**Pokemon Project Documentation**

***Net-Centric Programming***

**By Đinh Bảo Duy and Hà Tiến Đạt**

**Instructor: Le Thanh Son**

**May, 2022**

**Table of contents**

1. **Introduction**

The Net-Centric programming course term project: To develop a game inspired by the Pokemon franchise that allows players to battle and capture Pokemon using Socket Programming.

The game is made from 3 major features, with their short descriptions:

* Pokedex: Retrieves and stores Pokemon information.
* Pokebat: 2 players battling using their Pokemon online through Sockets.
* Pokecat: Open Pokemon world that allows many player connections and serves at the place to catch Pokemons.

Together they form a Pokemon game in which you can see the information of all available Pokemons, allows you to catch and battle other players online.

Creators:

Đinh Bảo Duy – ITITIU19107

Hà Tiến Đạt – ITITIU19095

1. **Project Documentation:**
   1. Technology:

Programming language: Python.

Python packages:

* JSON package: to convert python dictionaries to JSON strings and vice-versa to send through sockets.
* Sockets package: this project utilizes a TCP connection-based server for online connection.
* Requests package: to send HTTP requests to the online API for the Pokedex feature.
* Thread package: project also uses multithreading to handle multiple connections to the server.
  1. Features and Functions:
     1. Pokedex:

An online, on-demand Pokedex is implemented using the REST API provided by the website: <https://pokeapi.co/>. The site contains the information of all Pokemon. This is a better implementation of the Pokedex as a file because there are more Pokemon available in the API instead of saving a local file. This reduces the game file size and also the ability to query Pokemon dynamically. However, this will be slower to fetch Pokemon because a HTTP GET request will be dispatched so speed will be dependent on your connection.

Pokedex detailed feature:

* Have information about all Pokemon such as: Names, Stats, Types, ect… Also other information such as EXP to level up, Evolutions, Type weaknesses are also available.
* Everything is stored in JSON format for easy parsing and storing. Player teams and information is stored in an external file for saving progress between game sessions.
* Pokemon information includes features such as levelling up (stats increases accordingly), evolutions, EVs, …
  + 1. Pokebat:

Currently 2 player battles are supported. Each team has 3 Pokemons for battle. The 2 players has clients that connect to the server where the battle is computed. The battle calculations take place server-side and the inputs and displays take place in the client-side. The communication method between server and client is Polling, the client continuously sends packets to server and the server responds with the appropriate packets (more detail in the sequence diagram).

Pokebat detailed feature:

* Two players connect to a server and the battle commences. The players can pick their actions and send it to the server, then the server calculates their actions and sends the response.
* Players have 3 actions to take per turn: Attack, Switch Pokemon and surrender.
* On attacks the server will calculate attack damage and order through Pokemon stats (Pokemon typing multiplier taken into account). If Pokemon loses the player will have to switch Pokemon. Battle ends if there is surrender or player has no more available Pokemon.
* At the end of the battle the winner will receive EXP rewards accordingly.
  + 1. Pokecat:

//TODO//

1. **Implementation:**
   1. Important Classes and Pre-requisites:

* *Pokemon Object:*

The Pokemon class is represent a Pokemon in the game. It stores the Pokemon’s information like HP, EXP, EVs, types, attributes, type weaknesses,…

* *PokeTeam Object*:

The PokeTeam is essentially a list of Pokemons with helpful methods such as *printAllPokemon(), getPokemonFromID(), selectPokemonForBattle(),*… This class organizes and stores each players team and battling pokemons.

* *Battler Object:*

The Battler Object is held by the server, it simulates and conducts battles between Pokemon Teams.

