Milestone 1 - Production Documentation

Group: **Abstract**
Production Manager: **Kyle James**
Due Date: **2/16/20**

1. Production Prompts

1.1 When and where does your group meet?

The group meets on Wednesdays at 1:00 pm in the open Golisano lab to discuss concepts for the video game.

1.2 How will your group communicate throughout this project?

The group communicates regularly on an organized Discord server.

1.3 How will your group handle task management throughout the project?

The group uses Trello to organize the production of the assignment. The group also uses a channel in Discord to organize upcoming due dates to help keep communication and task management in the same location.

1.4 What are the minimal specifications you need to have a working version of the game by the end of the semester?

At the very minimum, the game needs to have a functioning player, functioning enemies, levels, collectibles, score tracking, basic game states such as the menu, game, and game over states, and proper transitions within the game states.

1.5 Which elements of the game fall under “if we have time”? (“stretch goals”)

The stretch goals for the game include bosses, experience points, various classes for the player character, random map generation, and a shop system.

1.6 Based on the above, make a rough timeline/estimate of which tasks will be completed each week until the final milestone is due.

Week One [2/16/2020 - 2/22/2020]: Game states, menu system, start external tool development

Week Two [2/23/2020 - 2/29/2020]: Character movement [enemies/players], attacks, god mode
Week Three [3/1/2020 - 2/5/2020]: Collisions, start wrapping up external tools, collectibles
*
Week Four [3/6/2020 - 3/14/2020]: Level drawing with an external tool (random map generation)
Week Five [3/15/2020 - 3/21/2020]: Score tracking, statistics tracking, map drawing (code)
Week Six [3/22/2020 - 3/28/2020]: Player-character classes (different skill sets), data stat storage
Week Seven [3/29/2020 - 4/4/2020]: Boss enemies, shop system, experience points

*Note: The first three weeks are concrete due to the known deadline. Any of the following weeks are subject to change and workload may be increased or decreased per week depending on the deadlines.

2. Milestone 2 Tasks

2.1 Bulleted List for Bare Bones Game

- Create game states [menu, game, game over, pause]
- Create a finite state machine to manage game states and transitions
- Start coding external tool
- Create parent class GameObject
- Create Enemy class
- Create Player class
- Create a Collectible class
- Code Weapon parent class
- Code Weapon subclasses
- Code Walls
- Add Player controls (WASD)
- Create a Collidable interface
- Code Collisions between Player, Enemies, and Collectibles
- Code Collisions between environment objects
- Code Player attacks
- Code a God mode for testing
- Code managers (i.e. EnemyManager)
- Read in data from external file for enemies and weapons

Milestone 2 - Documentation

Group: **Abstract**
Production Manager: Kyle James
Due Date: **3/6/20**

1. Your name and your team name

Kyle James, Team Abstract Production Manager.

2. What was your role for this milestone?

My role for this milestone was to work on the debug mode requirement, game states and transitions, code clean-up, and basic production management; I filled a role similar to the "Gameplay" role outlined in the group project document.

3. Explain the features you worked on for this milestone: the individual classes, what they do and how they interact with other classes/systems.

I designed the finite state machine that handles the game states in the Game1 class, worked in conjunction with the other classes to set up debug mode, and I had to work within each of the other classes for code clean-up.

4. Does your code have any bugs or issues that need to be fixed? Were any bugs introduced when you integrated your code with the overall game project? What is your plan for tracking down and fixing these bugs?

There were no bugs that were introduced when I integrated my part of the code with the overall game project. There were only a small handful of bugs that were introduced in general, all of which were solved within a day or were inspected by the group to devise what was causing it.

5. Has the design, architecture, look, or timeline of the overall game changed since milestone 1? If so, how has it changed?

Each part of the game has changed since milestone in some small way; however, I solely worked on the production management in milestone one where, as of the moment, not much has changed.

5.1 Update any parts of your original milestone 1 write-ups and include the updated version (for instance, if your architecture has changed or evolved, provide an updated class diagram). *You only need to do this for your part of milestone 1.*

The only part that changed from my section of milestone 1 was the removal of an item from the 2.1 Bulleted List for Bare Bones Game and a small change to the timeline which can be found below.

5.1.1 Timeline Update

…
Week Four [3/6/2020 - 3/14/2020]: Level drawing with an external tool (random map generation)
Week Five [3/15/2020 - 3/21/2020]: Score tracking, statistics tracking, map drawing (code)
Week Six [3/22/2020 - 3/28/2020]: **Animations**, data statistic storage
Week Seven [3/29/2020 - 4/4/2020]: Boss enemies, shop system, experience points, **player types**
…

5.1.2 Bulleted List for Bare Bones Game Update

- ~~Read in data from external file for enemies and weapons~~

5.2 If you've decided to swap those roles, submit the updated version of the materials for your current role.

Nobody in the group desired to swap roles and was content to keep the role from Milestone 1.

