# Log Viewer Application Manual

## Table of Contents

1. [Introduction](#introduction)

2. [System Requirements](#system-requirements)

3. [Installation](#installation)

4. [Usage](#usage)

- [Starting the Application](#starting-the-application)

- [Adding a New Log Tab](#adding-a-new-log-tab)

- [Managing Connections](#managing-connections)

5. [Code Structure](#code-structure)

- [LogHandler Class](#loghandler-class)

- [LogViewer Class](#logviewer-class)

6. [Troubleshooting](#troubleshooting)

7. [Future Enhancements](#future-enhancements)

8. [Contributing](#contributing)

9. [License](#license)

## Introduction

The Log Viewer Application is a tool designed to connect to multiple wall controllers or octrollers via Telnet, allowing users to view and manage logs in real-time. The application provides a graphical interface for adding and managing multiple log streams simultaneously.

## System Requirements

- Python 3.7 or higher

- PyQt5

- telnetlib3

- asyncio

- threading

- queue

## Installation

1. \*\*Clone the Repository:\*\*

```bash

git clone https://github.com/yourusername/log-viewer.git

cd log-viewer

```

2. \*\*Install Dependencies:\*\*

```bash

pip install -r requirements.txt

```

## Usage

### Starting the Application

To start the application, run the following command in your terminal:

```bash

python log\_viewer.py

```

### Adding a New Log Tab

1. Click the "Add Log Tab" button in the application.

2. Enter the host address of the wall controller or octroller in the prompt.

3. A new tab will be created, and logs from the specified host will start streaming in real-time.

### Managing Connections

- \*\*Disconnecting:\*\* Close the tab to disconnect from the log stream.

- \*\*Reconnecting:\*\* Re-add the tab with the same host information if needed.

## Code Structure

### LogHandler Class

- \*\*Purpose:\*\* Manages the Telnet connection to a log source, reads logs asynchronously, and places them in a queue.

- \*\*Key Methods:\*\*

- `async\_connect()`: Establishes a Telnet connection.

- `read\_loop()`: Continuously reads logs and places them in a queue.

- `connect()`: Starts the connection in a separate thread.

- `disconnect()`: Stops the connection and cleans up resources.

### LogViewer Class

- \*\*Purpose:\*\* Provides the GUI for managing multiple log streams using PyQt5.

- \*\*Key Components:\*\*

- `QTabWidget`: Manages multiple tabs for different log streams.

- `QTextEdit`: Displays logs for each connection.

- `add\_log\_tab()`: Handles the creation of new tabs and connections.

## Troubleshooting

- \*\*Connection Issues:\*\* Ensure the host address is correct and the device is reachable.

- \*\*Log Display Delays:\*\* Check network latency and ensure the application is not overloaded with too many connections.

## Future Enhancements

- \*\*Search and Filter Logs:\*\* Implement functionality to search and filter logs within each tab.

- \*\*Save Logs:\*\* Add the ability to save logs to a file for later analysis.

- \*\*Custom Commands:\*\* Allow users to send custom commands to the connected devices.