* *Client – Server Polling*:

The client class: Game and the server class: Server communicates using JSON packets. The client continuously sends polling packets to the server and the server replies with responses with information such as waiting for players, waiting for actions, current game state,…

* *Packet:*

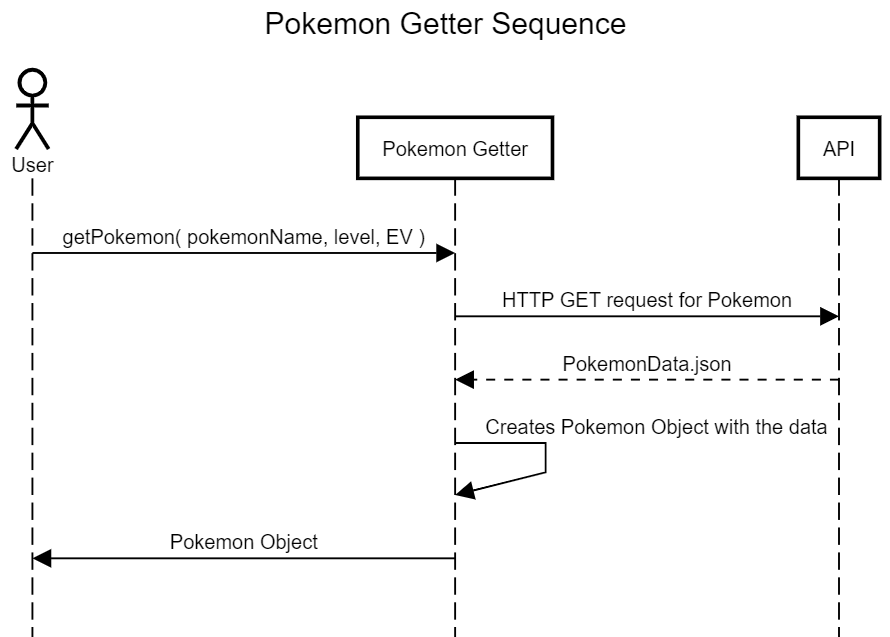
The packet class is the way that client and server communicate. It is a simple dictionary so it is easy to serialize to json and send through sockets.

* 1. Pokedex:

The main module for the implementation of Pokedex is APIRequest.py.

The module contains the PokemonGetter class which servers as a Pokemon factory. Through the *getPokemon(str)* method, you pass the name of the Pokemon with some optional parameters such as level and EV. The function sends a request to the API and parses the JSON information receive and returns a Pokemon object with the Pokemon information.

Sequence Diagram for *PokemonGetter.getPokemon():*



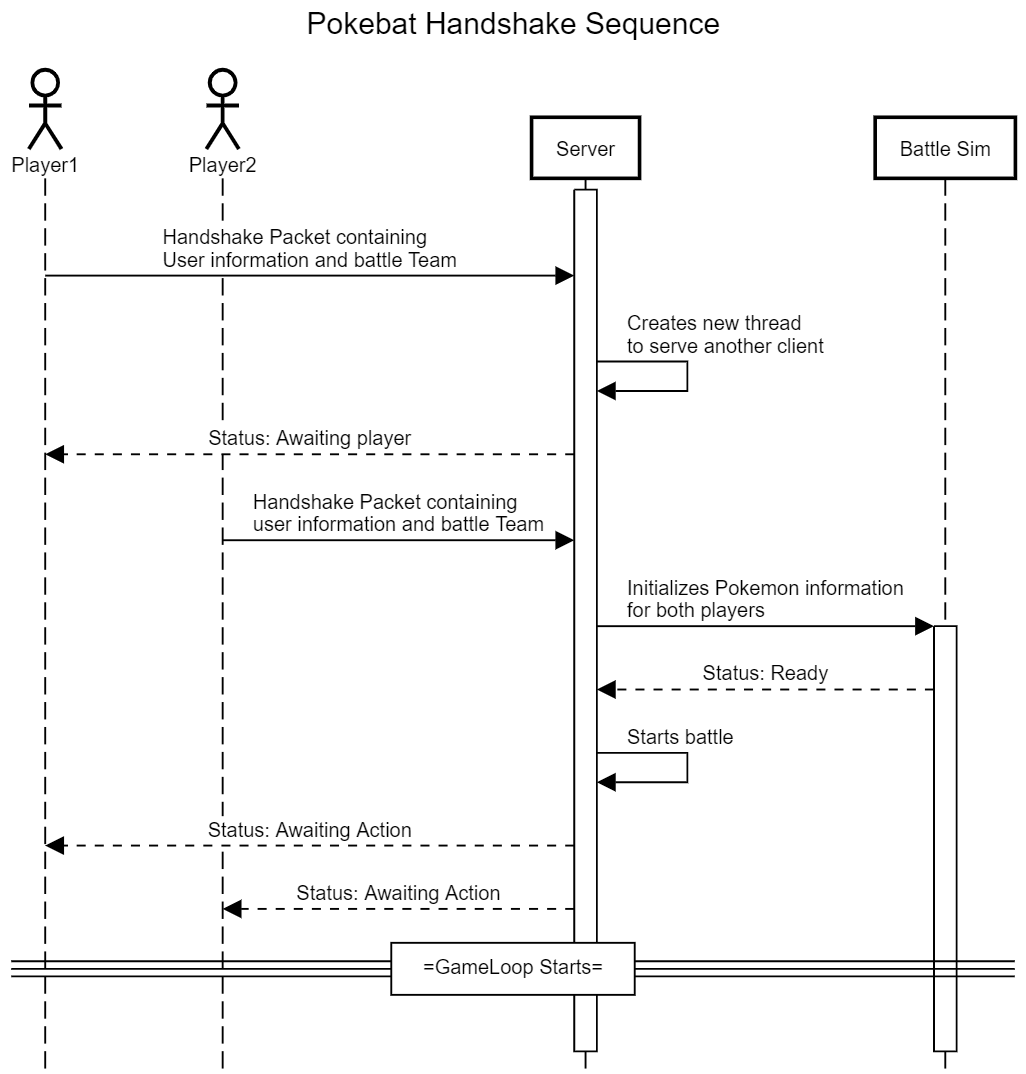
* 1. Pokebat:

