CP372 ASSIGNMENT 1
WINTER 2020

TINOTENDA NYAKURERWA
SHIMA IRANPARAST

CLIENT- SERVER PROTOCOL RCF

TABLE OF CONTENTS

1. Introduction

1.1 Purpose

This project will allow multiple users on the client side to connect to a server and post, pin, unpin and retrieve notes from a note board on the server side that have certain properties. The client will be able to establish connection using a Graphic User Interface (GUI). The purpose of this project is to help gain an understanding of socket programming in Python how clients and servers interact with each other using stream socket API (TCP) of Python.

1.2 Requirements

The client must be able to store and request notes on server that have certain properties this includes:

1. Setup connection with the server

2. Send diverse types of request messages through the established connection such as:

    a) CONNECT: the client connects to the server with the same port number and server name. Provides the server with board size and width as well as colours for notes.

    b) POST: client sends a request to the server to post a note on a predefined board.

    c) GET: client sends a request to the server and a list of notes that are pinned to the board is returned given specified parameters such as coordinates, colour and reference.

    d) PIN/UNPIN: client sends a request to the server with coordinates of a note for it to be either unpinned or pinned to board. Pinned count is either decremented or incremented.

    e) CLEAR: client send request to server and all notes that are not pinned to the board

    f) DISCONNECT: server will break connection with the client

The server must accept multiple requests for connections as well as maintain an appropriate data structure that holds the data received from clients. Board stored by the server exists only between server start and shutdown and does not need to be stored.

1.3 Overall Operation

This application will serve as a request/response mechanism for use of a virtual board to allow user to make certain changes to a data structure holding board information on the server until the client makes the choice to end the connection with the server.

2. Responsibilities

2.1 Server Responsibilities

The server is responsible for accepting client connections and being able to connect to multiple clients simultaneously. The server must be able to read messages from multiple clients as well as process multiple requests from clients using the protocol. The data provided by the client is to be stored by the server in a temporary data structure whose contents are only valid if the session between the server and the client is valid and not broken. The server should be able to respond to the client with a message if the request is completed otherwise an error will be thrown.

2.2 Client Responsibilities

The client is responsible for requesting a connection with the server, transform and forward user input according to the protocol. The client will read and convert user input in accordance to the protocol and forward the data to the server.

3. Format of Data

3.1 Format of Notes Object

Each note is treated as an object, and each note is stored in an array. If a note is pinned, it will also be added to a separate array to keep track of all pinned notes. Each note keeps track of all necessary attributes, but also tracks if a note is pinned or not so it is easier to distinguish a pinned note.

3.2 Client Request

The client requests will be sent as strings separated by a single space (“ “) character. The client is able to send multiple requests to the server such as CONNECT, DISCONNECT, GET, PIN, UNPIN, CLEAR and POST as described above. A connect request would be as follows: 4320 200 200 red green yellow

3.3 Server Responses

The response from the server after a request from ye client will let the server know if the request was completed or if an error occurred. For CONNECT, DISCONNECT, GET, PIN, UNPIN, CLEAR and POST, a message will be sent to the client indicating that the connection was successful, a list of pinned notes with the described properties would be returned for the GET request, and a message will be returned indicating that the post has been pinned or unpinned.

4. Synchronization Policies

The server will be multithreaded and will handle client requests based on a first come first serve basis so that there are no conflicts between different client requests.

5. Protocol Parameters

The parameters that are taken by the protocol are note, coord, params. The parameters are described as follows:

- <note>: similar to post-it note (coloured rectangular piece of paper containing a string)
- <coord>: pair of integers, describing x and y coordinates of a point on the board.
- <params>: field in GET request that takes two values, pin, colour or contains in order to retrieve posts

6. Data Structure

The data structure used to store all the note data will be an array list. This is because it is easier to implement given the requirement of the protocol. The array will be filled with note lists until the connection between the client and server ends.

7. Errors

7.1 Client Errors

7.2 Server Errors