Lucky B. Wavai

Project Proposal

**Project Description:**

**Name:** Collegiate Combat

**Description:** Collegiate Combat is a 2D third person fighting game. The game is made using Python’s third-party module, Pygame. In the first installment of Collegiate Combat, the player will be able to choose the character of their choice, and either vs other players in battle mode, or go off on the story mode to become immersed into the Carnegie Mellon college experience, facing off 1-on-1 against the heavy hitters of Freshmen fall and spring. The object of the game is to maintain an acceptable QPA (health in other similar games), and complete the infamous battle that is the collegiate experience

**Competitive Analysis:**

Despite the sheer originality that makes Collegiate Combat the game that it is, it however does share some common traits with other games of its genre. Collegiate combat shares some of the fighter style characteristics of well-known games such as Mortal Kombat, Street Fighter, and Tekken. These blockbuster games share gameplay and storyboard features with Collegiate Combat.

In terms of Gameplay, the player can choose characters who each have their own unique traits. Each character will obviously have different appearances, different special moves, fighting styles, and even different button combinations to achieve those moves. In the scope of the storyboard, players have the option to go through several similar screens. The player will first encounter a title screen, then a menu screen where they are free to choose which modes to play – single player, multiplayer, story mode, and settings – the character selection screen, and eventually the fighting screen; all the while a fire playlist plays in the background.

**Structural Plan:**

The final project will be organized in one main directory similar to Lukas’ Asteroids game in the following way:

* Main File: this python file will be written as a class, take in Pygame and pass information such as events to different level/game files similar to Lukas’ pygamegame file in Asteroids.
* Level/Game File: there will be several of these files; one for each mode of gameplay. These files will be written as class files which inherit from the main file similar to Lukas’ Game file. In these files, sprites will be initiated along with other mode specific content.
* Object File: this file will be written as a sprite class file which will be used as the parent class for initializing and updating specific objects in the game such as projectiles and characters similar to Lukas’ Game Object file.
* Projectile/Special Move File: this file will be written as a class file which special moves from different characters will inherit properties, and fire upon command. Similar to Lukas’ Bullet File
* Character File: this file will be the parent file for specific characters to inherit similar properties
* Specific character Files: similar to Luka’s Asteroid file, this file will be for the specific character, and implement there specific moves based on keypresses etc

**Algorithmic Plan:**

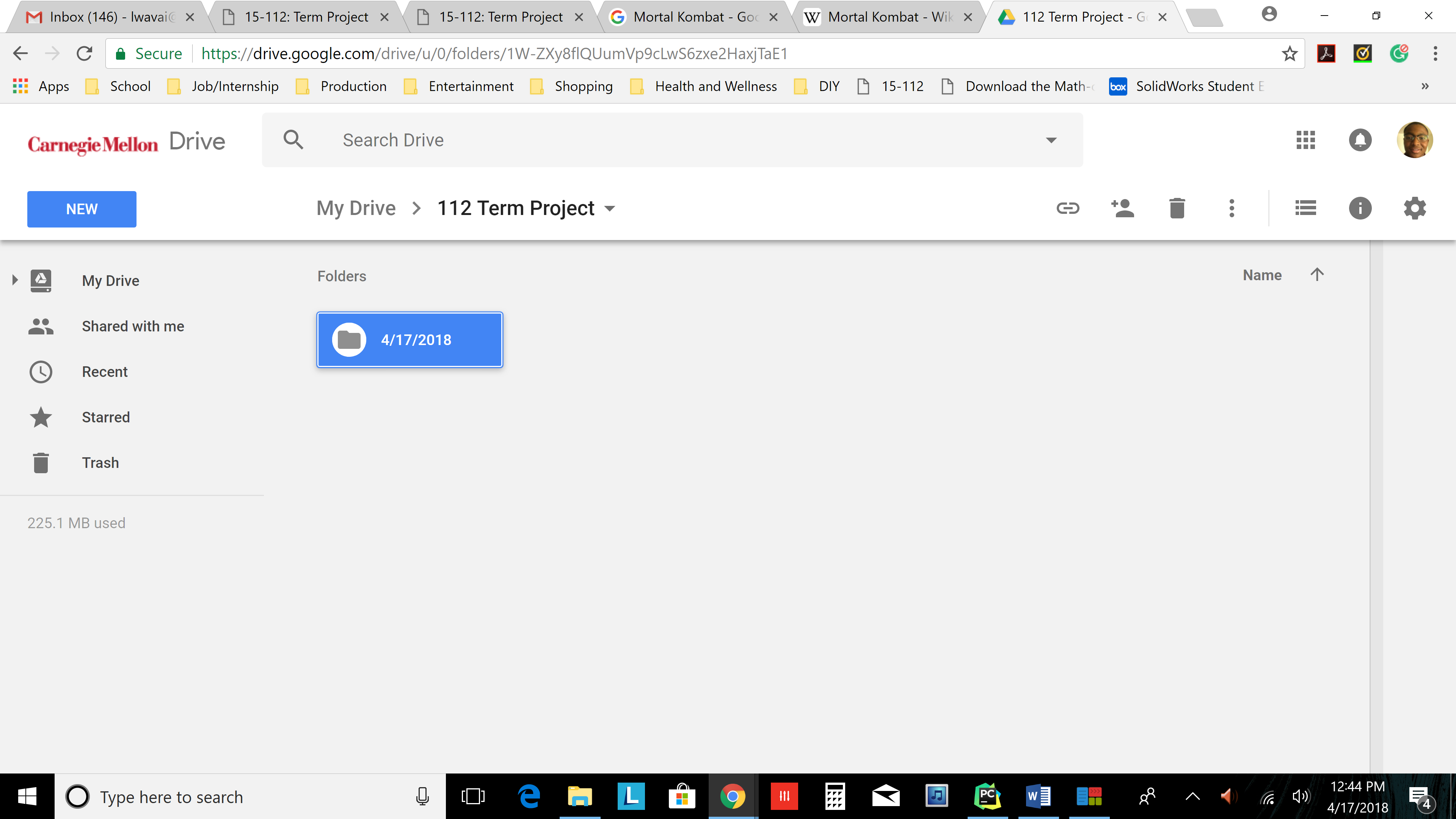
I think for my project there will be 2 main portions that could potentially cause me the most sleepless nights which includes the instances of levels and character selection. I think that the actual gameplay may be manageable, but customizing it by the selected characters, and the levels of story mode will prove difficult. I plan to approach this issue with the structural planning that I described above of instituting a class and inheritance system throughout the game. That way for different levels, I might initialize the different classes of levels, and for characters I will initialize different characters and their respective traits.