Milestone 3 - Documentation

Group: **Abstract**
Production Manager: Kyle James
Due Date: **4/12/20**

1. Your name and your team name

Kyle James, Team Abstract Production Manager.

2. What was your role for this milestone?

My role for this milestone was to work on user interface interactions, re-organize game states and transitions based on new parts of the game, code clean-up, and basic production management; I filled a role similar to the “Gameplay” role outlined in the group project document.

3. Explain the features you worked on for this milestone: the individual classes, what they do and how they interact with other classes/systems.

I reworked the finite state machine that handles the game states in the Game1 class and I had to work within each of the other classes for some code clean-up.

3.1 Estimate what percentage of the total coding you contributed. If the work was split evenly, your contribution should be around 25% (group of 4) or 33% (group of 3). Your exact contribution will probably not be a perfect even split, so give us your best estimate.

The percentage of total coding that I contributed was probably around 20% since Dean took it on himself to get the map drawing done, which was a large piece of this milestone for our group.

3.2 If you created the external tool, explain what has been added to the final version.

I did not code either of the external tools.

3.3 In either case, feel free to go into detail about any problems you faced and how you overcame them.

I did not run into any problems; however, a few groupmate did, and I was able to assist them where necessary.

4. Does your code have any bugs or issues that need to be fixed? Were any bugs introduced when you integrated your code with the overall game project? What is your plan for tracking down and fixing these bugs?

There were no bugs that were introduced when I integrated my part of the code with the overall game project. There are no notable bugs that came up overall; however, we will begin game testing soon to try to find more.

5. Has the design, architecture, look, or timeline of the overall game changed since milestone 2? If so, how has it changed?

Each part of the game has changed since milestone two in some small way, namely in deadlines. However, I solely worked on the production management in milestone one and two, where, as of the moment, not much has changed.

5.1 Update any parts of your original milestone 2 write-ups and include the updated version (for instance, if your architecture has changed or evolved, provide an updated class diagram). *You only need to do this for your part of milestone 2.*

The only part that changed from my section of milestone two was the date changes and the addition of one extra goal for the timeline.

5.1.1 Timeline Update

…

Week Four [**3/6/2020 - 3/28/2020**]: Level drawing with an external tool (random map generation)
Week Five [**3/29/2020 - 4/4/2020**]: Score tracking, statistics tracking, map drawing (code), **UI interaction**

Week Six [**4/5/2020 - 4/11/2020**]: Animations, data statistic storage
Week Seven [**4/12/2020 - 4/18/2020**]: Boss enemies, shop system, experience points, player types

…

5.2 If you've decided to swap those roles, submit the updated version of the materials for your current role.

Nobody in the group desired to swap roles and was content to keep the role from Milestone 2.

6. What do you need to do to finish the game? Which features must be implemented, started, finished, cut, etc.?

The main feature that's left to be implemented is varying weapon types and enemy movement (in progress) within our newly developed maps. Everything else is a stretch goal and can be cut if necessary.

Milestone 4 - Final Documentation

Group: **Abstract**

Production Manager: **Kyle James**

Due Date: **5/6/2020**

1. What went right?

1.1 Frequency of Communication

Within our group, we met very frequently. At the bare minimum, we met one time a week, but we would communicate a lot on Discord about what we were working on at the time. I think the frequency of communication was crucial in the successes that had with the project. Being able to talk so often, especially after school closed, was helpful for the project, and probably the sanity of each of us as well.

1.2 Work Assignment

Keeping the workload balanced between everyone was an important task, more on this in the “What went wrong?” section. I would argue that at every week’s meeting, we would split up the work very evenly. The work that we desired to get done for the week divided among four people lessened the workload, and *ideally*, we would have all had an equal workload. I think the assignment of work worked well for us, primarily as we could discuss what we were capable of for the week.

1.3 Debugging

One thing we did well as a group was debugging. At one point or another, everyone in the group pointed out a bug. Despite each person’s vigilance in bug testing, not everyone knew how to solve the bug. As a group, we did an excellent job working together to debug the game as a whole. I think the approach that we took in discussing possible causes followed by close inspection of the code where the errors could originate in those possible causes worked well for us.