There are 2 main modules in the Pokebat feature: pokeGame.py is the client-side and server.py is the server-side for the Pokebat game.

* Handshake:

A game is initiated when there are 2 players having 3 Pokemons in their battle list. Players can either choose to load the player profile through the player.json file or create a new player, creating a new player will result in a team without Pokemons so they would need to catch and assign their battle list before joining Pokebat.

There are 2 main processes in the Pokebat. The initial handshake process to start the battle and the main battle loop:

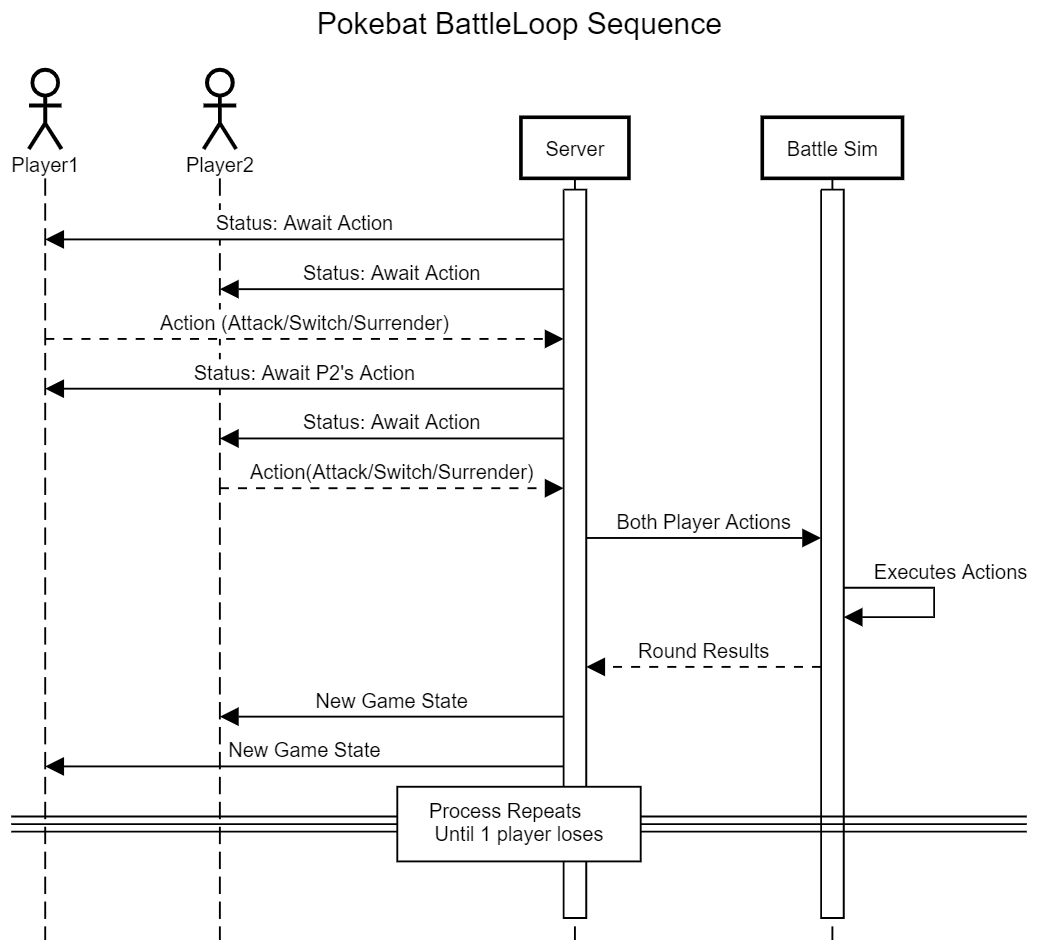


* Pokebat battle loop:

Players can choose between 3 actions: Attack, Switch or Surrender. Every round the server executes the action of both players. Here is the list of the interactions between actions:

* Attack – Attack: The player with the current battling Pokemon whose speed is faster will conduct the attack first. Attack is selected randomly between Special and Physical attack. Type effectiveness multiplier will take place if it is a Special Attack. If the slower Pokemon survives the Attack, it will attack back, else the player has to switch.
* Attack – Switch: Switch always have priority so switching will occur first. The new switched Pokemon will be attacked without attacking back because the action is used on Switching.
* Attack – Surrender or Switch – Surrender: the Surrender player will lose and the other wins and gets the reward EXP.
* Switch – Switch: both player switches current Pokemon.
* Surrender – Surrender: both side loses, no rewards given.

Sequence diagram of main battle loop:



* 1. Pokecat:

//TODO//

1. **JSON FORMATS:**
   1. Pokemon:

Pokemon is saved in JSON format with information such as type, attributes, weaknesses:

