**Stress Learning Games**

**Iteration 1**

* Team members and roles
  + Hanyang Li (Product Owner)
  + Ping Lu (Scrum Master)
  + Xin Hu (Programmer)
  + Zhiyu Yan (Programmer)
  + Chaoyang Zhu (Game Testing)
  + Yipeng Lu (Game Design and Game Testing)
  + Mufeng Xie (Report Editor)
* Project Results so far:

We have established the logical process of the game design, the functions of every feature and initialized our webpage. The detailed programming codes can be found here:

<https://github.com/pinglu0212/2020-Spring-CSCE-606>

* The Logical Process of the Game

1. Word Guessing with Category and Length Given:

The game randomly chooses a “category” of the words, and randomly chooses a word from the “category”. The gamer is supposed to guess a word from the hinted “category” of the word and the “length” of the word.

2. Word Guessing with the English Alphabet Table Given:

The Gamer can guess an alphabet that could potentially show up on the word of interest and click on the alphabet in the alphabet table. If the clicked alphabet is in the word of interest, then the alphabet would show up in the exact position of the word. If the clicked alphabet is not in the word of interest, then some “life points” would be deducted from the health or vitality attribute of the gamer. The gamer is initially granted with 10 life points as the health or vitality attribute. If the vitality of the gamer becomes 0 life point, then the game is over and the gamer loses. Each alphabet in the alphabet table can only be clicked once and never again until the game is over.

3. When the gamer guesses the entire word before the vitality value becomes 0 life point, then the gamer wins the game; otherwise, if the gamer loses all of the life points before the gamer can guess the entire word, the game is over and the gamer loses.

4. Every time the gamer plays, the word category is randomly chosen without prejudice.

* JavaScript Programming Logic

1. Website Loading and Rendering

a. Randomly chooses a word category and randomly chooses a word from the category.

b. Render the alphabet table.

c. Render the word of interest as “\_” for each alphabet. If there are white spaces between words, then use “-” between words.

d. Render the vitality of the gamer with 10 life points to begin with.

e. Render the category of the word of interests as a hint.

f. Initialize the attributes in the hangman canvas paint stroke parameters.

2. The Logic of the Hint Button

Use the randomly chosen category and word as the index, find the corresponding hint words, and render the hints on the page at the appropriate label position.

3. The Logic of the “Play Again” Button

a. Delete all the labels that JavaScript rendered in the previous game.

b. Delete all the canvas paint strokes that JavaScript created in the previous game.

c. Reload and render the page all over again.

4. The Logic of the “Alphabet Table Button”

We have created the process graph below.

