# Peer-2-Pocketmonster Design Document

Riley Abrahamson, Alexandra Burress, Lucia Ho

This project is publicly available to play at: <https://peer2pocketmonsters.web.app/>

## Overview

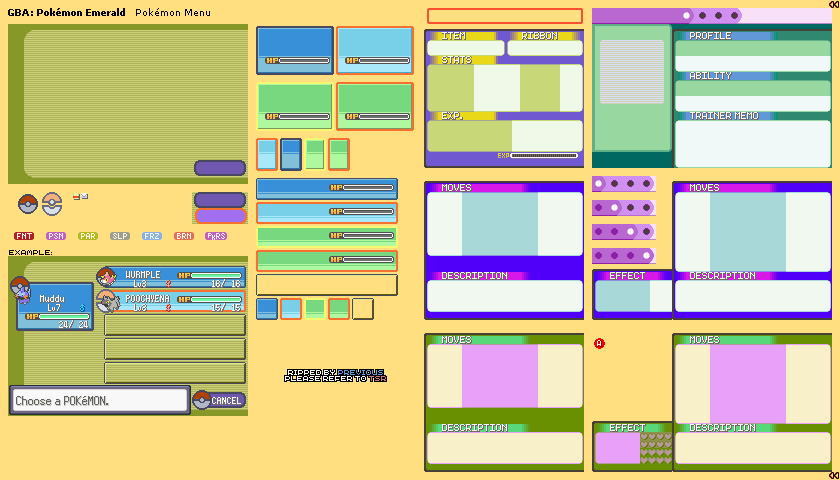
Peer-2-Pocketmonster is a browser based, peer-to-peer monster battle simulator. On the website, players can select characters and connect to one another directly to play. Players then take turns selecting moves in an attempt to bring their opponent’s health to 0 HP. There are a variety of monsters to choose from, each with their own set of moves and specializations. While battling, players can also use a chat feature to send messages back and forth. Peer-2-Pocketmonster is built on a stack of vanilla HTML/SCSS/JS. Because of this, Peer-2-Pocketmonster does not need a backend server (see section “Peer-2-Peer Connecting” for more info). Peer-2-Pocketmonster is fully responsive and works on all major mobile and tablet platforms.

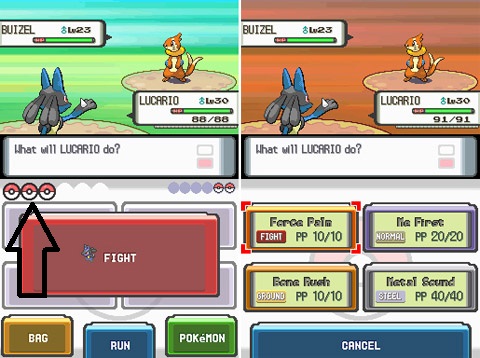
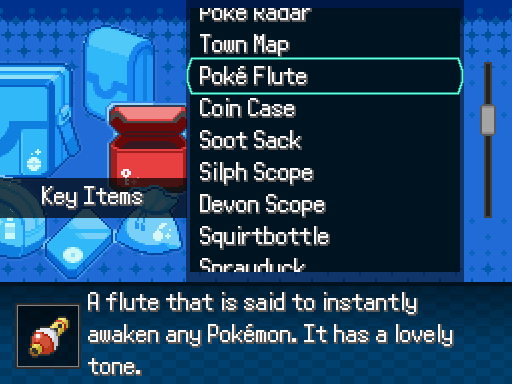
## Front-end Design

