# **INDIVIDUAL MILESTONE REPORT – V01**

**Every student** must submit a report at each milestone detailing all the work they have done for the class. This report is submitted through a submission link in Moodle (https://distance3.sg.digipen.edu).

**Subject Line**

The file name must be in the following format:

“**Mx\_ TeamName\_LastName\_FirstName \_IMR.docx**”. Replace **x** with a number between [1 … 6]

Incorrectly formatted files will be rejected, with high penalty!

**Contents**

The contents of this report must start with the following (in this order, with all caps for the section titles). Anything in italics here must be changed to whatever is appropriate for you (but should not be in italics in the actual submission you send).

MILESTONE: *Proof of Concept Prototype (or Functional Prototype, Pre-Alpha Prototype, Alpha, Beta, Gold)*

TEAM NAME: *My Team Name (if this has changed, then add “(formerly Old Team Name)” to this line)*

GAME NAME: *My Game Name (if this has changed, then add “(formerly Old Game Name)” to this line)*

JOBS/CHAMPIONING: *Product Manager, Game Designer, Graphics Programmer, etc. (if this has changed, make sure to note that)*

The next section contains a detailed listing of all the code you wrote this milestone. For every source file you wrote code in, state the number of lines of real code you wrote (within +/- 10 lines, you can use LOC to assist with this), the file name, and then a description of what the code you wrote does (this can be much longer than what is listed in the example below). Note that “lines of code” does not include comments, empty lines, code broken into multiple lines to inflate the count, repetitive cut-and-paste code, etc. Quality and difficulty of code counts more than quantity, so don’t just focus on writing lots of lines. Give us the TOTAL LOC at the end. Use a tool like **CLOC** to count the lines of code: “http://cloc.sourceforge.net/”.

Add the scripted files you’ve worked on. Example: .bat files, .lua files or C# files… [For custom engine side only]

*CODE:*

|  |  |  |  |
| --- | --- | --- | --- |
| **File** | **Lines of code** | **Describe your current work added/updated in the file** | **Reference (if any)** |
| *DataCenter.cpp/.h* | 181 + 41 | *Created system to read and write json file. Furthermore, this script will also contain all the master prefab of all object* | https://rapidjson.org |
| *Gameobject.cpp/.h* | *57 + 36* | *Created base Entity object that all object in the game will be base on* |  |
| *Main.cpp* | *5* | *Call helper function that contain in Datacenter to serialize, deserialize and set up the master version entity* |  |
| *ANC\_Game.cpp/h* | *39 + 203* | *Help create and setup game system. Create function to help render object and scenes* |  |

***TOTAL LOC: 562***

The next section contains a detailed listing of all the prototypes you made this milestone. Prototypes can be for gameplay, content, level design, or systems. Your prototypes can be made in any engine, although it makes sense to use your game's engine once it is available. For every prototype, state how many different versions you made and a description of the different versions. You must also have tested your prototypes, at the very least among your team members. Quality and difficulty of your prototypes counts more than quantity, so don’t just focus on making lots of very simple prototypes.

PROTOTYPE:

DefaultEnemyData.json contain all the main prefab data that all enemy in the game will base on. The system that is set will be use for all the other entity types/components like physics, window configuration etc.

GameObject.cpp/h contains all the data stats that are all present in the json file. A list of master copy for each type of enemy will be made using the data in the json file in which multiple copy can be made in from the master list

This section contains everything you did for this milestone that was not actual making of prototypes. This includes debugging, optimizing, testing, running playtest sessions, planning, meetings, designing, creating, or finding art, creating, or finding audio, preparing presentations, helping teammates, helping other teams, helping the instructors, buying food for the team, etc. Don’t worry if it “counts” or not—just include everything.

EXTRAS:

Help create the prototype in Unity in which our UXGD teammate is basing our game from

*Help our UXGD with the unity prototype (Game idea and mechanic. Introducing certain element like camera shake in unity prototype to better game experience)*

*Help bridge the gap between the artist, designer, and programmer to ensure minimum miscommunication and smooth development.*

*Helping reallocated task upon learning that a team member is planning to drop out. This is to ensure that the transition is as smooth as possible so that the team can continue with production whilst the person leaving will not feel any guilt for leaving the team.*

Finally, throw in anything else you think is relevant, including comments on how you think your teammates are doing (positive or negative), explanations for poor performance or absences, explanation of team changes, how the milestone/semester went, etc.

NOTES:

*As product manager, I tried my best to communicate and bridge the other discipline to the best of my ability. I make sure the line of communication between disciplines is open and transparent. Ensuring that no miscommunication or argument occurs. So far, morale seems quite high although there is still some uncertainty (a team members is dropping out, certain members are unable to catch up with other and sudden reallocation of task). I plan out with the best of my ability about the possible outcome and how to achieved/avoid them to ensure production is as smooth as possible.*