**Project Description**:

My project is called 112Fishing, a fishing game where the player can catch fish, sell them for money, and buy better rods, baits, and powerups to catch rarer fish.

**Competitive Analysis**:

There are a multitude of fishing games or minigames, such as Cat Goes Fishing, Stardew Valley (fishing minigame), and more basic, casual projects created for school assignments. Naturally, the complexity of the fishing varies in all of these games. The more basic fishing simulators simply had a random number generator choose a random fish that would be caught, with no graphics involved. A more complex game had animations with fish swimming back and forth, as well as a controllable fishing line to hook them. Stardew Valley’s fishing minigame does not have the fish swimming around, but when the player is near an eligible body of water, they can cast a line and catch fish by filling a tension gauge.

My game will be most similar to Cat Goes Fishing, where the line can be controlled and different baits can be used for different fish, and those caught fish can be sold to progress in the game. Unlike other games, my fishing game will incorporate real-world physics in the animations of the fishing line and fish. Similar to Cat Goes Fishing, I will also offer power-ups for the player to buy. Another mechanic I will be incorporating is seasonal and time-of-day changes in availability for fish, which was found in Stardew Valley, but not in other games. Unlike Stardew Valley, this mechanic will use probability distributions.

**Structural Plan**:

I anticipate using only one file with all classes and functions.

The fish will be organized under one Fish superclass, with the various types of fish as subclasses of the parent class. Similarly, the baits and rods are also stored in classes.

There are 6 different modes for the different screens: startMode, helpMode, prefishingMode, fishingMode, shopMode, and menuMode. startMode is the starting screen, helpMode is the instructions screen, prefishingMode allows the player to switch between modes and start fishing, fishingMode contains the actual fishing game, shopMode allows the user to buy bait, rods, or powerups, and the menuMode allows the user to view fish they have caught, switch between baits and rods, and toggle time and season.

Each of the modes will have individual keyPressed(), mousePressed(), and other cmu\_112\_graphics controller functions, as well as their individual redrawAll() functions. Variables used throughout the app are stored in appStarted().

**Algorithmic Plan**:

The first complex part of my project is managing the physical interactions between the bait, fish, and fishing rod. Most notably, the movement of these objects within the water and air will be most challenging. I will first calculate and store the different measurements of the baits and fish that I will need in order to manage their movement through the water and air. This includes measurements such as volume, mass, and area. Then, I will continually calculate the forces in fishingMode\_timerFired(), adjusting the acceleration, velocity and depth of the bait as needed.

Another major complex part of my project is managing the probability of different fish spawning based on time of day and season. I will approach this by creating four probability distributions: one for each season. Within each season, depending on the time, the probability of each fish appearing will vary; some fish will be more likely to appear during the morning, day, evening, or night. These probabilities will depend on the type of fish. Furthermore, some fish will only appear at certain depths within the lake.

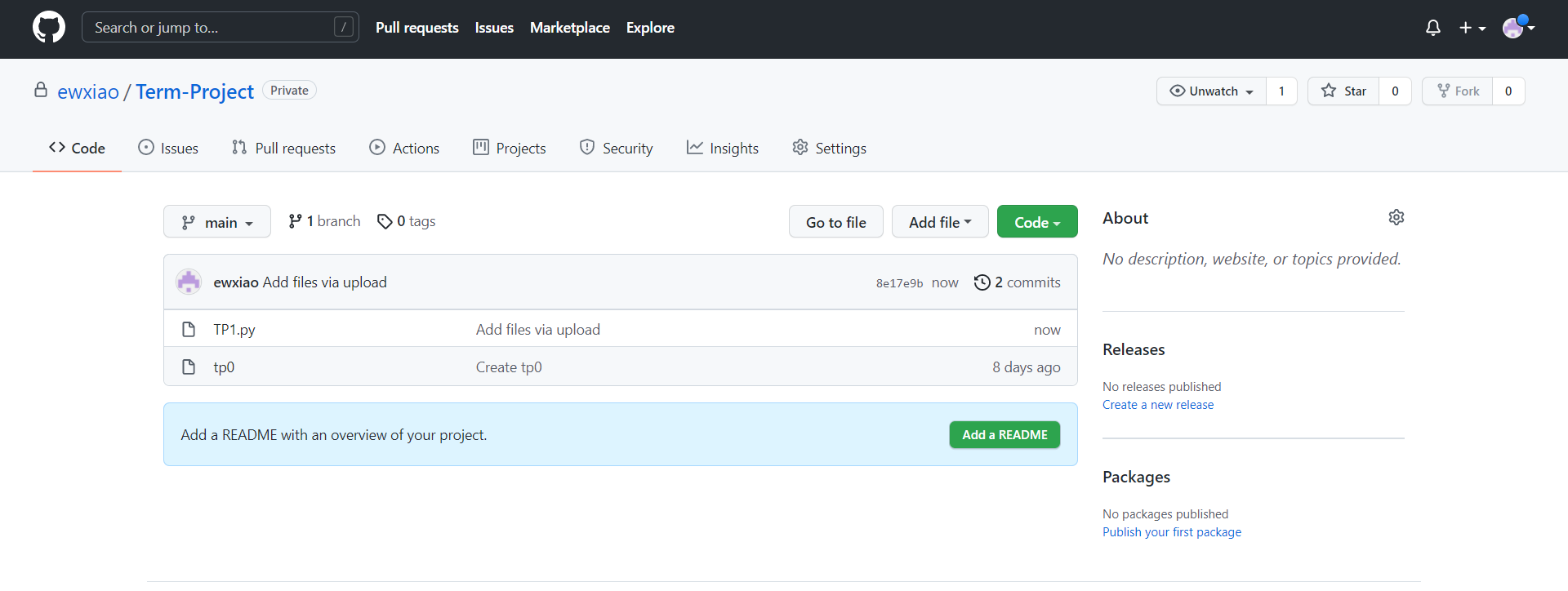
I will also incorporate randomly-spawning schooling fish, which will move according to the Boids algorithm. The school will be a subclass of the Fish superclass, and be composed of a list of boids. In fishingMode\_timerFired, loop through the boids and recalculate their acceleration, velocity, and acceleration. All boids must adhere to the separation, cohesion, and alignment rules, which is how their movement will be recalculated every 100 ms.

**Timeline Plan**:

By TP1, I will have completed the physics of the bait and fish, as well as the catching mechanism. By TP2, I will have completed the probability distributions of the fish, incorporated boids, and complete the function of each of the screens. Ideally, I would have also started importing images for fish and menus to use to improve the gameplay experience of the user. By TP3, the game will be complete with all images imported and functioning in my game.

**Version Control Plan**:

I will back up my work on github.com:



**Module List**:

I will not use modules.

**TP2 Update:**

I will instead focus on implementing boids and sideways motion instead of probability for TP2. I will also attempt to split up my code into multiple files for clarity and manageability.

**TP3 Update:**

I decided to not implement the powerups or probability appearance mechanisms. Instead, each type of fish has a chance to appear every 0.5 seconds. Instead of buying new rods or baits, the player can use the money they earn from catching fish to buy new rod colors, which they can toggle on the ‘prefishing’ screen menu. I also included images for the fish, hook, player, and splash screen.