**CIS427-Programming Assignment 1**

**Online Pokémon Cards Store**

Jacob Marchywka, Meher Jabbar, Andrew Leutzinger, Brandon Bailey

**Project Description**

This project implements a client-server, Online Pokemon Cards Store application. The two main programs, client.py and server.py, utilize network sockets in the Internet domain to communicate with each other. The client sends requests to the server, and the server sends the appropriate response back.

**Note:** This implementation only allows for one client to be connected at a time.

**Environment**

The code is compatible with the current latest version of Python, 3.12.0. Ensure you are using Python3 on your device. To run the code, an SHH client software will need to be utilized. For Mac and Linux you can run the ssh client in Terminal. For Windows, the Bitvise software may be used.

**Usage**

The client and server can be implemented directly in the command-line. To do this, type the following commands into 2 separate terminals, starting with the server that will start listening for client connections:

python3 server.py

python3 client.py

Then from the client you must enter the server IP address. Once connected you can use the BUY, SELL, LIST, BALANCE, QUIT, and SHUTDOWN commands to communicate with the server. Below are details on each of the commands, including formatting, examples of a success client-server interaction, and functionality:

**BUY**

BUY allows the client to purchase cards existing in the database. Below is an example, where the command format is 'BUY card\_name rarity price number\_of\_cards user\_id':

*client: BUY Pikachu Electric Common 19.99 2 1*

*server: Received: Pikachu Electric Common 19.99 2 1*

*client: 200 OK*

*BOUGHT: New balance: 2 Pikachu. User USD balance $60.02*

**SELL**

SELL allows the client to sell cards that it currently owns. Below is an example where the command format is 'SELL card\_name number\_of\_cards price user\_id':

*client: SELL Pikachu 1 34.99 1*

*server: Received: SELL Pikachu 1 34.99 1*

*client: 200 OK*

*SOLD: New balance: 1 Pikachu. User’s balance USD $95.01*

**LIST**

LIST allows the client to view all the cards it currently owns. Below is an example where the command format is 'LIST user\_id':

*client: LIST 1*

*server: Received: LIST 1*

*client: 200 OK*

*The list of records in the Pokémon cards table for current user, user 1:*

*ID Card Name Type Rarity Count OwnerID*

*1 Pikachu Electric Common 2 1*

*2 Charizard Fire Rare 1 1*

**BALANCE**

BALANCE allows the client to view its current balance. Below is an example where the command format is 'BALANCE user\_id':

*client: BALANCE 3*

*server: Received: BALANCE 3*

*client: 200 OK*

*Balance for user Jane Smith: $10.00*

**QUIT**

QUIT terminates the client, after receiving the '200 OK' response from the server.

*client: QUIT*

*client: 200 OK*

**SHUTDOWN**

SHUTDOWN allows the client to terminate the connection and shut down the server. The server will respond with '200 OK' before it closes all sockets and database connection, and terminates.

*client: SHUTDOWN*

*server: Received: SHUTDOWN*

*client: 200 OK*

**Files Used**

* client.py - implements the client, initiates connection with server and sends requests to it
* server.py - implements the server, opens socket and listens for client connections and processes their request
* command\_handler.py - handles the processing of requests sent to the server
* constant.py - defines constants used in code (server port number, command names)
* database\_manager.py - controls interactions with the SQLite database and creates the database tables
* pokemon.db - SQLite database that stores all the data (cards inventory, user records)
* utilities.py - utility functions for processing the commands, response formatting, and error handling