The 3 week development process of our recreation of The Legend of Zelda NES went through many useful stages and we learned a lot. We utilized each milestone well and worked hard to make those deadlines. At first for p1_milestone, we attempted to make everything bug-free right off the bat. This didn't go too well for us since we were still mostly getting our sea legs with Unity and bugs are going to come around no matter what. Pretty quickly into it, we realized that we needed to take what we could get and leave bug fixes until later when more stuff was implemented into the game. After this realization, we began to focus more on the big things and getting done what needed to be done and save the polish for p1_gold. Even though our game was very buggy after p1_alpha, we were able to create a nice, polished version of The Legend of Zelda at the end of it due to our prioritization and getting done what needed to get done at the right stages.

We utilized Jira very well during our project lifetime. At the very start of each deadline, we created all the issues as tasks that were required. Whenever we found a difficult bug with something, we would write it up so that when one of us had time, we could pick it up and complete it. Using this method allowed us to stay organized and keep track of what was finished when and who completed it. It also let us visualize how much work we were doing and what we had left as well as accomplished during our project.

Our project goal did not change much with playtesting and feedback. We always had the same goal, but there were many times we would playtest, find a bug, and write it up in Jira to be completed later. This cycle showed us what we still had to do as well as let us focus on what really mattered in the moment for the game making process.

The initial idea for the custom mechanic was to change the game into a platformer rather than a top down tile map game. This idea was inspired by the lecture given about Undertale.

Another idea was to change the size of Link based on inspiration from Mario mushrooms. In the end we decided to settle for a more puzzle-like mechanic with the pushing of different color-coded blocks.

The idea of color-coded pushable blocks where each block serves a different purpose came up when implementing pushable blocks in the normal dungeon. We found that in the normal dungeon, both blocks had similar functionality, each bringing Link to a different room. We thought why not bring these secret blocks to the next level? Thus we added different pushable blocks along with tiles that must be stepped on to activate different effects.

The goal behind this specific mechanic of the colored blocks was to increase the amount of puzzle elements in the game. While in the normal dungeon players do feel a thrill of actually being in a dungeon/labyrinth when they discover a secret movable block that leads to a secret room, we feel as if there are more fighting for survival elements than puzzle elements. We want players thinking before performing certain actions. We want players to actually feel like the dungeon is filled with mazes and puzzles to solve, not just a sequence of rooms where you are required to kill enemies to proceed.

By utilizing Jira tasks and constantly communicating with each other, we were able to finish and submit all the deliverables on time. We divided tasks between team members accordingly, with tasks relying/based upon other tasks being assigned to the same member. This allowed us to work efficiently and finish tasks in quick succession. We were also able to solve bugs quickly by immediately writing up a jira task or asking the member who had worked on the particular feature, allowing them to be worked on immediately.

Particularly during the milestone and gold deliverables, we did not playtest our game as extensively as we could have. This caused us to discover many bugs at the last minute, and we had to work very hard leading up to the submission deadline to fix these bugs. Another problem we encountered was with our compartmentalization of the tasks. While this allowed us to quickly finish some tasks in succession, it resulted in several cases of each of us developing components with identical functionality that could not interact with each other. We ended up having to delete several and go with the other.

In future projects, we will make sure to playtest each feature extensively, starting early in development instead of right before the deadline to make sure we can catch bugs and fix them early and throughout the development cycle. We will also strive to identify pieces of code and components that are the same throughout multiple tasks/objects so one person can work an a single implementation that can be used for all of them, saving time and removing and redundant work.