**Timeline Plan:**

1. By TP1:
   1. I want to have made progress in the actual fighting mode of the game, being able to use a character with keypress buttons, and implement different moves
2. By TP2:
   1. I want to have completed the fighting stage of the game, an implement switching between screens, character selections, and the levels of story mode
3. By TP3:
   1. I want to have implemented a respectable AI that can respond to different instances of difficulties chosen for single player and story mode
   2. I also want to add the feature where someone can make there own custom character with open-cv

**Version Control:**

I will use google drive to back up my code. I made an overall directory for the term project in which I will add other directories for each day that I work on the project. In the daily folder’s I will upload all of the files that I made changes to, to maintain a stable and consistent backup for my project.



(June 22, 2019 update) : I will use git and github for version control of this program under user lbw798.

**Module List:**

The main module that I will be using up until the MVP phase of my project is Pygame which I was approved for in the tech demo. I will then implement other features such as music with Pyaudio or stick to Pygame’s own sound mixer Pygame.mixer, and customization OpenCV.

**TP 2 Update:**

I added background music using Pygame.mixer, and an ogg music file, along with up to 4 characters, and my planned additions as described above.

**TP 3 Update:**

1. I now have 6 available characters for the user to experience
   1. I also went back and made sure that each sprite was almost pixel perfect, by using a photo editor
      1. I adjusted off-color pixels that decreased the quality of any image
      2. I cropped images to exact sizes to limit misfires in collision detection
   2. I added sound for each characters actions varying depending on the character
2. I added text throughout the game on various screens to improve the user experience
   1. In order to add text I created a class just for text in which I have functions for creating specified fonts, rendering text, and drawing text on to screens via pygame
3. I improved the user interface look and feel
   1. I made design and functionality improvements to the character select screen
      1. I added a board within which both players can move about to select their characters, including a wrap around
      2. I added instructions onto the screen, and character label names
   2. I added a background select screen
      1. The background screen is similar to the character screen in functionality and design i.e. board of choices with wrap around
      2. I added up to 14 different backgrounds that the user can experience with some fun names and a transitioning background surprise
   3. I added a main menu with a play, options, and credits tab available via mouse press
      1. In options the in game instructions are available and music can be turned on and off via mouse pressed
   4. I updated the look of the health and energy bars
      1. I added an icon to know which character is associated with which health and energy
      2. I added labels to the bars
   5. I updated the heath bars functionality
      1. Instead of a players’ health depleting to zero, and game over, I added three stages of health bars
         1. The health bars begin with a green color, and changes once depleted signifying a shift in the stage. The color changes from green to red, and the player has a full bar at the beginning of each stage
         2. Once the final red stage health bar is depleted, the player loses, thus elongating the game and improve overall user experience

**Work Cited:**

* See the document labeled **Term Project Work Cited** in the **Design Proposal** directory