Game Logic – Minimum Viable Product

**Action Gauge:**

* Make a counter variable
* Use setInterval to increase the height frame by frame (use a very small interval e.g. 10), increasing the counter at every frame: [**https://www.w3schools.com/howto/howto\_js\_progressbar.asp**](https://www.w3schools.com/howto/howto_js_progressbar.asp)
* Make height a percentage and have height correspond to the counter value
* Subtract 100 from the counter after attack/defend is executed

**2nd Action Gauge:**

* Connect the 2nd Gauge to values 101-200 of the counter (e.g. if counter <= 100, guage1.height ++. If counter > 100, gauge2.height ++)

**Hp Bar:**

* Similar logic to the action gauge but no need for setInterval.
* Have width be a percentage and correspond to the hp values of player/enemy
* Width = (Current hp/total hp) \* 100%

**Player & Enemy Class:**

* Assign hp variables
* Assign str variables
* Assign armour variables
* Assign Action Gauge counter
* Enemy Attack function
* Check if player defend variable === true
* Do damage calculation (factoring in armour values and defend state)
* Execute Text Log function, with parameters of (“enemyAttack”, dmg value, defend true/false)
* Set player defend variable = false
* Player Attack function
* Set player defend variable = false
* Do damage calculation
* Execute Text Log function, with parameter of (“playerAttack”, dmg value)
* Player Defend function
* Set defend variable to true. Show defend animation.

**Enemy Attack:**

* If Enemy Action Gauge === 100, execute attack function within the Enemy class

**Player Attack:**

* Check first if Attack Button is pressed && Action Gauge >= 100, then execute attack function within the player class

**Player Defend:**

* Check first if Defend Button is pressed && Action Gauge >= 100, then execute defend function within the player class

**Text Log:**

* HTML of many “div”. Each “div” is one line.
* If first parameter === playerAttack, change text1.innerText = “You dealt x damage to the enemy”
* If first parameter === enemyAttack, change text1.innerText = “The enemy dealt x damage to you”
* If defend = true, change text1.innerText = “You defended! The enemy dealt x damage to you. ”

**Attack/Defend Button:**

* Toggle between selected/unselected states(reflected visually) whenever either button is selected and also, after either action is executed

Game Logic – Stretch Goals

**Parry:**

* Add on to Enemy attack function:
* If player defend === true, execute parry function
* Parry function:
* Pass a Boolean into both player and enemy Action Gauge functions to set parry === true
  + Inside the player and enemy Action Gauge functions, add the parameter and stop the counter for setInterval of 1000 (1 second) when parry === true.
* Make a bar appear that ticks down within 1 second
* Make 3 random keys appear in the middle of the screen (letters and numbers)
  + Check for player input and toggle the colour/appearance of the keys on screen when any are inputted correctly
  + Hide the keys when 1 second is over or when the player is successful
* When successful, play parry animation, do damage calc and text log. If failed, do damage calc and text log.