CS 2340 — Milestone 4: Project Iteration 3

Monsters, HP Stats, End game, and SOLID/GRASP Principles

BACKGROUND: For the third iteration of the project, your team will implement the final set of requirements. Additionally, you will review your code for the OOD principles you have been learning in class.

PURPOSE: Reviewing your implementation of the dungeon crawler can be a helpful exercise. This assignment allows you to reflect on your implementation choices and identify code smells that may have crept into your code. Identifying what went well and what could have gone better may help you avoid similar problems in the future.

TASK: This milestone has one team design deliverable, one testing deliverable, and a required feature set for implementation. Other than the requirements outlined below, the details of your implementation are up to you. **There is no in-person demo for this milestone.** Your app implementation/functionality will be graded via a video highlighting your work throughout all the major milestones.

Design Deliverable: SOLID and GRASP principles

- For this milestone, you will need to show examples of some of the SOLID and GRASP principles in your code with the option to instead show code smells for some of your examples.
  - Specifically, you will need to provide concrete examples in your team's code of 3 SOLID principles and 3 GRASP principles (either GRASP "basic" or "advanced").
  - If you are having difficulties with identifying examples of design principles in your code, you can provide up to 3 examples of code smells in your code and their possible solutions in your team's project to replace concrete examples of SOLID or GRASP. The solution should include specific action items that would result in a better implementation.
  - Your team must highlight at least 1 SOLID principle and 2 GRASP principles represented in your final deliverable. Code smells are not required to be included in your team's submission.

    - To recap, you should have a total of 6 examples, and at minimum 1 SOLID principle and 2 GRASP principle examples.

- To get credit for this portion of the assignment, you must take a screenshot of the code for each of your examples.
    - Each principle must be demonstrated in a unique example. When you have compiled your examples of design principles and code smells from your code, add the pictures to a PDF document.
    - For each SOLID/GRASP principle in your document, label it with the principle it demonstrates and a brief description (3-5 sentences) of how the example fulfills the coding principle. If the picture is for a bad code smell, give a brief description (3-5 sentences) of how the example represents a bad code smell and highlight a possible solution to eliminate the code smell.
    - Do not forget to submit a PDF of your code examples for this assignment.

Implementation Requirements

1. Implement **monsters**
    - Monsters should have a health or HP stat
    - If a monster's HP goes down to 0, it should die/faint
    - There should be at least 3 types of monsters
    - All rooms except for the starting room should contain at least one monster
2. Implement the player's ability to attack **monsters**
    - Player should have a health or HP stat
    - Player should be able to attack a monster
    - When the player attacks a monster, there should be an ability to hit the monster and take away some of its HP
    - There should also be a way for the monster to attack back and decrease the player's HP
    - If the player's HP reaches 0, they should die and end the game, and the user should be able to restart the game
    - When in a room with living monsters in it, the player should be able to retreat to the previous room, but the player should not be able to use any doors to rooms that they have not yet visited.

3. Exit room
   - The exit room should contain more monsters than in other rooms
   - Once all monsters in the exit room are defeated, the player should only then be able to perform an action to exit the dungeon and end the game. This should trigger the end game screen.
4. End game screen
   - The end game screen should congratulate the player on escaping or console them if they were defeated.
   - It should display at least three statistics from the game, such as total monsters killed, total damage dealt, time taken, etc.
   - The player should be able to start a new game or exit the app from this screen.

Testing Requirements

1. Write **unit tests** to verify the functionality the newly implemented features.
   - There is no code coverage requirement, but you should make sure that your unit tests cover meaningful functionality of the implementation requirements.
   - ***Each*** *team member should add at least* ***two*** *unit tests.*
2. **Testing Deliverable**: Include with your submission a brief writeup describing your testing process for the milestone. Explain which components were chosen for testing and why. Additionally, explain how your tests verify that the code functions as expected. Submit this as an additional pdf on Canvas.

Video Demo and Competition:

In your video demonstration, you will cover all the implementation requirements from this milestone (M4) and highlight meaningful functionality from previous milestones. Your video should be 3-5 minutes in length. Think of this as an application showcase that you would present to a potential client or investor. The contents of your video must display:

1. All implementation requirements for M4 (Monsters, HP stats, End game).
2. At least **3** implementation requirements from previous milestones (M2-M3). Some examples of functionality that could be shown are the

welcome screen and the maze of rooms. These are typically shown as bullet points in the milestone descriptions.

Additionally, we will have an optional, video demo competition. Your team may opt into the competition and may also choose to remain anonymous. **Please include if you would like to compete and/or remain anonymous in competition in your submission.** For the competition, the class will vote on which projects they like the most to decide the winners.

Optional: Extra Credit (15 points)

1. Brainstorm ideas for future CS2340 projects for up to 15 points:
    a. Provide a high-level description of your proposed project (~5 sentences).
    b. Additionally, provide a brief description (3-4 sentences) for each of the **five** (5) iterative milestones and the features they will include. The milestones should include implementation-specific functional requirements. For inspiration, please look at this semester's M2-M4 implementation requirements. Be sure to exclude M1-like implementation requirements.

Milestone Tagging

Tags are a way of marking a specific commit and are typically used to mark new versions of software. To do this, use "**git tag**" to list tags and "**git tag –a tag_name –m description**" to create a tag on the current commit. ***Important:*** To push tags, use "**git push origin –tags".** Pushing changes normally will not push tags to GitHub. <u>*You will be required to checkout this tagged commit during demo.*</u> This is done with **"git fetch --all --tags"** and **"git checkout tag_name".**

Submission Requirements

In addition to your deliverable, ensure that you include a link to your GitHub repository in your submission. Also, ensure that you have added your grading TA(s) and instructor as collaborators so that they may view your private repository. **Repositories must be located on the Georgia Tech GitHub and must be set to private!** Checkstyle will not be required for this milestone. Points may be deducted if these guidelines are not followed! All deliverables must be submitted to Canvas prior to the deadline.