**Standard Documentation:**

**Game Name:** Stinky Pet

**Game Description Short:**

Raise your virtual pet and see how it grows up depending on how you care for it.

**Game Description Long:**

Is this a game? Is this a simulation? Who knows. But it is playable and not coded like an Italian dish, so it has some worth in the world. Inspired by silly little games like Tamagotchi and Pakka Pets, Stinky Pet aims to bring the highest quality of virtual pet game known to man. It has the only foods in existence, apples and pizza, and even has a Gameboy in it.

The pet will evolve depending on care, with each stage having three possible evolutions. If the pet is taken care of well, it evolves into a happy pet. Other than that, poor care makes a pet evolve to an angry pet, and horrible care makes a pet evolve to a corpse. Once the pet has survived into adulthood the player’s job is done and the pet will go out of the house to explore the world. If it’s a smart pet, it will get a degree in programming and actually use SOLID when designing applications.

**Credits:** Karen Spriggs (best programmer eva!!!)

**Genre:** Simulation

**Developer Log:**

The most important choice that I made regarding design was to split up the pet functionality into several classes that the pet would have instances of. That way, I could follow the single responsibility and encapsulation principles to create better code. Then I had a class that translated the information from the pet object to Unity and the rest of the classes. I also did that on a much smaller scale with the food and toy classes. The rest of the scripts were not written as effectively since I am still learning Unity (first class where I’m actually writing scripts for it and not following tutorials). I will probably be better at this stuff when I take game programming 2 next semester in person at the ungodly time of 6:30-9:30pm.

**Postmortem:**

**UML:**

**A picture containing text, computer, bunch, parking

Description automatically generated**

**Dependencies:**

A lot of classes were dependent on the pet class, which makes sense since the data from the pet has to get to other classes for the game to work. The UI manager, game manager, and all consumables save an instance of the pet class as a field for use with methods.

**Class Design:**

I feel like my class design was pretty good, and I don’t know what else I would change about it. Maybe as I learn more about Unity, I will develop a perspective that will allow me to find issues with what I’ve made. For now though, I think I did a good job designing the code structure.

**Code Reusability and Maintainability:**

The majority of the code relating to the virtual pet itself is not tied to Unity and would work in Monogame if there was an equivalent extension of it like there is with Unitypet. This includes the petstats, petmove, petevolution, evodatabase, and pet classes. The lossmanager class and the consumables are also not specific to Unity. It would be hard to give an accurate percentage of code that is reliant on Unity, but I would expect it to be under half.

It would be pretty easy to add new pets, toys, and food to the game. For all of these, I would only need to make more sprites and then more prefabs for the consumables with different values. I could also easily add more evolutions to the pet evolution database, but it would be a bit more complicated as I would have to decide what the new threshold of care was for each evolution.

**Is the Game Finished?:**

I did pretty much everything in the proposal, the only thing I can think of that I did not include was having the pet run over to food or toys that were placed. This was not added due to the way I had the consumables spawn in since that would be the same x value every time. I would also have to add a few more things before I would ever consider uploading it anywhere, like actual passage of time and more features. For what it is, I feel like I accomplished what I wanted to since I focused more on good code than crazy features. If I redid the proposal I would not change anything about it since it was all realistic in terms of scope and time frame.