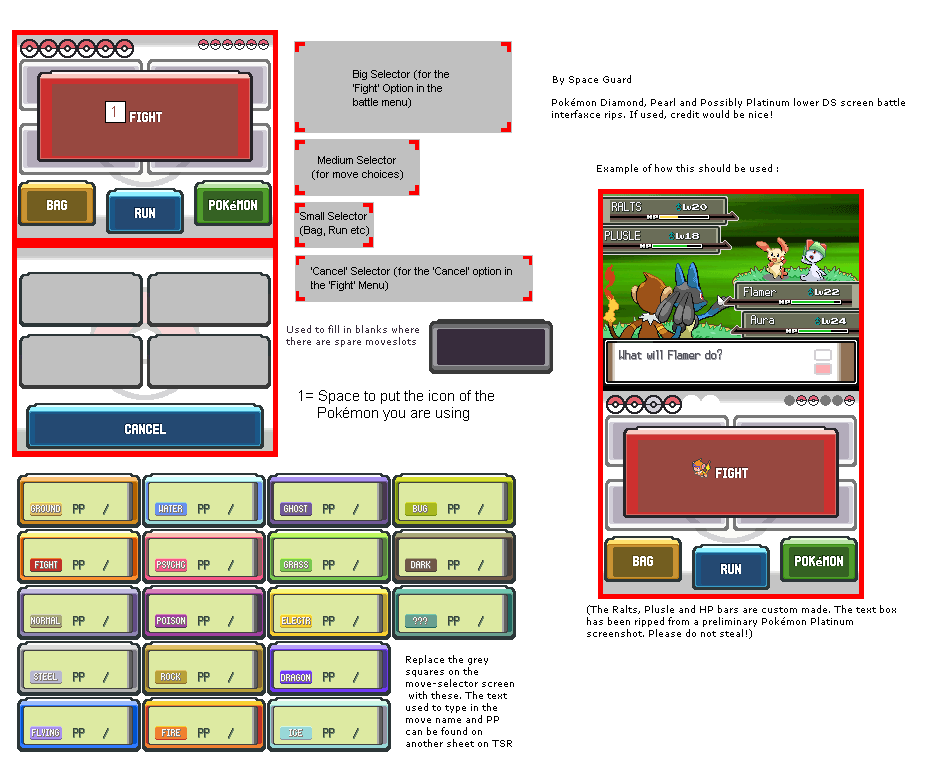
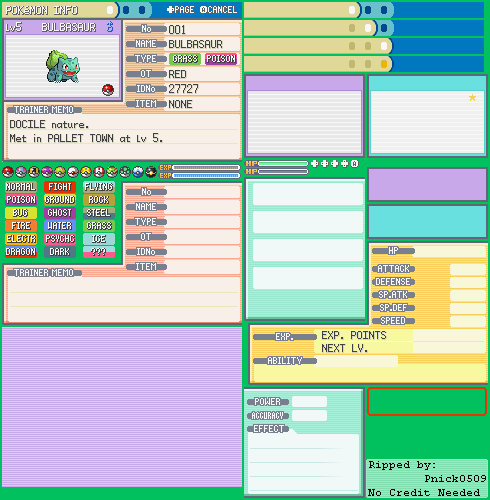
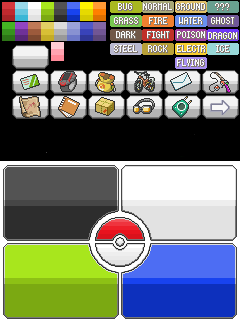
Much of the visual design for this project draws from the Pokémon series of video games. The general battle schema and layout, as well as specific visual elements are loosely based off of the UI from the mainline series titles. The most notable place for inspiration is the 5th generation Pokémon games: Pokémon Black & White/ Black 2 & White 2. These games’ black and blue cyber-tech theme served as a base for the flair applied to the UI elements of Peer-2-Pocketmonster.

Listed below is a collage of images used as inspiration.

### Inspiration







### Health Bar

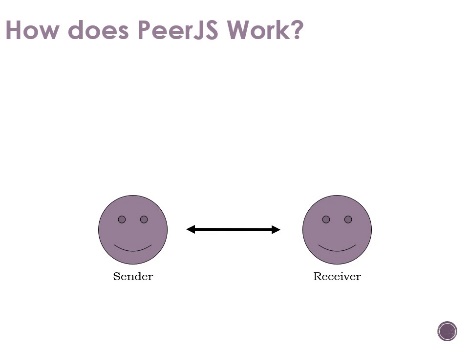
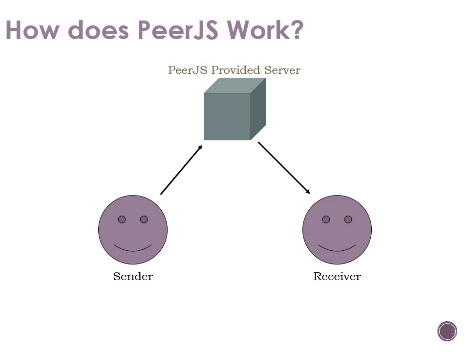
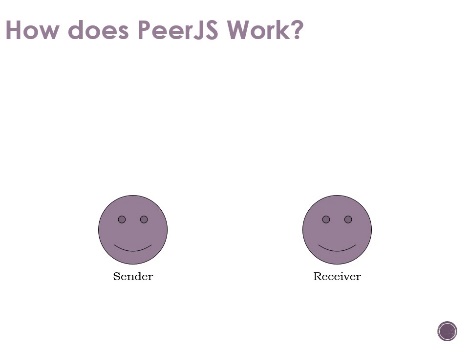
For the health bar, we used a combination of JavaScript and CSS to make it responsive to what was happening in the battle. Essentially, the JavaScript portion of the code calculated what percentage of the max health each player’s current health was at. Then, this code adjusted the CSS attribute correlating to the width of the green bar in the Health Bar and made the width percentage of this bar equal to the current health percentage. We got the inspiration and direction for this from <https://codepen.io/AndrewMcDowe/pen/akWBKj>, a CodePen example of a simple health bar.