1.4 Adapting to Adversity

Although this might not have been something that I would have pointed out in a different situation, I think it's worth noting here. I think the group did an excellent job transitioning into working at home to get the project polished. Although it may not be as good as we originally planned, I'd say that everyone worked to the best of their ability, given the presented adversity.

1.5 Ahead of Schedule

Another thing our group did well was getting work done ahead of schedule for the milestones. As the production manager, I laid out a precise road map for what we should be getting done by the milestones. Yet, we consistently finished goals milestones that were unnecessary until the next milestone. I thought it was beneficial, especially when we went to work from our homes. Finishing things ahead of time allowed us to focus on polishing it up; however, some mistakes came with this, more on this in the "What went wrong?" section.

2. What went wrong?

2.1 Value of Communication

Although we met frequently, I think the conversations we had could have had more value. There were a few times where we would meet, and we wouldn't have a lot to discuss. Unfortunately, this resulted in many questions coming minutes after ending a call. I often felt that we didn't make any progress in some of our meetings. I would argue that although we talked a lot, it wasn't as beneficial as a better-organized meeting.

2.2 Lone-Wolf Work

We experienced a decent amount of this problem. We split up work a lot so we could work separately, which worked well. Still, occasionally some people worked on some pieces of the project without consulting others. This problem helped contribute to section 1.5 in that it helped us get ahead, but it left many others in the dark. There were times that I read the code and had little understanding of what was going on or how it connected because of it. *Because of this*, I became guilty of the Lone-Wolf Work issue as well. I went through and did, basically, a whole rewrite of the code in Milestone 4 to clean it up and understand it myself.

2.3 Committing with Errors

Committing with errors and bugs to GitLab was a big pet peeve for me. I found that if we ever needed to roll back to a previous version that it would be *impossible* to dictate what would be usable. There was a

small series of commits with errors, but an awful lot with bugs. Being restrained from being able to load back didn't cause any significant issues for us. Still, I can see how it could in the future, especially since some commits lacked any testing of the newly written code.

2.4 Pre-Production Planning

We were probably a little too unclear in pre-production with what we were looking to do. We covered the very ground basics listed in the group project document. Still, it was nowhere near enough. In Milestone 4, some group members were getting frustrated with what other group members were asking of them. Some of the work we had supposedly finished working on two milestones ago until they needed to work on it again in Milestone 4. If we had been more clear in pre-production, we wouldn't have had these issues so late into programming the game.

2.5 Behind Schedule

Contradictory of section 1.5, we also found ourselves behind schedule. I think moving off-campus had a lot to do with this, but I also think that there was complacency with our work after we got ahead of schedule. We were ahead of where we wanted to be in Milestone 2, but after those couple of weeks in Spring break, we fell off. It took us a while to get back into a rhythm, and we fell behind. It was stressful. When some work didn't get done, it would hold up other portions of the project, but no one person was responsible for falling slightly behind at any point.

3. What did you learn from this process?

3.1 Coding Standard Importance

I came to understand the importance of establishing a coding standard within a project. I found it very difficult to go through some sections of code as they had different foundations than the rest of the code. Some people were explicit with comments and what was going on, while others were not, leaving some of us in the dark about what certain lines of code accomplish. If we establish a standard with what it should look like, it may have helped everyone stay involved a little better. More on this in section 4.

3.2 Balance Is Key

I learned that there is profound importance in keeping everyone's workload balanced. It's more challenging to keep people involved when they get less work than others. This issue came in part due to the Lone-Wolf Work issue. Sticking to a schedule may be more beneficial in the future so that no one person gets overwhelmed or underwhelmed. If a person feels like working on more, I think

communication would be crucial in allowing others to build off of the work with ease, relative to what was happening at times in this project.

4. What will you do differently next time (in terms of code, design, your team, etc)?

4.1 Pre-Production

In any future project, I'm planning on spending much more time in pre-production. I think that creating a well-designed document outlining everything would help prevent issues popping up in later milestones. If we elaborate more on the project in the earlier stages, there would be fewer misunderstandings and less frustration later on in the project.

4.2 GitLab & Code Standards

There were many issues with our code at various points in the process. I found it impossible to roll back to any former commit as there was no way to tell what the last stable version was. In the future, I think teams need to understand the purpose of GitLab is not just to save the program, but rather to make available to everybody else. There should be extensive testing before pushing the team to avoid issues. As for coding standards, they go hand-in-hand with this. Some people write with far fewer comments, then go back through to comment their code later. Unfortunately, some things don't get commented on later. If any other person needs to edit or use that section of code, it makes it very difficult. Establishing a coding standard for comments and name conventions may be very beneficial as a part of the pre-production stage to keep the project organized.