**Pokemon Battle Simulator:**

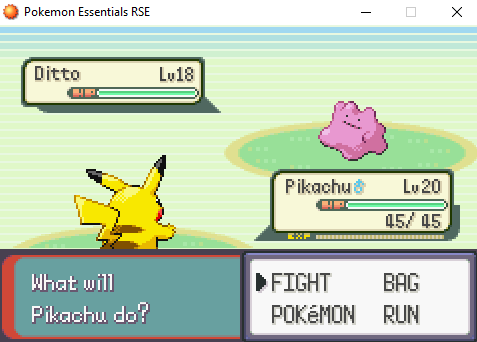
two Pokemon fight each until one of them loses all of their HP. Pokemon use different moves to cause damage, apply status effects, heal, or inflict buffs/debuffs. The goal of this simulator is to emulate these battles for the user so they can be more effective in the actual game.

**Generations:** The simulator will use the most up to date generation of Pokemon to pull its values from. It is possible to emulate all generations. However, the art style and font will come from the Gameboy Advanced generations of the game (Fire red, Leaf Green, Ruby, Sapphire, Emerald).

**Technology:** This app will be built using Python 3.11. Pygame will run the GUI and game style engine. PokeAPI will be used to gather the Pokemon data.

**Pokemon battle example (Gameboy advanced style):**

(Figure 1.1)

(Figure 1.2)



(Figure 1.3)

**App opening:** The app should open with a splash screen emulating game freak’s but possibly having my own branded name instead. Then the app will open with Gengar and Nidorino preset as the Pokemon fighting each other (Figure 2.1) Also there is a chance one or both could be shiny.

(Figure 2.1)

**Types:** The hardest part about a Pokemon battle for non-veteran players is the type system. There are 18 possible types in the game. Pokemon can have 1-2 types (different kinds). The types work on a Rock Paper Scissors style system. This is complicated by 18 different types and a possibility of any two type combinations. The app will allow the user to simulate a battle to figure out what types it should use against another Pokemon, as well as the types that their Pokemon are most vulnerable to.

**Primary App Function:** The app will emulate a wild Pokemon battle. The User will assume the role of the Pokemon in the bottom left corner. The Pokemon in the top right corner will be called the *wild Pokemon*. From the bottom context menu, the user can manipulate the battle. The layout will look like a traditional Pokemon battle where the options are *FIGHT*, *BAG*, *POKeMON*, and *RUN* (like figure 1.2). However, this app will have *FIGHT*, *DEFEND*, *CHANGE*, and *SET*.

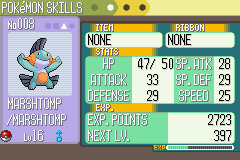
**FIGHT:** The FIGHT menu will bring up a new context menu for choosing moves (Figure 1.3). Initially, this will be populated with the names of each move type. Unlike the real game, the user can scroll down to see each type. Once the user clicks one, it will *attack* the wild Pokemon with this type. Attacking will result in the context menu informing the user about the effectiveness of the attack (Figure 3.1 Below) and the wild Pokemon will lose health accordingly (none if it is immune). The user then clicks any key (or specified button/mouse click) and it will return to the FIGHT menu. There will be a back button (or dedicated key press) to go back to the main menu. If time allows, a list of the possible moves the Pokemon and their types would be placed here instead and realistic damage from the game would be applied.

(Figure 3.1)

**DEFEND:** The DEFEND menu is almost the same as the FIGHT menu. However, it will give the user access to the wild pokemon instead. Initially the user can choose from one of the 18 types and see how the attack would effect the user’s Pokemon. If time allows, it would also be updated to include an accurate move set and realistic damage values from the game.

**CHANGE:** The CHANGE menu will allow the user to change the Pokemon in either position. The user will be able to type the name of the Pokemon to change them. (At this time, having a scrolling list of all the Pokemon seems cumbersome). If the name is typed wrong, it’ll just prompt the user to try again. Case shouldn’t matter.

**SET:**  If time permits,The SET menu will be implemented. It will allow the player to set and view the details of each pokemon. This will likely bring up an extra screen similar to the stats screen in the games (See Figure 4.1 Below). The player will be able to edit either Pokemon’s Level, Stats, Nature, EVs, IVs, and Held Item. This feature is the absolute last priority. Until then, this will be a QUIT menu for the User.



(Figure 4.1)

**Additional Notes:** Pokemon that are spawned onto the screen should have some chance to be their shiny version (probably more than the game’s chances so it happens more often. Could allow the SET option to let the user do this or keep it special, either way). It also wouldn’t be hard to implement them as male or female randomly and show those changes.

**Possible Features:** If time permits, some other features would be nice.

1. Could have animations when Pokemon appear, attack and get damaged like the game.
2. Could have a catching simulator that gives the chance a catch could happen based on status effects, health, level, ect.
3. Could change generations to simulate those games specifically as well.
4. Game accurate sounds to menus, fights, Pokemon cries ect. (some sounds may be implemented, but game accurate sounds in every way is also a last priority).