## Peer-2-Peer Connecting

PeerJS: <https://github.com/peers/peerjs/>

PeerJS is a JavaScript library that provides an easy to use framework for making direct peer-to-peer connections between browsers. PeerJS is the backbone of this project and is the piece that allows the players to battle asynchronously back and forth.

PeerJS creates a peer connection by first connecting both web browsers to a public PeerServer provided by the PeerJS staff.  Within the documentation there is the option to host your own server for this purpose, but the PeerServer provided worked perfectly for our methods. After a connection is formed by both peers to the server, they are paired up based on their pre-generated IDs. If the PeerServer finds a match, then the two peers’ browsers are connected directly and the connection to the PeerServer is dropped.

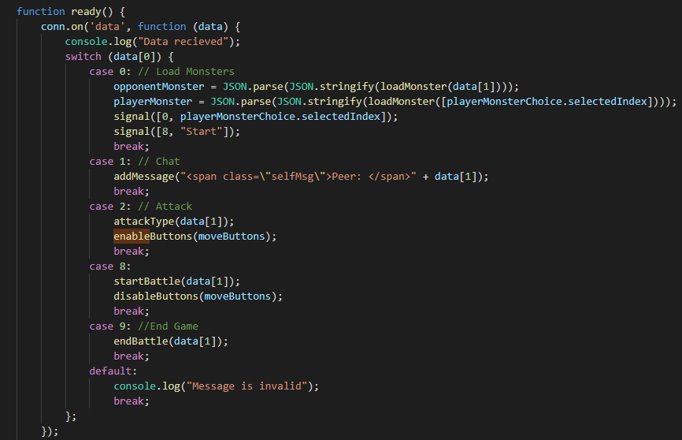


From here, the two peers can send packets of asynchronous data back and forth by using the *Conn.send()* function. This function allows any type of data to be sent back and forth between peers. When a peer receives an incoming message, it is processed in a *Conn.on(‘data’, function(data) { }* block. This block acts almost like a loop on a local machine, as it is constantly listening for new data to come in and processes it accordingly.

Because the *Conn.send()* function’s usage is so open to interpretation, we decided to create a message encoding paradigm for sending data back and forth. Each message would be encoded as a two-part list: *[code, value].* In this list, the *code* represents what type of message is being sent. Once the type of message is parsed, the *value* is used to execute whatever request was being sent. Our table of message types is listed below.

## 

Additionally, here is an example of how messages are parsed in our main connection code:



## Chat-bar

The chat-bar in Peer-2-Pocketmonster allows players to communicate to each other, it also shows the message about what move the monster uses, as well as a win/lose message when one player’s monster faints. We implemented an addMessage(msg) function for this feature. This function is called to show messages in the chat-bar, as well as to send it to other players. The function will generate the current time (hours and minute) then print out the time with message in message box. The message box has a scroll bar when there are too many messages.

There is a text box at the bottom of the page that allows each player to type a message they want to send to another player. Players can either press enter or click the “Send” button to send the message. After player pick a move to attack, the message box will also show the messages about which monster used what move, such as “Vulpix used Tackle!”. At the end of the game, when one player’s monster faint, the winner will receive a message “Congratulations, you win!” and the loser will receive a message “Game over, you lost!” in their message box to let both players know game over.

## Battle System

The battle system in Peer-2-Pocketmonster works similarly to many conventional RPG games, especially conventional monster battlers. There are currently four monsters to choose from when starting a match. Each monster possesses four moves to choose from. These moves are used to attack the opponent monster and bring its health to 0.

The players attack in turns. The player who starts the connection gets to attack first. After they attack, their moves are disabled, and the opponent gets a chance to react with a move of their own. Play continues in this manner until one of the players is eliminated.

### Monster Moves

Monsters have four (4) moves available to choose from. Moves can be of four (4) types: attack, status, attack/status and item.

* Attack (value 0)
  + These moves aim to do damage to the opponent. They can come in a wide range of styles, but they always do damage.
* Stat (value 1)
  + These moves add stat modifications to the target monster. Some stat changes are self-targeted (a buff) while others affect the opponent (a de-buff). Stat moves do not carry any damage with their action.
* Attack/Stat (value 2)
  + These moves both attack and can cause stat modifications to the opponent. While powerful, they often come with conditions to limit their effectiveness. For example, many attack/stat moves will have lower accuracy, damage, or the extra effect may only have a chance of triggering.
* Attack/Status (value 3)
  + These moves are similar to attack/stat, but instead cause status effects to the opponent. These can cause external afflictions to the monsters. While powerful, they often come with conditions to limit their effectiveness. For example, many attack/status moves will have lower accuracy, damage, or the extra effect may only have a chance of triggering.
* Item (value 4)
  + Items are single use moves with powerful effects. They effect the battle in ways that are stronger than regular moves but come with the cost of limited uses. Items can be thought of as band-aids to keep you battling. A common item for monsters is a Potion, which is a move that restores 50% of the monsters HP, but can only be used once.