{

“name”: “Pikachu”,

“type”: [“electric”],

“damage\_when\_attacked”: {

“ground”: 2,

Etc,..

}

“base\_hp”: …,

“base\_speed”: …,

Etc,…

}

* 1. player.json:

The current player and their PokeTeam is saved locally in the player.json file so progress is saved between game sessions.

{

“team”: [ List of Pokemon in their Team],

“name”: “Duy”,

“battle\_list”: [ List of IDs of battle Pokemon ],

}

* 1. Packet.json:

The way that client and server communicates, sockets sends strings and at the end it is converted to dictionaries using the Packet class structure.

{

“type”: “type of packet”,

“info”: “packet info”,

“payload”: “packet payload”

}

With only 3 fields the packet system carries all the information and communication between server and client.

1. **Host and Run:**
   1. Hosting:

The server can be hosted with any computer in the same Local Area Connection as the players, aka the same Wifi or router system.

Go to cmd and type the command:

ipconfig

The command prompt will show your current IP configurations, navigate to your current Internet Connection and look for the “IPv4 Address” field. Copy your IP.

Then navigate to the CONSTANTS.py file and paste the host IP as a string in the SERVER\_IP variable.

Then go into your terminal and run the server.py file:

python server.py

This should be the correct output:



Where the IP part is your IP address.

* 1. Playing:

Make sure your server host is in the same wifi or LAN as you. Copy their IP Address and paste it into the SERVER\_IP variable in the CONSTANTS.py file.

Then go to terminal and run the StartGame.py file:

python StartGame.py

The game should be started for you. Get 1 of your friends to do the same and then you can battle their Pokemon Team through socket programming.