IMAT3904 Report

Please write in the boxes below. Expand the boxes as you need to, however this report should not exceed 2 pages.

Name:	Adam Leonard Hubble	P Number:	P17175774
Github Username:	xGliff	Github Repo URL:	https://github.com/IMAT3904/cw-xGliff

Please summarise the functionality of your game engine (bullet points are fine):

- Logger system
- Timer system
- Windows system
- Event dispatching for all event types
- Input polling for all event types
- Running application window
- Application window resizes, minimizes, closes, moves, maximizes (windowed), full screen mode, pillar-box/ letterboxes and VSync capability
- All GLFW event call-backs
- All OpenGL related code abstracted from application
- OpenGL super simple renderer (3D)
- OpenGL basic renderer (3D)
- OpenGL text renderer (2D)
- OpenGL render commands
- Draw single character glyph
- Parse source code files - single or multiple files
- Resource manager for application assets
- FPS and orthographic cameras movable via mouse and key events and are switchable
- User Interface (UI) and game layer-related code separated and moved to sandbox application

What testing have you performed and what testing strategy was used?

Throughout the development of the game engine, I have predominantly explored the accuracy of my implementation through the use of the 'spdlog' logging system, in which, I have used the system to output the values of variables in the console window during program runtime. I used this method to observe gradual change and was mostly used for input polling and other input-related events. Furthermore, I have also made use of breakpoints within the IDE for debugging purposes, this has enabled me to observe the current values of variables during each iteration through the program; I believe this is an effective strategy for isolating errors, where it has enabled me to overcome difficulties on numerous occasions. In further relation of identifying errors in isolation, I have also conducted a series of unit test cases to identify the correctness or accuracy of units, as the result of arithmetic calculations, conditional statements or initialisation methods. I had approached to write unit tests after the engine's functionality was implemented, for which I utilised the test cases to refactor improper functionality; this allowed the software to behave as expected from a statistical basis. As previously mentioned, I made use of breakpoints, the logger system and the window to identify immediate errors and unexpected behaviours. Meanwhile, in relation to performance, I had conducted a series of performance profiling to analyse system performance when performing different tasks; this enabled my understanding of the systems efficiency and consumption. Similar to unit testing, performance profiling was also conducted after implementing functionality, to help reduce resource consumption where applicable. Lastly, I have also conducted a series of Blackbox test cases to identify functional errors immediately, from a visual basis; Blackbox testing has enabled me to acknowledge the working order of functional implementation, despite of not knowing each unit's correctness. Throughout the development of the engine, I have performed Blackbox testing regularly and before implementing further functionality.

Performance profiling testing available: [cw-xGliff/engineTests/PerformanceProfilingTests/]

Blackbox testing available: [cw-xGliff/engineTests/BlackboxTests/]

How have you approached your time management for this piece of work?

Originally when starting the coursework piece, I had invested the majoritive available time in each week to implement as much functionality as possible. This was abled by other modules coursework being issued later, I believe that this methodology made use of time available effectively. However, the ongoing weeks later required me to share time across of all my modules and their coursework pieces and so I aimed to continue to implement what I could gradually, each week. In which, I had made sure that I was making progress weekly; in earlier stages of the engine's development, I ensured that each week of the lectured content was implemented in the corresponding week. But as the work demand heightened and became more difficult, as well, other modules coursework pieces were due for submission earlier and required more time, I gradually deferred more time from the engine's development. However, I ensured to work on the engine's implementation one day per week at the minimum. For each day I had worked on the game engines implementation, I exhausted each available hour of the day in attempt to make as much progress possible, this also applied to other modules coursework pieces. Once most of the other module’s coursework pieces were submitted, I began to appropriate more time for the game engines development; I believe that my time allocation has been suitable throughout the engine’s development, nonetheless. It is inevitable that the time I have invested into my game engines development is the most I could possibly invest.

What have your learned from whilst building your game engine?

Throughout the development of the game engine, I have explored the concept of API abstraction, this was a new concept introduced to me; I now understand the importance of API abstraction for enabling the game engines use for building and deploying games on various platforms with varying system compatibilities/ requirements. Moreover, I have been provided insight into the way in which game engines are structured compositionally, in terms of all of the components which make up an engine and are considered when developing a game of varying scale. For which, I have become ever more confident in the application of OpenGL related code, as my previous knowledge of OpenGL application was only introductory, as learnt from a previous module. Also, I have been introduced to and become more confident within the application of Lambda functions; my knowledge and use for macro functions has been better informed as well, my previous implementation would only involve defining numerical data. Moreover, the concept of resource management has also been better informed, through which I was able to implement a resource managing system. In relation to systems, I have been introduced to a series of systems that can be used universally across the engine for debugging, arithmetic and initialisation-related purposes; of which, I have been newly introduced to a range of libraries that have been systematised, examples being ‘spdlog’ for the console window and ‘chrono’ for determining time between given periods. Meanwhile, in focus of the engine’s architecture, my implementation of singleton design patterned classes has become more proficient, as well has the use and initialisation of static class members. In addition, the design concept for using interface classes has been introduced to me and my application of these classes has enabled my understanding of their use and requirement for abstraction. Moreover, throughout the engines development I have also been introduced to multiple types of graphical renderer, which exist for different rendering purposes and for bettering system performance; system performance has been a focus of this assignment and has been explored thoroughly, I have become more knowledgeable in regard of managing system performance because of this. Lastly, it is without doubt that my knowledge of maps, vector containers, smart pointers and iterators has expanded due to their implementation and the difficulties I have had to overcome throughout the engine’s development.

In relation to my findings, throughout the implementation of the engine code I was mostly informed by the progressive delivery of lecture and lab content; from which, the basis of the functionality expected of me to implement, was given. Moreover, I was externally informed about the engine architecture and some of the featured content from the recommended-to-watch video series, the videos presented within the game engine YouTube playlist authored by ‘The Cherno’, was useful for implementing lectured content adapted from the series. The video series introduced me to some of the modules content and enabled me to develop an understanding of the functionality I was implementing, steadily. Watching the series on numerous occasions was essential for my development. Furthermore, I had also visited many online websites for assisting my use of newly introduced libraries and syntax, as well, I made use of websites and forums for help with implementing functionality that become problematic. Inevitably, all of the sources I have approached has dramatically impacted my programming ability over the development of the game engine.

If you were to undertake this piece of work again what would you do differently?

Given the outcome of my game engine, as I have not been able to fully implement all of the content taught, I would attempt to defer more time from other modules to satisfy more time available for implementing game engine content. This would aim to allow for all of the content’s implementation. Although I had revised all of the game engine series videos as advised to, before and during the semester, I would also aim to watch the videos as often possible where appliable; this may allow for enhancing the rate of the engines development. Over the course of development, I have also intervened with difficulties that seemed impossible to overcome at first and led to days of implementation being wasted before I had found a solution. To avoid this, I would seek for help as soon as possible. Moreover, in relation to testing, I would also aim to unit test the code I have implemented immediately before starting to implement the next week of content; this may also have enabled my development to be fastened.