### Stats

* HP (Health Points)
  + These represent how much damage a monster can take before it is defeated. If it is reduced to 0, the creature faints.
* AT (Attack)
  + This stat affects the power of attacking moves the monster uses. It cannot be reduced below 10. It has a maximum of 100.
* DF (Defense)
  + This stat reduces incoming damage to the monster. The lower the value is, the more damage is taken from incoming attacks. It cannot be reduced below 10. It has a maximum of 100.
* AC (Accuracy)
  + This stat affects the accuracy of moves used by the monster. By default, it is 100%. A monster’s AC is multiplied by a move’s accuracy to get the hit success chance. For example, if a monster has 100% AC and the move has a 90% accuracy, it will hit 90% of the time. If a monster has 50% AC and the move has 70%, it will hit 35% of the time. Accuracy cannot be reduced below 25%.
* EV (Evasion)
  + This stat determines if a monster is able to “dodge” an attack. This stat affects a moves accuracy after the (Move accuracy \* monster accuracy) calculation is made. For example, if an attacking monster with 100% AC uses a move with 90% accuracy against a monster with 10% EV, they would have an 81% to hit (90 – (90 \* 0.1)). Accuracy starts at 0% for every monster and has a maximum of 75%.
* SP (Speed)
  + Speed determines which monster attacks first in a single turn. If the speed stat is tied between the two monsters, it will be random which monster goes first.

## Monsters

### Fire Type

Visual Design:

Vulpix,

Stats:

HP: 100

AT: 50

DF: 50

AC: 100%

EV: 0%

SP: 5/10

Moves:

1. Tackle

Type: Attack

Base Power: 40

Base Accuracy: 100%

Effects:

None

1. Tail Whip

Type: Status

Base Power: --

Base Accuracy: 100%

Effects:

Lower Opponent Defense by 10

1. Ember

Type: Attack/Status

Base Power: 30

Base Accuracy: 80%

Effects:

20% to *Burn* opponent

1. Potion

Type: Item

Base Power: --

Base Accuracy: --

Effects:

Heal 50% of Max HP. Single Use

### Water Type

Visual Design:

 Poliwhirl

Stats:

HP: 100

AT: 50

DF: 50

AC: 100%

EV: 0%

SP: 5/10

Moves:

1. Pound

Type: Attack

Base Power: 40

Base Accuracy: 100%

Effects:

None

1. Bubble

Type: Attack/Stat

Base Power: 40

Base Accuracy: 70%

Effects:

Lower Opponent Speed by 1

1. Ice Ball

Type: Attack/Status

Base Power: 30

Base Accuracy: 80%

Effects:

15% to *Freeze* opponent

1. Potion

Type: Item

Base Power: --

Base Accuracy: --

Effects:

Heal 50% of Max HP. Single Use

### Electric Type

Visual Design:

Luxio,

Stats:

HP: 100

AT: 50

DF: 45

AC: 100%

EV: 0%

SP: 6/10

Moves:

1. Tackle

Type: Attack

Base Power: 40

Base Accuracy: 100%

Effects:

None

1. Growl

Type: Status

Base Power: --

Base Accuracy: 100%

Effects:

Lower Opponent Attack by 10

1. Spark

Type: Attack/Status

Base Power: 30

Base Accuracy: 90%

Effects:

40% to *Paralyze* opponent

1. Potion

Type: Item

Base Power: --

Base Accuracy: --

Effects:

Heal 50% of Max HP. Single Use

### Grass Type

Visual Design:

Cherrim,

Stats:

HP: 100

AT: 40

DF: 60

AC: 100%

EV: 0%

SP: 4/10

Moves:

1. Tackle

Type: Attack

Base Power: 40

Base Accuracy: 100%

Effects:

None

1. Worry Seed

Type: Status

Base Power: --

Base Accuracy: 100%

Effects:

Lower Opponent Accuracy by 10

1. Solar Beam

Type: Attack

Base Power: 65

Base Accuracy: 50%

Effects:

None

1. Sub-Potions

Type: Item

Base Power: --

Base Accuracy: --

Effects:

Heal 30% of Max HP. Three Uses

## Status effects

* *Burn*
  + Deals 10% of the target’s health as damage each turn
* *Freeze*
  + Prevents the target from acting for 1 turn
* Paralyze
  + Lowers the opponents speed to 0