7 in a large number of problematic issues.
5	5 opens the levels for “sufficient” results. **Code**. Contains a number of major points or not more that a couple of critical (such as *having unreleased dynamically allocated memory* or *attempting to dereference a wrong pointer*).
4	4 is a satisfactory mark which differs from 5 in a large number of problematic issues. This is the lower bound at which the work of the students may still be considered acceptable.
3, 2, 1	3 and below means that the work and/or discussion with the student cannot be considered acceptable. **Code**. The program does not work correctly/cannot be compiled/produces unhandled exceptions of (almost) unknown nature.

Grade	Requirements and notes
0	Special mark that is used for indicating any critical issues. They include (but are not limited to): -absence of submission, -any kind of plagiarism.